WHITEPAPER

7 effektive Schritte, um die Softwarequalität dauerhaft zu steigern

Wie Dein Unternehmen attraktiver für Mitarbeitende und Kunden wird und gleichzeitig zufriedenere Kunden mit wiederkehrenden Umsätzen generiert

procelo CONSULTING

Inhaltsverzeichnis

EXECUTIVE SUMMARY – WELCHE VORTEILE DU NUTZEN KÖNNTEST 3

EYE - Ein erprobtes Konzept aus der Praxis 3

ZUSAMMENFASSUNG - HELIKOPTERPERSPEKTIVE 4

FUNDAMENT 6

Fragst Du Dich auch warum Trainings oft nicht den gewünschten Effekt haben? 6

Warum EYE ist anders ist 6

Interessierst du dich auch für die Kosten, die deinem Unternehmen durch Unterbrechungen entstehen? 7

EYE IN DER PRAXIS ANWENDEN 9

Phasenübersicht – Schritt für Schritt zum Ziel 9

Die Erste Hauptphase: Analyse-, Schulungs-, und Strategiephase 9

Schritt 1: Engpass- und Potentialanalyse – Die grössten Hebel finden 10

Grabe tief 11

Rechne mit typischen Reaktionen 11

Gehe umsichtig vor 12

Schritt 2: Methodenüberblick – Erfahre, was du nutzen könntest 12

Schritt 3: Methoden-Auswahl– Nur das nutzen, was nützt 14

Die Zweite Hauptphase: Iterative Implementierungsphase mit begleiteter Anleitung 15

Die Komponenten jeder Iteration – Noch tiefer eingetaucht 16

Schritt 4: Iterationsstart-Meeting – Regelmässigkeit ist Trumpf 16

Schritt 5: Abarbeitung – Das Wissen anwenden 16

Schritt 6: Review-Meeting – Dürfen wir feiern? 17

Schritt 7: Mentalitätswandel – Die Geister die ich rief … 17

Tipps für den Erfolg – Und täglich grüsst das Murmeltier 18

Und danach? – Es gibt keine Wiederholungen, es gibt Fortsetzungen 18

DAS SAGEN ANWENDER 19

SCHÖNE NEUE WELT 20

ÜBER RALF TRAPP 21

Executive Summary – Welche Vorteile du nutzen könntest

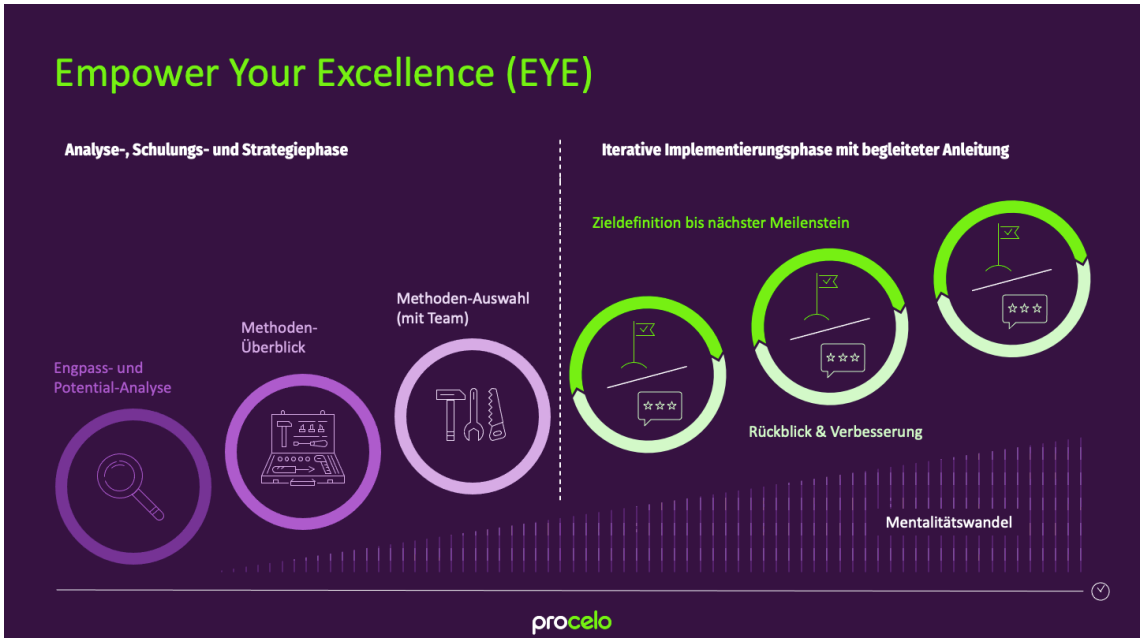
EYE - Ein erprobtes Konzept aus der Praxis

Kundenreklamationen durch Software-Qualitätsprobleme sind der **Hemmschuh für das Unternehmenswachstum** von Softwareunternehmen. Sie reduzieren die Kundenzufriedenheit und führen innerhalb des Unternehmens zu frustrierten Mitarbeitenden.

Es sei denn, es gibt **nachhaltige und angewandte Konzepte**, um die Softwarequalität dauerhaft zu verbessern.

Dieses Whitepaper zeigt mittelständischen Softwareunternehmen mit Schwerpunkt webbasierter Software, wie sie mein einzigartiges Konzept ***EMPOWER YOUR EXCELLENCE (EYE)*** zu diesem Zweck anwenden können.

Zusammenfassung - Helikopterperspektive



Starte[1] mit der **Engpass- und Potentialanalyse**, um Eure stärksten Engpässe zu identifizieren und sortiert sie der Schwere nach. Achtet bis hierher noch nicht auf mögliche Lösungswege.

Danach schliesst du den **Methodenüberblick** derjenigen Methoden an, die *mögliche* Lösungen für die Engpässe liefern. Bleibt auf hoher Flughöhe und geht nicht zu sehr ins Detail, sondern bleibt beim Überblick.

Anschliessend wählt Ihr in der **Methoden-Auswahl** eine Methode aus, die den wichtigsten Engpass löst. In dieser Methode lasst Ihr Euch im Detail schulen und greift den schwersten Engpass damit an.

