codurance | CRAFT AT HEART

Das Modernisierungs-strategie-Handbuch:

Ein Leitfaden zur Nutzung des Software-Modernisierungs-Canvas

von José E. Rodríguez Huerta
Spain Managing Director at Codurance

codurance.com

Über den Autor

José Enrique Rodríguez Huerta ist Managing Director von Codurance Spain, verantwortlich für die Strategie und das Wachstum des Unternehmens, verbindet er eine ausgeprägte Geschäftsperspektive mit tiefem technischem Verständnis. Im Laufe seiner Karriere hat er zahlreiche Organisationen in komplexen Technologie-Modernisierungsprozessen begleitet, mit einem besonderen Fokus auf Software-Nachhaltigkeit, Kultur der kontinuierlichen Verbesserung und Exzellenz in der Wertlieferung.

Neben seiner Tätigkeit als Führungskraft ist José Enrique eine anerkannte Persönlichkeit im Bereich des technischen Coachings. Er beleuchtet die Herausforderungen dieser Rolle realistisch – von den Risiken überhöhter Erwartungen bis hin zur Notwendigkeit von Balance zwischen Teamführung und Autonomie.

Er teilt sein Wissen regelmäßig auf Konferenzen, in Podcasts und in professionellen Fachforen. Dabei hat er zentrale Themen der Technologiebranche aufgegriffen: den Einfluss künstlicher Intelligenz auf Geschäftsmodelle, die Weiterentwicklung von Führungsrollen in agilen Umgebungen, technische Beratung als Treiber für Transformation sowie die Modernisierung von Unternehmenssystemen.

In den vergangenen Jahren hat er einen großen Teil seiner Arbeit Projekten im Umfeld von Mergers & Acquisitions (M&A) gewidmet und besonderes Know-how im Bereich Product & Technology Due Diligence (PTDD) aufgebaut. Sein Beitrag konzentriert sich darauf, Investoren bei der Risikoreduktion zu unterstützen, die Wertschöpfung zu beschleunigen und nachhaltige Chancen in einem komplexen, hochstrategischen Umfeld zu schaffen, in dem technische Exzellenz maßgeblich für den Erfolg einer Transaktion ist.

Sein kontinuierliches Engagement folgt einer klaren Überzeugung: Technologie entfaltet ihren nachhaltigen Nutzen für Organisationen nur dann, wenn sie von soliden Engineering-Praktiken, einer lernorientierten Kultur und bewusstem Leadership begleitet wird.

José E. Rodríguez Huerta
Spain Managing Director at Codurance
jrhuerta
jrhuerta@codurance.com

Einführung

Die Modernisierung von Software ist für moderne Unternehmen von entscheidender Bedeutung – nicht zuletzt aufgrund der wachsenden Abhängigkeit von Informationssystemen und der schnellen technologischen Entwicklungen. Systeme, die oft über viele Jahre hinweg gewachsen sind, werden zunehmend komplex und fragil, bleiben jedoch für den Geschäftsbetrieb unverzichtbar.

Modernisierung bedeutet nicht einfach, veraltete Technologien zu ersetzen. Sie ist ein strategischer Prozess, der darauf abzielt, Systeme so weiterzuentwickeln, dass sie die geschäftliche Agilität erhöhen und Risiken reduzieren. Im Kern geht es darum, die Fähigkeiten der IT mit den übergeordneten Zielen des Unternehmens in Einklang zu bringen.

Eine gute Modernisierungsstrategie sollte:

- **Ergebnisorientiert sein:** Sie konzentriert sich auf messbare geschäftliche Vorteile statt auf rein technische Updates.
- **Schrittweise und anpassungsfähig sein:** Sie vermeidet riskante „Big-Bang"-Ansätze und setzt stattdessen auf priorisierte, inkrementelle Verbesserungen.
- **Stakeholder-zentriert sein:** Sie bezieht Geschäfts-, Technik- und Betriebsteams von Anfang an ein.
- **Risikobewusst sein:** Sie identifiziert Sicherheits-, Budget- und Betriebsrisiken frühzeitig und begegnet ihnen proaktiv.

Die Planung einer Modernisierungsinitiative ist jedoch alles andere als trivial.

Viele IT-Verantwortliche stehen vor Herausforderungen, weil Entscheidungen isoliert, auf Basis veralteter Annahmen oder ohne ein strukturiertes Rahmenwerk getroffen werden. Ohne eine klare Strategie riskieren Teams, Millionen in Technologie zu investieren, die nicht mit den tatsächlichen Geschäftsanforderungen übereinstimmt.

Ziel dieses Dokuments ist es, das **Software Modernization Canvas** vorzustellen: eine einseitige Vorlage, die IT-Führungskräften hilft, ihre Modernisierungsstrategie schnell zu klären, zu strukturieren, zu validieren und zu kommunizieren.

Der *Software Modernisation Canvas* ist darauf ausgelegt, typische Fehler zu vermeiden, indem es die Modernisierungsplanung in sechs miteinander verknüpfte Bereiche gliedert und so einen ganzheitlichen Blick gewährleistet. Diese Bereiche führen Teams logisch durch den Prozess – von der Analyse des geschäftlichen Kontexts bis zur Definition eines klaren Umsetzungsansatzes – und stellen sicher, dass kein kritischer Aspekt übersehen wird.

Überblick über das Tool

Der *Modernisation Canvas* ist so aufgebaut, dass es den typischen Ablauf einer Modernisierungsinitiative widerspiegelt. Es unterstützt Teams dabei, die zentralen Fragen zum richtigen Zeitpunkt zu adressieren und vorschnelle Entscheidungen zu vermeiden. Jede Sektion ist bewusst positioniert, um eine logische und fundierte Entscheidungsfindung zu fördern:

- **Unternehmenskontext (1. Sektion):** Legt die Grundlage und klärt, warum die Initiative existiert.
- **Bestandsanalyse der Alt-Systeme (2. Sektion):** Bietet eine realistische Einschätzung der aktuellen Herausforderungen, bevor Lösungen geplant werden.
- **Modernisierungsziele (3. Sektion):** Definiert, was Erfolg bedeutet, und stellt sicher, dass die Bemühungen auf messbare Ziele ausgerichtet sind.
- **Modernisierungsansatz (4. Sektion):** Beschreibt den Umsetzungsplan unter Berücksichtigung notwendiger Entscheidungen und Kompromisse.
- **Ressourcen und organisatorische Aspekte (5. Sektion):** Stellt sicher, dass der Plan angesichts von Rahmenbedingungen und Einschränkungen realistisch ist.
- **Risikomanagement (6. Sektion):** Antizipiert kritische Risikopunkte und sichert den langfristigen Erfolg der Initiative.

Diese sechs Bereiche bilden zusammen das Kernstück jeder erfolgreichen Modernisierungsstrategie.

Jeder Bereich des Canvas wurde gezielt für einen bestimmten Zweck entwickelt. Ein kurzer Überblick über die Hauptsektionen erleichtert das Verständnis dieser Ziele. Von links nach rechts und von oben nach unten:

1. Unternehmenskontext
Hier wird die aktuelle Geschäftssituation betrachtet, die die Modernisierung notwendig macht. Diese Sektion ist entscheidend, da sie die Grundlage dafür bildet, was erreicht werden soll. Ist dieser Punkt unklar, sinken die Erfolgschancen der Initiative erheblich – er ist die *raison d'être* des Projekts.

2. Bestandsanalyse der Alt-Systeme
Bietet einen Überblick über die bestehenden Systeme, die transformiert werden sollen, und identifiziert sowohl die zentralen Herausforderungen als auch bereits erkannte Chancen. Beantwortet die Frage: Wo stehen wir heute?

3. Modernisierungsziele
Definiert den Erfolg der Initiative, legt die zu erreichenden Ergebnisse fest und gibt eine Vorstellung vom gewünschten Endzustand der Systeme. Beantwortet die Frage: Wohin wollen wir?

4. Modernisierungsansatz
Beschreibt den Plan zur Umsetzung der Modernisierung und zur Erreichung des Zielzustands. Enthält die zentralen Leitlinien sowie die wichtigen Entscheidungen und Kompromisse, die die Strategie begleiten.

5. Ressourcen und organisatorische Aspekte
Analysiert die Rahmenbedingungen und Einschränkungen, unter denen die Modernisierung durchgeführt wird: Budget, Teamstruktur, Fähigkeiten und Stakeholder. Stellt sicher, dass der Plan realistisch und umsetzbar ist.

6. Risikomanagement
Identifiziert die wichtigsten Risiken der Initiative und die erforderlichen Strategien zu deren Abschwächung, um die langfristige Nachhaltigkeit des Projekts zu sichern.

In Kombination liefern diese Bereiche einen umfassenden Überblick über die Modernisierungsinitiative und dienen als Ausgangspunkt für die notwendigen Gespräche, um eine fundierte Bewertung, Planung und Kommunikation der Strategie auf hoher Ebene zu ermöglichen.

Anleitung zur Nutzung des Canvas

Der Hauptzweck des Modernisation Canvas besteht darin, die Ausrichtung und das Verständnis einer Modernisierungsstrategie zu erleichtern. Es bietet einen Rahmen, der die notwendigen Gespräche leitet, insbesondere in den frühen Phasen einer Initiative.

Er ist keine Zauberlösung.
Damit der Canvas wirklich nützlich ist, müssen bestimmte Bedingungen erfüllt sein:

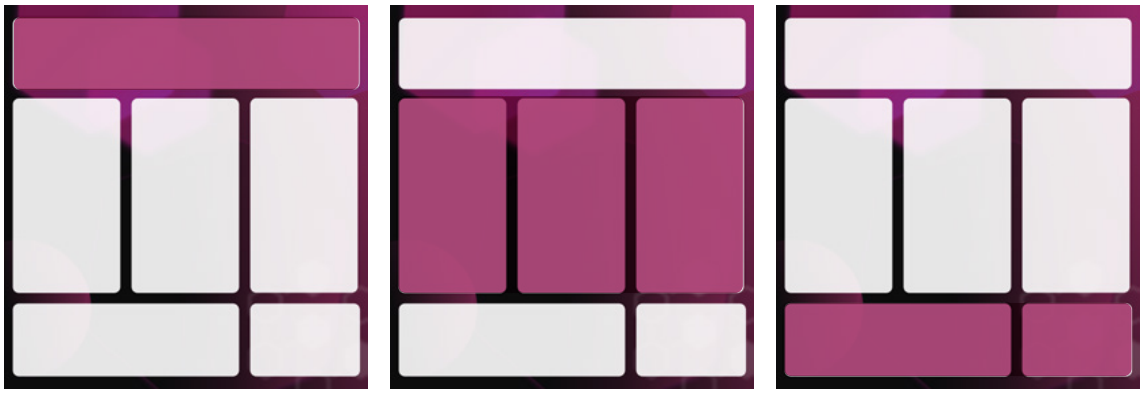
- **Die richtigen Personen müssen im Raum sein;** andernfalls geht der Sinn der Abstimmung verloren.
- **Ein erfahrener Moderator ist erforderlich;** andernfalls können Diskussionen leicht abschweifen und das eigentliche Ziel könnte aus den Augen verloren werden.

Die zentrale Idee des Canvas besteht darin, seine verschiedenen Bereiche nacheinander zu durchlaufen und die wichtigsten Themen zu adressieren. Ziel der Diskussion ist es, die Teilnehmer aufeinander abzustimmen und die entstehenden Erkenntnisse festzuhalten.

In jedem Abschnitt dieses Buches werden diese Erkenntnisse detailliert erläutert und durch praktische Beispiele veranschaulicht. Vorab lassen sich einige der häufigsten Erkenntnisse zusammenfassen:

- Bereiche, in denen weitere Informationen erforderlich sind, um fundierte Entscheidungen treffen zu können (z. B. Budgetfragen).
- Bereiche mit mehreren Optionen, für die zusätzliche Analysen oder sogar Proof-of-Concepts notwendig sind.
- Bereiche, in denen konkrete Entscheidungen getroffen werden müssen.
- Fehlende Schlüsselakteure in den Gesprächen und die möglichen Auswirkungen auf den Erfolg oder Misserfolg der Initiative.

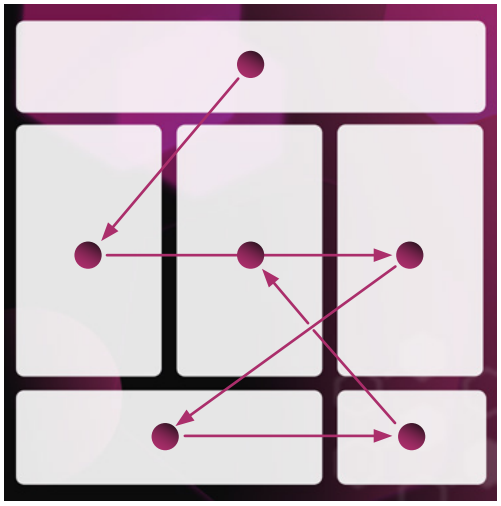
Beim Betrachten des Canvas erkennt man, dass es aus drei sich gegenseitig verstärkenden Hauptbereichen besteht:



Die drei Hauptbereiche des Canvas

Diese Bereiche werden schrittweise aufgebaut: Sie beginnen bei den Einschränkungen, gehen weiter über die Umsetzung und enden bei den Zielen, die erreicht werden sollen.

- Der erste Bereich behandelt den Geschäftskontext, der unterstützt werden soll. In der Regel umfasst er Ziele, Prozesse, die verändert werden sollen, und den Zweck der Initiative.
- Der zweite Bereich befasst sich damit, vom Punkt A zum Punkt B zu gelangen: A ist der aktuelle Stand, B der gewünschte Zielzustand.
- Die letzte Gruppe von Bausteinen ermöglicht es, dieses Ziel zu erreichen. Normalerweise handelt es sich um einzuhaltende Einschränkungen, wie z. B. das Budget, oder um proaktive Maßnahmen, die notwendig sind, um den Erfolg sicherzustellen.



Die empfohlene Reihenfolge, um die verschiedenen Abschnitte des Canvas zu durchlaufen, ist wie folgt:

Bei einer ersten Durchsicht ist es nicht unbedingt erforderlich, alle Bausteine auszufüllen; oft ist es einfacher, zu einigen später zurückzukehren, nachdem man in anderen Bereichen Fortschritte gemacht hat.

Der Plan zur Umsetzung sollte nur dann diskutiert werden, wenn ein ausreichendes Verständnis der Gesamtsituation besteht, um fundierte Entscheidungen treffen zu können. Praktisch bedeutet dies, dass die Arbeit am Plan erst beginnen sollte, wenn der notwendige Kontext vorhanden ist.

In Kapitel 7 werden einige mögliche Vorgehensweisen vorgestellt, die sich als sehr nützlich erweisen können.

Wie man dieses eBook effektiv nutzt

Jedes Werkzeug entfaltet seinen Wert nur, wenn es richtig eingesetzt wird. Ein Business-Plan kann Wachstum vorantreiben – oder nutzlos bleiben, wenn er falsch angewendet wird. Dieser Canvas funktioniert nach demselben Prinzip.

Dieses E-Book dient als Nachschlage- und Schulungsmaterial und soll Ihnen helfen, die Methode sowie ihre verschiedenen Aspekte klar zu verstehen.

Wo immer möglich, geben wir Beispiele und Hinweise, was Sie erwarten können. Gleichzeitig bleibt es Ihnen überlassen, das Material so zu nutzen, wie es für Sie am nützlichsten ist.

Wie man lernt, das Werkzeug zu benutzen

Ich bin ein starker Befürworter des Lernens durch praktische Erfahrung, daher schlage ich Ihnen zwei Ansätze vor, die je nach Ihrem Lernstil und Ihrer Toleranz für Chaos gut funktionieren können. Beide haben Vor- und Nachteile.

Lernansatz: Sehen, Machen, Lehren (mit Anleitung)
Nach der in der Medizin und bei Polizeikräften üblichen Methode *SODOTO** ist dieser Ansatz besonders nützlich. Die Idee ist: Zuerst studiert man das Material, siehst sich eine aufgezeichnete Sitzung an oder nimmt daran teil; danach versucht man, selbst eine Sitzung oder einen Teil davon zu leiten; und schließlich übt man, es anderen beizubringen, was den eigenen Lernprozess stärkt.

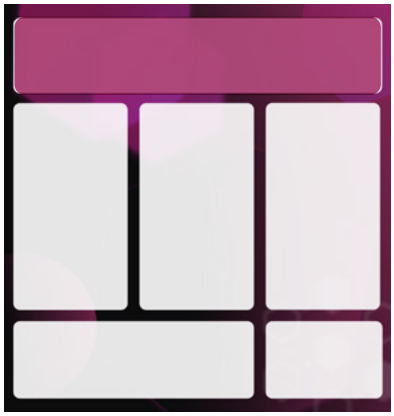
Lernansatz: Komplett auf eigene Faust
Dieser zweite Ansatz unterscheidet sich, da er weniger Orientierung bietet. Meine Empfehlung: Versuchen Sie zunächst, die Übung alleine durchzuführen – in der Regel ohne sie für andere zu leiten – entweder anhand eines aktuellen Falls oder eines, an dem Sie zuvor gearbeitet haben. Anschließend überprüfen Sie die Dokumentation, um Ihr Verständnis zu kontrollieren, passen Ihre Eintragungen im Modell an und vergleichen diese mit Ihren bisherigen Ergebnissen.

Der nächste Schritt besteht darin, das Gelernte mit anderen zu testen – entweder indem Sie sie bitten, Ihr Verständnis zu prüfen, oder indem Sie den Workshop selbst leiten. Ein letzter Tipp: Bei den ersten Versuchen sollten Sie vermeiden, Kunden als Teilnehmer einzusetzen, es sei denn, Sie sind sich Ihrer Sache sehr sicher.

*SODOTO: Akronym für See One, Do One, Teach One, ein praxisorientierter Lernansatz, der in der Medizin und von Polizeikräften häufig verwendet wird.

Kapitel 1:

Das Geschäftsumfeld verstehen

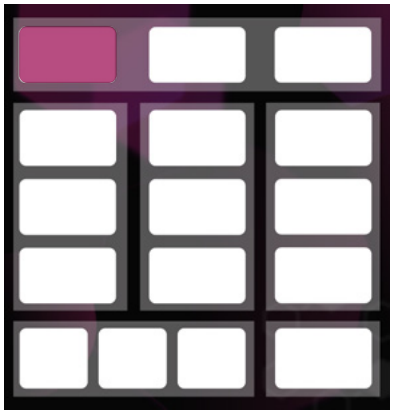


Wie bereits erwähnt, ist dies der eigentliche Zweck der Modernisierungsinitiative.

Es ist sehr häufig, dass Modernisierungsinitiativen gestartet werden, ohne das Geschäftsumfeld, in dem sie umgesetzt werden, wirklich zu verstehen. Wenn wir von „Unternehmen“ sprechen, meinen wir das, was aus wesentlicher Perspektive für das Unternehmen wichtig ist.

Dies wird höchstwahrscheinlich auf eine Form von „Profitabilität steigern“ hinauslaufen, es sei denn, Sie arbeiten in einer NGO oder einer gemeinnützigen Organisation,

Ziele und Vorhaben: Abstimmung der Modernisierungsbemühungen auf die Unternehmensprioritäten



Das Ziel dieses Abschnitts ist es, dass die Teilnehmer die allgemeinen Geschäftsziele klar formulieren, die durch die Modernisierung unterstützt werden sollen (z. B. Umsatzsteigerung, Verbesserung der Kundenzufriedenheit, Reduzierung der Time-to-Market oder Einhaltung gesetzlicher Vorschriften)

Unterschied zwischen Zielen und Vorhaben
Ein kurzer Hinweis: Ziele und Vorhaben sind nicht identisch – auch wenn sie zusammenhängen und oft fälschlicherweise synonym verwendet werden.

Ein Ziel ist ein erreichbares Ergebnis, das im Allgemeinen breit gefasst und langfristig ist, während ein Vorhaben kurzfristig angelegt ist und messbare Maßnahmen definiert, um ein übergeordnetes Ziel zu erreichen.

Einige Beispiele für Ziele sind:

- 20 Mio. Euro Umsatz in diesem Jahr erreichen
- Eine Marge von 20 % in allen Servicebereichen erzielen

Einige Beispiele für Vorhaben sind:

- Ein neues Produkt im ersten Quartal auf den Markt bringen
- Eine Marge von 20 % in allen Projekten halten, indem Verzögerungen reduziert werden

Unterschied zwischen diesen Zielen und denen im Abschnitt „Modernisierungsziele“

Noch ein kurzer Hinweis: Besonders im Austausch mit technischen Profilen ist es üblich, sofort an die Ziele oder Kennzahlen der Initiative oder des Systems zu denken. Diese Ziele sind ebenfalls wichtig, aber sie werden nicht als Erstes definiert. Hier sprechen wir über unternehmerische Ziele, also jene, die für das Gesamtunternehmen – oder in geringerem Maße für die jeweilige Geschäftseinheit – festgelegt werden.

