inkl.
Sommer-Fahrplan

SOMMER IN SOFTWAREPROJEKTEN

Warum der Sommer mehr als eine ruhigere Projektphase ist

openDEVS

Einleitung

In vielen Softwareprojekten fühlt sich der Sommer zunächst wie eine Zwischenzeit an. Einige Kolleginnen und Kollegen sind im Urlaub, Termine werden verschoben, Entscheidungen warten auf Rückmeldungen und die operative Dynamik nimmt spürbar ab. Für Projektleitende kann diese Phase auf den ersten Blick nach Stillstand wirken.

Tatsächlich liegt genau darin eine große Chance.

Denn wenn der tägliche Druck etwas nachlässt, entsteht Raum für Fragen, die im normalen Projektbetrieb oft zu kurz kommen: Sind unsere Prioritäten für die nächste Phase wirklich klar? Trägt unsere Architektur noch das, was wir im Herbst vorhaben? Gibt es technische Schulden, die uns bald ausbremsen werden? Und ist im Team genug Seniorität vorhanden, um kommende Entscheidungen sicher zu treffen?

Gerade erfahrene Projektleitende wissen: Ein starker Herbst beginnt nicht erst nach der Sommerpause. Er wird vorher vorbereitet. Wer die ruhigeren Wochen nutzt, um Muster zu erkennen, Risiken zu sortieren und kritische Abhängigkeiten sichtbar zu machen, startet kontrollierter in die nächste Umsetzungsphase.

Der Sommer ist deshalb keine Pause vom Projekt. Er ist ein strategisches Fenster.

Dieses Whitepaper zeigt, wie du diese Zeit bewusst nutzen kannst: für einen klaren Blick auf Teamstruktur, Architektur, Delivery-Fähigkeit und den gezielten Einsatz externer Softwareentwicklung. Es hilft dir dabei, typische Reibungspunkte frühzeitig zu erkennen, notwendige Entscheidungen vorzubereiten und dein Projekt stabil für den Herbst aufzustellen.

Denn je klarer du im Sommer sortierst, desto weniger musst du im Herbst improvisieren.

DER SOMMER ALS STRATEGISCHES FENSTER

Die Sommermonate verändern selten die Komplexität eines Softwareprojekts. Anforderungen bleiben bestehen, Releases verschwinden nicht aus der Planung und technische Herausforderungen lösen sich nicht durch Urlaubszeit auf. Was sich verändert, ist der Takt.

> Weniger Meetings, weniger Eskalationen und eine geringere Zahl paralleler Entscheidungen schaffen etwas, das im normalen Projektalltag kaum verfügbar ist: **Distanz zum operativen Geschehen.**

Diese Distanz ist wertvoll, denn sie ermöglicht einen Blick auf Themen, die während laufender Sprints, enger Deadlines und permanenter Abstimmungen meist zu kurz kommen. Sie gibt dir die Chance, nicht nur über die nächsten Tickets nachzudenken, sondern darüber, was in den nächsten 6 bis 12 Monaten zählt, z.B.:

- Ist die aktuelle Architektur bereit für die geplanten Erweiterungen?
- Tragen die bestehenden Teamstrukturen die nächste Wachstumsphase?
- Sind Qualitätsprobleme unter Kontrolle oder werden sie lediglich auf spätere Releases verschoben?
- Entstehen kritische Abhängigkeiten von einzelnen Personen?
- Reichen die vorhandenen Kompetenzen aus, um die kommenden Anforderungen sicher umzusetzen?

Wer diese Fragen erst im Herbst stellt, tut dies häufig unter Zeitdruck. Dann stehen bereits neue Roadmap-Ziele im Raum, Budgets müssen umgesetzt werden und die nächsten Meilensteine rücken näher. Entscheidungen werden reaktiv getroffen, obwohl sie Monate früher hätten vorbereitet werden können.

Der Sommer bietet die Gelegenheit, das zu vermeiden.
Er schafft Raum für eine ehrliche Standortbestimmung.

Mithilfe von diesen Fragen gelingt dir eine solche Standortbestimmung:

- Welche Ziele sind tatsächlich geschäftskritisch?
- Welche Entscheidungen wurden mehrfach verschoben?
- Wo entstehen technische Schulden, die die nächste Entwicklungsphase ausbremsen könnten?
- Welche Abhängigkeiten innerhalb des Teams bergen Risiken für Geschwindigkeit, Qualität oder Stabilität?

Ebenso wichtig ist der Blick nach vorne. Wer im Herbst zusätzliche Entwicklungskapazität benötigt, eine Architektur modernisieren möchte oder spezielles Know-how für eine kritische Projektphase braucht, sollte diese Fragen nicht erst dann adressieren, wenn der Handlungsdruck bereits steigt. **Deshalb lohnt es sich, die ruhigere Phase bewusst zu nutzen.** Nicht als Pause vom Projekt, sondern als Vorbereitung auf die nächste Etappe.

