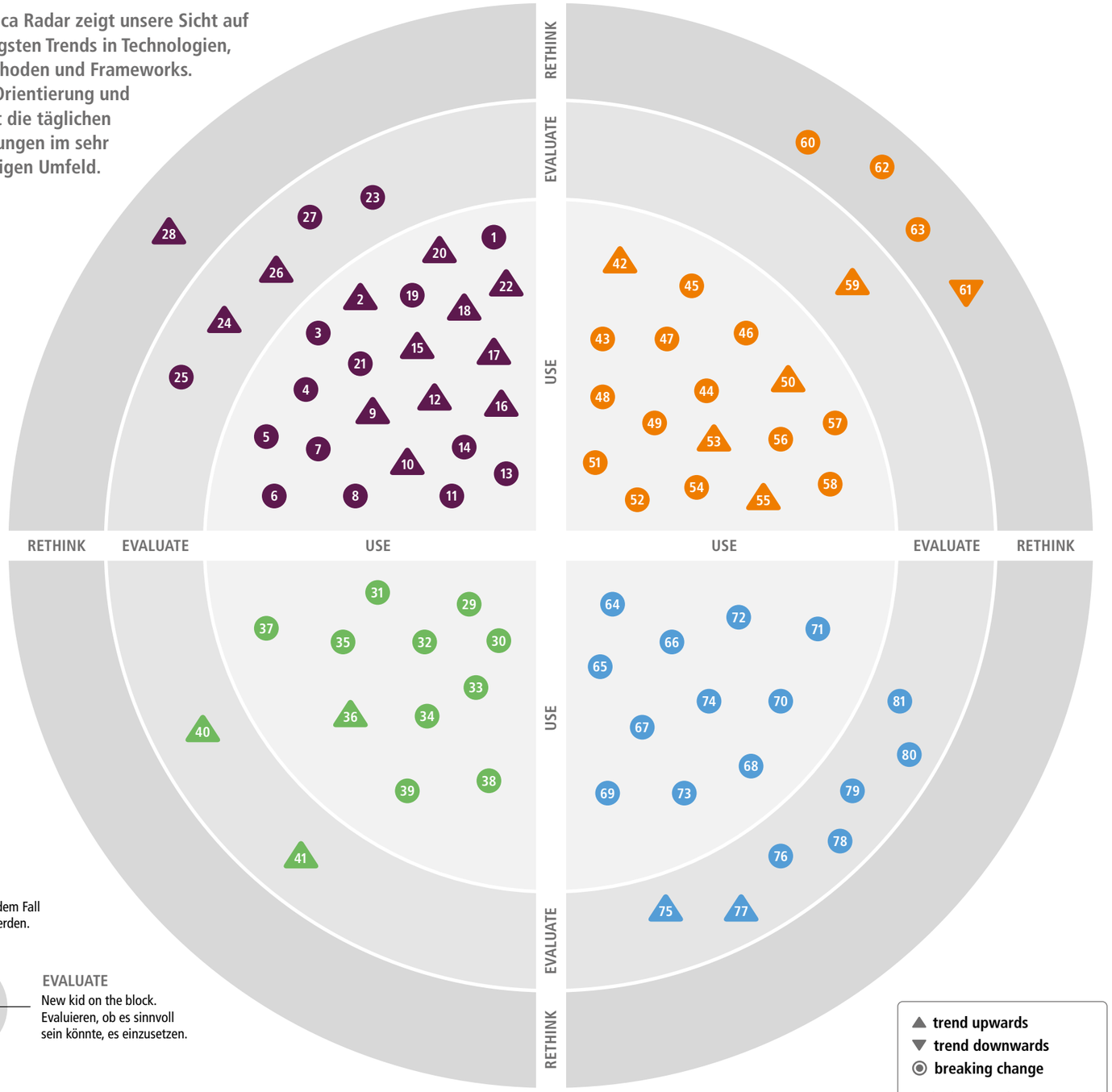
bbv
MAKING VISIONS WORK.

TECHNICA RADAR – VOL. 3

Der Technica Radar zeigt unsere Sicht auf die wichtigsten Trends in Technologien, Tools, Methoden und Frameworks. Er bietet Orientierung und erleichtert die täglichen Entscheidungen im sehr schnelllebigen Umfeld.