Eine der größten Herausforderungen bei Modernisierungsinitiativen besteht darin, dass es häufig nicht gelingt, die Initiative mit dem tatsächlich angestrebten Unternehmensziel zu verknüpfen. Solange das nicht klar ist, wird es schwierig sein, die Initiative intern zu verkaufen, insbesondere gegenüber nicht-technischen Stakeholdern, und die notwendige Unterstützung – wie beispielsweise die Finanzierung – sicherzustellen.

Liste häufiger Geschäftsziele und KPIs

Zur Orientierung finden Sie hier einige typische Geschäftsziele und KPIs:

- 20 neue Märkte vor Jahresende eröffnen
- Den Umsatz um 10 % steigern, indem unterversorgte Marktsegmente entwickelt werden
- Die operativen Kosten um 10 % senken

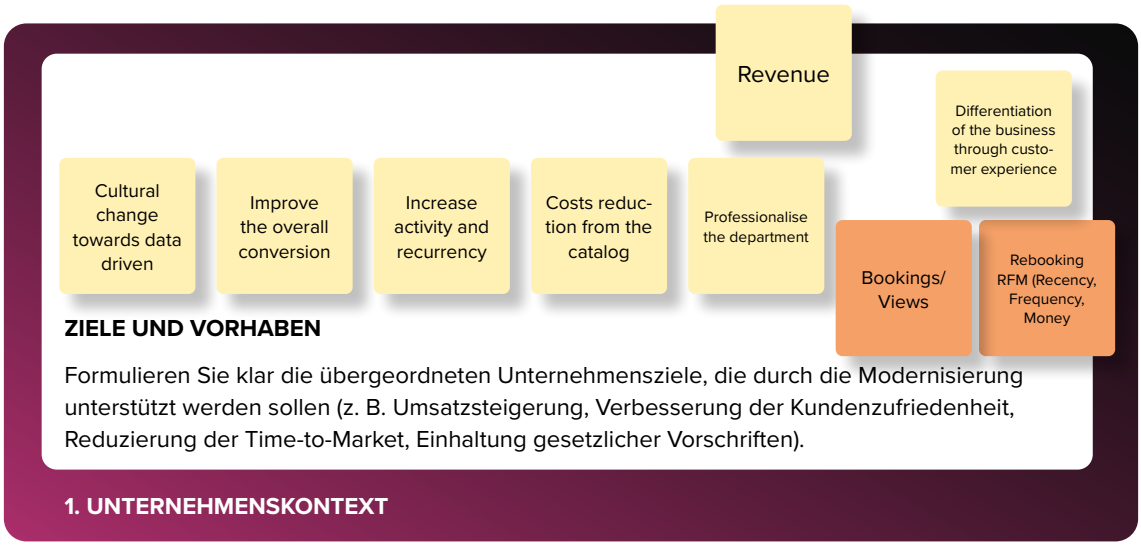
Warum mit diesem Punkt beginnen?

Ein wesentlicher Aspekt dieses Abschnitts ist zu verstehen, was das Unternehmen zu erreichen versucht. Denn das, was Teil der Unternehmensstrategie ist, erhält Priorität und entsprechende Unterstützung.

Beispiel

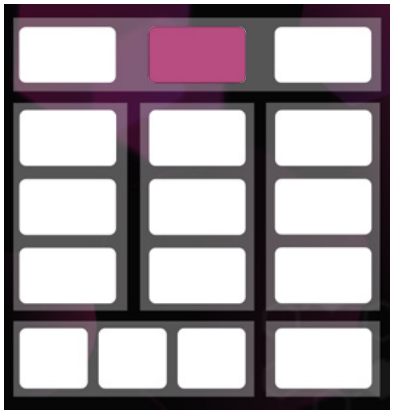
Das folgende Beispiel bezieht sich auf eine E-Booking-Plattform, bei der der Marketingleiter zuvor die Data-Science-Funktion verantwortete. Nach seinem Weggang entschied seine Nachfolgeperson, Änderungen in den Prozessen vorzunehmen.

Beispiel: Ziele und Vorhaben – Workflow im Bereich Data Science



Wie man sehen kann, haben einige Vorhaben nicht unbedingt einen zugehörigen Indikator. Meine Empfehlung ist, dies als Teil der Arbeit zu dokumentieren und sicherzustellen, dass der entsprechende Indikator definiert ist, bevor die Initiative startet.
Das wird entscheidend sein, um den ROI, die Auswirkungen und den Fortschritt der Modernisierung nachweisen zu können.

Wesentliche Geschäftstreiber: Prozesse und Systeme analysieren, um Abhängigkeiten zu verstehen



In diesem Abschnitt werden die geschäftskritischen Herausforderungen oder Chancen beschrieben, die den Bedarf an Modernisierung auslösen – beispielsweise der Wettbewerbsdruck im Markt, der Bedarf an Skalierbarkeit oder die Einhaltung regulatorischer Anforderungen.

Unterschied zwischen Treibern und Zielen

Vielleicht fragen Sie sich: Warum sowohl Treiber als auch Ziele verwenden? Wirkt das nicht etwas redundant?

Die Antwort liegt in der Definition eines Treibers: Ein Treiber ist ein Ereignis, das stattgefunden hat, stattfindet oder stattfinden wird und das alles Weitere auslöst.

Einige Beispiele für Treiber sind:

- Neues Führungsteam übernimmt die Leitung der Abteilung
- Rückgang der Unternehmensmargen
- Kürzliche Ausfälle oder Vorfälle, die den Ruf oder die Einnahmen erheblich beeinträchtigt haben
- Wachstumsstagnation

Liste typischer Gruppen von Treibern (Impulsfaktoren)

Beim Analysieren dieser Treiber lassen sich oft bestimmte Muster erkennen. Ohne Anspruch auf Vollständigkeit finden Sie hier die wichtigsten Kategorien, aus denen sie üblicherweise hervorgehen:

Nachhaltiges Wachstum: Konzentriert sich darauf sicherzustellen, dass das Unternehmen weiter wächst und dieses Wachstum auch halten kann – unter Berücksichtigung der Geschwindigkeit, mit der es expandieren möchte.

Nachhaltige Veränderung: Bezieht sich auf die Notwendigkeit, die Zusammenarbeit der Teams und die Art und Weise der Softwarebereitstellung zu verbessern. Dazu gehört das Beheben von Ineffizienzen, Lieferproblemen und hohen Wartungskosten. Das Ziel ist es, ein System zu schaffen, in dem sich Funktionen einfach hinzufügen oder ändern und ohne Reibung in Produktion bringen lassen.

Kostenreduzierung: Technologie kann genutzt werden, um Kosten zu senken – etwa durch die Abschaffung alter, proprietärer oder herstellerabhängiger Systeme.

Risikomanagement: Besonders relevant in regulierten Branchen. Es geht darum, sicherzustellen, dass Systeme weiterhin den Vorschriften entsprechen, während sie sich weiterentwickeln, und umfasst Themen wie Sicherheit und Skalierbarkeit.

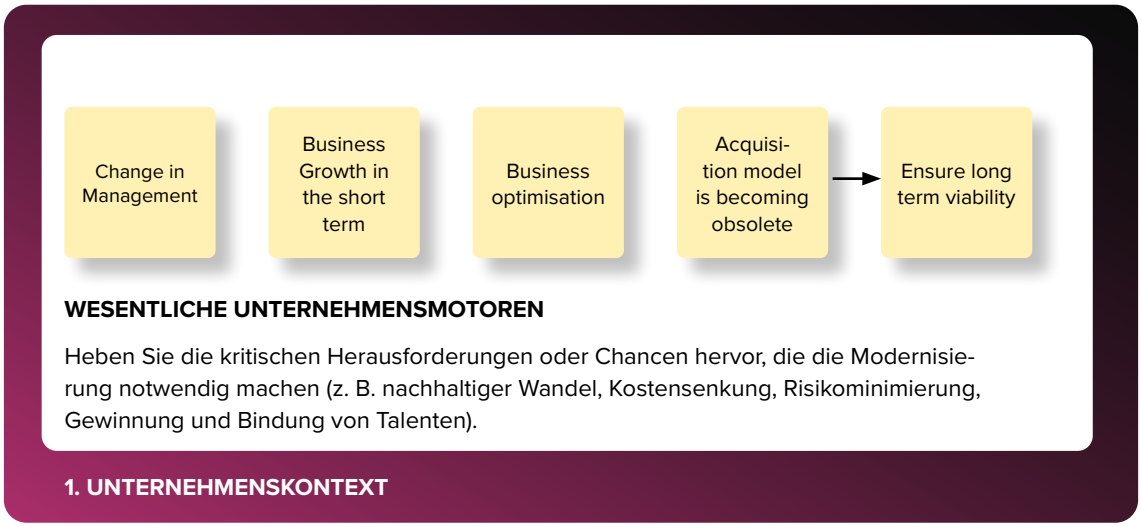
Ausrichtung auf das Geschäft: Bezieht sich darauf sicherzustellen, dass Technologie und Geschäftsziele miteinander im Einklang stehen, insbesondere im Hinblick auf das Conway'sche Gesetz.

Geschäftliche Agilität und Geschwindigkeit: Modernisierung soll es dem Unternehmen ermöglichen, sich schneller zu bewegen. Dies kann bedeuten, Einschränkungen zu entfernen, die die schnelle Einführung neuer Funktionen behindern, oder neue Zahlungsmethoden und andere Veränderungen zu ermöglichen.

Experimentieren und Innovation: Hier geht es um Treiber, die Innovation erleichtern sollen, indem sie Proofs of Concept und das Experimentieren mit neuen Ideen vereinfachen.

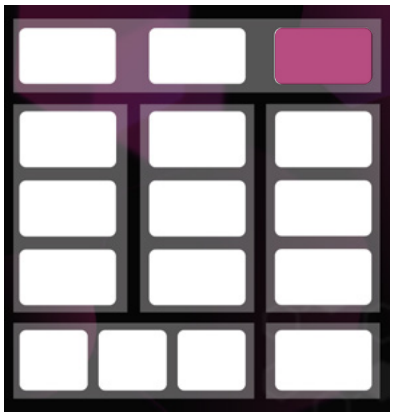
Menschen und Kultur: Umfasst den menschlichen Aspekt der Unternehmensstrategie. Typischerweise hängt er mit den Verhaltensweisen zusammen, die die Organisation fördern möchte, um ihre Strategie zu unterstützen.

Beispiel: Wesentliche Unternehmensmotoren – Data-Science-Workflow



Wie bereits erwähnt, umfassen einige der genannten Treiber sowohl geschäftliche Faktoren als auch situative Aspekte, wie zum Beispiel die Übernahme der Abteilung durch eine neue Führung.

Kritische Geschäftsprozesse: Prozesse und Systeme abbilden, um ihre Abhängigkeiten zu verstehen



Dieser Abschnitt konzentriert sich darauf zu verstehen, welche Teile der Organisation – über den technischen Bereich hinaus – von der Modernisierungsinitiative betroffen sein werden. Dies ermöglicht es, frühzeitig zu erkennen, wer den Erfolg des Projekts blockieren, verzögern oder unterstützen könnte.

Wenn man dies versteht, lässt sich erkennen, wie die internen Mechanismen des Unternehmens beeinträchtigt werden könnten, und es wird möglich, frühzeitig zu planen, wie man damit umgehen sollte.

Zu den typischen Beispielen kritischer Geschäftsprozesse zählen unter anderem: Zahlungen, Versand, Bestandsauffüllung, Preisgestaltung, Marketingkampagnen und Kundenservice.

Welche Bereiche?

Im Folgenden finden Sie eine Liste von Bereichen innerhalb der Organisation, die direkt oder indirekt von der Modernisierungsinitiative betroffen sein können.

Betrieb und Ausführung

- Auftragsabwicklung
- Bestandsmanagement
- Koordination der Lieferkette
- Logistik und Versand
- Rücksendungen und Erstattungen

Finanzen und Buchhaltung

- Rechnungsstellung
- Umsatzrealisierung
- Ausgabenmanagement
- Budgetierung und Forecasting
- Procure-to-Pay-Zyklus

Vertrieb und Kundenmanagement

- Kundenbeziehungsmanagement (CRM)
- Angebotserstellung und Verwaltung von Geschäftsabschlüssen

- Vertragslebenszyklusmanagement
- Kundenservice und technischer Support (Helpdesk)
- Kontoerstellung und -konfiguration (Onboarding)

Marketing

- Planung und Durchführung von Kampagnen
- Lead-Management
- Personalisierung und Empfehlungen
- Marketinganalyse und Reporting
- Verwaltung der digitalen Customer Experience

Personalwesen (HR)

- Recruiting und Onboarding neuer Mitarbeitender
- Lohn- und Gehaltsabrechnung
- Arbeitszeit- und Anwesenheitskontrolle
- Leistungsbeurteilung
- Schulung und Weiterentwicklung

Compliance, Risiko und Rechtliches

- Audit-Erfassung und Reporting
- Verwaltung von Datenschutz und Einwilligungen
- Einhaltung regulatorischer Anforderungen (z. B. GDPR, HIPAA)
- Dokumentenaufbewahrung und Legal Hold
- Identitäts- und Zugriffsmanagement

Daten und Analytik

- Business Intelligence und Reporting
- Data Warehousing und Data Lakes
- ETL/ELT-Prozesse (Pipelines)
- Machine-Learning- und AI-Pipelines
- Data Governance und Qualitätskontrolle

IT und Infrastruktur

- Deployment und Release-Management
- Monitoring und Incident Response
- Supportsysteme und Ticketmanagement
- Konfigurations- und Change-Management
- Backups und Disaster Recovery

Häufige Herausforderungen

Jeder dieser Bereiche kann von einer anderen Abteilung abhängen und daher eigene Herausforderungen mitbringen, die berücksichtigt werden müssen. Diese Schwierigkeiten sind nicht immer ausschließlich technischer Natur.

Einige Beispiele

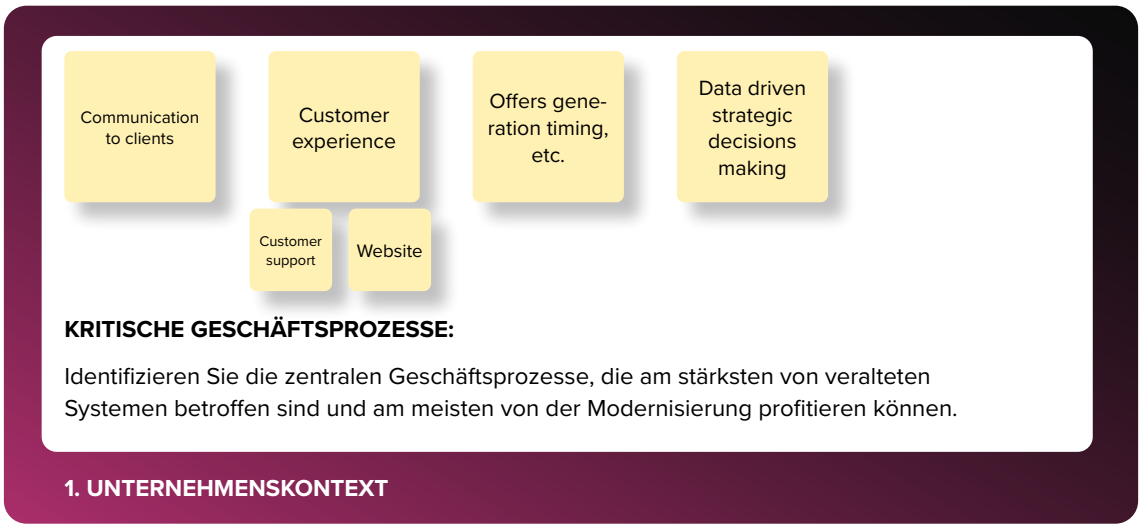
Geografisch verteiltes Personal: Beispielsweise Filialen in verschiedenen Ländern oder Entwicklungsteams, die über mehrere Kontinente verteilt sind.

Organisatorische Ziele der prozessverantwortlichen Abteilung: Finanzen, Compliance, Sicherheit oder Recht sind gute Beispiele für Bereiche mit eigenen Zielen innerhalb der Organisation, die oft zu spät in die Diskussion einbezogen werden – was Verzögerungen verursacht oder die Initiative sogar vollständig blockieren kann.

Manuelle Prozesse: Einige Abläufe können einen hohen manuellen Anteil haben, wie etwa die Beantwortung von Telefonanrufen oder die Überprüfung eines Fahrzeugmotors.

Wie man sieht, kann die Art dieser Herausforderungen sehr unterschiedlich sein und liegt oft außerhalb deines Einflussbereichs oder des Umfangs der Initiative.

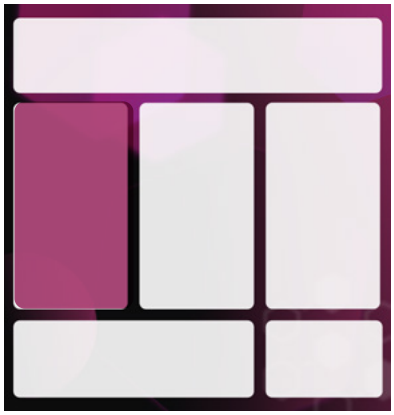
Beispiel: Kritische Geschäftsprozesse – Data-Science-Workflow



Dieses Beispiel zeigt ebenfalls schnell, dass andere Abteilungen wie Marketing und Kundenservice betroffen sein werden. Dies hilft dabei zu erkennen, was in der Stakeholder- und Kommunikationssektion berücksichtigt oder weiter untersucht werden sollte, und festzustellen, wer so früh wie möglich in die Gespräche einbezogen werden muss.

Kapitel 2:

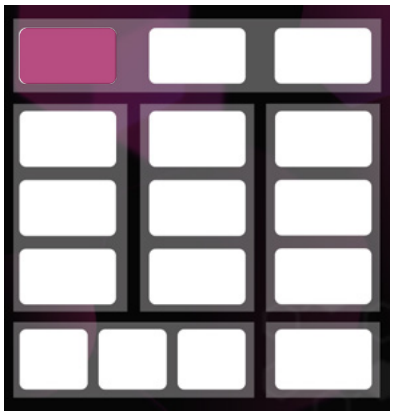
Bewertung der Altsysteme



Dieser Abschnitt des Canvas konzentriert sich auf den aktuellen Zustand des Systems. Ziel ist es, einen klaren Überblick zu geben und die wichtigsten Verbesserungsmöglichkeiten oder Risiken zu identifizieren, die im Rahmen der Modernisierungsinitiative adressiert werden müssen.

Die Idee besteht darin, dass dieser Abschnitt als Ausgangspunkt für die weitere Diskussion dient, indem er alle Beteiligten zusammenbringt, damit sie ihr Verständnis des aktuellen Zustands des Systems oder der Systeme teilen können, die modernisiert werden sollen.

Systemübersicht: Bewertung der Architektur und des Zwecks der Altsysteme



In diesem Abschnitt wird der aktuelle Zustand des Systems auf hoher Ebene untersucht. Ziel ist es, die strukturellen Muster zu klären und zu identifizieren, was funktioniert und was nicht funktioniert.

Es wird empfohlen, Informationen über das System im Voraus anzufordern, um bessere Ergebnisse zu erzielen. Dabei ist jedoch Vorsicht geboten, dass man sich nicht in komplexen Architekturdiagrammen oder veralteter Dokumentation verliert, wie im nächsten Abschnitt erläutert wird.

Was dieser Abschnitt nicht ist

Wie bereits erwähnt, kann das Anfordern von Informationen über das System den Eindruck erwecken, dass hier alle Details zum System festgehalten werden müssen. Das ist nicht der Fall.

Es werden nicht zu viele Informationen benötigt; halten Sie es einfach und verweisen Sie nur bei Bedarf auf Details.

Ein weiterer häufiger Trend ist, Systemattribute anzugeben, wie zum Beispiel: „Das System hat keine Tests“ oder „Es ist 7 Jahre alt“. Das gehört nicht in diesen Abschnitt.

Dieser Abschnitt soll eine ausreichende strukturelle Übersicht bieten, um definieren zu können, was als Zielarchitektur vorgeschlagen wird.

Aspekte wie fehlende Tests oder die Schwierigkeit, das System zu prüfen, passen besser in andere Abschnitte, wie Risiko oder technische Schulden.