WAS VOR DEM HERBST GEKLÄRT SEIN SOLLTE

Nach der Sommerpause nimmt das Tempo oft schneller wieder Fahrt auf, als man denkt. Roadmaps werden konkreter, Budgets sollen sichtbar etwas bewegen, Stakeholder möchten Fortschritte sehen und die Teams gehen vom Sortieren wieder ins Machen über. Umso besser, die Sommerzeit zu nutzen, um die wichtigsten Fragen nicht nur festzuhalten, sondern gemeinsam klare und verbindliche Antworten darauf zu finden.

NUMMER 1: WELCHE PROJEKTZIELE WIRKLICH PRIORITÄT HABEN

Nicht jedes Ziel auf der Roadmap hat denselben Wert für die nächste Projektphase. Manche Funktionen sind geschäftskritisch, weil sie Umsatz sichern, regulatorische Anforderungen erfüllen oder operative Prozesse stabilisieren. Andere Themen sind wichtig, aber nicht entscheidend für den nächsten Entwicklungsschritt.

Vor dem Herbst sollte klar sein, welche Ziele tatsächlich Priorität haben und welche bewusst zurückgestellt werden. Diese Klärung schützt Teams vor parallelen Erwartungshaltungen, überladenen Sprints und Entscheidungen, die später wieder korrigiert werden müssen.

Für Projektleitende bedeutet das: **Priorisierung darf nicht nur auf Feature-Ebene stattfinden.** Sie muss auch Architektur, Qualität, technische Risiken und verfügbare Kompetenzen berücksichtigen. Ein Ziel ist nur dann realistisch priorisiert, wenn klar ist, welche technischen Voraussetzungen, Rollen und Entscheidungen dafür gebraucht werden.

NUMMER 2: WELCHE ENTSCHEIDUNGEN ZU LANGE VERTAGT WURDEN

Vertagte Entscheidungen wirken zunächst harmlos. Ein Architekturthema wird auf den nächsten Sprint verschoben. Eine Rollenklärung bleibt offen. Eine Technologieentscheidung wird weiter diskutiert. Eine externe Unterstützung wird erst einmal nicht eingeplant.

Mit jeder Verschiebung wächst jedoch die Unsicherheit im Projekt. Teams entwickeln Zwischenlösungen, Verantwortlichkeiten bleiben interpretierbar und kritische Themen wandern von Meeting zu Meeting, ohne verbindlich gelöst zu werden. Vor dem Herbst sollte deshalb sichtbar werden, welche Entscheidungen längst fällig sind. Dazu gehören Fragen wie:

- Wer übernimmt **technische Verantwortung** für kritische Komponenten?
- Welche **Architekturprinzipien** gelten für die nächste Ausbaustufe?
- Welche **Qualitätsstandards** sind nicht verhandelbar?
- An welchen Stellen braucht das Team zusätzliche **Seniorität**, um Entscheidungen belastbar treffen zu können?

Gerade diese Fragen entscheiden darüber, ob der Herbst mit Tempo beginnt oder mit erneuten Grundsatzdiskussionen.

NUMMER 3: WO TECHNISCHE SCHULDEN BREMSEN KÖNNTEN

Technische Schulden sind nicht automatisch ein Problem. Sie entstehen in jedem anspruchsvollen Softwareprojekt, besonders wenn unter Zeitdruck geliefert wird.

Kritisch werden sie dort, wo sie neue Entwicklung verlangsamen, Fehleranfälligkeit erhöhen oder die Wartbarkeit zentraler Systembereiche gefährden.

Vor der nächsten Wachstumsphase sollte klar sein, welche technischen Schulden akzeptabel sind und welche aktiv bearbeitet werden müssen.

Nicht jede Altlast rechtfertigt sofort ein Refactoring. Aber jede relevante Altlast braucht eine bewusste Entscheidung.

Typische Warnsignale sind wiederkehrende Bugs in denselben Modulen, steigender Aufwand für kleine Änderungen, instabile Schnittstellen, schwer verständliche Abhängigkeiten oder Tests, denen das Team nicht mehr vertraut.

Wenn solche Muster bereits vor der Sommerpause sichtbar waren, werden sie im Herbst nicht verschwinden. Sie werden durch neue Anforderungen verstärkt.

Ein wirksamer Sommer-Check priorisiert technische Schulden deshalb nach Projektrisiko:

- Was gefährdet die nächsten **Releases**?
- Was blockiert **Skalierung**?
- Was bindet **unverhältnismäßig** viel Senior-Zeit?
- Was muss **stabilisiert** werden, bevor zusätzliche Entwicklungskapazität sinnvoll eingesetzt werden kann?