Concept or theme

USE
1 Advanced Analytics / Machine Learning
2 Beyond Scrum ▲
3 Clean Architecture / Ports & Adapters / Hexagonal Architecture / Onion Architecture
4 Content Strategy
5 Customer Experience
6 Design-System
7 DevOps
8 Domain Driven Design
9 Edge Computing ▲
10 Emergent Architecture ▲
11 Event Sourcing
12 Evolutionary Architecture ▲
13 Functional Programming
14 Hybrid Cloud
15 Inclusive Design ▲
16 Internet of Things ▲
17 IT-Security ▲
18 Product Lifecycle-based Development ▲
19 Serverless Architecture
20 Strategy engineered ▲
21 Web Components
22 Wertstromanalyse ▲

EVALUATE
23 Data Mesh
24 CUPID ▲
25 Distributed Cloud
26 Mixed Reality ▲
27 WebAssembly

RETHINK
28 Democratizing Programming ▲

Tools

USE
29 Cloud Computing
30 CMake
31 Cumulocity IoT
32 Dependency Vulnerability Scanning
33 Docker
34 Figma
35 Graph Database
36 Infrastructure as Code ▲
37 Kubernetes
38 Miro
39 Software as a Service (SaaS)

EVALUATE
40 3D-Visualisierung ▲
41 Programming Support with AI-Technologies ▲

Method or technique

USE
42 Acceptance Test Driven Development (ATDD) / Behavior Driven Development (BDD) ▲
43 arc42
44 ATAM – Architecture Tradeoff Analysis Method
45 Continuous Delivery
46 Continuous Refactoring / RefactorMercilessly
47 Cost of Delay
48 Event Storming
49 Large-Scale Scrum (LeSS)
50 Liberating Structures ▲
51 Pair Programming
52 Risk Management
53 Security Engineering ▲
54 Service Design
55 Systems Thinking ▲
56 Test Driven Development (TDD)
57 User Research
58 User Centered Design

EVALUATE
59 Ethical OS ▲

RETHINK
60 Design-Thinking
61 Effort Estimation ▼
62 Mob (Ensemble)-Programming
63 Scaled Agile Framework (SAFe)

Libraries, Frameworks, Programming Languages

USE
64 .NET
65 Angular
66 Azure IoT Edge
67 C++
68 Cumulocity IoT
69 Flutter
70 gRPC
71 Java
72 Python
73 React
74 Spring Framework

EVALUATE
75 Blazor ▲
76 dapr
77 F# ▲
78 Go Programming Language
79 Kotlin
80 R
81 Rust

USE
Dies kann in jedem Fall angewendet werden.

EVALUATE
New kid on the block. Evaluieren, ob es sinnvoll sein könnte, es einzusetzen.

RETHINK
Technologien und Rahmenbedingungen ändern sich und müssen manchmal neu betrachtet werden.

▲ **trend upwards**
▼ **trend downwards**
◉ **breaking change**

Beobachtete Trends im Markt und in Projekten. Die Analyse unserer Experten dazu auf der Folgeseite.

Vol. 3 – Published: September 2022

Die Analyse der wichtigsten Trends

▲ trend upwards ▼ trend downwards ◉ breaking change

Concept or theme

USE

2 Beyond Scrum ▲

Wenn Teams mit einer hohen Maturität, einer effizienten Arbeitsweise und einem Verständnis der agilen Werte und Prinzipien die kontinuierliche Verbesserung weiterführen, kann es sinnvoll sein, nur gewisse Teile von Scrum zu adaptieren und somit Scrum ein Stück weit hinter sich zu lassen.

6 Design-System

Ein Design-System ist ein Regelwerk, um Elemente und Strukturen von Benutzeroberflächen zu konzipieren. Dabei werden Verhalten, Aussehen und Kombinierbarkeit von Attributen festgelegt, sodass das Design eines neuen Elements einer Benutzeroberfläche aus den Regeln abgeleitet werden kann. Dies ermöglicht es Teams, neue Elemente zu entwickeln, ohne die Konsistenz der Benutzerführung zu gefährden oder die Design-Prinzipien zu verletzen.

9 Edge Computing ▲

Edge Computing ist ein Lösungspattern, das Logik, Sicherheits- und Datenschutzaspekte an den Rand zur Cloud-Lösung verlagert. Zusätzlich eignet sich das Pattern, um das Datenvolumen in der Kommunikation mit der Cloud zu steuern. Im Kontext von IoT-Projekten ist Edge Computing nützlich, um Daten einzelner Gerätegruppen zusammenzufassen und die Kommunikation zu strukturieren.

10 Emergent Architecture ▲

In einer komplexen Umgebung ist es unmöglich, eine Architektur zu finalisieren, dennoch sollte sie die aktuellen architekturellen Probleme lösen. Solch eine Architektur wird fortwährend weiterentwickelt (emerge), indem Architekturprinzipien wie Modularisierung und Clean Architecture kontinuierlich angewendet werden, statt die Architektur vor der Entwicklung zu entwerfen.

12 Evolutionary Architecture ▲

In einer komplexen Entwicklungsumgebung muss die Software-Architektur flexibel bleiben. Aufgrund konstanter Änderungen der eingesetzten Technologien, der Cloud-Dienste und Business-Prozesse müssen gewisse Teile der Architektur ersetzt, andere verbessert werden.

13 Functional Programming

Es ist einfacher geworden, mit funktionalen Sprachen zu arbeiten. Immer mehr Entwickler nutzen funktionale Programmiersprachen und deren Vorteil der Typsysteme, um die Geschäftsdomäne zu modellieren. Der Fokus auf «Immutability» und auf zustandslosem Design vereinfacht die Implementation und Verständlichkeit der Businesslogik.

16 Internet of Things ▲

IoT hat sich etabliert und ist zum Teil des IT-Lösungsraumes geworden. IoT wächst stetig und entwickelt sich weiter.

17 IT-Security ▲

In Zeiten, in denen eine effektive Ransomware-Software für ein paar hundert Dollar im Darknet zu erwerben ist, ist das Thema IT-Security wichtiger denn je. Diverse Fälle in den Medien bestätigen die reale Bedrohung dieser Angriffe. Es gilt, sein Sicherheitsnetz gesamtheitlich zu überdenken, von Netzwerksegmentierung über die Passwortstrategie (keine universalen Passwörter) bis hin zur 2-Faktor-Authentisierung.

18 Product Lifecycle-based Development ▲

Wie in der klassischen Industrie muss sich die Software-Entwicklungsindustrie über den ganzen Software-Produktlebenszyklus Gedanken machen. Dies kann die Gesamtheit der Kosten einer Investition, die über ihren kompletten Lebenszyklus anfallen (total cost of ownership), verkleinern. Im Vergleich zu stabilen Projektteams führen stabile Produktteams zu weniger Spannungen. Weitere Resultate sind eine kontinuierliche, stabile Weiterentwicklung, sich evolvierende Systeme und stabileres Domänenwissen.

20 Strategy engineered ▲

Die Technologie wird ein immer grösserer Treiber in der Gesamtstrategie eines Unternehmens. Heute sind Business und Technologie eng miteinander verwoben. Damit eine realistische Strategie erarbeitet werden kann, sollte eine enge Zusammenarbeit zwischen Business und Technologie etabliert werden. Beide Seiten müssen sich besser verstehen und mehr aufeinander eingehen. Daher sollte die Person mit Technologie-Verantwortung aktiv in die Strategieentwicklung miteinbezogen werden. Die Entwicklung einer Strategie wird so zu einem gemeinsamen und anhaltend iterativen Prozess.

21 Web Components

Web Components verkleinern die Komplexität der Web-Entwicklung. Durch das Definieren von Web Components werden etliche Teile der Webseite einfacher.

22 Wertstromanalyse ▲

Die Wertstromanalyse hat zum Ziel, den Fluss von Entscheidungen, Informationsvermittlung und Arbeitsschritten zu optimieren. Dazu wird in einer Wertstromanalyse bei einer Produktentwicklung geprüft, wie lange die einzelnen Schritte dauern. Dabei geht es nicht nur um die Implementierungszeit, sondern auch darum, wie lange es dauert, bis ein Team eine Antwort auf eine Frage hat, oder wie lange es dauert, bis eine Entscheidung, welches Produktfeature als nächstes angegangen wird, getroffen ist. Die Absicht ist, alle Zeiten in der Organisation so kurz wie möglich zu halten, um so Produkte effizienter entwickeln zu können.

EVALUATE

24 CUPID ▲

Mit CUPID frischt Dan North die Diskussion zu gut designtem Code auf und wechselt im Paradigma von einer Gut/Schlecht-Beurteilung von SOLID zu einer zielorientierten Beschreibung, was gutes Codedesign ausmacht.

RETHINK

28 Democratizing Programming ▲

Democratizing Programming und Low Code sind eine Möglichkeit, ohne Studium oder Ausbildung in Informatik die Businesslogik einer Applikation in Teilen zu automatisieren oder gesamte Applikationen zu entwickeln. Democratizing Programming ersetzt keine klassische IT-Entwicklung und sollte zusammen mit der Softwareentwicklung im Unternehmen eingeführt werden, um keine falschen Erwartungen zu wecken und Missverständnisse zu vermeiden. Dieses Konzept kann dabei unterstützen, Softwareentwicklung und Business einander näherzubringen.

Tools

USE

32 Dependency Vulnerability Scanning

Mit jeder Abhängigkeit besteht die Gefahr, dass eine Sicherheitslücke in eine Software eingebunden wird. Der erste wichtige Schritt ist, die Abhängigkeiten immer auf dem aktuellen Stand zu halten. Der zweite Schritt besteht darin, mögliche Sicherheitslücken automatisiert zu erkennen. In vielen Ökosystemen gehört das zum Standard und ist gut integriert. Seit .NET 5 wird eine Überprüfung der Abhängigkeiten gegen potenzielle Sicherheitslücken als Standardfunktion angeboten. Wir empfehlen dringend, diese zu nutzen.

36 Infrastructure as Code ▲

Infrastrukturen werden komplizierter. Um die Qualität aufrechtzuerhalten und Migrationen zu automatisieren, werden Infrastrukturen in maschinenlesbarem Format definiert und versioniert. Ein Trade-off zwischen manuell und IaC sollte bedacht werden. Der IaC-Lösung sollte der Vorzug gegeben werden.

EVALUATE

40 3D-Visualisierung ▲

Die Maturität von 3D-Engines ist schon länger hoch. Deutlich verbessert haben sich die Editoren, um 3D-Visualisierungen zu gestalten. Es ist leichter geworden, 3D-Modelle zu generieren und zu animieren. So werden 3D-Engines, die ursprünglich für die Spieleentwicklung gedacht waren, immer mehr in anderen Anwendungsgebieten, wie z. B. in der Industrie zur Realisierung eines Digital Twin, benutzt.

41 Programming Support with AI Technologies ▲

Mit Programming Support with AI Technologies wird der erste Schritt in Richtung AI Engineering genommen. Mittlerweile entwickeln wir nicht mehr nur AI, sondern AI unterstützt uns auch bei der Entwicklung – mit Werkzeugen wie z. B. GitHub Copilot oder askjarvis.io. Aktuell ist der Stand der Tools mit «Command Completion Next Level» zu bezeichnen. Basierend auf Kontext, wie z. B. einer geschriebenen Funktionssignatur, wird eine Implementierung vorgeschlagen.

Method or technique

USE

43 arc42

Architektur-Dokumentationen wurden leider in letzter Zeit vernachlässigt. Arc42 bietet ein Template für die Struktur und Hilfestellung zum Erstellen der Architektur-Dokumentation. Dadurch ist die Dokumentation formalisiert und bietet die Möglichkeit, einzelne Aspekte der Architektur strukturiert zu diskutieren und zu bewerten. Arc42 liefert ein solides Grundgerüst in einem pragmatischen Ansatz, ist an individuelle Bedürfnisse flexibel anpassbar und lässt sich mit anderen Ansätzen wie z. B. dem C4-Modell kombinieren.

47 Cost of Delay

Immer mehr Firmen sehen die Vorteile einer schnellen Produkteinführung (time-to-market), um früh Geld zu verdienen. Der ökonomische Effekt von Verzögerungen (cost of delay) wird kontinuierlich ein wichtigerer Faktor für die Priorisierung von Produktinkrementen.

48 Event Storming

Machen Sie abstrakte Dinge für das gesamte Produktteam (Entwicklung, Buisness-Verantwortliche, Vertrieb usw.) fassbar, indem Sie zusammen die Geschäftsdomäne erforschen und verstehen. Gleichen Sie das verwendete Vokabular einander an, und definieren Sie gemeinsam verwendete Begriffe.

50 Liberating Structures ▲

Die «liberating structures» haben sich als hilfreich herausgestellt. Die Selbstorganisation in abgestecktem Rahmen hat gezeigt, dass Mitarbeitende sich stärker mit den Arbeitsergebnissen identifizieren. Wenden Sie es an, und Sie werden über den erzeugten Mehrwert staunen.

53 Security Engineering ▲

Security Engineering bedeutet, Softwareentwicklung unter dem Paradigma Secure-by-design. Es gibt heute kaum noch abgeschottete Softwaresysteme, somit sind Attacken garantiert und die Security unabdingbar. Durch Zero Trust, Threat-Analysen und Risiko-Mitigationen kann ein Shift-Left erreicht werden. Zusätzlich braucht es eine Validierung durch ein Security Testing (z. B. Penetration Test, Security Review).

55 Systems Thinking ▲

Systems Thinking ist sehr machtvoll für die Analyse und Konzeption von komplexen Systemen. Es bildet eine Brücke zwischen Businesslogik, dem Fluss und der Verarbeitung der Daten in Systemen. Die Präzision von systematischen Einschränkungen (es gibt kein «Hier passiert die Magie») unterstützt Teams, Entscheidungen zu finden und bei Design-Kompromissen (trade-off) zu entscheiden.

58 User Centered Design

Nutzer werden immer selbstbewusster. Mit den zunehmenden Alternativen an Lösungen, um seine Ziele zu erreichen, wird der Nutzer diejenige Lösung wählen, die am einfachsten zu verwenden ist.

EVALUATE

59 Ethical OS ▲

Ethical OS ist ein Framework, das Start-ups und Produktentwicklern einen Fragenkatalog an die Hand gibt, damit sie das Risiko möglicher moralischer Desaster ihrer Produkte minimieren können. Diese Fragen befassen sich mit Aspekten geplanter Produktfeatures, die das Produkt in einen moralisch fragwürdigen Kontext bringen können. Beispiel: Das Feature zur Ortung eines verlorenen Mobiltelefons wird schlussendlich zur Überwachung einer Person eingesetzt.

RETHINK

60 Design-Thinking

Immer mehr Unternehmen verfügen über Innovationsabteilungen, die gute Produktideen generieren. Doch meistens findet nur eine kleine Anzahl der Ideen den Weg in die Produktentwicklung. Auch wenn Workshops auf dem Markt mit dem Titel «Design-Thinking» angeboten werden, ist zu beachten, dass der gesamte Design-Thinking-Prozess Wochen oder Monate benötigt, bis die Methode Wirkung zeigt.

61 Effort Estimation ▼

In einem komplexen Umfeld ist vieles nicht voraussehbar. Daher ist eine zuverlässige Schätzung über einen längeren Zeitraum unmöglich. Anstelle von vermeintlich genauen Plänen und der Frage «Wie aufwändig ist das?» sollten Teams eher den Fokus auf «Was gewinnen wir?» legen und kontinuierlich an Produktinkrementen arbeiten, die den bestmöglichen Wert generieren.

62 Mob (Ensemble)-Programming

Wenn alle Teamfähigkeiten und das komplette Wissen bei einer massgeblichen Softwarekomponente einbezogen werden, lohnt es sich, Mob-Programming anzuwenden.

Libraries, Frameworks, Programming Languages

USE

69 Flutter

Die Maturität von Flutter wächst. Seit 2.0 können Applikationen für das Web entwickelt werden. Es vereinfacht die Entwicklung von Services über mehrere Plattformen hinweg.

70 gRPC

Ein Standard – basierend auf den Nutzungszahlen – für synchrone Kommunikation zwischen Services.

72 Python

Aufgrund der Fähigkeiten, einfach strukturierte Daten zu prozessieren, erlebt Python eine Renaissance in den Bereichen Machine Learning und Big Data.

EVALUATE

75 Blazor ▲

Eine in das .NET-Ökosystem integrierte Web-Entwicklungsumgebung. Der Pluspunkt: bessere Zugänglichkeit der Webentwicklung für .NET-Entwickler.

76 dapr

Dapr verspricht als Distributed Application Runtime Entwicklern, sich auf die Businesslogik fokussieren und die Querschnittsthemen delegieren zu können, wie die Anbindung und Kommunikation mit anderen Systemen. Wir sehen Analogien zu Serverless Computing. Daher könnte Dapr eine gute Grundlage für zukünftige verteilte Systeme bieten.

77 F# ▲

Die prägnante, robuste «functional-first»-Sprache F# hat eine viel kleinere Sprachspezifikation als C# und ist damit wohl einfacher zu erlernen und einzusetzen. Trotzdem ist sie als Vorreiter oft die Quelle für die neueren Sprachfeatures von C#, auch wenn sie nicht immer gleichwertig umgesetzt werden. F# kann für die Entwicklung eines Frontends, eines Backends oder zur Infrastruktur verwendet werden.

80 R

Die Themen Data Science und Machine Learning sind mittlerweile in der Entwicklung von digitalen Produkten sehr verbreitet. Im Zuge dessen gewinnt die Programmiersprache R für die statistische Auswertung und Verarbeitung von Daten an Bedeutung.

Vol. 3 – Published: September 2022