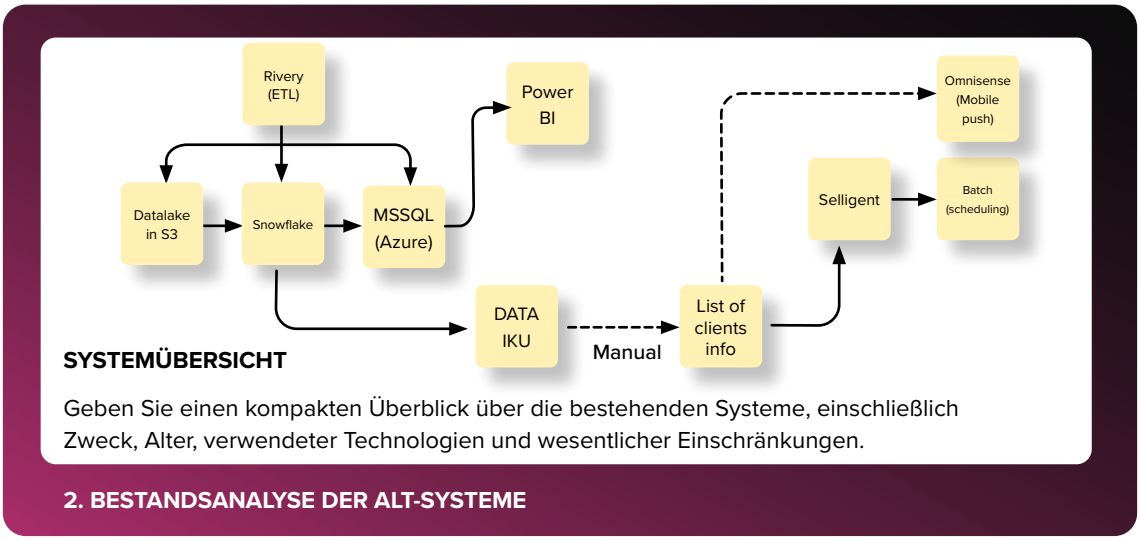
Wie detailliert sollte dieser Abschnitt sein?
Die kurze Antwort: so viel, dass man Folgendes versteht:

- Was das System macht
- Welche Hauptkomponenten es hat
- Welche allgemeinen Muster es verwendet
- Wo die wichtigsten Engpässe liegen
- Wo die größten Schmerzpunkte liegen
- Welche Haupttechnologien verwendet werden

Sie können Details hinzufügen oder weglassen, wie Sie es für nötig halten, aber eine gute Faustregel lautet: zu wenig Informationen können es erschweren, die Modernisierungsstrategie und den Ansatz korrekt zu planen.

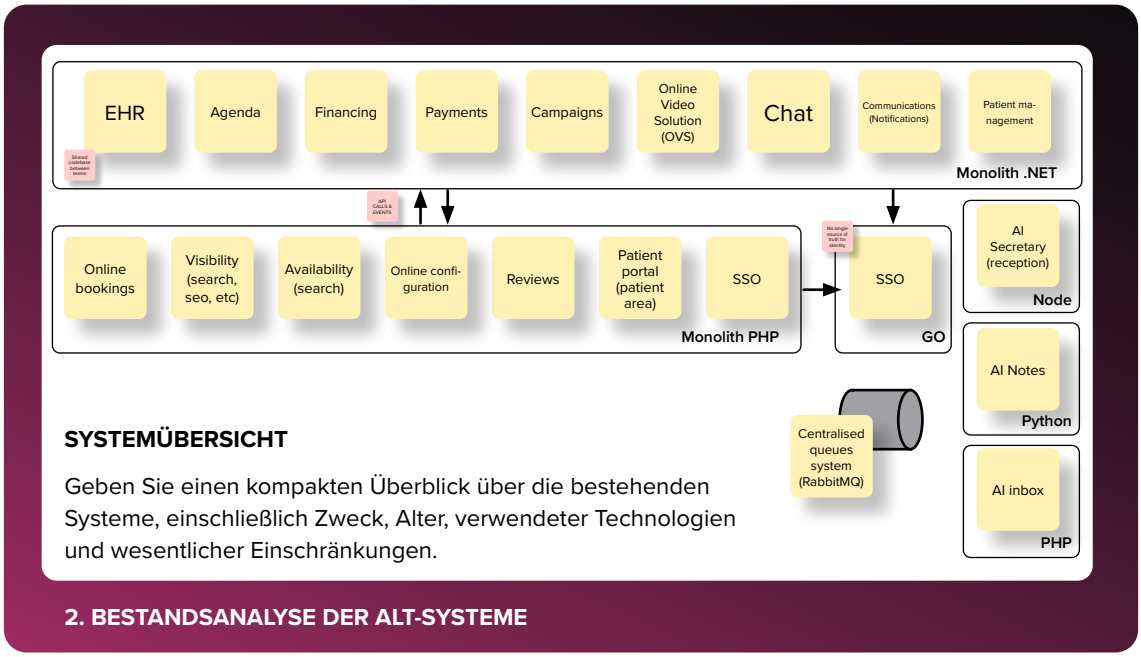
Eine solide Systemübersicht ermöglicht eine fundierte Diskussion über die Probleme, die durch die Modernisierung gelöst werden sollen.

Beispiel: Systemübersicht – Data-Science-Workflow



In diesem Beispiel beschrieb der Kunde seinen allgemeinen Workflow des Data-Science-Departments kurz und erwähnte die wichtigsten verwendeten Technologien und SaaS-Plattformen. Die gestrichelten Linien stellen manuelle Prozesse außerhalb jeglicher Systeme dar. Dieses Beispiel könnte durch die verwendeten Technologiearten in jedem Schritt ergänzt werden.

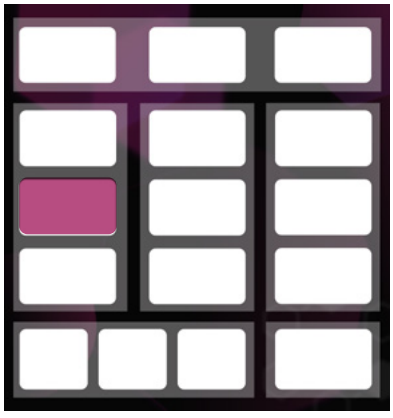
Beispiel: Systemübersicht – Doppelte SSO



Hier beschrieb der Kunde die Architektur seiner Systeme zusammengefasst, die Hauptmodule und deren Abhängigkeit von SSO-Systemen. Die roten Post-its heben Schmerzpunkte innerhalb der aktuellen Lösung oder Verbesserungspotenziale hervor.

Dieses Diagramm zeigt auch die Übergangsnatur der aktuellen Architektur, bei der die Altsysteme einer monolithischen Struktur folgen und die neuen Systeme kontextbezogen stärker autark sind, wie es bei KI-Systemen der Fall ist.

Technische Schulden und Risiken: Die Kosten und Risiken veralteter Systeme verstehen



In diesem Abschnitt werden die mit dem Altsystem verbundenen technischen Schulden dokumentiert, einschließlich veralteter Technologien, komplexem Code, Sicherheitslücken und potenzieller Risiken.

Technische Schulden beziehen sich auf die impliziten Kosten, die entstehen, wenn man sich jetzt für eine suboptimale oder einfachere Lösung entscheidet, anstatt einen besseren Ansatz zu wählen, der mehr Zeit in Anspruch nehmen würde.

Technische Schulden können verschiedene Risiken und Probleme bei der Softwareentwicklung und -wartung verursachen, und es ist wichtig zu verstehen, dass technische Probleme auch Geschäftsprobleme sind. Außerdem häufen sich technische Schulden im Laufe der Zeit an, und wenn sie nicht adressiert werden, nehmen die Probleme und Risiken nur zu.

Einige Beispiele für technische Schulden und damit verbundene Risiken:

Kodierte Prozesslogik ohne Dokumentation: Macht Systeme schwer verständlich, wartbar und modifizierbar, erhöht das Risiko, Fehler bei Änderungen einzuführen, und erschwert die Integration mit anderen Systemen.

Fehlende Dokumentation: Erschwert es Entwicklern, das System zu verstehen, und erhöht das Risiko von Fehlern bei Änderungen. Dieses Problem tritt häufiger bei Altsystemen auf und macht Migrationen riskanter.

Komplexe Systemarchitektur: Nicht-evolutionäre Architekturen erschweren das Hinzufügen neuer Funktionen oder die Anpassung an Geschäftsänderungen. Dies kann das System fragil und schwer wartbar und testbar machen.

Fehlende Tests: Das Fehlen von Tests erhöht das Risiko, fehlerhafte Software freizugeben, und erschwert Änderungen mit Vertrauen. Fehlende automatisierte Tests verhindern das frühe Erkennen von Defekten und erfordern mehr Aufwand für spätere Korrekturen.

Duplizierter Code: Kopieren und Modifizieren von Code erhöht das Risiko, Defekte im System zu verbreiten. Ein Fehler in einer Kopie ist wahrscheinlich auch in anderen vorhanden, und das Beheben an einer Stelle löst das Problem nicht in den anderen.

Veraltete Technologien: Die Verwendung nicht gewarteter Technologien stellt ein Geschäftsrisiko dar, da das System von Software abhängt, die keine Sicherheitsupdates oder Fehlerkorrekturen erhält. Außerdem erschwert es die Gewinnung und Bindung von Talenten, da Entwickler moderne Tools bevorzugen.

Monolithische Architektur: Große monolithische Anwendungen sind schwer zu ändern, zu testen und bereitzustellen, was die Geschäftsgeschwindigkeit einschränkt und Innovation erschwert. Fehlende Modularität kann Blockaden zwischen Teams verursachen.

Unnötige Nacharbeit: Häufiges Problem in Organisationen mit niedriger oder mittlerer IT-Leistung, die viel Zeit damit verbringen, Fehler zu beheben und Arbeit zu wiederholen. Diese Zeit könnte für neue Funktionen genutzt werden.

„Hot Spots“ im Code: Bereiche, die von vielen Teams modifiziert werden müssen, erzeugen Engpässe und erhöhen das Risiko von Konflikten und Fehlern.

Fehlende Fähigkeiten und Erfahrung: Das Fehlen von Experten für Altsysteme und von Personal mit ausreichenden Kompetenzen erhöht das Risiko von Fehlern und Projektversagen. Dies verschärft sich, wenn kein Wissen zwischen Teams übertragen wird oder implizites Wissen verloren geht.

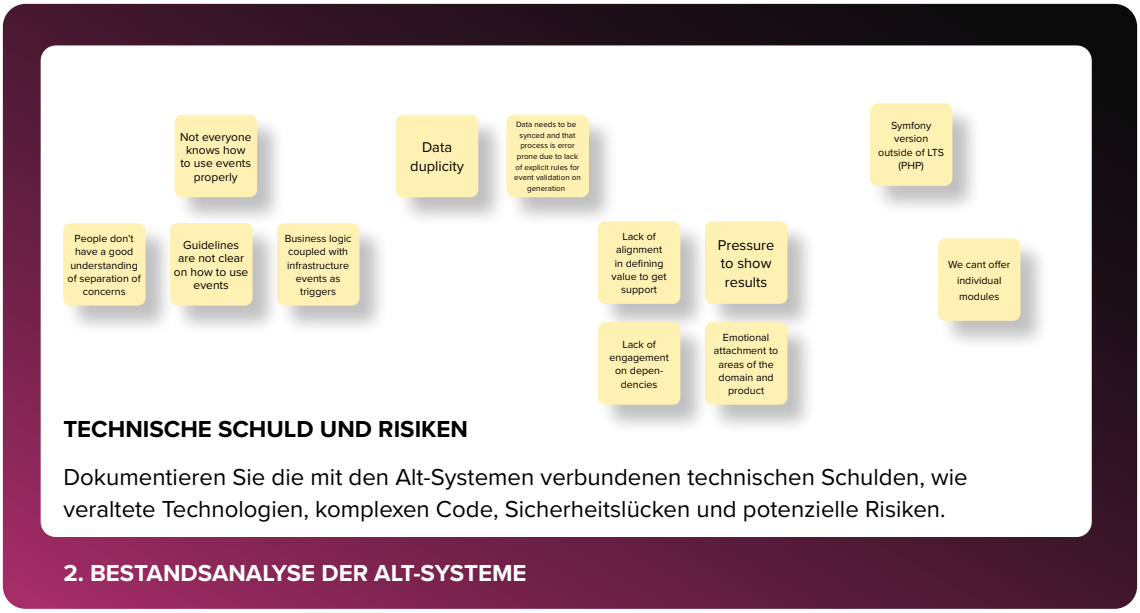
Sicherheitslücken: Schwachstellen in Altsystemen oder Drittanbieter-Code bergen erhebliche Risiken, z. B. Exploits wie Log4J. Unbekannte Schwachstellen können ausgenutzt werden, wenn sie nicht behoben werden, und je länger die Behebung dauert, desto größer ist das Risiko für die Nutzer.

Installationsdefekte: Probleme bei der Installation entstehen oft durch Schnittstellen zu Altsystemen oder die Interoperabilität mit vorinstallierter Software, was Produktivität, Umsatz und Kosten beeinträchtigt.

Veraltete oder starre Technologie: Altsysteme, insbesondere in alten Sprachen wie COBOL, sind oft unflexibel und schwer an neue Anforderungen anzupassen. Außerdem ist es schwierig, ausreichend qualifiziertes Personal dafür zu finden.

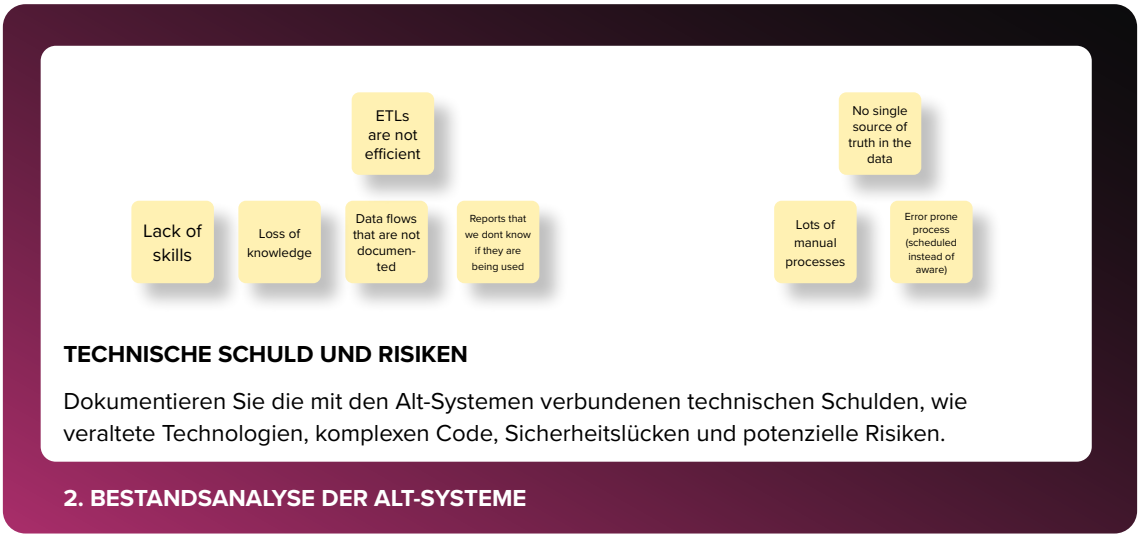
Fehlerhafte Korrekturen: Wenn ein Defekt behoben wird, kann die Korrektur neue Fehler erzeugen. Dies kann ein ernstes Problem werden, wenn es nicht sorgfältig verwaltet wird.

Beispiel: Technische Schulden und Risiken – Doppelte SSO



Einige Risiken und technische Schulden hängen nicht nur mit dem Code zusammen; einige können direkt mit den Modernisierungszielen verbunden sein.

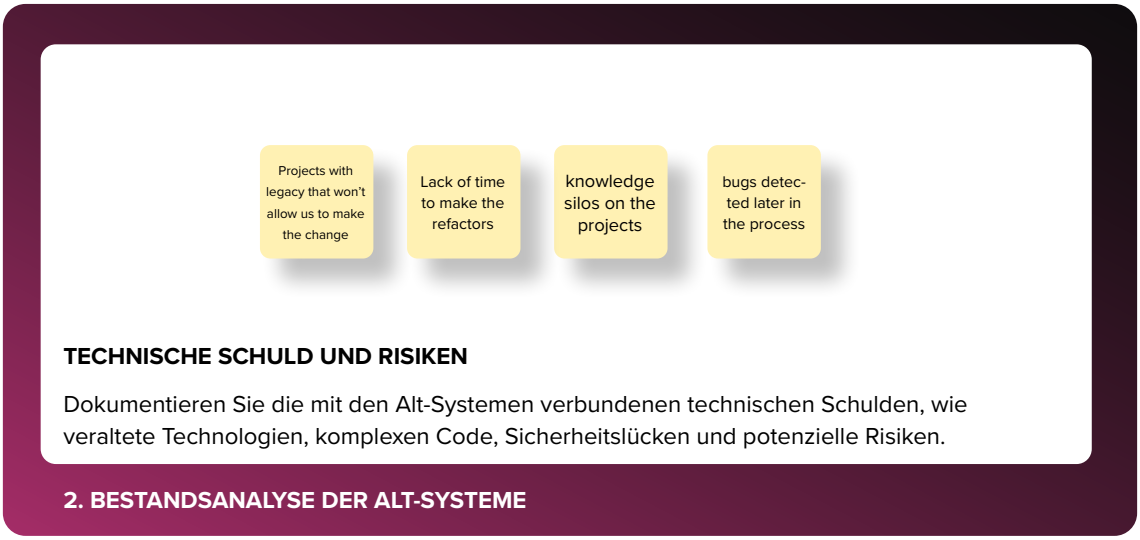
Beispiel: Technische Schulden und Risiken – Data-Science-Workflow



Einige Probleme stehen im Zusammenhang mit Arbeitsweisen, nicht unbedingt mit Werkzeugen. Fehlende Dokumentation und Wissensverlust können schwerwiegende Konsequenzen haben, wie z. B. die Stilllegung von Arbeiten.

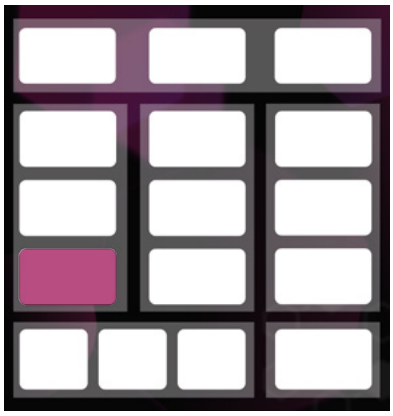
In diesem Fall sind bestimmte Prozesse nicht dokumentiert, und die Personen, die sie erstellt haben, arbeiten nicht mehr im Unternehmen. Das neue Team stößt auf ETL-Prozesse, bei denen es nicht weiß, ob sie noch verwendet werden, aber niemand entfernt sie aus Angst, etwas zu beschädigen. Diese „Paralyse durch Angst“ kommt bei nicht dokumentierten Altsystemen sehr häufig vor.

Beispiel: Technische Schulden und Risiken – Trunk Based Development



Dieses Beispiel zeigt, wie Wissensmanagement und Qualität als Risiken in der Modernisierung erscheinen können.

Performance- und Effizienzengpässe:
Kritische Engpässe identifizieren



Dieser Abschnitt konzentriert sich darauf, die Bereiche des Altsystems zu identifizieren, in denen Leistungsprobleme, Ineffizienzen oder Skalierungsbeschränkungen die Geschäftsabläufe beeinträchtigen.

Leistungs- und Effizienzengpässe können die Softwareentwicklung und -wartung sowie die allgemeinen Geschäftsabläufe erheblich behindern. Sie entstehen häufig durch technische Schulden, Architekturprobleme oder Ineffizienzen in Prozessen.

Einige Beispiele:

Langsame Deployment-Zeiten: Verlängerte Deployment-Zyklen können verhindern, dass das Unternehmen neue Funktionen oder Updates schnell an Kunden ausliefert. Dies kann an komplexen Deployment-Prozessen, fehlender Automatisierung oder veralteter Infrastruktur liegen. Beispielsweise ist ein Team, das Monate benötigt, um Änderungen bereitzustellen, weniger agil als eines, das mehrere Deployments pro Tag durchführen kann.

Es ist wichtig zu beachten, dass Geschwindigkeit relativ ist. In bestimmten Kontexten, wie regulierten Branchen, schränken externe Faktoren wie Audits oder Zertifizierungen die Deployment-Geschwindigkeit ein. In anderen Fällen ist die Automatisierung aufgrund spezieller Hardware oder Edge-Standorte nicht vollständig möglich.