NUMMER 4: WELCHE ABHÄNGIGKEITEN IM TEAM SICHTBAR WERDEN

Softwareprojekte geraten selten nur deshalb ins Stocken, weil schlicht zu wenig Kapazität da ist. Häufiger sind es Abhängigkeiten, die erst dann auffallen, wenn sie bereits zum Problem werden.

Vielleicht kennt nur eine Person die zentrale Legacy-Komponente. Architekturentscheidungen laufen informell über einzelne Köpfe.

Ein Senior Developer hängt dauerhaft in Reviews, Eskalationen und Abstimmungen fest. Oder ein Product Owner muss technische Prioritäten mitentscheiden, weil die Rollen im Team nicht klar genug voneinander getrennt sind.

Solange diese Schlüsselpersonen verfügbar sind, fallen solche Abhängigkeiten im Alltag oft kaum auf. In der Sommerzeit wird jedoch schnell sichtbar, wo es hakt: Übergaben dauern länger als gedacht, Entscheidungen bleiben liegen, Reviews stapeln sich oder Aufgaben kommen nicht weiter, weil genau das Wissen fehlt, das gerade nicht verfügbar ist.

Deshalb lohnt es sich, noch vor dem Herbst genauer hinzuschauen: Wo sind Wissen, Verantwortung und Entscheidungsspielraum zu stark auf einzelne Personen konzentriert?

Aus dieser Analyse lassen sich ganz konkrete Schritte ableiten. So lassen sich die Dokumentation verbessern, Verantwortung klarer verteilen und Review-Prozesse entlasten. An manchen Stellen kann es sinnvoll sein, gezielt externe Expertinnen und Experten dazuzuholen.

Diese Transparenz ist wichtig für alles, was danach kommt. Denn erst, wenn Projektziele, offene Entscheidungen, technische Schulden und Abhängigkeiten im Team klar auf dem Tisch liegen, lässt sich realistisch einschätzen, ob das Projekt aus eigener Kraft stabil weiterlaufen kann oder ob zusätzliche Unterstützung gebraucht wird.

Damit rückt die nächste Frage in den Mittelpunkt: Ist im Team genug Seniorität vorhanden, um die kommenden Anforderungen souverän zu tragen?

GENUG KAPAZITÄT, ABER AUCH GENUG SENIORITÄT?

Ein starkes Team braucht nicht nur verfügbare Hände, sondern auch Erfahrung, klare Verantwortung und die Fähigkeit, im richtigen Moment gute Entscheidungen zu treffen.

FOLGE UNS AUF LINKEDIN FÜR NOCH MEHR BRANCHENINFOS UND BEST PRACTICES!

• SCAN HERE •

DER TEAM-CHECK: IST GENUG SENIORITÄT FÜR DIE NÄCHSTE PHASE VORHANDEN?

Ein Softwareprojekt ist immer nur so stabil wie die Entscheidungen, die zum richtigen Zeitpunkt getroffen werden. Vor einer intensiven Umsetzungsphase lohnt es sich deshalb, nicht nur auf freie Kapazitäten zu schauen. Mindestens genauso wichtig ist die Frage: Welche Erfahrung bringt das Team mit, wie klar sind Verantwortlichkeiten verteilt und wer kann Entscheidungen sicher mittragen?

Vor dem Herbst ist dafür ein guter Moment. Denn nach der Sommerpause ziehen in vielen Projekten Tempo und Erwartungsdruck spürbar an. Roadmaps werden verbindlicher, Stakeholder möchten Fortschritte sehen und technische Fragen, die bisher noch offen waren, müssen auf einmal konkret entschieden und umgesetzt werden.

Dann zeigt sich, ob im Team genug Seniorität vorhanden ist, um die nächste Etappe souverän zu meistern.

Dabei geht es bei Seniorität um weit mehr als um viele Jahre Berufserfahrung. Sie zeigt sich darin, technische Zusammenhänge schnell einzuordnen, Risiken früh zu erkennen, Prioritäten kritisch zu prüfen und auch dann gute Entscheidungen zu treffen, wenn noch nicht alle Antworten auf dem Tisch liegen.

Senior Developer kümmern sich nicht nur um besonders komplexe Aufgaben. Sie ordnen ein, geben Orientierung, halten dem Team den Rücken frei und achten darauf, dass die Qualität stimmt. Gleichzeitig sorgen sie dafür, dass technische Entscheidungen nicht losgelöst getroffen werden, sondern zu den geschäftlichen Zielen des Projekts passen.