In der **Iterativen Implementierungsphase mit begleiteter Anleitung** nehmt Ihr Euch kleine Häppchen der Lösung an und wiederholt in möglichst kurzem Abstand die Sequenz:

- Iterationsstart-Meeting
- Abarbeitung der im Iterationsstart -Meeting festgelegten Aufgaben
- Review-Meeting

so lange, bis der Engpass ausreichend gut gelöst ist.

[1] Fragst du Dich, warum ich Dich so unverfroren duze, obwohl wir uns noch gar nicht kennen? Den Grund dafür findest du auf meiner Website: https://www.procelo.ch/duzen/

Achtung: «Ausreichend gut» muss nicht bedeuten «perfekt». Es reicht sogar, dass der angegriffene Engpass so klein geworden ist, dass er in der Liste der Engpässe weit genug nach hinten gerutscht ist.

Wenn das passiert ist, prüft die Engpass-Liste auf Aktualität, sortiert um, falls notwendig und nehmt Euch auf dieselbe Art den nächsten Engpass vor.

Während der Arbeit, wird ein **Mentalitätswandel** bei den Beteiligten stattfinden. Nach anfänglicher Skepsis werden die Erfolge letztendlich auch die grössten Skeptiker überzeugen.

Es gibt also insgesamt 7 Schritte beziehungsweise Komponenten, deren Zusammenspiel den Erfolg bringen:

1. Engpass- und Potentialanalyse
2. Methodenüberblick
3. Methodenauswahl
4. Iterationsstart-Meeting
5. Abarbeitung
6. Review-Meeting
7. Mentalitätswandel

Fragst Du Dich auch warum Trainings oft nicht den gewünschten Effekt haben?

Die meisten Schulungs- und Weiterbildungskonzepte fokussieren sich auf punktuelle Verbesserungspotentiale. Sie schulen Methode X oder trainieren Verfahren Y. Das Zusammenspiel der Methoden, Prozeduren und Verfahren wird nicht beleuchtet. Und wenn doch, dann werden die Unternehmen nach dem Training **mit den geschulten Inhalten alleine gelassen**.

Die Menschen, die an einem Training teilnehmen wissen zwar dann viel, wenden dieses Wissen in der Praxis noch nicht an.

Dein **Investment** in das Training **versandet**.

Hast Du das auch schon erlebt?

Denn ohne *kontinuierliche* Anwendung und Erinnerung an die neuen Methoden werden die ausgebildeten Inhalte oft zugunsten des Altbekannten zurückgestellt. Das **Tagesgeschäfts gewinnt** über alle guten Vorsätze und die teuer eingekauften Inhalte entfalten ihr Potential nicht. Es ist fast so, als würden die Berater das Wissen wieder mitnehmen, wenn der Beratungsauftrag vollendet ist.

Denn aus Sicht der Mitarbeitenden gibt es oft Entscheidungsprämissen, die das ‘alte’, nun unerwünschte, Verhalten logischer und sinnvoller erscheinen lassen.

Warum EYE ist anders ist

Die durch *EYE* angeregten Veränderungen fokussieren sich auf die Verbesserung der Softwarequalität. Gleichzeitig verfolgt *EYE* jedoch einen ganzheitlichen Ansatz und betrachtet die **gesamte Wertschöpfungskette** von den Anforderungen über die Programmierung bis hin zum Kundensupport und dem Betrieb der Lösung.

Denn das schwächste Glied dieser Kette bestimmt die Widerstandskraft Deines Unternehmens gegenüber Störungen.

Wenn Du durch Einzelmassnahmen in Insellösungen investierst, gewinnst Du schon viel. Aber erst wenn alle Zahnräder reibungslos ineinandergreifen und gut zusammenwirken, erst dann **entfaltet sich die ganze Kraft von *EYE***.

Durch die fest integrierte iterative Implementierungsphase mit begleiteter Anleitung wird sichergestellt, dass einmal erarbeitete Ideen und Methoden auch **tatsächlich den Weg in die gelebte Praxis** finden.

Genau diese Phase garantiert nicht nur den Erfolg der Massnahme, sondern führt vor allem zu einem Mentalitätswandel bei den Beteiligten.

Dieser **Mentalitätswandel** ist der entscheidende **Schlüsselfaktor** für den langfristigen Erfolg. Er ist die Basis für die dauerhafte Verbesserung der Softwarequalität in Deinem Unternehmen.

Denn dadurch wandeln sich die Entscheidungsprämissen der Mitarbeitenden zugunsten des Neuen, besser funktionierenden.

Interessierst du dich auch für die Kosten, die deinem Unternehmen durch Unterbrechungen entstehen?

Chris Parmin vom Game Developer Magazine hat in einer viel zitierten Untersuchung[2] von 10'000 Programmier-Sessions herausgefunden, dass ein Programmierer **10-15 Minuten** benötigt, um wieder mit der Bearbeitung von Code zu beginnen, nachdem er die Arbeit nach einer Unterbrechung wieder aufgenommen hat. In nur in 10% der Fälle nimmt er die Arbeit in weniger als einer Minute wieder auf.

[2] https://www.gamasutra.com/view/feature/190891/programmer_interrupted.php?page=2 abgerufen am 20.2.2021

Das bedeutet: Je mehr Unterbrechungen Deine Software-Entwickler durch Bugs, Notfälle, oder Kundenanfragen erfahren, desto grösser ist **Dein Verlust**.

Gute Softwarequalität führt also gleich doppelt dazu, dass Du gewinnst: Zum einen werden Deine Entwickler weniger unterbrochen und können kontinuierlicher produktiv sein. Zum anderen steigt die **Zufriedenheit der Anwender**.

Das wiederum führt zu erheblich reduzierten Kundenreklamationen und mehr zufriedenen Stammkunden mit langjährigen Vertragsverhältnissen und regelmässigem wiederkehrenden Umsatz.