Manuelle Prozesse: Manuelle Prozesse bei der Softwareentwicklung und -bereitstellung sind langsam und fehleranfällig. Zum Beispiel beanspruchen manuelle Tests erhebliche Zeit und Ressourcen. Fehlende automatisierte Tests verhindern die frühzeitige Fehlererkennung und erhöhen den Aufwand für Korrekturen später. Auch die manuelle Bereitstellung und Konfiguration von Infrastruktur verursacht Verzögerungen.

Monolithische Architektur: Große, monolithische Anwendungen sind schwer zu verwalten und zu deployen, was die Entwicklungszyklen verlangsamt und die Agilität reduziert. Änderungen in einem Teil der Anwendung können andere beeinflussen, wodurch die schnelle Einführung neuer Funktionen oder Fehlerbehebungen erschwert wird.

Abhängigkeiten zwischen Teams: Wenn mehrere Teams dieselben Codebereiche bearbeiten müssen, entstehen sogenannte „Hot Spots“. Dies kann Merge-Konflikte, Code-Blockierungen und Verzögerungen bei neuen Versionen verursachen und erhebliche Engpässe erzeugen.

Schlecht gestaltete APIs: Ineffiziente APIs können zu Engpässen werden, insbesondere wenn sie komplexe oder langsame Datenverarbeitung erfordern. Fehlende oder unzureichende Dokumentation verlangsamt ebenfalls die Entwicklung.

Unzureichende Tests: Fehlende, einschließlich automatisierter, Tests können zu häufigen Fehlern führen, kostspielige Nacharbeit verursachen und Zeit beanspruchen. Dieses Problem verschärft sich, wenn Testumgebungen nicht mit Produktionsumgebungen übereinstimmen oder Tests zu spät im Entwicklungszyklus durchgeführt werden.

Komplexe Build-Prozesse: Lange Build-Zeiten beeinträchtigen die Produktivität des Teams, insbesondere wenn der Prozess unzuverlässig ist und fehlerhafte Builds zusätzliche Zeit für Korrekturen erfordern.

Langsame Fehlerbehebung: Die Zeit, die benötigt wird, um Fehler zu identifizieren, zu beheben und zu deployen, kann ein Engpass sein, insbesondere wenn Fehler nicht frühzeitig erkannt werden. Fehler in der Produktion sind deutlich teurer zu beheben als solche, die während des Design-Reviews gefunden werden. Einige Fehler sind schwer reproduzierbar, was die Behebung erschwert.

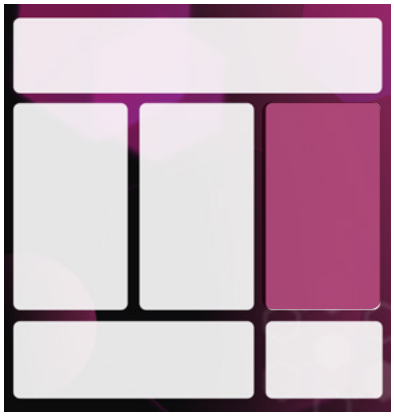
Datenübertragung oder -duplizierung: Die Datenmigration ist eine häufige Herausforderung bei der Systemmodernisierung. Informationen zwischen Systemen zu verschieben, kann komplex und zeitaufwendig sein und Engpässe verursachen.

Fehlende Experten oder Dokumentation für Altsysteme: Das Fehlen von Wissen über das bestehende System oder von erfahrenem Personal kann ein erhebliches Hindernis sein. Es kann notwendig sein, externe Experten während des Modernisierungsprojekts hinzuzuziehen, um den Code, die Prozesse und die alte Datenbank zu verstehen, was die Kosten erhöht.

Ineffiziente Teamstrukturen: Schlecht organisierte Teams können Engpässe erzeugen. Silos und fehlende Koordination zwischen Gruppen verlangsamen die Entwicklung durch unnötige Reibungen und Abhängigkeiten.

Kapitel 3:

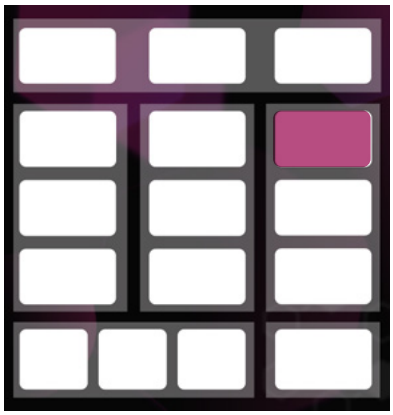
Definition der Modernisierungsziele



In diesem Kapitel analysieren wir die übergeordneten Ziele der Modernisierungsinitiative, wie sie mit den Geschäftszielen übereinstimmen, wie deren Fortschritt verfolgt werden kann und wie ein potenzieller Zielzustand des Systems aussehen könnte.

Dieser Abschnitt konzentriert sich hauptsächlich darauf, abzustimmen, was die Modernisierungsinitiative erreichen muss, um als erfolgreich zu gelten. Ähnlich wie beim aktuellen Systemzustand wird empfohlen, ein relativ hohes Detaillierungsniveau für die Zielarchitektur beizubehalten, damit sowohl technische als auch nicht-technische Stakeholder teilnehmen und den Umfang der Modernisierung verstehen können.

Erwünschte Geschäftsergebnisse: Die Modernisierung mit messbaren Mehrwert verknüpfen



Mangelnde Abstimmung ist einer der Hauptgründe, warum viele Modernisierungsinitiativen nicht die notwendige Unterstützung erhalten.

Heute hat sich der Markt verändert: In den meisten Organisationen treffen die Geschäftsbereiche die Kaufentscheidungen und verwalten das Budget. Dies führt dazu, dass viele Modernisierungsgespräche stärker auf den ROI (Return on Investment) als auf die Technologie fokussiert sind.

Dieser Abschnitt soll sicherstellen, dass die Ziele der Modernisierungsinitiative klar und greifbar direkt mit dem Geschäft verbunden sind.

Was macht ein gutes Ziel in diesem Abschnitt aus?
Neben der Einhaltung des bekannten SMART-Prinzips (Spezifisch, Messbar, Erreichbar, Realistisch und Terminiert) ist es wichtig, sich zu fragen, wie diese Ziele mit den zuvor definierten Geschäftszielen verbunden sind. Eine gute Frage lautet: Inwiefern helfen oder ermöglichen sie das, was die Organisation erreichen möchte?

Beim Formulieren ist es ideal, wenn die Ziele als erwartete Geschäftsergebnisse ausgedrückt werden können. Wenn dies nicht möglich ist, handelt es sich wahrscheinlich um zu technische Details (Implementierungsebene), die nicht zur Diskussion beitragen und nicht-technische Stakeholder sogar verwirren oder abschrecken könnten.

Beispiele für Formulierungen:

- Kosten in der AWS-Dateninfrastruktur reduzieren ➜ korrekt, als Geschäftsergebnis formuliert.
- Daten aus unserem Datalake von S3 nach Glacier migrieren ➜ inkorrekt, zu technisch und niedrigstufig.

Zu viele Ziele

Es kann vorkommen, dass zu viele Ziele formuliert werden. In diesem Fall:

- Halte die Ziele auf einem hohen Abstraktionsniveau.
- Wenn sie nach Implementierungsdetails aussehen, gehören sie nicht hierhin.
- Als Faustregel sollten nicht mehr als fünf Ziele definiert werden. Die Wahrscheinlichkeit, alle zu erreichen, sinkt drastisch nach drei.

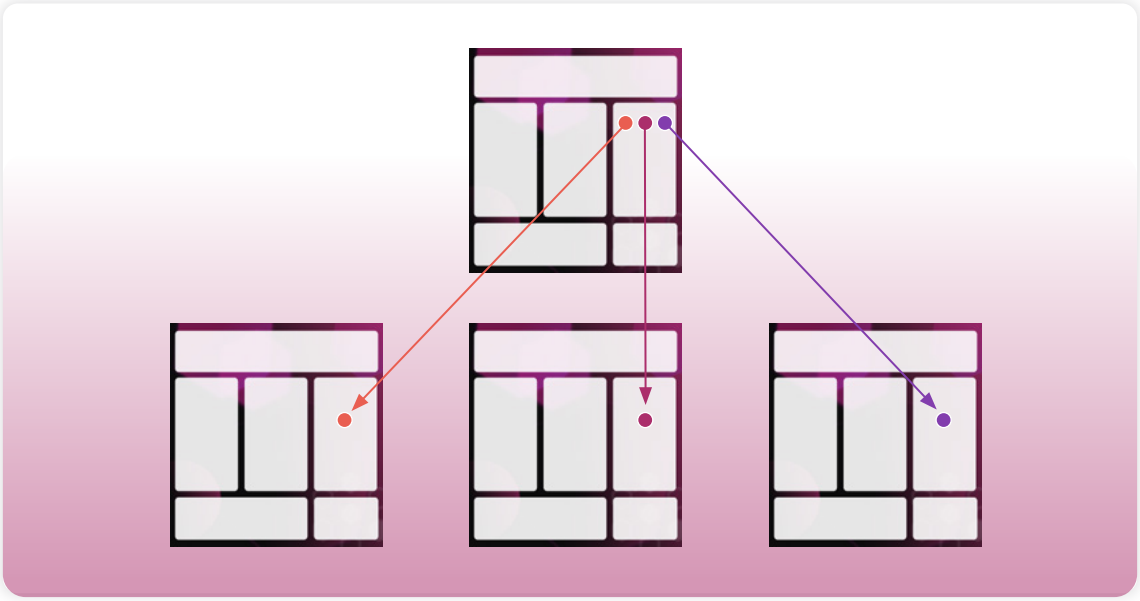
Wenn das Problem nicht die Anzahl der Ziele, sondern der zu große Umfang der Initiative ist, kann dies die Analyse des Canvas erschweren, insbesondere bei Einschränkungen wie Budget oder Fokus.

In diesem Fall empfiehlt es sich, den Canvas zu teilen, um den Umfang besser handhabbar zu machen und die Diskussion gezielter zu führen.

Canvas nach Zielen aufteilen

Mehrere Ziele deuten bereits darauf hin, dass es innerhalb der Modernisierungsinitiative mehrere Unterinitiativen geben könnte.

Beispielsweise könnte ein Ziel die Skalierbarkeit des Systems betreffen, während gleichzeitig Kosten reduziert oder die Liefergeschwindigkeit erhöht werden soll.

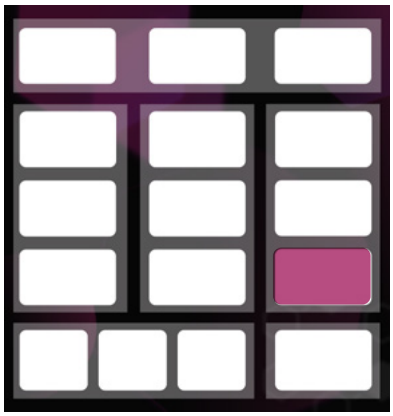


In diesem Fall könnten die Kosten durch die Migration bestimmter Systemteile auf Open-Source-Software gesenkt werden, während das Skalierbarkeitsproblem durch eine Neugestaltung der Architektur unter Nutzung nativer Cloud-Patterns gelöst wird. Die Verbesserung der Liefergeschwindigkeit könnte durch mehr Automatisierung zur Reduzierung von Engpässen erreicht werden.

Jede dieser Initiativen kann unabhängig untersucht werden, um die Arbeit besser zu priorisieren, Budgetanforderungen zu verstehen, betroffene Prozesse zu identifizieren und weitere Aspekte zu berücksichtigen.

An diesem Punkt kann es hilfreich sein, der Canvas in mehrere separate Canvas-Darstellungen aufzuteilen, um jede Initiative besser widerzuspiegeln und die Diskussion zu erleichtern.
Beim Definieren der Geschäftsergebnisse sollte sichergestellt werden, dass sie mit den Erfolgsmessungen in der KPI & Metriken-Sektion abgestimmt sind und direkt mit den in der Kontextsektion definierten Geschäftszielen verbunden sind.

Erfolgsmessung und KPIs: Erfolg definieren und nachverfolgen (z. B. Verfügbarkeit, TCO-Reduzierung)



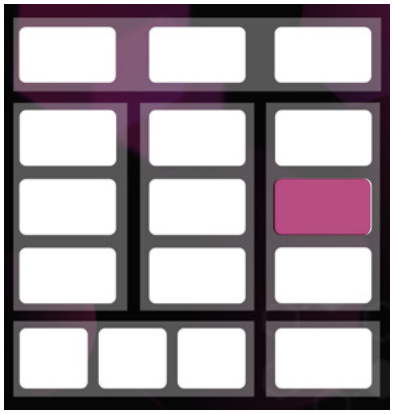
Unmittelbar nach der Diskussion der Ziele oder sogar während ihrer Definition ist es sinnvoll, diesen Abschnitt auszufüllen. Ziel ist es, klarzustellen, welche Metriken bestimmen, ob die Initiative als erfolgreich betrachtet wird oder nicht. Erfolg sollte aus einer Geschäftsperspektive betrachtet werden.

Ein häufiger Fehler besteht darin, ausschließlich an die Systemleistung zu denken. Obwohl Leistung wichtig ist, haben wir bereits erwähnt, dass die Ziele der Initiative mit den übergeordneten Zielen der Organisation verknüpft sein müssen. Daher sollten die hier ausgewählten Metriken direkt an diese Ziele gekoppelt und in messbare Geschäft-KPIs übersetzbar sein.

Beispiele: Wenn das Ziel Kostensenkung ist, stellen Sie sicher, dass diese Metrik in eingespartes Geld übersetzt wird. Wenn die Kundenzufriedenheit verbessert werden soll, messen Sie, wie sich der Support-Service direkt als Ergebnis der Modernisierung entwickelt. Wenn das Ziel die Verbesserung der Systemleistung ist, quantifizieren Sie, wie viele zusätzliche Nutzer nun bedient werden können.
Generell liegt die Berechnung des ROI einer Modernisierungsinitiative außerhalb des Rahmens dieser Anleitung, daher gehen wir nicht ins Detail.

Wichtig ist, dies vor Beginn der Initiative im Blick zu haben: Legen Sie Ausgangswerte fest und messen Sie nach den Änderungen. Andernfalls wird es sehr schwierig sein, den Erfolg aus Geschäftsperspektive zu bewerten.

Zielarchitektur oder Technologien: Den potenziellen Endzustand erkunden



Ziel ist es, eine klare Vorstellung davon zu haben, wie das System nach der Modernisierung aussehen wird. Es geht nicht darum, von Anfang an eine vollständige Architektur festzulegen.

Es ist sehr hilfreich, wenn Sie Diagramme oder andere Dokumentationen als Teil der Vorarbeit zum Canvas haben, da sie die Diskussionen beschleunigen und Kontext liefern. Eine detaillierte Systemaufgliederung, wie z. B. ein vollständiges C4-Diagramm, liegt außerhalb des Rahmens dieses Tools und wird wahrscheinlich später während der Planung und Umsetzung der Modernisierung behandelt.

Ebenso wichtig ist es, das Publikum des Workshops zu berücksichtigen. Detailgrad und Sprache sollten für alle Stakeholder zugänglich sein, damit nicht-technische Teilnehmer nicht aus der Diskussion ausgeschlossen werden.

Wie bei der Systembewertung soll auch hier genügend Information vorliegen, um die Art der Lösung zu erkennen, die erforderlich ist, um die Probleme zu beheben.

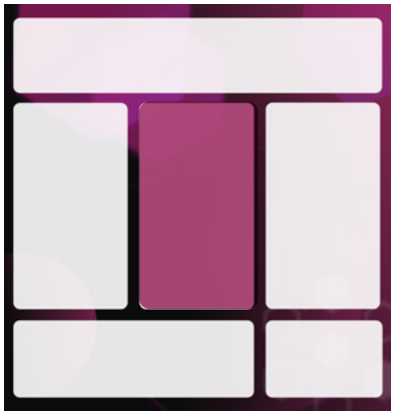
Wie detailliert sollte dieser Abschnitt sein?
Die kurze Antwort: so viel wie nötig, um Folgendes zu verstehen:

- Welche allgemeinen und spezifischen Muster eingeführt werden
- Welche Paradigmen integriert werden
- Wie die wichtigsten Engpässe konkret adressiert werden
- Wie die zentralen Schmerzpunkte konkret gelöst werden
- Welche Haupttechnologien eingesetzt werden

Sie können Elemente nach Bedarf hinzufügen oder weglassen, aber dies stellt generell eine gute Basis dar.

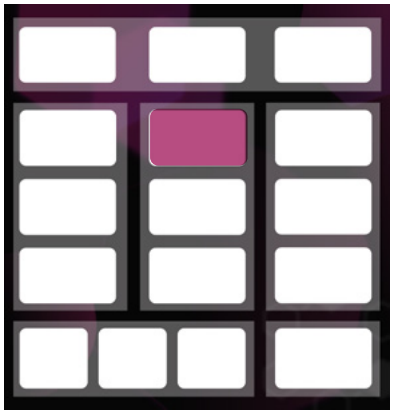
Kapitel 4:

Gestaltung des Modernisierungsansatzes



Dieser Abschnitt des Canvas befasst sich mit der Umsetzung der Modernisierungsinitiative. Es geht nicht um einen detaillierten Plan, sondern um einen Überblick über den Ansatz oder die Strategie, auf der die Initiative basiert, die wichtigsten definierten strategischen Prioritäten sowie die bereits getroffenen Kompromisse oder Entscheidungen, die den Plan beeinflussen.

Ansätze für die Modernisierungsstrategie



Dieser Teil konzentriert sich auf die verschiedenen Wege, die eingeschlagen werden können. Jede Modernisierungsinitiative ist unterschiedlich, aber es gibt Muster und Ansätze, die häufig und effektiv angewendet werden.

Gängige Modernisierungsansätze

Hier betrachten wir einige der gängigsten Ansätze zusammen mit ihren Vor- und Nachteilen. Es handelt sich nicht um eine vollständige Liste; je nach Kontext können sie kombiniert oder angepasst werden.

Rehosting: Auch bekannt als „Lift and Shift“ oder „Redeploy“. Dabei wird eine Anwendung auf eine andere Infrastruktur (physisch, virtuell oder in der Cloud) verschoben, ohne den Code oder die Funktionalität zu ändern. Dies ist eine gute Option, wenn die zugrunde liegende Technologie Probleme verursacht (z. B. veraltete Hardware oder Software), die Architektur und Funktionalität der Anwendung jedoch weiterhin gültig sind. Rehosting kann Migrationskosten und den Betriebsaufwand reduzieren.

Replatforming: Ebenfalls auf die zugrunde liegende Technologie fokussiert, beinhaltet dieser Ansatz die Übertragung der Anwendung auf eine neue Plattform mit minimalen Änderungen. Wie beim Rehosting hilft er, Probleme durch veraltete Technologien zu beheben.

Encapsulation: Besteht darin, die bestehende Anwendung mit einer Schnittstelle (API) zu umhüllen, um ihre Funktionalität als Dienste bereitzustellen. Dies ermöglicht eine Modernisierung der Anwendung und den Zugriff über moderne Schnittstellen, ohne das zugrunde liegende System zu verändern.

Refactoring: Beinhaltet die Umstrukturierung des Codes, um seine interne Qualität, Wartbarkeit und Leistung zu verbessern, ohne die externe Funktionalität zu ändern. Nützlich, wenn bestehender Code aufgrund technischer Schulden schwer wartbar oder erweiterbar ist, und kann sowohl technologische als auch architektonische Probleme lösen.

Rearchitecting: Bedeutet, die Architektur der Anwendung neu zu gestalten, um Skalierbarkeit, Flexibilität und Wartbarkeit zu verbessern. Wird eingesetzt, wenn die bestehende Architektur Probleme verursacht, wie z. B. bei monolithischen Anwendungen. Zudem ermöglicht es die Einführung moderner Architekturstile wie Microservices.

Komponentisierung oder Modularisierung: Besteht darin, eine große oder monolithische Anwendung in kleinere Komponenten oder Module zu unterteilen, um Wartung, Entwicklung und Wiederverwendbarkeit zu erleichtern.

Rebuilding/Rewriting: Beinhaltet die Neuentwicklung der Anwendung von Grund auf unter Beibehaltung des Umfangs und der Spezifikationen. Nützlich, wenn die aktuelle Anwendung die Geschäftsanforderungen nicht erfüllt und ein vollständiges Redesign erforderlich ist.

Replacing: Beinhaltet das vollständige Entfernen der bestehenden Anwendung und den Ersatz durch eine neue Lösung. Dies ist die beste Alternative, wenn die Anwendung nicht mehr den funktionalen Anforderungen entspricht. Allerdings ist es auch die teuerste und risikoreichste Option.