Fehlt diese Seniorität, fällt das im Alltag oft nicht sofort auf. Stattdessen zeigen sich nach und nach typische Anzeichen: Entscheidungen ziehen sich, technische Diskussionen kommen nicht richtig voran und Reviews landen immer wieder bei denselben Schlüsselpersonen.

Architekturfragen bleiben offen, weil sich niemand klar dafür verantwortlich fühlt. **Weniger erfahrene Teammitglieder bringen sich engagiert ein, brauchen bei komplexen Aufgaben aber häufiger Rücksprache, Feedback und Orientierung.**

Das hat meist wenig mit fehlender Motivation zu tun. Viel häufiger fehlt dem Team die passende Balance aus Umsetzungskraft und erfahrener technischer Führung. Ein guter Team-Check startet deshalb mit einigen zentralen Fragen:

- Wer trifft aktuell die **wirklich kritischen** technischen Entscheidungen?
- Welche Personen werden **immer wieder** in Reviews, Eskalationen oder Architekturfragen eingebunden?
- Welche Aufgaben landen bei Teammitgliedern, die dafür eigentlich noch nicht ausreichend **Erfahrung oder Kontext** haben?
- Welche Kompetenzen **fehlen** für die nächsten Roadmap-Ziele?
- Wo **hängt** Projektfortschritt an einzelnen Schlüsselpersonen?

Besonders kritisch wird es, wenn Seniorität nur punktuell vorhanden ist. Eine einzelne erfahrene Person kann kurzfristig viel stabilisieren. Langfristig entsteht dadurch jedoch ein Risiko.

Wenn dieselbe Person Architekturentscheidungen vorbereitet, Reviews übernimmt, Bugs analysiert, Stakeholderfragen beantwortet und gleichzeitig selbst entwickeln soll, wird sie **schnell zum Engpass**. Das Projekt wirkt nach außen besetzt, aber intern ist die Entscheidungsfähigkeit zu stark konzentriert.

Die Sommermonate machen solche Muster oft sichtbarer. Sobald Schlüsselpersonen im Urlaub sind oder weniger Abstimmungen stattfinden, zeigt sich, ob Wissen und Verantwortung gut verteilt sind.

- Können Aufgaben weiterlaufen?
- Sind Entscheidungen dokumentiert?
- Versteht das Team die technischen Leitplanken?
- Bleibt Arbeit liegen, weil bestimmte Fragen nur von einer Person beantwortet werden können?

DER ARCHITEKTUR-CHECK: TRÄGT DAS SYSTEM ENTWICKLUNGSSCHRITTE?

Wenn ein Softwareprojekt wächst, wächst nicht nur der Funktionsumfang. Auch die Anforderungen an Architektur, Schnittstellen, Wartbarkeit und Skalierbarkeit steigen. Was in einer früheren Projektphase ausreichend war, kann in der nächsten Etappe zum begrenzenden Faktor werden.

Deshalb ist der Sommer ein guter Zeitpunkt, um die Architektur nicht nur technisch, sondern strategisch zu betrachten.

Im laufenden Projektbetrieb bleibt dafür oft zu wenig Raum. Neue Anforderungen müssen umgesetzt, Fehler behoben, Releases vorbereitet und Stakeholder eingebunden werden. Architekturentscheidungen entstehen dann häufig nebenbei. Sie werden pragmatisch getroffen, verschoben oder durch kurzfristige Lösungen ersetzt, weil die operative Umsetzung Priorität hat.

Das ist nicht automatisch problematisch. Softwareentwicklung lebt von pragmatischen Entscheidungen. Kritisch wird es jedoch, wenn sich über längere Zeit Muster bilden, die die nächste Wachstumsphase gefährden.

Typische Hinweise sind steigende Aufwände für eigentlich kleine Änderungen, unklare Schnittstellen, instabile Integrationen, wiederkehrende Fehler in denselben Bereichen oder Komponenten, die nur noch wenige Personen wirklich verstehen. Auch lange Diskussionen über technische Grundlagen können ein Warnsignal sein. Wenn jedes neue Feature zuerst Grundsatzfragen auslöst, fehlt dem System möglicherweise eine klare architektonische Richtung.

Ein Architektur-Check muss deshalb nicht bedeuten, das gesamte System infrage zu stellen. Vielmehr geht es darum, gezielt zu prüfen, ob die bestehende Struktur die nächsten sechs bis zwölf Monate tragen kann.

Dafür sind einige Fragen besonders relevant:

- Welche Teile des Systems sind für die nächste Projektphase **geschäftskritisch**?
- Welche Komponenten verursachen **regelmäßig** Aufwand, Fehler oder Abstimmungen?
- Sind Schnittstellen ausreichend stabil und verständlich **dokumentiert**?
- Lässt sich das System erweitern, ohne bestehende Funktionen unverhältnismäßig stark zu **gefährden**?
- Sind Skalierbarkeit, Performance und Wartbarkeit für die kommenden Anforderungen **realistisch** abgesichert?
- Gibt es Architekturentscheidungen, die bereits mehrfach **vertagt** wurden?