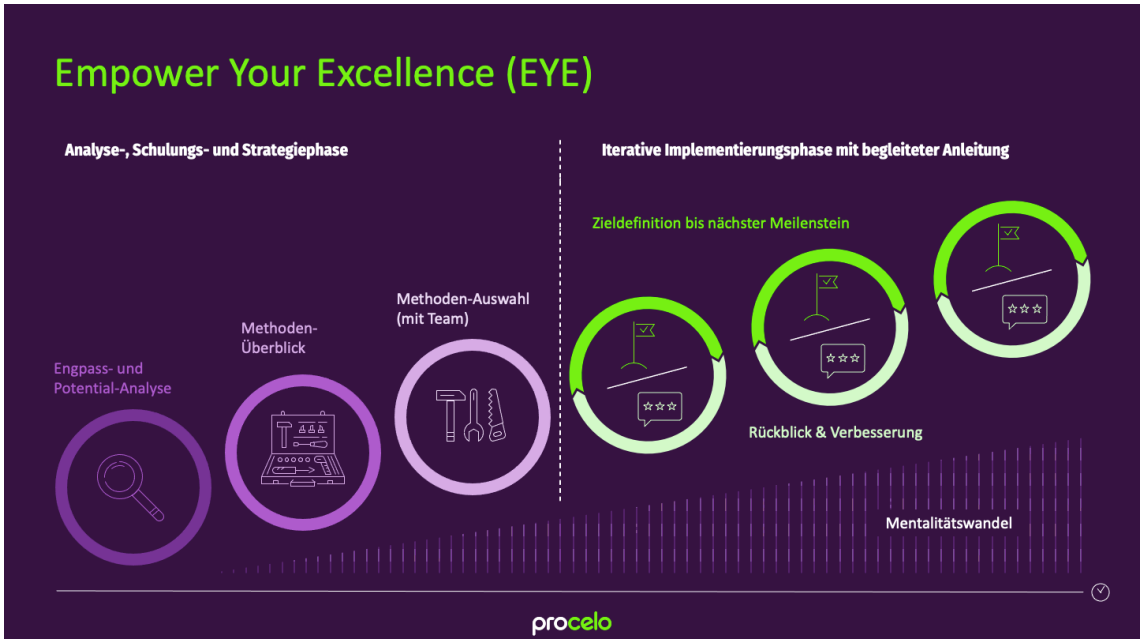
Da so der Aufwand für Bugfixing, Reklamationsbearbeitung und Support sinkt, steigt gleichzeitig der Deckungsbeitrag. Zusätzlich kannst Du mit **derselben Mannschaft mehr Kundenprojekte umsetzen**.

Grund genug, sich dieses Themas im Rahmen von EYE ebenfalls anzunehmen.

Phasenübersicht – Schritt für Schritt zum Ziel

EYE arbeitet mit **zwei Hauptphasen**, die jeweils in weitere Unterphasen untergliedert sind:

- Analyse-, Schulungs- und Strategiephase
- Iterative Implementierungsphase mit begleiteter Anleitung



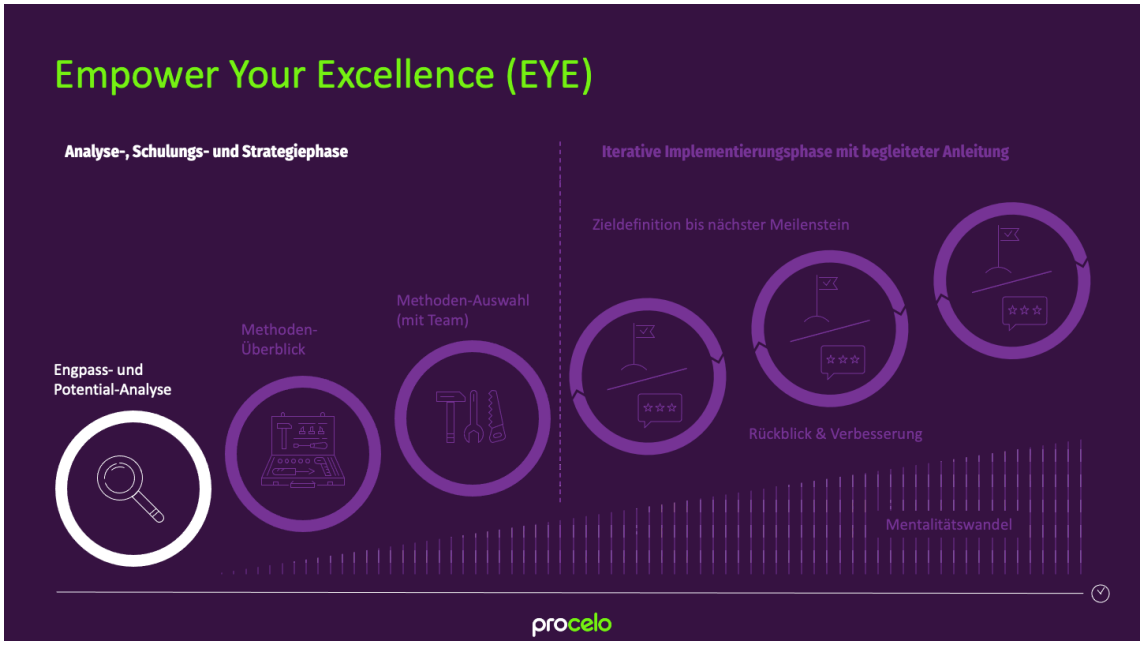
Die Erste Hauptphase: Analyse-, Schulungs-, und Strategiephase

In dieser ersten Haupt-Phase erarbeitest Du den konkreten **Bedarf** Deines Unternehmens anhand der real existierenden **Engpässe**.

Gewinne einen Überblick über die in Frage kommenden Methoden und Techniken und wähle den für Dein Unternehmen passenden Methodenmix aus.

Dazu gehst Du in 3 Schritten vor.

Schritt 1: Engpass- und Potentialanalyse – Die grössten Hebel finden



Starte klassisch mit einer **Engpass- und Potentialanalyse**, in der Du identifizierst, was genau Dein Unternehmen am Wachstum hindert. Hierbei ist es wichtig, dass die verschiedensten Teams wie z.B. Account Manager, Projektleiter, Requirement-Engineers, Programmierer und Support-Personen einen Beitrag leisten. Falls Du grosse Teams hast, kann jedes Team einen Vertreter bestimmen, der das Team repräsentiert.

Um einen guten Überblick zu bekommen, eignen sich Interviews als Vorbereitung sehr gut. Hier einige mögliche Interviewfragen:

- Was sind Deine **offiziellen Aufgaben**?
- Was siehst Du **darüber hinaus** als Deine **Aufgaben** an?
- Gibt es etwas, was Dich darin **behindert**, diese auszuüben und was wäre das?
- Was **erwartest** Du von anderen Personen im Unternehmen in Bezug auf bessere/effizientere Erledigung Deiner Aufgaben?
- An welchen **Projekten** arbeitest Du zurzeit aktiv (mit)?
- Welche Projekte/Aufgaben/Tätigkeiten hast Du aktuell in der **Pipeline**?

Die Interviewergebnisse fasst Du anschliessend zusammen, gewichtest sie und identifizierst die grössten Engpässe.

Sei nicht überrascht, wenn die Liste der gewichteten Engpässe so oder so ähnlich aussieht:

- Unspezifische Anforderungen
- Unzureichende Tests
- Unzureichende Dokumentation
- Fehlendes Know-How
- Fehlende Projekt Kick-Offs/Projektüberblick
- Fehlende Organisation
- Zu knappe Timeline-Vorgaben
- Zu viel Support
- Zu viele CRs
- Nicht gut genug kalkulier(bar)
- Zu viele/nicht optimal genutzte Kommunikationskanäle
- Zu viel auf dem Tisch
- Zu viele Absprachen mit anderen Teams
- Zu viel beim Kunden

Grabe tief

Scheue Dich nicht, tiefer zu graben, um herauszufinden, ob es nicht womöglich **strukturelle Ursachen** für Engpässe gibt (siehe Box).

Deine Ergebnisse stellst Du in der Gesamtrunde der interviewten Personen vor. Ziel dieser Vorstellung ist, dass alle Personen auf *denselben* Stand bezüglich der **real existierenden Engpässe** gebracht werden.

Beispiel: Vertriebsprovisionen

Der Vertrieb ist durch Provisionen angehalten, möglichst schnell das nächste Projekt zu verkaufen und hat deshalb kein Interesse daran die Anforderungen ausreichend einzuholen oder Fachpersonen hinzuzuziehen, um den Angebotsumfang und die Machbarkeit vorab zu prüfen.

Rechne mit typischen Reaktionen

Üblicherweise kannst Du von der Mehrzahl der Personen mit einer dieser drei Reaktionen rechnen:

1. «Das ist doch **kalter Kaffee**, das wusste ich schon. Das ist ja offensichtlich.»
2. «Davon reden wir doch schon seit Jahren, bei uns **ändert sich aber nichts**.»
3. «Aha. *Das* ist ja interessant. **Das hätte ich nicht gedacht**.»

Lasse Dich von Nummer 1 nicht aus dem Konzept bringen. Denn wenn es für *alle* Beteiligte so offensichtlich und klar wäre, wäre der Engpass schon längst beseitigt und er wäre kein Engpass mehr.

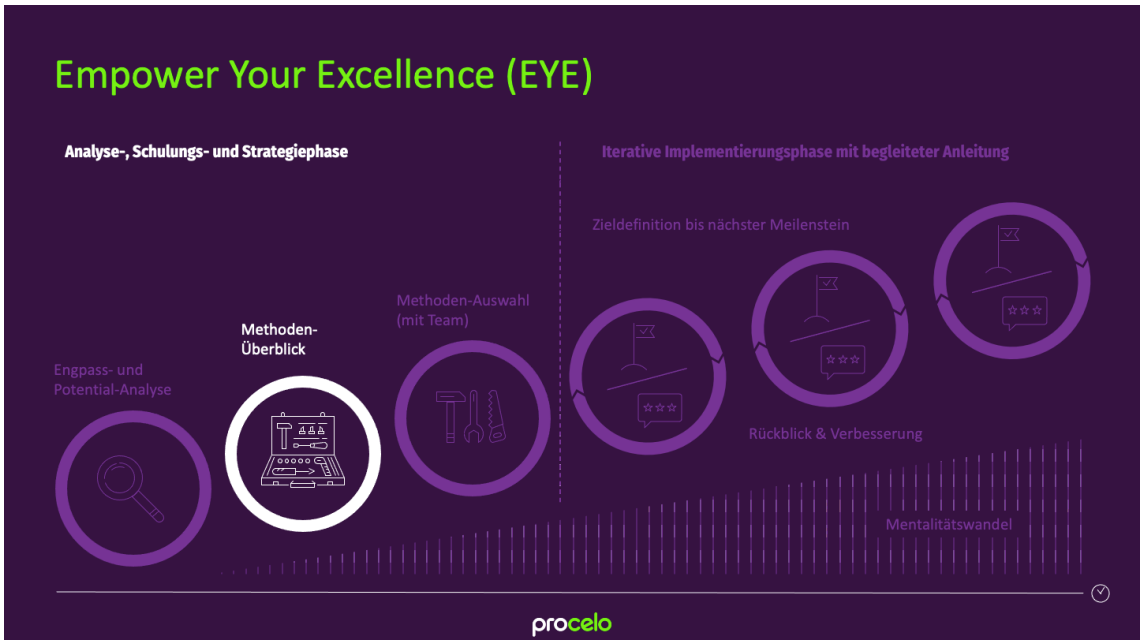
Und für Nummer 2 bietet Dir *EYE* die Lösung, deren Umsetzung die herbeigewünschten Verbesserungen zur Realität werden lassen.

Gehe umsichtig vor

Gehe umsichtig vor und **vermeide**, mögliche Verursacher für die Engpässe jetzt schon zu identifizieren und «an den Pranger» zu stellen. Hier und heute geht es zunächst um eine **Bestandsaufnahme** ohne jegliche Bewertung.

Falls möglich, lasse diesen Teil von einem neutralen Diskussionsleiter moderieren. Dazu eignen sich moderationserfahrene Mitarbeitende, die *nicht* aus dem Kreis der Interviewten Personen kommen dürfen. **Externe Moderierende** können auch eine hervorragende Unterstützung in dieser Phase sein.

Schritt 2: Methodenüberblick – Erfahre, was du nutzen könntest



Im anschliessenden **Methodenüberblick** geht es darum, dass Dein Team auf hoher Flughöhe eine Übersicht über die am Markt existierenden Methoden wie z.B. Scrum, XP, Kanban, Test Driven Development, XP u. v. a. bekommt.

Stelle die Methoden gegenüber und fokussiere Dich dabei auf die Elemente der Methoden, die einen **Lösungsansatz** für die in der Engpass- und Potentialanalyse identifizierten **Engpässe** bieten. Achte dabei darauf, dass Du diejenigen Engpässe zuerst betrachtest, deren Lösung die grössten Auswirkungen haben.

Auch die Elemente, die allen Methoden gemeinsam sind, sind einen Blick wert, sofern sie einen Deiner Engpässe lösen (siehe Box).

Um einen möglichst nützlichen Methodenüberblick zu gewinnen, benötigst Du **Fachpersonen**, die sich in der Breite in modernen Methoden für die Softwareentwicklung auskennen. Vielleicht hast Du solche Personen sogar im Unternehmen oder du kannst externe Fachleute dafür beauftragen.

Zur Erinnerung: Suche Dir hier Trainer, die viele Methoden beherrschen aber keine Präferenz für eine bestimmte Methode haben. Es ist wichtig einen **neutralen Überblick** zu bekommen.

Beispiel: Backlog

Ein Backlog kommt in allen heutigen Methoden vor. Hier lohnt es sich einen Blick darauf zu werfen, was die grundlegenden Eigenschaften eines Backlog sind: Nämlich eine visuelle Darstellung von Elementen, die Dein Team liefern kann oder eben nicht.

Allen Methoden ebenfalls gemeinsam ist, dass in einer idealen Welt nur die Elemente des Backlogs in die Implementierung gelangen, deren Details klar genug sind, um erfolgreich erledigt werden zu können.

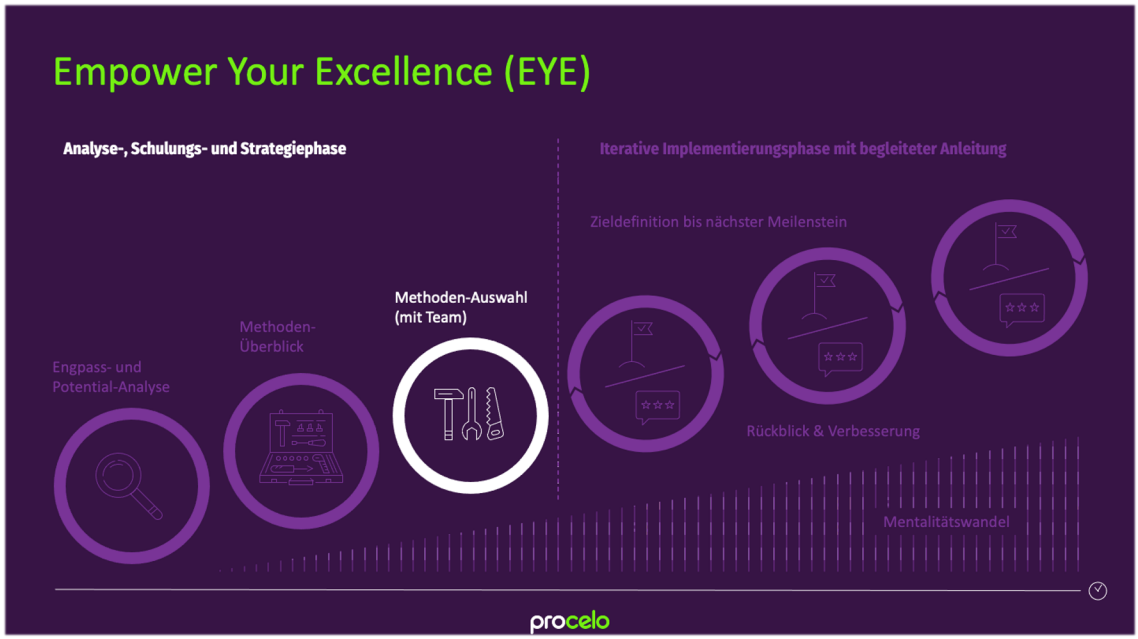
In der realen Welt ist es allerdings oft so, dass Entwickler auf Basis unklarer Anforderungen anfangen müssen zu entwickeln. Steht dann der Projektleiter oder Kunde nicht für Rückfragen bereit, entstehen Verzögerungen und Deine Entwickler können nicht weiterarbeiten.

Im schlimmsten Fall wird sogar auf Basis von Annahmen entwickelt und am Ende hörst Du wieder die so oft schon gehörte Aussage: «So hatte ich mir das aber nicht gedacht».

Es lohnt sich also in diesem Fall einen generellen Überblick über den Sinn und Zweck eines Backlog zu geben. Weiterhin lohnt es sich darauf hinzuweisen, dass nur ausreichend gut spezifizierte Backlog-Elemente in die Entwicklung gehen. Beides würde auf den Engpass «unklare Anforderungen» einzahlen.

Was hingegen *nicht* in den Methodenüberblick gehört, ist die jeweilige spezifische Handhabung des Backlog in den jeweiligen Methoden. Das Backlog wird beispielsweise in Scrum anders organisiert als in Kanban. Es ist effizienter, dass Ihr Euch erst eine Methode entscheidet, und Euch dann näher damit beschäftigt, wie die jeweilige Methode das Thema «Backlog» konkret handhabt.

Schritt 3: Methoden-Auswahl– Nur das nutzen, was nützt



Beispiel: Umsetzungsplan

Angenommen, Eure Top 3 Engpässe sind:

1. Fehlende Organisation
2. Unspezifische Anforderungen
3. Unzureichende Tests

Wobei sich «Fehlende Organisation» darin äussert, dass Eure Entwickler zu häufig wegen aktueller Dringlichkeiten unterbrochen werden.

In diesem Szenario könnte es sinnvoll sein

a) zuerst mit Kanban für die Organisation der Schnittstelle Support ⇔ Entwicklung zu beginnen
b) danach denkt Ihr über den Einsatz von Requirement-Engineers für das Schreiben der Anforderungen nach
c) und zum Schluss des Umsetzungsfahrplans setzt Ihr auf die Einführung von automatisierten Software-Tests.

Mit dem Wissen über die einzelnen Methoden gehst Du gemeinsam mit Deinen Teams in den nächsten Schritt: Die **Methodenauswahl**.

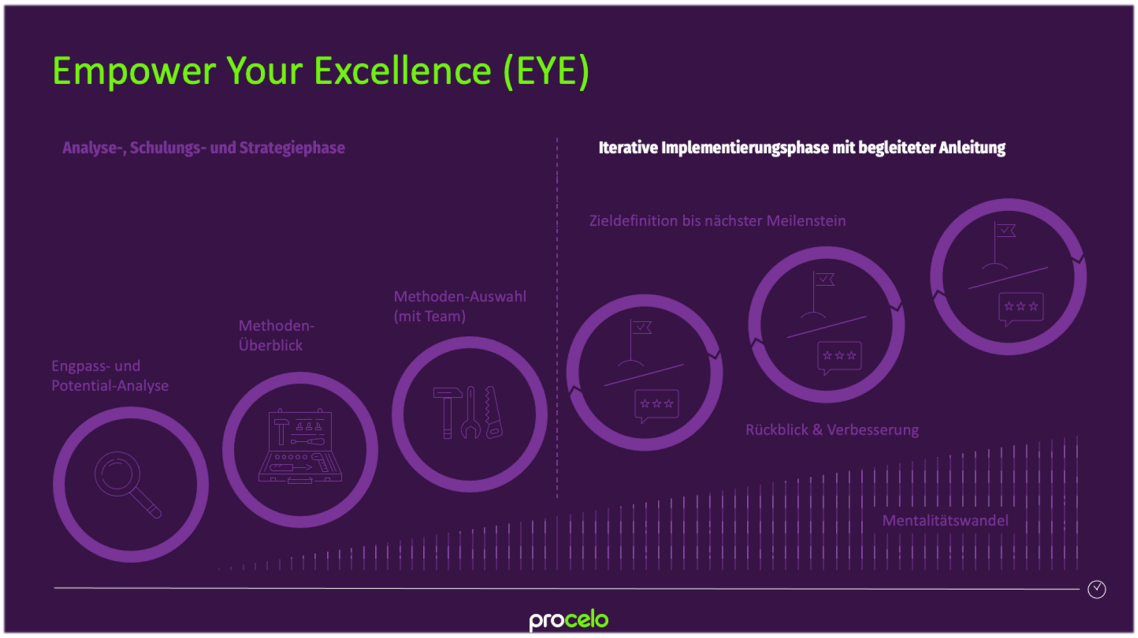
Wähle Methoden oder Methoden-Elemente, die Deine wichtigsten Engpässe zuerst lösen.

Danach entscheidet Ihr Euch für einen Umsetzungsplan (siehe Box). Dieser sollte zunächst lediglich die groben **Meilensteine** enthalten und sich strikt an den gefundenen Engpässen orientieren. Steht der Plan einmal, **beginnt** Ihr mit der **ersten** definierten Massnahme.

Für diese lasst Ihr Euch im Detail schulen. Ist diese Schulung erfolgt, setzt Euch eine Deadline, bis wann die Massnahme abgeschlossen und im praktischen Betrieb sein soll.

Für **Tipps zur konkreten** Umsetzung der Massnahme lies bitte weiter unten in der zweiten Hauptphase, der iterativen Implementierungsphase, nach.

Die Zweite Hauptphase: Iterative Implementierungsphase mit begleiteter Anleitung



Diese Phase ist **entscheidend** für den Erfolg jeder Massnahme im Umsetzungsplan. Denn die Schulung ist essentiell, aber nur die halbe Miete.

Erst kontinuierliche Anwendung unter Begleitung und mit Monitoring **transformiert** theoretisches Wissen in gelebte Handlung.

In der oben beschriebenen ersten Hauptphase wurden die **Vorbereitungen** für die Umsetzung der Veränderung geschaffen. In der nun folgenden zweiten Hauptphase werden die Veränderungen **in der Praxis umgesetzt**.

Sie besteht aus vielen, regelmässigen **Iterationen**[3] in denen die nachfolgend beschriebene Sequenz so oft durchlaufen wird, bis die beschlossene Veränderung zur Zufriedenheit der Beteiligten vollständig umgesetzt ist.

[3] https://de.wikipedia.org/wiki/Iteration, abgerufen am 25.02.2021 definiert: *„Iteration (von lateinisch iterare, wiederholen') beschreibt allgemein einen Prozess mehrfachen Wiederholens gleicher oder ähnlicher Handlungen zur Annäherung an eine Lösung oder ein bestimmtes Ziel.“.*

Wie lang die jeweiligen Iterationen jeweils sein wird, hängt von der konkreten Situation in Deinem Unternehmen ab.

Zwei bis vier Wochen haben sich als nützliche Länge herausgestellt.

Wie viele Iterationen Du brauchst, hängt von der Grösse der Massnahme ab.

Für Meetings gilt: Die **Moderation** der Meetings sollte immer von **derselben Person** durchgeführt werden. Diese Person sollte über die gesamte Zeit gleichbleiben. Natürlich gibt es Ausnahmen. Zum Beispiel bei Urlaub oder Krankheit. Die Regelmässigkeit der Iterations-Längen hat in solch einem Fall Vorrang vor gleichen Moderierenden. Demzufolge kann in solch einem Fall durchaus eine Vertretung herangezogen werden.

Die Komponenten jeder Iteration – Noch tiefer eingetaucht

Jede Iteration wiederum besteht aus drei Komponenten, den Schritten 4, 5 und 6, die in jeder Iteration in derselben Reihenfolge wiederholt werden.

Schritt 4: Iterationsstart-Meeting – Regelmässigkeit ist Trumpf

Der Start jeder Iteration ist ein Meeting der dem die in der Iteration mitwirkenden Mitarbeitenden als aktive Teilnehmer beiwohnen. Zuhörer sind erlaubt und erwünscht.

Beim Iterationsstart wird **gemeinsam entschieden**, welche Ergebnisse zum Ende der aktuellen Iteration erreicht werden sollen.

Hierbei kann es sich auch um Teilergebnisse oder um Machbarkeitsstudien handeln.

Wichtig ist:

1. dass der Umfang des **Ergebnisses** klar definiert ist,
2. und die **Verantwortlichkeit** für dieses Ergebnis klar definiert ist und
3. dass die für den Task verantwortliche Person sich dazu bekannt hat, das Ergebnis bis zum Ende der Iteration **abzuliefern** und auch **vorzustellen**.

Schritt 5: Abarbeitung – Das Wissen anwenden

Die Abarbeitung der vereinbarten Zwischenschritte geschieht in Eigenregie durch die jeweils beteiligten Mitarbeitenden. Die moderierende Person steht in dieser Zeit für Rückfragen oder als Unterstützung an der rechten und linken Seite der

Mitarbeitenden zur Verfügung. Sie greift aber **nur dann** aktiv ein, wenn bemerkt wird, dass die Dinge «aus dem Ruder zu laufen». Dann wird reagiert und entsprechend und korrigierend.

Schritt 6: Review-Meeting – Dürfen wir feiern?

Im Review wird geprüft, ob die vereinbarten Ziele erreicht wurden.

Falls ja: Es empfiehlt sich, für den Fall der Zielerreichung ein Ritual einzuführen. Es hängt von Deiner Unternehmenskultur ab, ob das ein gemeinsamer Apéro, Standing Ovations oder andere Rituale sind.

Falls nein: Stellt Euch zwei Fragen, und zwar ohne Wertung und ohne Verurteilung:

1. Woran hat es gelegen?
2. Was muss in der nächsten Iteration geändert werden, damit die Zielerreichung wahrscheinlicher wird?

Schritt 7: Mentalitätswandel – Die Geister die ich rief …

Rechne mit anfänglicher Skepsis aller Beteiligten, wenn das Immunsystem Deines Unternehmens anspringt und **Veränderungen** zunächst **abwehrt**.

Darüber hinaus gibt es bei Euch vermutlich einen gewissen Leidensdruck, der durch die die wiederkehrenden Probleme verursacht wurde. Üblicherweise führt genau dies zu einer Überlastung der Mitarbeitenden. Und daher werden sie Veränderungen, die zunächst ja **weiteren Aufwand** bedeuten, wenig aufgeschlossen sein.

Ausserdem werden sich die wenigsten Personen vorstellen können, dass die geplanten Veränderungen, so schön sie auch klingen mögen, **wirklich** zu einer **Verbesserung** der Situation beitragen werden.

Erfahrungsbericht

In einer Transformation wie der oben Beschriebenen sagte mir der grösste anfängliche Skeptiker nach ungefähr einem Jahr:

> «Ich weiss nicht, wie wir jemals Software anders entwickeln konnten als jetzt.»

Und das war erst der Anfang. Danach war es nicht mehr notwendig, weitere Veränderungsphasen von meiner Seite aus anzustossen.

Denn die Verbesserungsideen und Veränderungswünsche kamen aus dem Team.

Es wird die Aufgabe der moderierenden Person sein, die notwendige Motivation herzustellen und die Teilnehmenden zumindest davon zu überzeugen, dass es einen Versuch wert ist.

Denn offensichtlich haben die bisherigen Massnahmen wenig gebracht. Sonst wärt Ihr nicht an dem Punkt, an dem Ihr gerade seid.

Stellen sich dann die ersten Erfolge ein, wird deine Mannschaft selbst darauf drängen, weitere Verbesserungen, weitere Optimierungen und weitere Qualitätsmassnahmen **selbst einzuführen**.

Tipps für den Erfolg – Und täglich grüsst das Murmeltier

Achte darauf, dass die jeweiligen Meetings immer am **selben Wochentag** und zur **selben Zeit** stattfinden. Durch die **Regelmässigkeit** ergibt sich mit der Zeit ein fester Rhythmus und alle Beteiligten können sich so besser darauf einstellen.

Damit nicht zu viele Meetings stattfinden und die Entwickler nicht unnötig mehrfach unterbrochen werden, kannst Du ab der zweiten Iteration das Start- und Review-Meeting direkt hintereinander stattfinden lassen.

Wenn alle Iterationen komplett, die Veränderung abgeschlossen und im praktischen Einsatz sind, führe ein Review Meeting für die **gesamte Massnahme** durch. Stelle wieder dieselben Fragen:

1. Habt Ihr den ursprünglich geplanten Zeithorizont erreicht?
2. **Wenn nein**: Warum und was muss für die nächste Veränderung geändert werden, damit der nächste Zeithorizont punktgenauer erreicht wird?

Und danach? – Es gibt keine Wiederholungen, es gibt Fortsetzungen

Um beim Beispiel zu bleiben: Angenommen Kanban wäre nun erfolgreich eingeführt. Was kommt nun? Es wäre fatal, dem ursprünglichen Umsetzungsplan nun blindlings zu folgen. Denn **Prioritäten** könnten sich **verändert** haben, **Engpässe verschoben** oder weggefallen sein.

Am besten fahrt Ihr mit einer **erneuten Engpassanalyse** fort, mindestens aber mit einer Abstimmung der Beteiligten und dem Entscheid über den nächsten Schritt. Ist der ehemals definierte Umsetzungsplan noch aktuell? Ist der vormals geplante zweite Schritt immer noch der nun wichtigste? Oder hat sich ein anderer Engpass in der Zwischenzeit als noch wichtiger herauskristallisiert und vorgedrängelt?

Nach dieser Abstimmung geht Ihr mit der gemeinsam entschiedenen zweiten Veränderung in die Implementierung.

Im Beispiel von oben wäre das die Einführung von Requirements-Engineers, um den Engpass «Unspezifische Anforderungen» anzugreifen.

Das sagen Anwender

Steigerungsraten von bis zu 30% bereits in den ersten 3 Monaten

«Durch die gewonnene Klarheit nach dem Strategiegespräch konnte ich das Projekt endlich in der Praxis angehen, was ich zwar schon lange im Kopf hatte, aber über die Theorie nie hinausgekommen ist. Ergebnis: Steigerungsraten mit unserem renovierten Webshop von bis zu 30% bereits in den ersten 3 Monaten nach dem Relaunch – im Corona-Jahr»

Oliver Merklinghaus, CEO tro.net GmbH

Jetzt haben wir mit der Projektdurchführung keinen Stress mehr

«Am Anfang konnten wir noch nicht glauben, welchen Benefit wir haben werden. Das Ergebnis hat meine Erwartungen weit übertroffen: Jetzt haben wir mit der Projektdurchführung keinen Stress mehr, keine Qualitätsprobleme mehr, die Aufwandsabschätzungen passen wieder. Wir können in einem sehr geregelten Arbeitsmodus arbeiten und Projekte wie angeboten umsetzen und abrechnen.»

Jürg Weber, CEO censhare (Schweiz) AG

In einer idealen Welt wird Software in guter Qualität und **ohne die Angst** vor Problemen weiterentwickelt.

Programmierer könnten Änderungswünsche mit Freude, schnell und fehlerfrei umsetzen. Und alle Kunden würden durch die Lieferung von passgenauer, zuverlässiger und reibungsloser Software zu **begeisterten Stammkunden**.

Stell Dir vor, wie Dein Unternehmen aussieht, wenn diese Ideale Welt ein Stück näher gerückt ist:

- √ Die von Euch entwickelte Software ist **stabil**, die **Fehlerquote** ist dramatisch **gesunken**.
- √ **Kundenreklamationen** gehören weitestgehend der **Vergangenheit** an.
- √ Die **Zufriedenheit** Deiner Teams ist **gestiegen** und zufriedene **Stammkunden** mit langjährigen Vertragsverhältnissen bringen **regelmässigen** wiederkehrenden **Umsatz**, der sich auch im Deckungsbeitrag niederschlägt.

Nicht zuletzt: Du besitzt ein erfolgreiches und profitables **Unternehmen**, das von Mitarbeitenden, Kunden und potentiellen Käufern des Unternehmens als sehr **attraktiv** wahrgenommen wird.

Dein Unternehmen hat die Aufholjagd geschafft und zu den jungen und frischen Unternehmen aufgeschlossen, die bei 0 angefangen haben und mit viel Geld ausgestattet links und rechts an Dir vorbeigezogen waren.

Endlich haben die Verzweiflung und Frustration über den **Strom von Problemen**, der nicht abzureissen scheint, **nachgelassen**.

Und du hast mehr Zeit für dich, für **Regeneration** und für Deine Familie. Beginne **heute** damit, Deine Identität als Unternehmer wiederzuentdecken und Deine Zukunft zurückzubekommen.

Überzeugt? Du benötigst weitere Infos oder Unterstützung?

Gerne gebe ich dir im kostenlosen Strategiegespräch auf deine Situation dein Unternehmen abgestimmte weitere Tipps.

Jetzt unter https://eye.procelo.ch/
Termin reservieren!

Über Ralf Trapp

Vor 14 Jahren bekam ich von meinem damaligen Arbeitgeber, einem Softwarehersteller, den Auftrag, die **Softwarequalität** wieder auf ein akzeptables Niveau zu bringen.

Denn die Situation war untragbar geworden. Folgende Herausforderungen galt es zu lösen:

- ⊗ Manuell gepflegte und ausgeführte Tests auf Excel-Tapeten, die zu horrendem manuellem Aufwand beim Testing und Packaging führten
- ⊗ Updates inkl. der notwendigen Tests dauerten im Extremfall Monate und benötigten einen sehr hohen Personalaufwand
- ⊗ Mangelhafte Dokumentation führte zu frustrierten Kunden und Entwicklern

All dies hat zu schlechter Softwarequalität geführt: Wir wussten nach einem Release nie, ob vielleicht grobe Fehler darin waren und **beteten**, dass keine auftraten.

Denn Softwarefehler, die beim Kunden entdeckt werden, sind die teuersten Softwarefehler überhaupt.

Darüber hinaus wurde unser Support von Kundenreklamationen überrannt und kam mit der Bearbeitung der Fälle nicht mehr hinterher.

Mein **Schlüsselerlebnis** war, dass ich zwei Dinge erkannte:

- √ Dass all diese Probleme eine gemeinsame Ursache hatten
- √ Dass wir in der Softwarebranche in der Pole-Position sind, um so viele manuelle Schritte wie möglich zu automatisieren. So können gleichartige Schritte in hoher Frequenz und gleicher Qualität mit reproduzierbarem Ergebnis ausgeführt werden

Nach der Transformation hatten wir eine vollautomatisierte Software-Test-Strasse geschaffen, die die Qualitätssicherung der Software sofort nach Änderungen am Code automatisiert durchführt. Die neuen Softwarepakete wurden ohne menschliches Zutun zusammenstellt. Die Entwicklerdokumentation wurde automatisch erzeugt und es wurde ebenso automatisiert überprüft, dass die Software performant genug ist.

Getestete Updates konnten wir nun in **Stunden statt in Monaten** bereitstellen.

Anders ausgedrückt: Wir **wussten** zu jedem Zeitpunkt, dass die Software gut ist.

Wer einmal erlebt hat, wie entspannend dieses Wissen sein kann, möchte nie wieder anders arbeiten.

Denn in einer fairen Welt könnten sich Software-Entwickler darauf konzentrieren, die Software von morgen zu entwickeln, anstatt die Software von gestern zu reparieren. In der realen Welt ist es oft noch genau andersherum.

Meine Überzeugung ist, dass jede Software-Entwicklung optimiert werden kann. So bekommen Software-Ersteller die Wertschätzung und den Respekt der Kunden, welche sie verdienen.