Service Identification: Wird verwendet, wenn auf eine serviceorientierte Architektur (SOA) migriert wird. Beinhaltet die Identifikation wiederverwendbarer Funktionen der bestehenden Anwendung, die in Dienste umgewandelt werden könnten.

Wrapping: Eine Black-Box-Technik, bei der das Legacy-System mit einer Software-Schicht umgeben wird, die seine Komplexität verbirgt und eine moderne Schnittstelle bereitstellt. Wird eingesetzt, um Inkompatibilitäten zwischen den Legacy-Schnittstellen und aktuellen Integrationspraktiken zu lösen.

Migration: Besteht darin, eine Anwendung von einer Plattform auf eine andere zu verschieben, was Datenmigration oder Sprachkonvertierung (z. B. von COBOL zu Java oder von Java zu C#) beinhalten kann.

Auswahl eines Modernisierungsansatzes

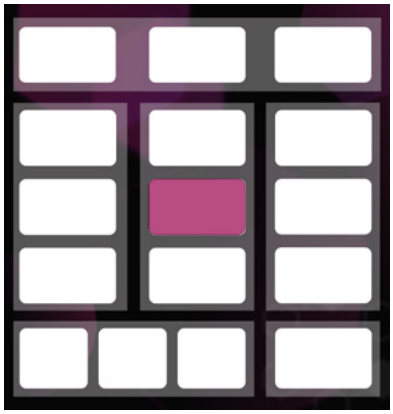
Wie bereits erwähnt, wird die endgültige Strategie höchstwahrscheinlich hybrid sein, indem mehrere Ansätze kombiniert werden, um ein komplexes System zu modernisieren. Eine Organisation kann beispielsweise entscheiden, einige Teile der Anwendung zu kapseln, während andere neu architektonisch gestaltet werden.

Die passende Kombination hängt von der Art des Problems, den Geschäftszielen, dem akzeptablen Risikoniveau sowie von Einschränkungen wie Budget oder Zeit ab. Risiko ist ein entscheidender Faktor: Rehosting ist eine kostengünstige und risikoarme Alternative, während das Ersetzen einer Anwendung mit hohen Kosten und hohem Risiko verbunden ist.

In jedem Fall wird ein „Big Bang“-Ansatz, bei dem die gesamte Anwendung auf einmal ersetzt wird, nicht empfohlen. Es ist eine Strategie mit hohem Risiko.

Unsere Empfehlung ist ein inkrementeller Ansatz, der sich darauf konzentriert, schrittweise Änderungen am Legacy-System einzuführen oder architektonische Muster wie das „Strangler Fig Pattern“ anzuwenden, die die Koexistenz beider Systeme während der Übergangsphase ermöglichen. Auf diese Weise werden die Erfolgschancen maximiert und Störungen im normalen Geschäftsbetrieb minimiert.

Priorisierung und Phasen: Planung inkrementeller Schritte für schnelle Ergebnisse und nachhaltigen Wert



In dieser Phase können verschiedene Übungen durchgeführt werden. Am häufigsten beginnt man mit einem Brainstorming und wendet anschließend eine Priorisierungsmatrix (Aufwand – Wert) an.

Unter Berücksichtigung der Ziele der Initiative ist es hilfreich, sich folgende Fragen zu stellen:

- Welches Problem ist am schwierigsten zu lösen oder wo herrscht die größte Unsicherheit?
- Wie können wir das Risiko minimieren?
- Was wird uns den größten Nutzen bringen?
- Welche Einschränkungen existieren, die die Priorisierung beeinflussen?
- Welche Ressourcen sind für diese Aufgabe eingeplant?

Jede Modernisierungsinitiative ist im Wesentlichen ein Transformationsprojekt. Die Balance zwischen kurzfristigen Ergebnissen und langfristigen Vorteilen ist entscheidend, um Engagement zu erzeugen und die Nachhaltigkeit der Initiative sicherzustellen. Wenn zu lange auf den Nutzen gewartet werden muss, fällt es schwer, das Team motiviert zu halten. Andererseits erschwert fehlende langfristige Planung, den Fokus auf die wirklich wichtigen Geschäftsziele zu bewahren.

Ressourcenstrategie

Ressourcen sind immer begrenzt – sei es Zeit, Geld oder andere Faktoren. Deshalb ist es wichtig, eine Ressourcenstrategie festzulegen, die während der Modernisierung angewendet wird. Einige gängige Ansätze sind:

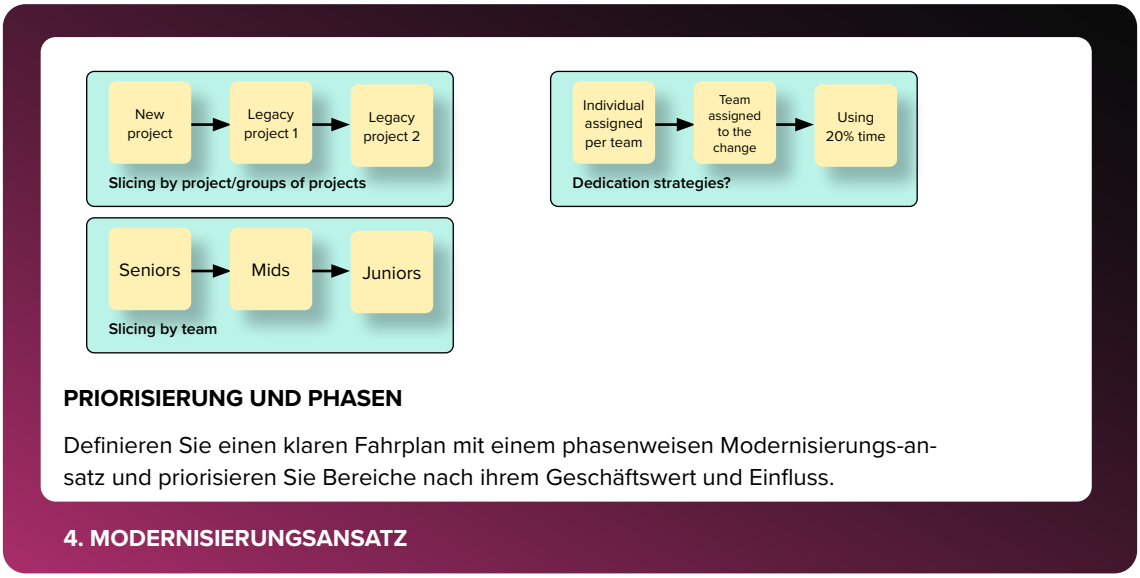
20%-Zeit: Von Google populär gemacht, erlaubt es Personen, 20 % ihrer Arbeitszeit einem persönlichen Projekt zu widmen, unter zwei Bedingungen: Die geleistete Arbeit ist geistiges Eigentum des Unternehmens und, falls erforderlich, muss erklärt werden, wie das Projekt einen direkten oder indirekten Nutzen für das Unternehmen bringt.

Timeboxed-Verbesserung: Ein spezifischer Zeitraum wird festgelegt, um eine Verbesserung umzusetzen. Nach Ablauf dieser Zeit wird die Arbeit gestoppt und zur nächsten Aufgabe übergegangen. Dieser Ansatz hilft, Risiken wie Scope Creep oder endlose Korrekturschleifen zu begrenzen.

Dedizierte Person pro Iteration: Bestimmte Personen werden während einer gesamten Iteration einem spezifischen Thema zugewiesen. Dies ermöglicht, dass die Hauptarbeit des Teams nicht durch unterbrechende Aufgaben oder Ablenkungen von den täglichen Aktivitäten beeinträchtigt wird.

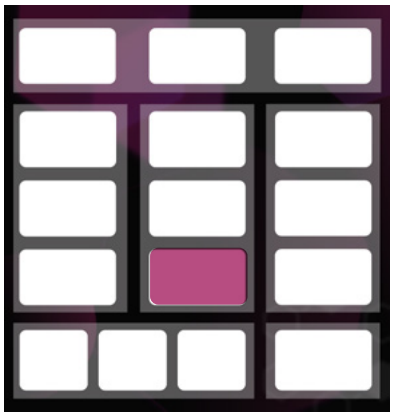
Dediziertes Team: Ein Team arbeitet ausschließlich und in Vollzeit an der Verbesserung. Einheitliches, nach Wert priorisiertes Backlog: Mehrere Teams arbeiten gleichzeitig an der regulären Arbeit und an der Initiative, wobei sie Aufgaben aus einem einheitlichen Backlog übernehmen, das nach Wert priorisiert ist.

Beispiel: Priorisierung und Phasen – Trunk Based Development



Dieses Beispiel zeigt verschiedene Möglichkeiten, die schrittweise Implementierung einer Prozessänderung zu planen – in diesem Fall die Einführung von Trunk Based Development – und einige der bewerteten Strategien hinsichtlich der Teamressourcenzuweisung.

Entscheidungen und Abwägungen: Kosten, Zeit und technologische Optionen ausbalancieren



Dieser Abschnitt konzentriert sich darauf, die wichtigsten Leitlinien für die tägliche Priorisierung zu sammeln sowie bereits getroffene Entscheidungen zu dokumentieren, die einen hohen Einfluss auf das Projekt haben werden.

Beispiele für Entscheidungen, die dokumentiert werden sollten:

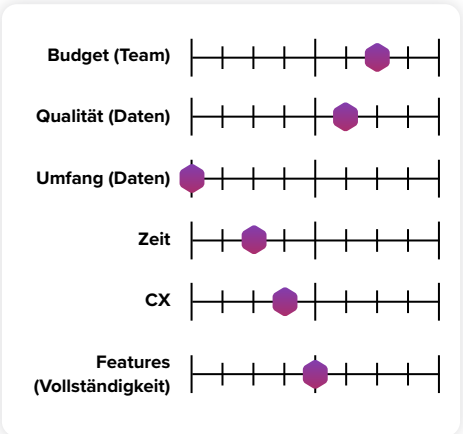
- Prioritäten zwischen den Projekteigenschaften.
- Welche Aspekte nicht betroffen sein dürfen (welche Teile des BAU keine Probleme verursachen dürfen).
- Konkrete Fristen (z. B. Gesetzgebung, die zu einem bestimmten Datum in Kraft tritt).
- Jegliche individuelle Strategie, die umgesetzt werden soll.

Die Dokumentation und Diskussion dieser Entscheidungen hilft dabei:

- Die Abstimmung und das Verständnis der Auswirkungen verschiedener Optionen für alle Stakeholder (technisch und geschäftlich) sicherzustellen.
- Die Kommunikation zu erleichtern, indem ein strukturiertes Rahmenwerk für die Diskussion von Optionen mit Kunden bereitgestellt wird.
- Die Bewertung von Kompromissen zu unterstützen und den gewählten Ansatz zu begründen.
- Herausforderungen vor der Umsetzung vorherzusehen und zu mindern.
- Sicherzustellen, dass die Entscheidungen mit den langfristigen Zielen übereinstimmen und nicht nur temporäre Lösungen betreffen.

Gemeinsam festlegen, welche Aspekte flexibel sind

In der Softwareentwicklung ist es üblich, Entscheidungen treffen zu müssen, die Kompromisse erfordern, meist in Bezug auf Attribute wie Qualität, Budget oder Zeit.



Es ist empfehlenswert, von Anfang an klarzustellen, welche Attribute flexibler sind und welche Priorität haben, damit alle wissen, wie sie ihre Entscheidungen ausrichten sollen.

Eine einfache, aber effektive Übung dafür sind die sogenannten „Tradeoff-Slider“.

Bei dieser Übung werden die verschiedenen zu priorisierenden Attribute definiert, wie Qualität, Datenumfang oder Budget. Anschließend platzieren die Teilnehmer sie auf einer Skala von 1 bis 10 (ohne

Wiederholungen). Die Positionen werden diskutiert, bis ein Konsens erreicht wird, der dann als Leitfaden für die tägliche Entscheidungsfindung oder das Risikomanagement dient.

Beispiele für Entscheidungen und Abwägungen

Rehosting vs. Refactoring vs. Rebuilding vs. Replacing

- **Entscheidung:** Sollte das System ohne Änderungen verschoben werden (rehosted), teilweise modernisiert werden (refactored), vollständig neu entwickelt werden (rebuilt) oder durch eine neue Lösung ersetzt werden?
- **Abwägungen:**
  - **Rehosting** (Lift & Shift) ist schneller und risikoärmer, adressiert aber nicht die technische Schuld.
  - **Refactoring** gleicht Aufwand und Verbesserung aus, erfordert jedoch sorgfältige Planung.
  - **Rebuilding** ermöglicht ein optimales und zukunftsfähiges System, ist aber kosten- und zeitintensiv.
  - **Replacing** beseitigt die Einschränkungen des Altsystems, kann aber Kompatibilitätsprobleme mit bestehenden Prozessen erzeugen.

Monolith vs. Microservices vs. Modularer Monolith

- **Entscheidung:** Sollte das System monolithisch bleiben, auf Microservices umgestellt oder ein modularer Monolith eingeführt werden?
- **Abwägungen:**
  - **Microservices** verbessern Skalierbarkeit und Flexibilität, erhöhen jedoch Komplexität und Betriebsaufwand.
  - **Modularer Monolith** gleicht Struktur und Wartbarkeit aus, erlaubt aber keine vollständige Unabhängigkeit der Services.
  - **Monolith** ist einfacher und leichter bereitzustellen, begrenzt jedoch Agilität und Skalierbarkeit.

Öffentliche vs. private vs. hybride Cloud

- **Entscheidung:** Sollte das System in einer öffentlichen Cloud (AWS, Azure, GCP), privaten Cloud oder hybriden Umgebung gehostet werden?
- **Abwägungen:**
  - **Öffentliche Cloud** bietet Skalierbarkeit und Kosteneffizienz, stellt jedoch Compliance-Herausforderungen dar.
  - **Private Cloud** bietet mehr Kontrolle und Sicherheit, aber höhere Betriebskosten.
  - **Hybrid Cloud** bietet Flexibilität, erfordert jedoch komplexe Integration und Governance.

Big Bang vs. inkrementelle Datenmigration

- **Entscheidung:** Sollen Daten auf einmal (Big Bang) oder schrittweise (inkrementell) migriert werden?
- **Abwägungen:**

- **Big Bang Migration** ist schneller, aber riskant, wenn Probleme auftreten.
- **Inkrementelle Migration** reduziert Risiken, dauert jedoch länger und kann Synchronisationsmechanismen erfordern.

SQL vs. NoSQL vs. Polyglot Persistence

- **Entscheidung:** Sollte die Datenbank relational (SQL) bleiben, auf NoSQL wechseln oder mehrere Technologien kombiniert werden?
- **Abwägungen:**
  - **SQL Datenbank** gewährleistet Konsistenz und strukturierte Daten, kann jedoch Skalierungsprobleme haben.
  - **NoSQL Datenbank** bewältigt unstrukturierte Daten und Skalierung gut, kann aber Konsistenz beeinträchtigen.
  - **Polyglot Persistence** bietet Flexibilität, erhöht jedoch die Komplexität im Datenmanagement.

REST vs. GraphQL vs. Event-Driven Architecture

- **Entscheidung:** Wie sollte das System Daten und Services bereitstellen?
- **Abwägungen:**
  - **REST** ist weit verbreitet, kann aber bei komplexen Abfragen ineffizient sein
  - **GraphQL** bietet mehr Flexibilität, erhöht jedoch die Implementierungskomplexität.
  - **Event-Driven Architecture** unterstützt Skalierbarkeit, erfordert jedoch robuste Event-Processing-Mechanismen.

Zero Trust vs. rollenbasierter Zugriff vs. hybrider Ansatz

- **Entscheidung:** Welches Sicherheitsmodell soll übernommen werden?
- **Abwägungen:**
  - **Zero Trust** bietet hohe Sicherheit, erfordert aber umfangreiche Änderungen im Zugriffsmanagement.
  - **Rollenbasierter Zugriff (RBAC)** ist einfacher zu verwalten, möglicherweise jedoch nicht granular genug.
  - **Hybride Ansätze** bieten Flexibilität, erfordern jedoch sorgfältige Governance.

Vollständige Automatisierung vs. schrittweise Einführung

- **Entscheidung:** Sollte die DevOps-Pipeline von Anfang an vollständig automatisiert werden oder schrittweise eingeführt werden?
- **Abwägungen:**
  - **Vollständige Automatisierung** beschleunigt die Lieferung, erfordert jedoch Investitionen in Tools und Prozesse.
  - **Schrittweise Einführung** reduziert den anfänglichen Aufwand, verzögert jedoch den vollen Nutzen von DevOps.

Vendor Lock-In vs. Open Standards

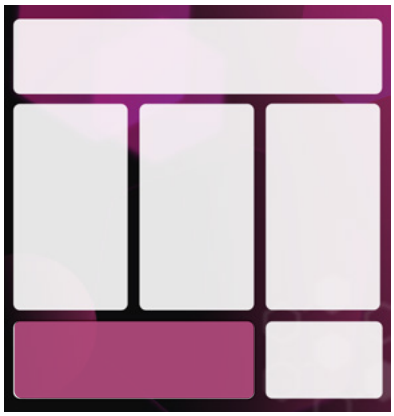
- **Entscheidung:** Sollte die Modernisierungsstrategie auf spezifische Anbieterlösungen (vendor-native, z. B. AWS Lambda oder Azure Functions) setzen oder auf Open-Source-Alternativen?
- **Abwägungen:**
  - **Anbieter-spezifische Lösungen** bieten solide Integration, erzeugen jedoch Abhängigkeiten.
  - **Offene Standards** bieten Flexibilität, erfordern aber mehr Entwicklungsaufwand.

Parallele vs. phasenweise vs. Big Bang-Implementierung

- **Entscheidung:** Wie sollte das neue System implementiert werden?
- **Abwägungen:**
  - **Big Bang** ist schnell, aber risikoreich.
  - **Phasenweise Implementierung** reduziert Risiken, erhöht jedoch die operative Komplexität.
  - **Parallele Implementierung** sichert Stabilität, erfordert jedoch doppelte Infrastruktur.

Kapitel 5:

Ressourcen und organisatorische Faktoren

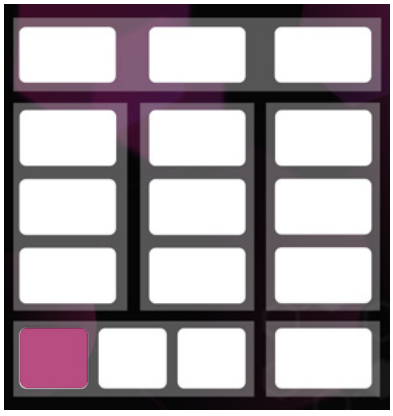


Diese Sektion konzentriert sich darauf, die Einschränkungen zu klären, innerhalb derer die Modernisierungsinitiative erfolgreich umgesetzt werden muss.

Die wichtigsten Einschränkungen, die üblicherweise behandelt werden, beziehen sich auf die Unterstützung – sei es finanziell, strukturell oder politisch.

Ohne zu sehr ins Detail jeder dieser Bereiche zu gehen, wird hier ein Überblick gegeben, und es werden weitere Werkzeuge und Konzepte aufgezeigt, die sehr nützlich sein können, um diese Gespräche zu führen und zu dokumentieren.

Stakeholder-Einbindung und Kommunikation: Die richtigen Personen von Anfang an einbinden



Das Ziel dieses Abschnitts ist es, die wichtigsten Stakeholder zu identifizieren und ihre Haltung zur Modernisierungsinitiative zu verstehen. Auf dieser Grundlage können Sie einen klaren Plan entwickeln, wie Sie sie dorthin führen, wo Sie sie benötigen.

Ein Stakeholder kann unter anderem jemand sein, der direkt von den Ergebnissen betroffen ist, ein Sponsor oder jemand, der die Initiative blockieren oder erleichtern kann.

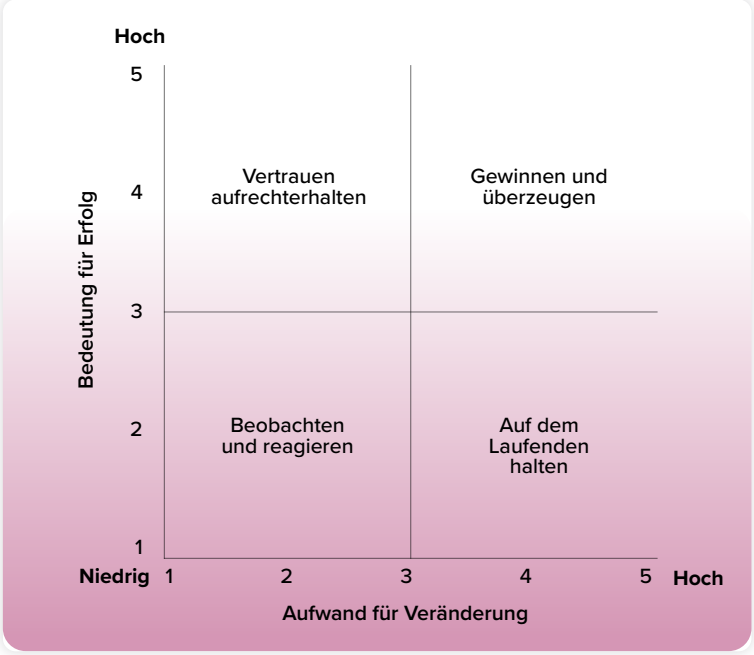
Das folgende Diagramm zeigt einen möglichen Funnel, den Sie nutzen können, um das Adoptionsniveau Ihrer verschiedenen Zielgruppen oder Stakeholder in Bezug auf die Initiative zu verstehen und zu bestimmen, wo Sie sie haben möchten.

Stakeholder-Gruppe	Unwissend	Bewusst	Verstehen	Zusammen-arbeiten	Engagieren	Befürworten
Geschäftseinheiten					●	○
Geografische Einheiten			●		○	
Funktionale Einheiten		●		○		
Mitarbeitende & Kunden		●		○		
Andere Stakeholder	●	○				

● Aktuell ○ Desired

Nicht alle Stakeholder erfordern denselben Aufwand, um überzeugt zu werden. Daher ist auch deren Priorisierung entscheidend.

Hier ein kurzes Beispiel, wie Sie vorgehen könnten:

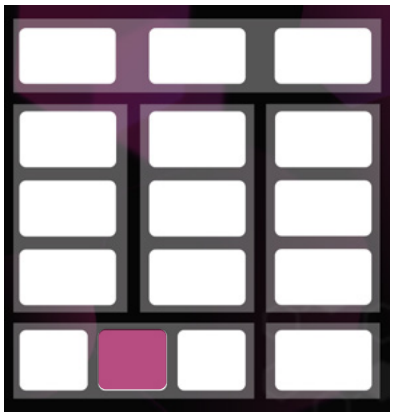


Die Idee ist, diese Priorisierung zu verwenden, um die geeignete Kommunikationsstrategie zu definieren und einen spezifischen Kommunikationsplan für jede Stakeholder-Gruppe zu erstellen.

	Bewusstsein (Awareness)	Verständnis (Understanding)	Zusammenarbeit (Collaboration)	Engagement (Commitment)	Befürwortung/ Promotion < (Advocacy)
Definition	Stakeholder kennen die Veränderung und verstehen deren Zweck und Fortschritt.	Stakeholder haben ein klares Verständnis der Vorteile und Auswirkungen der Veränderung für sie.	Stakeholder unterstützen die Veränderung, glauben an ihren Wert und würden handeln, wenn sie aufgefordert werden.	Stakeholder kommunizieren proaktiv und führen die notwendigen Maßnahmen zur Unterstützung der Veränderung aus.	Stakeholder übernehmen die Initiative, um Leistung zu verbessern und aufrechtzuerhalten.
Strategie	Informieren	Am Projekt beteiligen	Bedeutungsvolle Rollen zuweisen	Verantwortung übertragen	Eigentum an der Veränderung gewähren

Am Ende des Buches finden Sie ein Beispiel für einen Kommunikationsplan, den Sie als Referenz verwenden können.

Teamstruktur und Expertise: Multifunktionale Teams für nachhaltigen Erfolg aufbauen



In diesem Abschnitt wird das aktuelle Team analysiert: wie es strukturiert ist, welche Fähigkeiten vorhanden sind oder fehlen und welche Verantwortungsbereiche sie innerhalb der Initiative oder Teilen davon haben können.

Es ist wichtig, nicht nur das Team oder die Teams zu berücksichtigen, die direkt an der Initiative arbeiten, sondern auch diejenigen, die sie unterstützen.

Beispielsweise kann es bei einer Initiative, die die Art und Weise ändert, wie Kunden Zahlungen tätigen, hilfreich sein, einen Ansprechpartner aus dem Kundenservice-Team einzubeziehen, damit sie über die Änderungen informiert sind und in der Lage sind, auftretende Fragen zu beantworten.

Teamstruktur

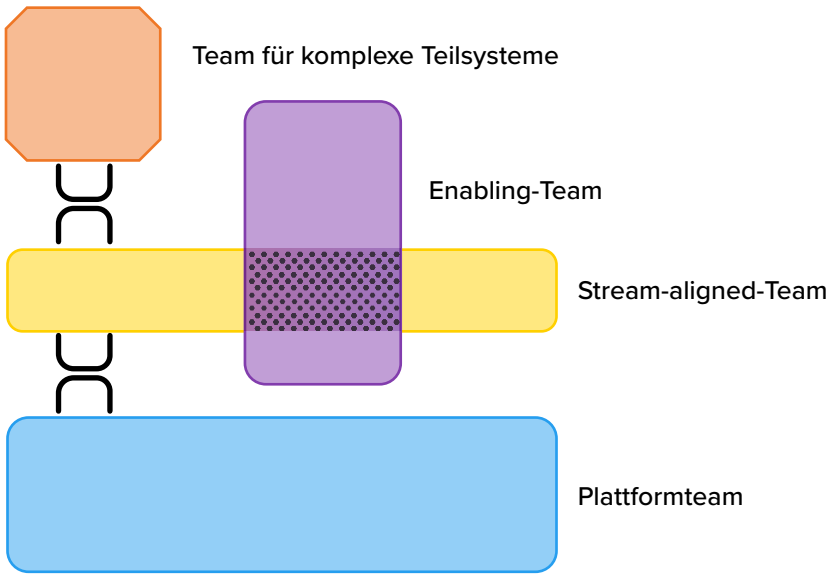
Bei der Analyse der Team-Perspektive im Rahmen des Modernisierungsplans ist das aktuelle Organisationsdesign entscheidend, um sicherzustellen, dass die Initiative nicht durch Abhängigkeiten oder fehlende Ressourcen blockiert wird. Insbesondere sollte berücksichtigt werden, ob die aktuelle Teamstruktur auf Wertschöpfung oder auf Effizienz ausgerichtet ist.

Wenn wir von Optimierung für Wertschöpfung sprechen, meinen wir, dass alles Notwendige für die Umsetzung der Initiative innerhalb des Verantwortungsbereichs des Teams oder der Teams liegt. Dadurch wird der Koordinationsaufwand und die Abhängigkeit von Dritten, die nicht direkt beteiligt sind, reduziert. Dies unterscheidet sich von der Optimierung auf Effizienz oder Auslastung, bei der verschiedene Teile des Projekts von außerhalb des verantwortlichen Teams gesteuert oder koordiniert werden.

Ein typisches Beispiel sind Organisationen, in denen Teams nach funktionalen Bereichen getrennt sind, wie UX, Frontend, Backend oder Mobile. Dies erhöht den Koordinationsaufwand zwischen den Funktionen und erschwert die Abstimmung von Prioritäten zwischen Funktionen und Initiativen. Es ist immer wichtig, innerhalb der organisatorischen Einschränkungen zu arbeiten, und gelegentlich ist es möglich, Strukturen temporär anzupassen, um den Workflow zu verbessern und Reibungen zu reduzieren. Entscheidend ist, sich der Risiken, des zusätzlichen Aufwands und des Ansatzes bewusst zu sein, der erforderlich ist, um die Initiative erfolgreich umzusetzen – abhängig davon, wie das Team organisiert ist.

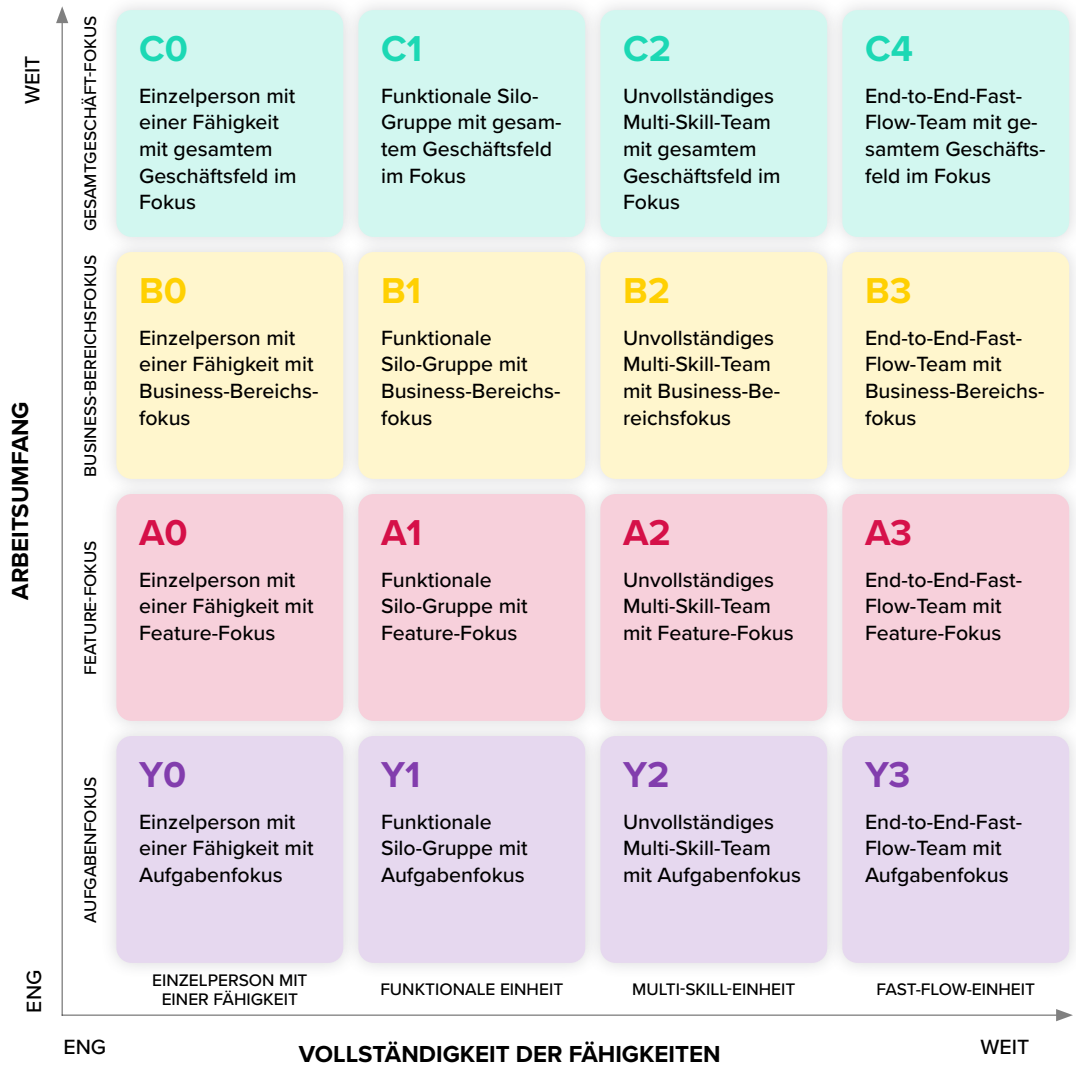
Einige gängige Muster werden im Buch *Team Topologies* beschrieben (**https://teamtopologies.com**).

Für einen strukturierten Ansatz bei der Implementierung dieser Muster können die Muster des Unfix-Modells ergänzt werden (**https://unfix.com/**)



Ebenso ist es entscheidend, die aktuelle Teamstruktur mit der im zukünftigen Plan vorgesehenen Struktur zu vergleichen. Dabei geht es nicht nur um die Anzahl der Mitglieder, sondern auch um das erforderliche Fachwissen und die Expertise, damit das Team seine Aufgaben erfüllen kann. Beispielsweise sollte, wenn die Initiative fundierte Kenntnisse in Cloud-Infrastruktur erfordert – sei es für Deployment oder Skalierbarkeit – sichergestellt werden, dass das Team über dieses Wissen verfügt oder zumindest genügend Autonomie und Kenntnisse hat, um die meisten auftretenden Probleme eigenständig zu lösen.

Neben der Überprüfung der Teamtopologie und einiger gängiger Muster kann auch die organisatorische Topologie berücksichtigt werden (https://www.orgtopologies.com) um zu verstehen, wie sich die Struktur je nach Reifegrad der Organisation entwickeln könnte.



Quelle: von Alexey Krivitsky und Roland Flemm, www.orgtopologies.com

Teamkompetenzen

Wenn wir über Teamkompetenzen sprechen, geht es vor allem darum zu verstehen, ob das Team über das notwendige Wissen und die Fähigkeiten verfügt, um das Projekt erfolgreich umzusetzen, und welche Strategien angewendet werden könnten, um dies zu erreichen.

Einige nützliche Fragen sind:

- Sind sie mit der Technologie, dem Fachbereich oder dem System vertraut?
- Verfügen sie über die erforderlichen unterstützenden Fähigkeiten, um das Ziel zu erreichen?
- Können sie innerhalb des verfügbaren Zeitrahmens geschult werden, oder sollten wir externes Fachwissen einbeziehen?
- Wenn Schulungen möglich sind, welche Optionen gibt es dafür?

In diesem Bereich sollten auch die Optionen des Enablement-Plans berücksichtigt werden, der umgesetzt werden könnte, um die wichtigsten identifizierten Wissenslücken zu schließen.

Beispiele für Strategien zum Wissensmanagement und Training, die in Betracht gezogen werden könnten:

Formale Schulungen und Zertifizierungen: Die Organisation investiert darin, Einzelpersonen in den erforderlichen Fähigkeiten oder Technologien zu schulen und zu zertifizieren.

Interne Akademien: Die Organisation etabliert ein eigenes groß angelegtes Trainingsprogramm. Communities of Practice: Die Organisation schafft einen Raum, in dem Personen über spezifische Domänen diskutieren, üben und lernen können.

Technische Coaches: Die Organisation stellt externe Fachkräfte ein oder sucht sie, um Teams beizubringen, wie bestimmte Aufgaben ausgeführt oder Praktiken angewendet werden.

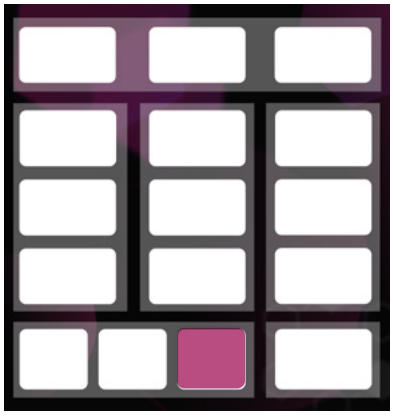
Internes Programm für technische Coaches: Die Organisation entwickelt ein eigenes internes Programm, um technische Coaches einzustellen oder auszubilden, die die Initiative unterstützen.

Externes spezialisiertes Fachwissen: Die Organisation beauftragt spezifische Berater, die Teams beitreten, denen die erforderlichen Fähigkeiten fehlen.

Externe Teams: Die Organisation bindet Dritte ein, um unabhängige Teams oder Co-Sourcing-Teams mit dem notwendigen Fachwissen für bestimmte Aufgaben zu schaffen.

Programme von Centers of Excellence: Die Organisation entwickelt ein Center of Excellence, das als zentrale Stelle für Wissensmanagement, -verteilung und -entwicklung in einem spezifischen Fachbereich innerhalb der Organisation dient.

Budget: Kostenschätzung und Sicherstellung der Finanzierung



Die Kostenschätzung für ein Modernisierungsprojekt unterscheidet sich nicht wesentlich von anderen Projekten. Die Sicherstellung der Finanzierung ist jedoch etwas komplexer, und solange kein überzeugender Business Case vorliegt, wird die Priorität des Vorhabens innerhalb der Organisation immer von anderen, stärkeren Initiativen beeinflusst.

Darüber hinaus herrscht – zumindest nach unserer Erfahrung und gemäß Untersuchungen von Gartner und anderen – die weit verbreitete Meinung, dass Modernisierungsprojekte tendenziell eine negative ROI aufweisen, ein hohes Scheiterrisiko bergen und in der Regel eine erhebliche Investition erfordern. Daher ist es entscheidend, diese Diskussion von Anfang an zu führen, um den Gesamterfolg einer Modernisierungsinitiative zu sichern.

Grundsätzlich sollte angestrebt werden, sicherzustellen, dass die für jede Phase der Modernisierung erforderliche Investition durch einen Wertzuwachs und eine positive, direkte Auswirkung auf die Organisation gerechtfertigt ist. Weitere Informationen dazu finden Sie in unserem Abschnitt über den Ansatz im Canvas.

Vielleicht fragen Sie sich: „Aber wo fangen wir an?“

Beginnen Sie mit dem, was aktuell geschieht. Die Analyse der Kosten, des investierten Aufwands und des durch das aktuelle System gelieferten Werts liefert eine erste Referenz, um zu beurteilen, ob der Plan aus Geschäftsperspektive sinnvoll ist. Denke daran, dass bei der Nutzung unterschiedlicher Ansätze jeder Ansatz eigene Kosten- und Budgetanforderungen hat und separat analysiert werden sollte, um die jeweiligen Trade-offs zu verstehen.

Wichtige Konzepte und Überlegungen bei der Kostenschätzung der Initiative:

- **Aufwand:** Ein großer Teil der Kosten in Softwareprojekten hängt vom Aufwand ab, der in Euro oder Dollar umgerechnet werden kann (z. B. Zeitaufwand für Fehlerbehebung, Kundenservice usw.).
- **Technologiekosten:** Kosten für neue Hardware, Software, Lizenzen und Tools (z. B. Mainframe-Lizenzen, AWS-Kosten, MSSQL Server-Lizenzen usw.).
- **Schulung und Beratung:** Kosten für die Schulung von Mitarbeiter:innen in neuen Technologien und die Beauftragung externer Berater (z. B. Schulungszeit, Zertifizierungskosten usw.).
- **Migrationskosten:** Kosten für die Überführung der Anwendung, Datenbank und Infrastruktur in die neue Umgebung (z. B. erforderliche Spezialtools, Ausfallzeiten).
- **Betriebskosten:** Kosten für den Betrieb des neuen Systems, einschließlich Wartung (z. B. Wartung, AWS-Kosten usw.).
- **Stilllegungskosten:** Kosten im Zusammenhang mit der Abschaltung des alten Systems (z. B. spezifische Entsorgungsprozesse oder Zertifizierungen; potenzieller Einfluss auf die Nutzerbasis).
- **Kosten für parallelen Betrieb:** Kosten für den gleichzeitigen Betrieb des alten und neuen Systems (z. B. Aufrechterhaltung von On-Premise- und Cloud-Infrastruktur, Datenreplikation usw.).
- **Produktivitätsverlust:** Kosten im Zusammenhang mit der Lernkurve, Integration neuer Technologien und Zeitaufwand für Re-Architektur oder Systemersatz (z. B. Umschulung von Anwendern).
- **Nacharbeit:** Kosten für die Korrektur von Fehlern und Problemen (z. B. je nach Ansatz kann es notwendig sein, bestehende Lösungen zu iterieren, was insbesondere in Übergangsphasen zu Nacharbeit führt, wenn Zwischenlösungen vorübergehend ausreichen).

Weitere Faktoren, die die Kosten beeinflussen:

- **Modernisierungsansatz:** Unterschiedliche Ansätze (Rehosting, Replatforming, Refactoring, Rebuilding, Replacing) haben unterschiedliche Kostenprofile. Rehosting ist in der Regel kostengünstiger und risikoärmer, während Replacing höhere Investitionen und Risiken bedeutet.

- **Systemkomplexität:** Sehr komplexe Systeme sind teurer zu modernisieren.
- **Kenntnisstand:** Eine Modernisierung, die tiefes internes Systemwissen erfordert (White-Box), ist aufwändiger als eine Black-Box-Modernisierung, die nur Kenntnisse der externen Schnittstellen benötigt.
- **Projektumfang:** Die Breite und Tiefe der Modernisierungsinitiative wirkt sich erheblich auf die Kosten aus, da sie die Komplexität der Budgetzuweisung und -genehmigung erhöht.
- **Risiko:** Modernisierungsansätze mit höherem Risiko verursachen auch höhere Kosten. Die Wahrnehmung dieses Risikos durch Stakeholder ist entscheidend; ein unterschätztes Risiko kann erhebliche negative Auswirkungen haben. Eine klare Kommunikation der Risiken ist einer der wichtigsten Faktoren, um den Erfolg der Initiative sicherzustellen.