Was die häufigsten Gründe für das Scheitern von IT Projekten sind, liest du in diesem Blogartikel.

openDEVS

Doch mit jeder neuen Funktion, jedem Workaround und jeder zusätzlichen Abhängigkeit wird es schwieriger, später sauber zu korrigieren. **Ein ruhigerer Projektmodus schafft hier die notwendige Distanz.** Statt nur die nächste Anforderung zu betrachten, lässt sich fragen:

- Welche **Struktur** braucht das Projekt, damit die kommenden Anforderungen nicht jedes Mal neue Reibung erzeugen?
- Welche **Prinzipien** sollten für Erweiterungen, Schnittstellen, Tests oder Deployment gelten?
- Welche **Altlasten** müssen vor dem Herbst adressiert werden, damit neue Entwicklungskapazität überhaupt sinnvoll wirken kann?
- Wo müssen technische **Leitplanken** klarer werden?

Mehr Entwickler:innen lösen architektonische Probleme nicht automatisch. Ist ein System schwer zu warten, sind Verantwortlichkeiten unklar oder zentrale Komponenten instabil, sorgt zusätzliche Kapazität oft vor allem für mehr Abstimmung.

Neue Teammitglieder brauchen länger, um sich einzuarbeiten, Änderungen werden riskanter und Reviews kosten mehr Zeit. Das Team wird dadurch nicht zwangsläufig schneller, denn die Arbeit verteilt sich lediglich auf mehr Koordination.

Deshalb sollte die Architektur immer Teil der Vorbereitung auf die nächste Projektphase sein. Nicht als technisches Randthema, sondern als wichtige Grundlage dafür, verlässlich planen und umsetzen zu können.

Ein guter Architektur-Check hilft Projektleitenden dabei, Risiken früh zu erkennen und sinnvoll zu priorisieren. **Dabei muss nicht jede Schwachstelle sofort behoben werden.**

Wichtig ist vielmehr, bewusst zu entscheiden, wie damit umgegangen wird: weiter beobachten, stabilisieren, refactoren, besser dokumentieren, neu strukturieren oder gezielt externe Expertise dazuholen.

Aus der technischen Bestandsaufnahme entsteht so eine fundierte Entscheidungsgrundlage für das Management. Sie zeigt, ob das System bereit für den nächsten Entwicklungsschritt ist oder ob zunächst mehr Stabilität geschaffen werden sollte.

Lass uns gemeinsam besprechen, wie du Codequalität, Stabilität und Geschwindigkeit in deinem Projekt nachhaltig stärkst.

ROBERT GITTENBERGER

GESCHÄFTSFÜHRER

SCAN ME!

+43 677 629 084 09

www.opendevs.net

... oder buche direkt einen Videocall per QR-Code

DER DELIVERY-CHECK: WIE PLANBAR IST EURE UMSETZUNG WIRKLICH?

Planbare Umsetzung entsteht nicht durch einen Projektplan, der auf dem Papier gut aussieht. Sie entsteht dann, wenn ein Team realistisch einschätzen kann, was es leisten kann, welche Risiken im Weg stehen und welche Abhängigkeiten den Fortschritt beeinflussen.

Ein Delivery-Check macht sichtbar, wie verlässlich ein Projekt tatsächlich liefern kann. Dabei geht es nicht darum, das Team möglichst eng zu kontrollieren. Viel wichtiger ist die Frage, wie gut Arbeit vom ersten Konzept bis zum Release durch den Prozess kommt.

Erste Warnsignale zeigen sich meist schon lange vor größeren Problemen.

Releases werden mehrfach verschoben, Buglisten wachsen schneller als sie abgearbeitet werden und selbst kleine Änderungen brauchen erstaunlich viel Abstimmung. Aufgaben wechseln zwischen Rollen hin und her, weil Verantwortlichkeiten nicht klar genug sind.

Prioritäten ändern sich häufig, ohne dass die Folgen für Architektur, Qualität oder Kapazität wirklich mitgedacht werden.

Auch viele spontane Abstimmungen können ein Hinweis darauf sein, dass etwas fehlt. Vielleicht sind Prozesse nicht klar genug, technische Leitplanken fehlen oder Entscheidungen können nicht schnell genug getroffen werden. Das Team arbeitet dann zwar mit viel Einsatz, aber nicht unbedingt planbar.

Besonders heikel wird es, wenn der Fortschritt stark an informellem Wissen hängt. Solange bestimmte Personen verfügbar sind, läuft die Umsetzung.

Sobald sie ausfallen oder mit anderen Themen beschäftigt sind, gerät der Prozess ins Stocken. Aufgaben bleiben liegen, Entscheidungen werden vertagt und das Team verliert an Tempo.

Ein ehrlicher Delivery-Check schaut deshalb auf Fragen wie:

- Wie **zuverlässig** werden geplante Releases tatsächlich erreicht?
- Welche Themen **verschieben** sich regelmäßig von Sprint zu Sprint?
- Wo entstehen die meisten **Rückfragen, Fehler oder Nacharbeiten?**
- Sind **Verantwortlichkeiten** zwischen Entwicklung, Produkt, Qualitätssicherung und Projektleitung klar verteilt?
- Wie **stabil** sind Schätzungen und Planungen im Vergleich zur tatsächlichen Umsetzung?
- Welche **Hindernisse** sind dem Team längst bekannt, wurden aber bisher nicht konsequent gelöst?

Wichtig ist dabei, nicht nur auf Symptome zu schauen. Ein stockendes Release ist selten nur ein Release-Problem.

Warum fehlende Senior Devs mehr als ein "Luxusproblem" sind, liest du in diesem kostenlosen Whitepaper.

openDEVS

Ein stockendes Release ist selten nur ein Release-Problem. Dahinter können unklare Anforderungen, technische Schulden, fehlende Testabdeckung, zu wenig Seniorität, instabile Schnittstellen oder zu viele parallele Prioritäten liegen.

Ebenso ist eine wachsende Bugliste nicht nur ein Qualitätsproblem. Sie kann zeigen, dass Architekturentscheidungen fehlen, Reviews überlastet sind oder **das Team unter dauerhaftem Zeitdruck arbeitet.** Wer Delivery verbessern möchte, muss deshalb die Ursachen hinter den sichtbaren Reibungen verstehen.

Die Sommermonate bieten dafür einen guten Moment. Wenn der operative Takt etwas ruhiger ist, lassen sich wiederkehrende Muster leichter erkennen.

- # Welche Verzögerungen waren Ausnahmen?
- # Welche waren strukturell?
- # Welche Probleme entstanden durch einzelne Situationen?
- # Welche werden sich im Herbst sehr wahrscheinlich wiederholen, wenn sie jetzt nicht angesprochen werden?

Für Projektleitende ist diese Unterscheidung entscheidend. Nicht jedes Problem erfordert sofort eine große Veränderung. Aber wiederkehrende Reibung braucht eine klare Antwort.

Das kann eine bessere Priorisierung sein, eine klarere Rollenverteilung, gezielte technische Stabilisierung, eine Anpassung der Release-Prozesse oder zusätzliche Erfahrung im Team.

Ein Delivery-Check macht Umsetzung dadurch realistischer. Er zeigt nicht nur, was geplant ist, sondern wie belastbar diese Planung tatsächlich ist. Genau das ist vor einer neuen Projektphase entscheidend.

Denn ein starker Herbst beginnt nicht mit optimistischen Annahmen. Er beginnt mit einem klaren Bild davon, was das Team zuverlässig liefern kann und was zuerst stabilisiert werden muss.

EXTERNE SOFTWAREENTWICKLUNG VORAUSSCHAUEND EINBINDEN

Externe Softwareentwicklung kommt oft erst dann auf den Tisch, wenn der Druck bereits spürbar steigt: Ein Release gerät ins Wanken, das interne Team arbeitet am Limit, wichtiges Know-how fehlt oder technische Schulden bremsen die Umsetzung. Manchmal konzentrieren sich zu viele Aufgaben auf einzelne Schlüsselpersonen. Sie treffen wichtige Entscheidungen, übernehmen Reviews und stimmen sich laufend mit anderen Beteiligten ab. Auf Dauer lässt sich diese Verantwortung kaum allein tragen.

Externe Unterstützung kann das Team an solchen Stellen gezielt entlasten. Besonders wirksam ist sie, wenn sie früh genug startet und nicht erst dann dazukommt, wenn das Projekt bereits unter hohem Druck steht.

Erfahrene Expertinnen und Experten brauchen etwas Zeit, um Architektur, Codebasis, Prozesse und fachliche Zusammenhänge wirklich zu verstehen. Werden sie früh eingebunden, können sie sich gründlich einarbeiten und wichtige Entscheidungen fundiert mittragen.

Das schafft nicht nur zusätzliche Kapazität, sondern stärkt auch die Stabilität, die Qualität und die Sicherheit im Projekt.

Ein Team-, Architektur- und Delivery-Check zeigt meist recht deutlich, wo externe Unterstützung den größten Nutzen bringt. Vielleicht fehlt die nötige Seniorität für eine wichtige Architekturentscheidung. Vielleicht sollte ein kritischer Systembereich erst stabilisiert werden, bevor neue Features entstehen.

Möglicherweise ist das interne Team fachlich gut aufgestellt, aber mit Reviews, Bugfixing und Abstimmungen so stark ausgelastet, dass strategische Themen kaum vorankommen.

Auch spezielles Know-how kann gefragt sein, ohne dass es dauerhaft intern aufgebaut werden muss.

Externe Softwareentwicklung ist in solchen Situationen keine Notlösung. Sie ist eine bewusste und vorausschauende Entscheidung.

Dabei geht es um mehr als zusätzliche Arbeitskraft. Anspruchsvolle Projekte brauchen Menschen, die sich schnell einarbeiten, technische Risiken früh erkennen, klar kommunizieren und Verantwortung für die Qualität übernehmen.

Besonders wertvoll sind erfahrene Senior Developer, Architektinnen und Architekten sowie Spezialistinnen und Spezialisten, die nicht nur Aufgaben erledigen,

sondern Orientierung geben, Wissen teilen und das Projekt nachhaltig stabilisieren.

Auch der Zeitpunkt spielt eine wichtige Rolle. Beginnt die Suche erst im Herbst, wenn der Projektdruck bereits steigt, gehen schnell mehrere Wochen für Bedarfsklärung, Auswahl, Abstimmung und Einarbeitung verloren. Unter Zeitdruck rückt außerdem oft die schnelle Verfügbarkeit in den Vordergrund.

Ob die Unterstützung fachlich und menschlich wirklich zum Projekt passt, bekommt dann leicht zu wenig Aufmerksamkeit. Wer hingegen im Sommer vorbereitet, kann gezielter entscheiden:

- Welche **Kompetenzen** fehlen wirklich?
- Welche Aufgaben sollen intern bleiben und welche lassen sich **sinnvoll extern ergänzen**?
- Welche Systembereiche brauchen **besondere Erfahrung**?
- Welche Entscheidungen müssen externe Expertinnen und Experten verstehen, bevor sie effektiv arbeiten können?
- Welche **Form der Zusammenarbeit** passt zur Projektlage: punktuelle Beratung, temporäre Verstärkung, technische Führung für einen Teilbereich oder längerfristige Entwicklungskapazität?

Diese Vorbereitung sorgt dafür, dass externe Softwareentwicklung deutlich schneller und gezielter wirken kann. Neue Teammitglieder starten nicht in unklaren Strukturen, sondern wissen von Anfang an, worauf es ankommt.

Bleiben Ziele, Rollen, Architekturthemen oder Probleme im Delivery-Prozess offen, geraten auch externe Kräfte schnell in dieselben Abstimmungsschleifen wie das bestehende Team.

Vorausschauende Einbindung beginnt deshalb mit einer sauberen Vorbereitung des Projekts. Gut eingebunden, entlasten externe Spezialistinnen und Spezialisten nicht nur das interne Team.

Sie bringen neue Perspektiven in technische Entscheidungen ein, sichern kritische Umsetzungsschritte ab und helfen dabei, Tempo und Qualität in ein gutes Gleichgewicht zu bringen.

Gleichzeitig können sie Engpässe auflösen, fehlendes Wissen ergänzen und dafür sorgen, dass wichtige Themen nicht bis zur nächsten Eskalation liegen bleiben.

Besonders wertvoll ist externe Softwareentwicklung deshalb in Übergangsphasen. Also dann, wenn ein Projekt vor dem nächsten Entwicklungsschritt steht, aber noch nicht alle Voraussetzungen dafür stabil genug sind.

Der Sommer bietet dafür einen guten Rahmen. Der Bedarf lässt sich in Ruhe und mit strategischem Blick klären, bevor der Druck wieder steigt.

So wird externe Unterstützung nicht zur spontanen Reaktion auf Überlastung, sondern zu einem bewussten Baustein für einen kontrollierten und belastbaren Start in den Herbst.

Am Ende des Whitepapers findest du einen Sommerfahrplan für Softwareprojekte

KURZFAZIT

Je klarer du die wichtigsten Themen im Sommer ordnest, desto souveräner kannst du im Herbst starten. Dieser Fahrplan zeigt dir, was vor der nächsten Umsetzungsphase geklärt sein sollte: die aktuelle Projektlage, Rollen und Verantwortlichkeiten, vorhandenes Wissen, Architektur und Qualität, mögliche externe Unterstützung sowie die ersten 30 Tage nach der Sommerpause.

Wurden mehrere Bereiche als „angespannt“ oder „kritisch“ eingestuft, lohnt es sich, jetzt konkrete Unterstützung zu planen. So werden bekannte Risiken früh aufgefangen, bevor daraus akuter Projektdruck entsteht.

5 Schritte, um dein Projekt stabil für den Herbst aufzustellen

Nutze diesen Fahrplan als kompakte Standortbestimmung vor der nächsten intensiven Projektphase. Er hilft dir, Risiken früh zu erkennen, Prioritäten klarer zu setzen und rechtzeitig zu entscheiden, wo dein Team mehr Klarheit, zusätzliche Seniorität oder gezielte Unterstützung braucht.

1. Projektlage ehrlich bewerten

Ziel: Ein realistisches Bild davon gewinnen, wo das Projekt vor dem Herbst tatsächlich steht.

Leitfragen:

- **Welche Ziele sind für die nächste Phase wirklich geschäftskritisch?**
- **Welche Themen wurden zuletzt mehrfach verschoben?**
- **Wo entstehen regelmäßig Verzögerungen, Nacharbeit oder Abstimmungen?**
- **Welche Risiken sind bekannt, aber noch nicht verbindlich priorisiert?**
- **Welche Annahmen über Kapazität, Qualität oder Architektur sollten überprüft werden?**

Unsere Einschätzung:

- ☐ stabil
- ☐ angespannt
- ☐ kritisch

Wichtigste Erkenntnis:

Nächste Entscheidung:

2. Kritische Rollen und Wissenslücken identifizieren

Ziel: Sichtbar machen, ob Verantwortung, Wissen und Seniorität ausreichend verteilt sind.

Leitfragen:

- **Wer wird bei kritischen technischen Fragen immer wieder eingebunden?**
- **Welche Aufgaben bleiben liegen, wenn bestimmte Personen nicht verfügbar sind?**
- **Wo fehlt Erfahrung für die nächsten Roadmap-Ziele?**
- **Welche Rollen sind offiziell nicht klar benannt, praktisch aber entscheidend?**
- **Welches Wissen ist zu wenig dokumentiert oder zu stark an einzelne Personen gebunden?**

Unsere Einschätzung:

- ☐ stabil
- ☐ angespannt
- ☐ kritisch

Kritische Rolle oder Wissenslücke:

.....

.....

Nächste Entscheidung:

.....

.....

.....

3. Architektur- und Qualitätsrisiken priorisieren

Ziel: Technische Risiken nach ihrer Bedeutung für die nächste Projektphase ordnen.

Leitfragen:

- **Welche Systembereiche sind für den Herbst besonders geschäftskritisch?**
- **Welche Komponenten verursachen regelmäßig Fehler, Aufwand oder Abstimmungen?**
- **Welche Schnittstellen, Integrationen oder Abhängigkeiten sind instabil?**
- **Welche technischen Schulden könnten neue Entwicklung verlangsamen?**
- **Welche Qualitätsrisiken gefährden Releases, Skalierung oder Wartbarkeit?**

Unsere Einschätzung:

- ☐ stabil
- ☐ angespannt
- ☐ kritisch

Größtes technisches Risiko:

Nächste Entscheidung:

4. Externe Unterstützung frühzeitig planen

Ziel: Klären, ob und wo externe Softwareentwicklung das Projekt gezielt stabilisieren kann.

Leitfragen:

- **Welche Kompetenzen fehlen intern für die nächste Phase?**
- **Welche Aufgaben sollten nicht dauerhaft bei überlasteten Schlüsselpersonen liegen?**
- **Wo braucht es zusätzliche Seniorität, Architekturkompetenz oder Spezialwissen?**
- **Welche Aufgabenpakete eignen sich für externe Expertinnen und Experten?**
- **Was muss vorbereitet werden, damit externe Unterstützung schnell wirksam wird?**

Unsere Einschätzung:

- ☐ kein Bedarf
- ☐ möglicher Bedarf
- ☐ konkreter Bedarf

Benötigte Expertise:

Nächste Entscheidung:

5. Die ersten 30 Tage nach der Sommerpause vorbereiten

Ziel: Den Wiedereinstieg nach der Sommerpause mit klaren Prioritäten, Verantwortlichkeiten und Entscheidungen starten.

Leitfragen:

- **Welche Entscheidungen müssen direkt nach der Sommerpause getroffen werden?**
- **Welche Risiken sollen zuerst reduziert werden?**
- **Welche Rollen oder Verantwortlichkeiten müssen geschärft werden?**
- **Welche technischen Themen kommen auf die Agenda?**
- **Welche externen Spezialistinnen oder Spezialisten sollen wann eingebunden werden?**

Unsere Einschätzung:

- ☐ klar vorbereitet
- ☐ teilweise vorbereitet
- ☐ noch offen

Top-Priorität für die ersten 30 Tage:

Erster konkreter Schritt: