

Warum dauert Softwareentwicklung (immer noch) so lange?

Die Frage nach der Geschwindigkeit (oder Langsamkeit) von Softwareentwicklungsprozessen beschäftigt Organisationen, Teams und Entscheider*innen seit Jahren – trotz neuer Methoden, Frameworks und Technologien. In dieser Sitzung wurden drei Perspektiven auf die Ursachen und Lösungsansätze diskutiert – von der Plattformstrategie über KI-gestützte Modernisierung bis hin zum strukturellen Umgang mit Anforderungen.

1. Skalierung durch Plattformen – Geschwindigkeit durch Wiederverwendung

Felix Becker, DB Systel GmbH

DB Systel hat in den letzten Jahren eine technische Plattform aufgebaut, die heute über 10.000 Entwickler*innen unterstützt. Die Zahlen sprechen für sich: 70.000 Repositories, 2,4 Millionen CI/CD-Pipeline-Runs und täglich zwei Deployments.

Im Zentrum steht eine hochgradig standardisierte Infrastruktur, die Wiederverwendung fördert und Entwicklungszeit reduziert – unterstützt durch »blueprints«, Cloud-native Tools und automatisierte Governance-Dienste für Security & Compliance. Gleichzeitig bleibt die Plattform freiwillig – wer sie nutzt, zahlt intern dafür.

Kulturell wird die Plattformidee durch Communities, Hackathons und Lernräume lebendig gehalten. Dennoch entstehen neue Herausforderungen: z. B. kognitiver Overload durch »Shift Left« oder die Frage, wie sich Abhängigkeiten im Lebenszyklus sowie »End-of-Life«-Entscheidungen managen lassen.

Key Takeaways:

- Plattformen schaffen Geschwindigkeit – durch Standardisierung und Skaleneffekte
- Autonomie der Teams bleibt zentral: Plattformnutzung ist optional
- Der Plattformgedanke braucht Pflege, Community und aktives Lifecycle-Management
- Hohe Autonomie durch Shift-Left erfordert neue Verantwortung und Aufmerksamkeit für mentale Last

2. KI-gestützte Software-Modernisierung – Innovation aus Altbestand

Thomas Immich, Centigrade GmbH

In einer interaktiven »Life-Hacking-Session« zeigte Thomas Immich, wie generative KI dabei helfen kann, Altanwendungen nicht nur zu modernisieren, sondern systematisch weiterzudenken.

Der Ansatz: Mit Hilfe von KI Personas entwickeln, Features explorieren, Screens analysieren und daraus Epics und Nutzungsszenarien ableiten – quasi ein »User Story Reverse Engineering«. Besonders spannend: Der Einsatz von KI-Agenten zur Reflexion bestehender Anforderungen und zur strukturierten Exploration von Verbesserungspotenzialen.

Dabei bleibt klar: KI kann viele Schritte beschleunigen – aber echte Produktentscheidungen brauchen menschliche Intuition, Einordnung und Verantwortung.

Key Takeaways:

- KI ermöglicht neue Ansätze zur Analyse und Weiterentwicklung von Altsystemen
- Personas können durch KI erzeugt, verglichen und als Diskussionspartner genutzt werden
- KI ersetzt nicht die Produktverantwortung – »Human-in-the-Loop« bleibt essenziell

3. Epics als Verzögerungsfalle – und wie man sie auflöst

Gunnar Harde, adesso SE

In vielen Projekten verhindern nicht fehlende Methoden, sondern ungünstige Strukturen schnelle Wertgenerierung. Gunnar Harde zeigte auf, wie Epics – ursprünglich als Strukturhilfe gedacht – zu einem Problem werden können, wenn sie fachliche und zeitliche Aspekte vermischen.

Gerade in agilen Projekten führt dies dazu, dass Product Owner versucht sind, Sprints mit »kleinen« User Stories zu füllen – unabhängig vom tatsächlichen Wert (Cost of Delay). Das kann dazu führen, dass »wichtige« Anforderungen übergangen werden, weil ihre Umsetzung komplexer oder weniger greifbar ist.

Sein Vorschlag: Epics sollten ausschließlich als fachliche Container fungieren, während zeitliche Steuerung über prozessuale Container erfolgt. Diese Trennung erlaubt es, User Stories nach Business Value und Umsetzbarkeit zu priorisieren – unabhängig von ihrer ursprünglichen Epic-Zugehörigkeit. So wird das Sprint-Backlog nicht mehr von der Epics-Struktur dominiert, sondern kann gezielter auf Wertschöpfung ausgerichtet werden.

Außerdem wurde diskutiert, wie User Story Mapping und der Einsatz von Containern zusammenwirken können – etwa zur besseren Nachverfolgbarkeit in großen Projekten.

Key Takeaways:

- Epics und Container klar trennen: Raum (fachlich) vs. Zeit (prozessual)
- User Stories sollten nach Wert, nicht nur nach Größe oder Epic-Zugehörigkeit priorisiert werden
- Container helfen, Stories gezielt zu bündeln und Komplexität im Reporting zu reduzieren
- Ziel: Mehr Flexibilität für Product Owner – weniger goldene Wasserhähne, mehr echte Wirkung

Fazit der Sitzung

Warum dauert Softwareentwicklung so lange? Eine einfache Antwort gab es nicht – aber viele konkrete Einsichten.

Ob Plattformstrategie, KI oder Anforderungsmanagement: Die Komplexität liegt nicht nur im Code, sondern in mehreren Dimensionen, die ineinandergreifen:

- Prozesskomplexität in der Organisation – von Governance über Rollen bis zur Priorisierung
- Toolkomplexität – viele Werkzeuge, Schnittstellen und fragmentierte Toolchains
- Produktkomplexität und Vielfalt der Nutzer*innen – unterschiedliche Bedürfnisse und Nutzungskontexte

Klar ist: Geschwindigkeit entsteht nicht durch Methoden allein – sondern durch das Zusammenspiel von Technik, Prozessen und Kultur. Wer Softwareentwicklung wirklich beschleunigen will, muss alle drei Dimensionen aktiv gestalten.



Felix Ansmann

Referent Software &
IT-Services

T 030 27576-098

f.ansmann@bitkom.org