Allgemeiner Ansatz zur Kostenschätzung

Auch wenn es keine einfache Formel gibt, umfasst ein allgemeiner Ansatz zur Schätzung der Modernisierungskosten die folgenden Schritte:

1. **Festlegung einer Ausgangsbasis:** Bestimmen Sie die Kosten der aktuellen Umgebung, einschließlich Hardware, Software, Wartung und Integrationskosten. Dies dient als Ausgangspunkt für den Vergleich.
2. **Schätzung der Kosten der neuen Umgebung:** Berechnen Sie die Kosten der neuen Umgebung, einschließlich Hardware, Software, Migration, Wartung und neuer Technologien.
3. **Einmalige Kosten erfassen:** Bestimmen Sie die Kosten, die nur einmal während des Projekts anfallen, zum Beispiel Migrationskosten.
4. **Wiederkehrende Kosten berücksichtigen:** Beziehen Sie die laufenden Kosten des neuen Systems ein, wie Wartungsaufwand.
5. **„Weiche Kosten“ berücksichtigen:** Berücksichtigen Sie Kosten für Schulung, Produktivitätsverlust und potenzielle Nacharbeit.
6. **Risiko einbeziehen:** Berücksichtigen Sie die Möglichkeit unerwarteter Probleme und Verzögerungen.
7. **Kosten vergleichen:** Vergleichen Sie die Gesamtkosten des aktuellen Systems mit den geschätzten Kosten des neuen Systems und berechnen Sie die Differenzen.

Budget und ROI

Die Herausforderung bei der Rechtfertigung einer Modernisierung besteht darin, darzustellen, wie viel eingespart oder wie viel eingenommen wird.

In der Regel interessieren sich Organisationen für vier Arten von Wert:

- Umsatzsteigerung durch Verbesserung bestehender Operationen und Geschäftsprozesse (z. B. höhere Produktivität, schnellere Time-to-Market usw.)
- Umsatzsteigerung durch ungenutztes Potenzial (z. B. Eröffnung neuer Einnahmequellen)
- Kostensenkung durch Optimierung (z. B. geringere Wartungs- oder Infrastrukturkosten)
- Reduzierung von Risiken mit hohem Einfluss (z. B. Minderung, Reduktion oder Eliminierung von Risiken, die kritische Geschäftsbereiche betreffen)

Während Ausgaben normalerweise kontrolliert und gut dokumentiert sind, ist die Zuordnung von Einnahmen – insbesondere wenn die Arbeit nicht direkt mit Vertrieb verbunden ist – deutlich schwieriger zu messen.

Beispielsweise ist die Frage
„Wie viel haben wir gespart, indem wir diese Änderung vorgenommen haben, die die AWS-Ausgaben reduziert hat?“ relativ leicht zu beantworten.

Komplexer wird es bei:
„Wie viel Geld hat diese neue Funktionalität generiert, die nicht direkt mit Verkäufen verknüpft ist?“

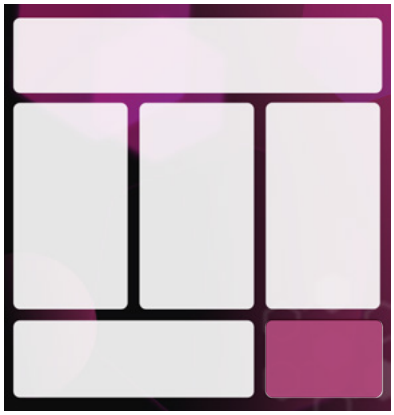
Und noch schwieriger wird es bei:
„Wie viel Umsatz wird dieses Refactoring generieren?“

In jedem Fall wird empfohlen, die Auswirkungen anhand der zuvor definierten globalen KPIs oder anderer geschäftsrelevanter KPIs auszudrücken.

Die folgende Tabelle zeigt illustrative Beispiele:

Geschäftsziel	Modernisierungsziel	KPI
Kosten senken (operative Effizienz verbessern)	Infrastrukturkosten in AWS reduzieren	% monatliche AWS-Kostenreduzierung im Vergleich zum gleichen Monat des Vorjahres oder einem ähnlichen Zeitraum
Kosten senken (operative Effizienz verbessern)	Manuelle Prozesse reduzieren/ eliminieren	Mean Time to Restore (Service-Wiederherstellungszeit) * Opportunitätskosten eines Ausfalls Änderung im Throughput (gelieferte Arbeit in einem Zeitraum) * generierter Umsatz pro Arbeitseinheit
Umsatz steigern durch Erschließung von 5 neuen Märkten	Go-to-Market-Operationen optimieren	Anzahl der geöffneten Märkte im ersten Quartal % Marktanteil im Verhältnis zum Gesamtmarkt
Umsatz steigern durch neue Kundensegmente	Skalierbarkeit der Plattform erhöhen, um die Nachfrage zu bedienen	Kosten pro Anfrage (Infrastrukturkosten im Zeitraum / bearbeitete Anfragen) % Steigerung der bearbeiteten Anfragen im Zeitraum % Steigerung der Transaktionen
KI nutzen, um Personalisierung zu verbessern	KI-Experimente im Produkt ermöglichen	Anzahl der durchgeführten KI-Experimente im Zeitraum (Quartal) Verbesserung der Conversion-Raten oder anderer KPIs, die mit dem Experiment verknüpft sind

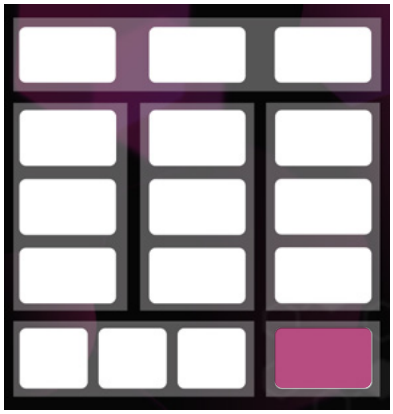
Kapitel 6:
Risikomanagement



Dieser Bereich des Canvas ist darauf ausgelegt, als Unterstützung für alle anderen Bereiche zu dienen. Hier sollen die verschiedenen Risiken erfasst werden, die im Laufe des Canvas identifiziert wurden.

Die Idee ist, diese anschließend zu überprüfen, um sie angemessen zu klassifizieren und sich auf sie vorzubereiten. Mindestens ermöglicht es, ein allgemeines Verständnis dafür zu schaffen, welche Risiken das Vorhaben am stärksten behindern könnten. Im Idealfall wird ein Maßnahmenplan zur Risikominderung für die wichtigsten Risiken entwickelt, basierend auf einer geeigneten Priorisierung.

Potenzielle Risiken und Minderungsstrategien: Identifizierung und Umgang mit häufigen Risiken



In diesem Abschnitt sammeln wir die wichtigsten Risiken, die während des Workshops besprochen wurden.

Wenn Sie an diesem Punkt angekommen sind, haben Sie wahrscheinlich bereits viele der in anderen Gesprächen identifizierten Risiken festgehalten. Falls nicht, ist es wichtig, sich die Zeit zu nehmen und die Diskussionen schriftlich festzuhalten.

Der Fokus dieses Bereichs liegt eher darauf, die verschiedenen Minderungsstrategien zu analysieren, die umgesetzt werden, um die Erfolgschancen zu erhöhen.

Beachten Sie, dass je nach Kontext einige der hier genannten Risiken möglicherweise nicht auf Ihre Organisation zutreffen.

Ein Beispiel: Vendor Lock-in ist möglicherweise kein Problem, wenn die Organisation bereits eine langfristige Bindung an einen bestimmten Cloud-Anbieter hat oder wenn Infrastructure as Code so implementiert wird, dass ein Wechsel zu einer anderen Cloud mit ähnlicher Virtualisierungstechnologie problemlos möglich ist.

Die folgenden Beispiele dienen als illustrative Orientierung für einige Risiken und ihre Minderungsstrategien:

Mangel an Fähigkeiten: Ein bedeutendes Risiko, insbesondere bei älteren Technologien, ist der Mangel an qualifiziertem Personal.

- **Mitigation:** Investition in die Weiterbildung des bestehenden Personals oder Hinzuziehen externer Experten mit den erforderlichen Fähigkeiten. Berücksichtigung von Technologien, für die sich leichter Talente finden lassen, was die Modernisierung unterstützen kann. Einbindung von Spezialisten in die Teams, um Wissen an das bestehende Personal zu übertragen.

Zeit- und Budgetverzögerungen: Modernisierungsprojekte – insbesondere solche mit komplexen Systemen – können länger dauern und teurer werden als ursprünglich geplant.

- **Mitigation:** Einen inkrementellen Ansatz verfolgen, der Wert in kleinen Iterationen liefert. Dies reduziert Risiken und stellt sicher, dass die Modernisierung mit den Geschäftsanforderungen abgestimmt bleibt. Einen klar definierten Plan erstellen und den Fortschritt eng überwachen.

Datenmigrationsprobleme: Die Migration von Daten aus Legacy-Systemen auf moderne Plattformen kann komplex und fehleranfällig sein.

- **Mitigation:** Die Datenmigration von Anfang an planen, um sicherzustellen, dass sie ohne Verlust oder Beschädigung erfolgt. Den Migrationsprozess umfassend testen.

Organisatorischer Widerstand: Mitarbeitende können sich gegen neue Technologien sträuben, aus Sorge, dass ihr Wissen und ihre Erfahrung mit Legacy-Systemen an Bedeutung verlieren.

- **Mitigation:** Die Gründe für die Modernisierung klar kommunizieren und alle Stakeholder von Beginn an einbeziehen. Die Vorteile hervorheben, die die Modernisierung der Organisation bringt.

Sicherheit und Compliance: Die Modernisierung kann Sicherheitslücken einführen und sich auf die Einhaltung von Vorschriften auswirken.

- **Mitigation:** Sicherheit und Compliance während des gesamten Modernisierungsprozesses berücksichtigen, nicht erst im Nachgang. Technologien wählen, die einen klaren Upgrade-Pfad und guten Support bieten, um Risiken im Zusammenhang mit End-of-Life oder begrenztem Support zu mindern.

Funktionsverlust: Beim Wechsel auf eine neue Plattform besteht das Risiko, wichtige Funktionen zu verlieren.

- **Mitigation:** Eine vollständige „As-is“-Analyse des Systems durchführen und alle aktuellen Funktionen dokumentieren, damit nichts verloren geht. Sicherstellen, dass während eines phasenweisen Ansatzes ausreichend Geschäftsabdeckung vorhanden ist.

Unklarer Umfang: Wenn Ziele und Umfang der Modernisierung nicht klar definiert sind, können Verzögerungen und Mehrkosten auftreten.

- **Mitigation:** Zu Beginn klare und messbare Ziele festlegen, die die Modernisierung leiten. Sicherstellen, dass alle Stakeholder den Zielen und dem Projektumfang zustimmen.

Technische Schulden: Wenn die strukturelle Qualität während der Modernisierung nicht gut verwaltet wird, kann die resultierende Anwendung mehr Probleme aufweisen als das ursprüngliche System.

- **Mitigation:** Technische Schulden während des Modernisierungsprojekts schrittweise abbauen. Sicherstellen, dass der Code wartbar und erweiterbar ist und das System gründlich getestet wird.

Vendor Lock-in: Legacy-Systeme basieren oft auf proprietären Technologien, die die Optionen einschränken. Modernisierungsprojekte sollten eine neue Form von Vendor Lock-in vermeiden.

- **Mitigation:** Wenn möglich, offene und standardbasierte Technologien wählen. Technologien mit gutem Support und klarem Upgrade-Pfad auswählen.

Mangelnde Transparenz: In großen IT-Organisationen kann es schwierig sein, einen vollständigen Überblick über alle Systeme zu erhalten, die modernisiert werden müssen.

- **Mitigation:** Eine umfassende Portfoliobewertung durchführen, um alle Anwendungen, ihren Geschäftswert, technischen Zustand und Risiken zu identifizieren.

„Analyseparalyse“: Es ist möglich, im Analyseprozess steckenzubleiben, was zu Verzögerungen führt und den Fortschritt stoppt.

- **Mitigation:** Die Analyse fokussiert und zeitlich begrenzt halten, um Stillstand zu vermeiden. Sich darauf konzentrieren, durch kleine Schritte inkrementellen Wert zu schaffen.

Schwierigkeiten bei der ROI-Prognose: Der Return on Investment von Modernisierungsprojekten kann schwer vorherzusagen sein.

- **Mitigation:** Business Cases entwickeln, die ein klares Verständnis der aktuellen Kosten, der Modernisierungskosten und des erwarteten Mehrwerts zeigen. „Immaterielle“ Vorteile wie verbesserte Wartbarkeit oder Leistung einbeziehen.

Kapitel 7:

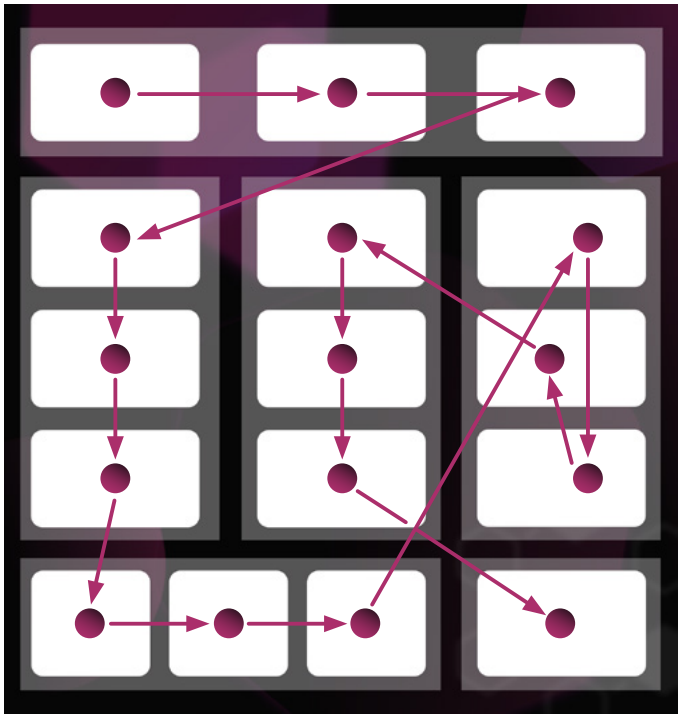
Workshop mit dem Canvas leiten

Schritt-für-Schritt-Anleitung zur Durchführung eines Strategie-Workshops für Modernisierung

Hier stellen wir eine Reihe spezifischer Reihenfolgen vor, in denen wir gerne jede Sektion des Canvas durchgehen – und erklären, warum.

Die Inhalte sollen inspirieren; wie bereits erwähnt, kann das Gespräch viele verschiedene Wege einschlagen, abhängig vom strategischen Niveau der Teilnehmenden sowie von den ihnen zur Verfügung stehenden Informationen.

Ablauf mit strikten Rahmenbedingungen

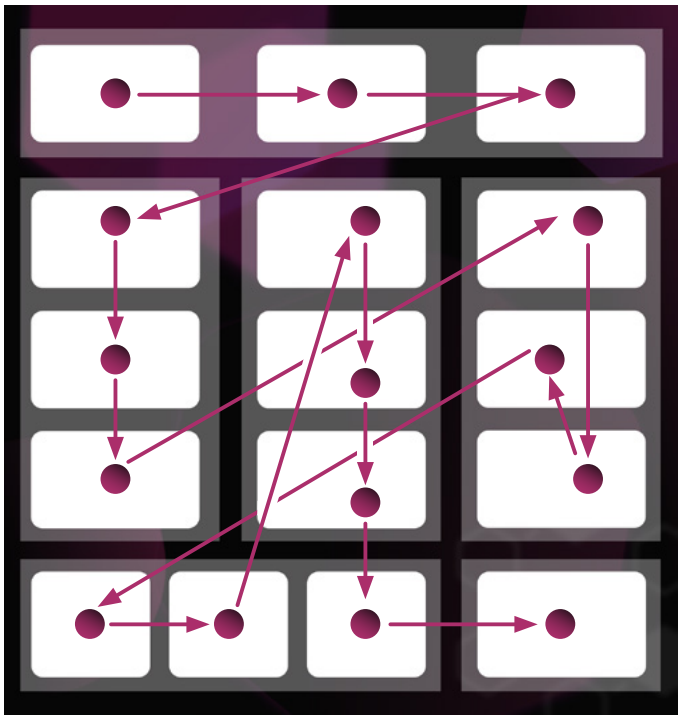


Die Idee hierbei ist, das Canvas durchzugehen, um zu verstehen, welche strikten Rahmenbedingungen bestehen, und anschließend innerhalb dieses vorgegebenen Kontexts Ideen zu entwickeln.

Die Idee hinter diesem Ablauf ist:

1. Geschäftskontext, alle Abschnitte der Reihe nach: Sich über die Geschäftsziele und -prioritäten abstimmen.
2. Aktueller Systemzustand: Ein gemeinsames Verständnis über die Ziele schaffen.
3. Überlegungen und Ressourcen: Die strikten Rahmenbedingungen identifizieren, die wir haben.
4. Zielzustand des Systems: Definieren, was wir erreichen wollen.
5. Modernisierungsansatz: Optionen entwickeln, wie wir dies erreichen können.
6. Risikomanagement: Anschließend die Risiken und die entsprechenden Minderungsstrategien analysieren.

„Shopping“-Pfad:
Die Idee hinter dieser Route besteht darin, verschiedene Optionen zu bewerten und anschließend Budgets und potenzielle Einschränkungen für jede davon zu analysieren.



Die Idee hinter diesem Ablauf ist:

1. Geschäftskontext, alle Abschnitte der Reihe nach: Sich über die Geschäftsziele und -prioritäten abstimmen.
2. Aktueller Systemzustand: Ein gemeinsames Verständnis über die Ziele schaffen.
3. Zielzustand des Systems: Definieren, was wir erreichen wollen.
4. Überlegungen und Ressourcen: Strikte Einschränkungen identifizieren, die nicht mit dem Budget zusammenhängen.
5. Modernisierungsansatz: Optionen entwickeln, wie wir dies erreichen können, typischerweise durch Aufteilung des Canvas für verschiedene Optionen.
6. Überlegungen und Ressourcen: Sich spezifisch auf das Budget konzentrieren.
7. Risikomanagement: Anschließend die Risiken und die entsprechenden Minderungsstrategien analysieren.

Tipps für die Zusammenarbeit und Entscheidungsfindung während des Workshops

1. Wenn Sie die Sitzungen mit verschiedenen Stakeholdern oder Teilnehmenden leiten – je nach den zu behandelnden Bereichen –, ist es immer hilfreich, im Voraus Kontext zu vermitteln.
2. Bevor Sie die Sitzung durchführen, prüfen Sie, wie viel des Canvas Sie bereits im Voraus ausfüllen können – basierend auf den Informationen, die Ihnen schon vorliegen. Das spart Zeit, falls Sie schneller vorankommen müssen, weil einzelne Teilnehmer blockiert sind. Selbst eine Diskussion über Ihre vorausgefüllten Einträge – auch wenn sie nicht vollständig korrekt sind – bietet einen guten Ausgangspunkt.

   Es spart außerdem Zeit, wenn bereits vorab Informationen bereitgestellt wurden. Sie möchten schließlich niemandem unnötig Zeit nehmen. Achten Sie nur darauf, dass alle verstehen, was Sie eingetragen haben.

Allgemeine Hinweise

Auch wenn es in einer Modernisierungsinitiative viele mögliche Maßnahmen gibt, sollten Sie diese zentralen Aspekte auf keinen Fall übersehen.

Fokussieren Sie sich auf die Business-Agilität, nicht nur auf Kostensenkung:
Auch wenn Kostensenkung ein häufiger Treiber ist, sollte Modernisierung vor allem als Mittel zur Erhöhung der geschäftlichen Agilität betrachtet werden. Dazu gehört, schnell auf Marktveränderungen reagieren zu können, die Time-to-Market neuer Produkte zu verkürzen und mehr Flexibilität zu schaffen, um neue Geschäftschancen zu nutzen.

Quantifizieren Sie, wie das aktuelle System spezifische Geschäftsinitiativen behindert und wie die Modernisierung diese Hindernisse beseitigen wird.

Richten Sie die Modernisierung an den Geschäftsfähigkeiten aus:
Modernisierung sollte durch die Anforderungen des Geschäfts geleitet werden, nicht nur durch technische Anliegen. Konzentrieren Sie sich darauf, die Softwarearchitektur an die Geschäftsfähigkeiten anzupassen – das kann bedeuten, ein monolithisches System in Microservices oder andere modulare Komponenten zu zerlegen.

Dieser Ansatz verbessert die Wartbarkeit und erleichtert es, das System an veränderte Geschäftsanforderungen anzupassen.

Nutzen Sie Value Stream Mapping zur Identifikation von Engpässen:
Betrachten Sie Technologie nicht isoliert. Mappen Sie stattdessen den gesamten Wertstrom – vom Erkennen eines Geschäftsbedarfs bis zur Lieferung eines funktionierenden Systems.

Dies hilft, Ineffizienzen jenseits des Codes zu identifizieren und den geschäftlichen Einfluss dieser Engpässe zu verstehen.

Kommunizieren Sie das „Warum“ an alle Stakeholder:
Es ist entscheidend, die Gründe für die Modernisierung und die Folgen des Nicht-Handelns an alle Stakeholder zu kommunizieren. Ziel ist es, zu zeigen, wie sie zu den Geschäftszielen beiträgt, statt sie als rein technische Übung darzustellen.

Nutzen Sie Business Cases, um aktuelle Kosten, Modernisierungskosten und den erwarteten ROI hervorzuheben – einschließlich der „intangiblen“ Vorteile.

Technische Vision und Strategie

Denken Sie in kontinuierlicher Modernisierung: Modernisierung ist kein einmaliges Projekt, sondern ein kontinuierlicher Verbesserungsprozess. Vermeiden Sie „Big Bang“-Ansätze und konzentrieren Sie sich auf kleine, inkrementelle Schritte, die kontinuierlich Wert liefern.

Priorisieren Sie Modularität und lose Kopplung:
Ziel ist ein System, in dem einzelne Komponenten möglichst geringe Abhängigkeiten voneinander haben. Modularität erleichtert Wartung und Änderungen – sei es durch Microservices oder durch Zerlegung monolithischer Anwendungen in klar definierte Module.

Ziehen Sie einen hybriden Ansatz in Betracht:
Sie müssen sich nicht auf eine einzige Strategie festlegen. Ein hybrider Ansatz – einige Teile re-hosten, andere refactoren, neue Funktionalität modern bauen – kann optimal sein. Wichtig ist, pro Komponente den geschäftlich und technisch sinnvollsten Ansatz zu wählen.

Nicht für zukünftige Anforderungen überengineeren
Planen Sie für zukünftige Anforderungen, aber übertreiben Sie es nicht. Beginnen Sie mit einer klaren Vision, aber lieferen Sie in kleinen, wertorientierten Inkrementen. Das reduziert Risiken und sorgt dafür, dass die Modernisierung eng an den Geschäftsbedarf gekoppelt bleibt.

Fokussieren Sie auf Wartbarkeit, nicht nur auf neue Features: Stellen Sie sicher, dass das Ergebnis leichter zu warten ist. Berücksichtigen Sie technische Schulden und ihre langfristigen Auswirkungen auf Wartbarkeit und TCO (Total Cost of Ownership).

Analyse des aktuellen Zustands

Dokumentieren Sie den „As-Is“-Zustand: Bevor mit der Modernisierung begonnen wird, sollte der aktuelle Zustand des Systems gründlich dokumentiert werden, einschließlich der Architektur, der Datenflüsse und der technischen Schulden. Dies hilft dabei, die Bereiche des Systems zu identifizieren, die die größten Probleme verursachen.

Identifizieren Sie „Hotspots“ im Code: Finden Sie die Bereiche im Code, in denen mehrere Teams häufig Änderungen vornehmen müssen und in denen Änderungen schwer umzusetzen sind. Diese sind in der Regel die größten Engpässe und die besten Bereiche, auf die man bei der Modernisierung abzielen sollte.

Analysieren Sie die Herausforderungen der Datenmigration: Die Datenmigration ist häufig ein wesentlicher Stolperstein. Verstehen Sie die Komplexität der beteiligten Daten und erstellen Sie frühzeitig einen soliden Migrationsplan.

Erfassen Sie die Perspektiven von Nutzern und Stakeholdern: Beziehen Sie Endnutzer und andere Stakeholder in die Analyse des aktuellen Zustands ein, um ihre Sichtweisen zu Schmerzpunkten, gewünschten Verbesserungen und Schlüsselfunktionen zu sammeln. Dadurch wird sichergestellt, dass die Modernisierungsbemühungen mit ihren Bedürfnissen übereinstimmen.

Bewerten Sie die Qualifikationslücke: Überlegen Sie, ob Sie die richtigen Personen haben, um die Modernisierung durchzuführen, und wo Sie externe Unterstützung oder Schulungen benötigen.

Definieren Sie klare und messbare Ziele: Legen Sie klare und messbare Ziele für den Zielzustand fest und nutzen Sie sie, um Ihre Modernisierungsaktivitäten zu steuern. Statt unklarer Ziele wie „modernisieren“ sollten spezifische Metriken definiert werden, z. B. die Verkürzung der Bereitstellungszeiten, die Steigerung der Systemleistung und die Reduzierung der Fehlerquote.

Priorisieren Sie klar definierte APIs: Beim Übergang zu einer Microservices- oder modularen Architektur sollten Sie sich darauf konzentrieren, gut definierte APIs zu erstellen, die die Funktionalität jeder Komponente kapseln. Dies erleichtert die Integration mit anderen Services und macht sie effizienter.

Berücksichtigen Sie technische und nicht-funktionale Anforderungen: Bei der Auswahl der Zieltechnologien sollten sowohl technische Fähigkeiten als auch nicht-funktionale Anforderungen wie Skalierbarkeit, Sicherheit und Compliance berücksichtigt werden.

Entwerfen Sie für Continuous Delivery: Ziel sollte eine Architektur sein, die Continuous Integration und Continuous Delivery unterstützt. Dies umfasst die Automatisierung von Build-, Test- und Deployment-Prozessen, um Änderungen schnell ausliefern zu können.

Beginnen Sie mit einem Proof of Concept: Wenn Sie einen neuen Ansatz ausprobieren, starten Sie mit einem Proof of Concept oder Experiment, um Ihren Ansatz zu validieren und Risiken zu mindern. Das hilft Ihnen, zu lernen und Ihre Pläne zu verfeinern.

Arbeiten Sie in kleinen, wertorientierten Schritten: Konzentrieren Sie sich darauf, in jeder Iteration echten Mehrwert zu liefern. Wählen Sie beispielsweise eine kleine, abgegrenzte Funktionalität aus und bringen Sie diese erst sauber zum Laufen, bevor Sie zu etwas Größerem übergehen.

Nutzen Sie automatisierte Tests und Code-Reviews: Verwenden Sie automatisierte Tests, um die Softwarequalität sicherzustellen, und Code-Reviews, um das System konsistent und verständlich zu halten.

Überwachen Sie und messen Sie den Fortschritt: Es ist entscheidend, den Fortschritt zu überwachen und die Leistung anhand der definierten Metriken zu messen. Das hilft dabei, den aktuellen Stand zu verstehen und den Ansatz bei Bedarf anzupassen.

Gehen Sie technische Schulden schrittweise an: Behandlen Sie technische Schulden im Rahmen der Modernisierungsmaßnahmen, indem Sie den Code im Laufe der Zeit refaktorieren und die Codequalität verbesseren. Ziel ist es, den Code wartbarer und skalierbar zu machen.

Seien Sie sich des organisatorischen Widerstands bewusst: Modernisierung kann Disruption verursachen, und Sie können auf Widerstand von Nutzern oder anderen Stakeholdern stoßen. Planen Sie dafür und kommunizieren Sie die Vorteile und die Bedeutung der Veränderung klar.

Erkennen Sie das Risiko der „Analyse-Paralyse“: Es ist wichtig, das System zu analysieren und sorgfältig zu planen, aber vermeiden Sie es, sich in zu vielen Details zu verlieren, da dies das gesamte Projekt verzögern kann. Halten Sie die Analyse fokussiert und zeitlich begrenzt, um Stillstand zu vermeiden.

Mildern Sie technologische Risiken: Wählen Sie Technologien, die gut unterstützt werden und einen klaren Upgrade-Pfad haben. Vermeiden Sie Abhängigkeiten von Technologien, die sich dem End-of-Life nähern oder nur begrenzten Support haben.

Stellen Sie regulatorische Compliance während des gesamten Prozesses sicher: Wenn Sie in einer regulierten Branche tätig sind, stellen Sie sicher, dass alle Modernisierungsaktivitäten den geltenden Vorschriften entsprechen.

Schaffen Sie funktionsübergreifende Teams: Modernisierung erfordert oft, Personen aus verschiedenen Disziplinen und Bereichen zusammenzubringen. Stellen Sie sicher, dass das Team über die richtigen Fähigkeiten und Erfahrungen verfügt, um erfolgreich zu sein.

Integrieren Sie Spezialisten in die Teams: Es kann hilfreich sein, Spezialisten in die Teams einzubinden, um Anleitung und Unterstützung zu bieten und Wissen an das bestehende Team weiterzugeben.

Ermöglichen Sie den Teams, Entscheidungen zu treffen: Geben Sie den Teams die Möglichkeit, die beste Vorgehensweise für die Modernisierung ihrer Systeme zu bestimmen, und fördern Sie eine Kultur der Experimentierfreude und des Lernens.

Steuern Sie den kulturellen Wandel: Beachten Sie, dass die Modernisierung von Systemen oft Änderungen in der Arbeitsweise der Menschen erfordert. Konzentrieren Sie sich auf die Schulung der Nutzer und die Förderung einer Kultur der kontinuierlichen Verbesserung.

Am Ende gibt es keine „einzigartigen“ Lösungen für die Modernisierung. Es ist wichtig, die Geschäftsbedürfnisse mit der Modernisierungsstrategie abzustimmen und den Wert sowie die Konsequenzen der Modernisierung allen Stakeholdern zu kommunizieren.

Vorlagen und Checklisten für Aktivitäten vor und nach dem Workshop

Hier ist eine einfache Liste, um sicherzustellen, dass Sie auf dem richtigen Weg sind.

Vor dem Workshop:

- ☐ Über den Zweck und die Dauer des Workshops informieren.
- ☐ Informationen über den Canvas und das Ziel des Workshops senden.
- ☐ Das benötigte Material vorbereiten; für Online-Sitzungen kann ein Miro-Board oder Ähnliches hilfreich sein, für Präsenzveranstaltungen Papier für Flipcharts, Marker usw.
- ☐ Vorhandenes relevantes Material für den Workshop anfordern, wie z. B. Architekturdiagramme auf hoher Ebene, Team- und Organisationsdesign, Metriken, strategische Ziele usw.
- ☐ Die Teilnahme der Teilnehmenden bestätigen (sicherstellen, dass die richtigen Personen dabei sind).
- ☐ Die Liste der Treiber, Ansätze usw. überprüfen und verschiedene Szenarien erstellen, die als Basis verwendet werden können.
- ☐ Prüfen, wie viel des Canvas Sie bereits mit den vorhandenen Informationen ausfüllen können. Sie sollten einen Canvas für die bereitgestellten Informationen (ohne Lösungen) und einen Canvas für jede Lösung, die diskutiert werden soll, bereithalten.

Nach dem Workshop:

- ☐ Eine Zusammenfassung der wichtigsten Erkenntnisse aus dem Workshop bereitstellen.
- ☐ Eine PDF-Version des Arbeitsbereichs in Miro oder hochaufgelöste Fotos der Boards/Flipcharts liefern.
- ☐ Die vereinbarten nächsten Schritte definieren. Dies beinhaltet normalerweise das Einholen von Informationen, die nicht verfügbar waren, sowie vorläufige Kostenschätzungen für jede Option usw.

Mögliche Fragen um eine Konversation zu beginnen

SEKTION	FRAGEN
Geschäftskontext • ZIELE UND VORHABEN • WESENTLICHE UNTERNEHMENSMOTOREN • KRITISCHE GESCHÄFTSPROZESSE	**GZIELE UND VORHABEN** • Was sind die wichtigsten Geschäftsziele, die dieses Modernisierungsprojekt unterstützen soll? • Wie verhindert das aktuelle System, dass wir diese Ziele erreichen? **WESENTLICHE UNTERNEHMENSMOTOREN** • Welche neuen Möglichkeiten könnten durch die Modernisierung des Systems eröffnet werden? • Welche Merkmale des Systems sind aus Geschäftssicht am wichtigsten? • Was sind die strategischen Gründe für eine Modernisierung? • Welche Trends, Ereignisse oder Auslöser führen dazu, dass wir heute darüber sprechen? • Was ist das gewünschte Ergebnis der Modernisierung in Bezug auf Geschäftsfähigkeiten und Leistung? **KRITISCHE GESCHÄFTSPROZESSE** • Welche kritischen Geschäftsbereiche würden positiv oder negativ betroffen sein, wenn diese Modernisierungsinitiative umgesetzt wird?
Bestandsanalyse der Alt-Systeme • SYSTEMÜBERSICHT • TECHNISCHE SCHULD UND RISIKEN • PERFORMANCE- UND EFFIZIENZENGPÄSSE	**SYSTEMÜBERSICHT** • Wie ist der aktuelle Zustand der Technologie, Architektur und Funktionalitäten des bestehenden Systems? • Welche Abhängigkeiten bestehen zwischen diesem System und anderen Systemen? • Welche Compliance-Anforderungen, Sicherheitsaspekte und sonstigen Vorschriften müssen wir einhalten? • Wie ist der aktuelle Stand unserer Tests und Automatisierung? Kann dies verbessert werden? • Wie verläuft der Deployment-Prozess? **TECHNISCHE SCHULD UND RISIKEN** • Gibt es technische Einschränkungen, über die wir informiert sein müssen? • Was sind die wichtigsten technischen Beschränkungen des bestehenden Systems? • Welche Schlüsselbereiche technischer Komplexität müssen adressiert werden? • Wie hoch ist der Stand der technischen Schuld und wie wirkt sich diese auf unsere Fähigkeit aus, geschäftlichen Mehrwert zu liefern? **PERFORMANCE- UND EFFIZIENZENGPÄSSE** • Was sind die größten Schmerzpunkte in unserem aktuellen System aus Geschäftsperspektive? • Was sind die Hauptineffizienzen im aktuellen System und welche Auswirkungen haben sie? • Wie ist unser derzeitiger Entwicklungsprozess und wo können Verbesserungen vorgenommen werden? • Wie ist der aktuelle Stand unserer Tests und kann dieser verbessert werden? • Wie verläuft der Deployment-Prozess?

Modernisierungsziele • ERWÜNSCHTE UNTERNEHMENSERGEBNISSE • ZIELARCHITEKTUR ODER TECHNOLOGIEN • ERFOLGSMESSUNG UND KPIs	**ERWÜNSCHTE UNTERNEHMENSERGEBNISSE** • Was sind die messbaren Ziele des Modernisierungsprojekts und wie erkennen wir, ob wir erfolgreich waren? **ERFOLGSMESSUNG UND KPIs** • Welche Metriken sollten wir verwenden, um den Erfolg des Projekts zu messen und wie stehen diese im Zusammenhang mit dem geschäftlichen Mehrwert? • Welche Metriken sollten wir verfolgen, um sicherzustellen, dass die Modernisierungsbemühungen Mehrwert erzeugen? **ZIELARCHITEKTUR ODER TECHNOLOGIEN** • Welche Eigenschaften sollte die neue Lösung haben? • Wie sieht die Lösung im Allgemeinen aus? • Welche allgemeinen und spezifischen Muster wurden in der neuen Lösung eingeführt? • Welche neuen Paradigmen werden eingeführt? • Wie adressiert die neue Architektur oder Lösung die wichtigsten Engpässe? (bitte spezifizieren) • Wie adressiert sie die wichtigsten Pain Points? (bitte spezifizieren) • Welche Haupttechnologien werden verwendet? • Welche technischen Einschränkungen und Risiken bestehen bei der vorgeschlagenen Lösung?
Modernisierungsansatz • ANSATZ • PRIORISIERUNG UND PHASEN • ENTSCHEIDUNGEN UND ABWÄGUNGEN	**ANSATZ** • Aus welchen Perspektiven lassen sich die Herausforderungen und Lösungsmöglichkeiten dieses Systems betrachten? • Welche zentralen Prinzipien werden den Modernisierungsprozess leiten? • Welche verschiedenen Modernisierungsansätze sollten wir in Betracht ziehen? (z. B. Rehosting, Replatforming, Refactoring, Rebuilding, Replacing) • Wie können wir sicherstellen, dass wir keine neue Form von Vendor Lock-In einführen? **PRIORISIERUNG UND PHASEN** • Welche inkrementellen Schritte können wir unternehmen, um das System zu modernisieren, anstatt einen „Big Bang“-Ansatz zu verfolgen? • Wie können wir sicherstellen, dass der gewählte Ansatz mit unserer langfristigen Technologie- und Geschäftsstrategie übereinstimmt? • Welche Wert-Meilensteine streben wir an? • Welche Art von Feedback-Zyklen werden wir implementieren? **ENTSCHEIDUNGEN UND ABWÄGUNGEN** • Was sind die Vor- und Nachteile jeder Modernisierungsoption in Bezug auf Kosten, Risiko und Geschäftswert? • Welche zentralen Verpflichtungen und nicht verhandelbaren Prinzipien ergeben sich aus dem gewählten Ansatz?

Ressourcen und organisatorische Aspekte • STAKEHOLDER-EINBINDUNG UND KOMMUNIKATION • TEAMSTRUKTUR UND EXPERTISE • BUDGET	**STAKEHOLDER-EINBINDUNG UND KOMMUNIKATION** • Wer sind die wichtigsten Stakeholder, die von dieser Modernisierung betroffen sind, und welche Bedürfnisse haben sie? (spezifische Rollen, Gruppen oder Einzelpersonen) • Wie priorisieren wir die Kommunikation für jeden Stakeholder und wie häufig? • Wie wird die Modernisierung die verschiedenen Nutzergruppen beeinflussen? • Welche Bedenken haben die Stakeholder in Bezug auf die Modernisierungsinitiative? • Wie können wir die Gründe für die Modernisierung allen Stakeholdern am besten vermitteln? • Wie können wir das Engagement aller Stakeholder sicherstellen, einschließlich Entwickler, Business-Anwender und Management? • Wie können wir gewährleisten, dass der Modernisierungsprozess transparent ist? **TEAMSTRUKTUR UND EXPERTISE** • Wie ist das aktuelle Design und die Zusammensetzung der involvierten Teams? • Welche Fähigkeiten haben wir intern und welche müssen wir extern erwerben oder hinzuziehen? • Welche Abhängigkeiten bestehen für diese Teams? Wie riskant sind sie und wie könnten sie gemindert werden? • Wie würde der allgemeine Fahrplan für die Entwicklung der benötigten Fähigkeiten aussehen? • Wie würde dies umgesetzt werden? (Ist es im Rahmen dieser Initiative enthalten?) **BUDGET** • Wie messen wir den vom Projekt gelieferten Wert? • Wie hoch sind die aktuellen Kosten für den Betrieb des Legacy-Systems, einschließlich Wartung, Support und sonstiger Kosten? • Wie hoch ist der potenzielle Return on Investment (ROI) für dieses Modernisierungsprojekt? • Welche harten und weichen Kosten sind mit den verschiedenen Modernisierungsoptionen verbunden? • Wie können wir die Vorteile der Modernisierung quantifizieren, einschließlich finanzieller und nicht-finanzieller Vorteile? • Welche möglichen Einsparungen kann die Modernisierung erzeugen? (z. B. Senkung der Wartungskosten, Effizienzsteigerung, geringere Ausfallzeiten) • Welches Budget steht für diese Initiative zur Verfügung? • Welcher Amortisationszeitraum wird angestrebt?
Risikomanagement • POTENZIELLE RISIKEN UND MITIGATIONSSTRATEGIEN	**POTENZIELLE RISIKEN UND MITIGATIONSSTRATEGIEN** • Was sind die wichtigsten technischen Risiken bei der Modernisierung des Systems? • Wie hoch ist die Risikobereitschaft innerhalb der Organisation und wie wirkt sich dies auf den Ansatz aus? • Wie erfolgt die Priorisierung der verschiedenen Risiken? • Gibt es Risiken, die wir bewusst eingehen? • Welche Strategien zur Risikominderung gibt es für jedes Risiko?

Vorlage für einen Kommunikationsplan

Aktion	Schlüsselbotschaften/Details	Zielgruppe	Lieferdatum	Kanal	In Bearbeitung	In Prüfung	Geliefert	Status

codurance

hallo@codurance.com
www.codurance.com

Codurance ist eine globale Softwareberatung, die Unternehmen dabei unterstützt, nachhaltige technische Exzellenz aufzubauen und Innovation sowie Wachstum aktiv voranzutreiben.

Wir entwickeln Software, die zuverlässig, sicher und flexibel ist, sich leicht anpassen lässt und dabei Ressourcen spart, Kosten senkt und Entwicklungsdauer verkürzt.

Mehr erfahren unter: www.codurance.com/de

Folgen Sie uns: