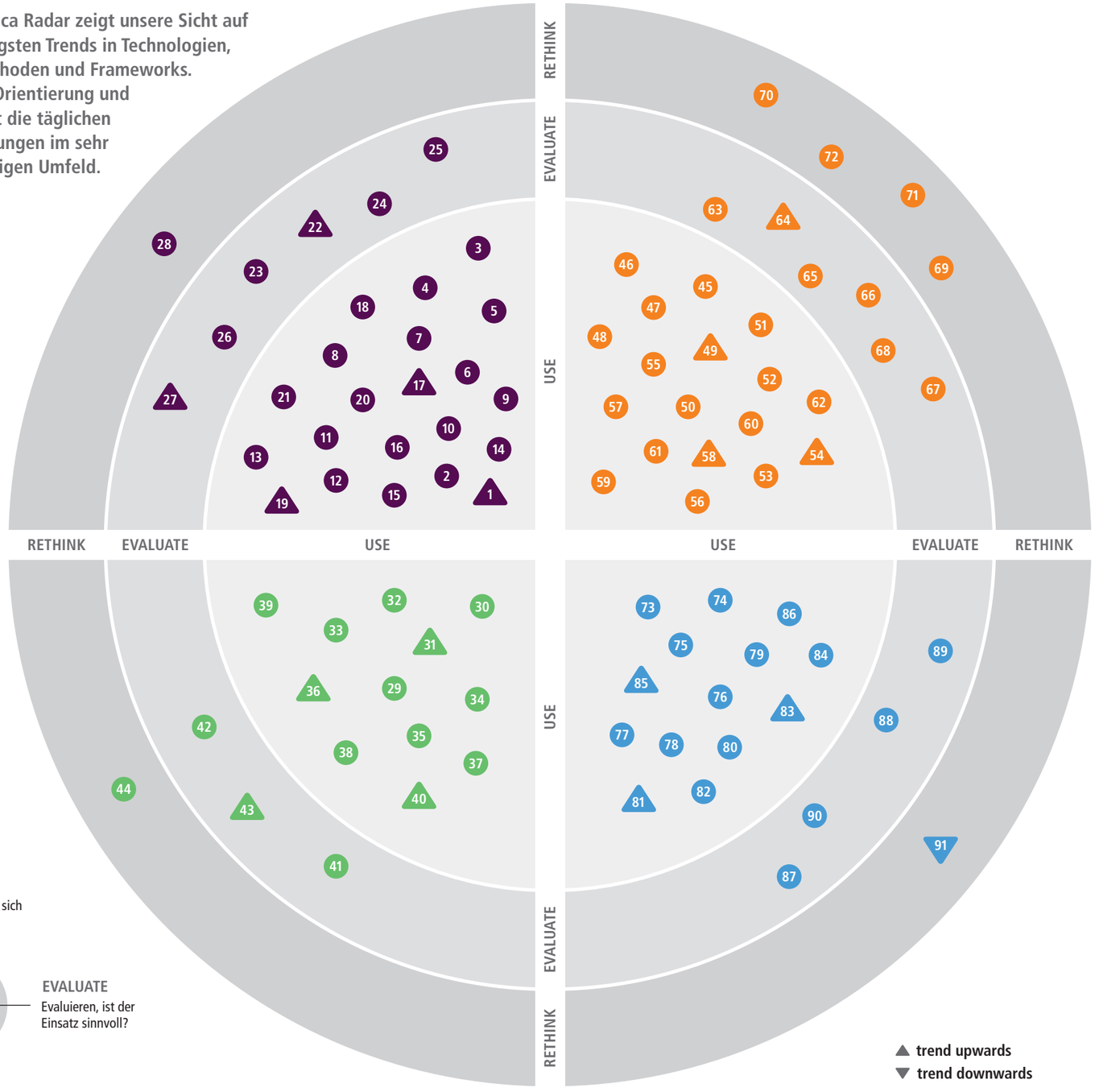
TECHNICA RADAR – VOL. 5/2025

bbv
MAKING VISIONS WORK.

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 Clean Architecture / Ports & Adapters / Hexagonal Architecture / Onion Architecture
3 Clean Code
4 Cloud Native
5 Design system
6 DevOps
7 Domain Driven Design
8 Edge Computing
9 Emergent Architecture
10 Evolutionary Architecture
11 Functional Programming
12 Hybrid Cloud
13 Inclusive Design
14 Internet of Things
15 Product Life Cycle Based Development
16 Retrieval Augmented Generation (RAG)
17 Security Engineering ▲
18 Serverless Architecture
19 Software Bill of Materials (SBOM) ▲
20 Web Components
21 Wertstromorganisation

EVALUATE
22 Developer Experience (DevEx) ▲
23 Mixed Reality
24 Platform Engineering
25 AI at the Edge
26 Multiagent-Systems
27 Democratizing Programming ▲

RETHINK
28 Data Strategy/Governance

Tools

USE
29 Cloud Computing
30 Conan
31 Dependency Vulnerability Scanning ▲
32 Docker
33 ESLint
34 Figma
35 Graph database
36 Infrastructure as Code ▲
37 Miro
38 Prettier
39 Software as a Service (SaaS)
40 Vite ▲

EVALUATE
41 Cursor
42 GitHub Copilot
43 Programming Support with AI-Technologies ▲

RETHINK
44 Kubernetes

Method or technique

USE
45 Acceptance Test Driven Development (ATDD) / Behavior Driven Development (BDD)
46 arc42 [Documentation]
47 Continuous Refactoring / RefactorMercilessly
48 Cost of Delay
49 Dependency scanning for vulnerabilities ▲
50 Design Thinking
51 Domain Storytelling
52 Ethical OS
53 Event Storming
54 Incident response process ▲
55 Pair Programming
56 Risk management
57 Service Design
58 Systems Thinking ▲
59 Test Driven Development (TDD)
60 Technical debt management
61 User Research
62 User-Centered-Design

EVALUATE
63 Continuous compliance
64 API Security Scanning ▲
65 Crowd-based Requirements Engineering
66 Point security
67 Technical Dept management
68 Worldmodels

RETHINK
69 Continuous Delivery
70 Scrum
71 SAFe – Scaled Agile Framework
72 Zone Security

Libraries, Frameworks, Programming Languages

USE
73 .NET
74 Azure IoT Edge
75 C++
76 F#
77 Flutter
78 Java
79 Jest
80 Kotlin
81 LLM ▲
82 Python
83 Preact ▲
84 React
85 Rust ▲
86 Spring Framework

EVALUATE
87 Aspire
88 Go Programming Language
89 Model Context Protocol (MCP)
90 R

RETHINK
91 Angular ▼

USE
Anwenden, hat sich bewährt.

EVALUATE
Evaluieren, ist der Einsatz sinnvoll?

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

▲ **trend upwards**
▼ **trend downwards**

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

Vol. 5 – Published: April 2025
This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Die Analyse der wichtigsten Trends

▲ trend upwards ▼ trend downwards

Concept or theme

USE

17 Security Engineering ▲

Bezogen auf Security gilt heute folgende Aussage: «Die Frage ist nicht, ob ein System angegriffen wird, sondern wann.» Investition in Security sollte so selbstverständlich sein wie die Entwicklung zum Softwareengineering gehört. Das Risiko ist zu hoch, um nicht in die entsprechenden Mitigationen gegen das Risiko zu investieren. Sprich: Weiterbildung der Engineers inklusive aller Beteiligten, Zeit für die Umsetzung, entsprechende Priorisierung der Umsetzungspakete sowie einen Secure Software Development Lifecycle. Ein externes Penetration Testing ist für jedes System eine gute Investition.

19 Software Bill of Materials (SBOM) ▲

Um dem Cyber Resilience Act gerecht zu werden, wird die Erstellung, Pflege und anschliessende Analyse eines SBOMs notwendig. Mit einer zusätzlichen Software Composition Analysis werden Softwarekomponenten gegen Listen von bekannten Schwachstellen geprüft, damit sie vom Team richtig behandelt werden können. CycloneDX definiert in diesem Umfeld einen möglichen Standard, der von diversen Tools unterstützt wird.

EVALUATE

22 Developer Experience (DevEx) ▲

Mit der DevEx wird die Effektivität von Software Ingenieure in einem Unternehmen beleuchtet. Eine positive DevEx reduziert kognitive Belastung, fördert Kreativität und beschleunigt Entwicklungszyklen. Somit kann eine gute Experience Talente anziehen und Fluktuation reduzieren. Der Fokus sollte auf die Automatisierung und kleine Feedback-Loops gelegt werden. Dies kann zu einem Wettbewerbsvorteil werden.

26 Multiagent-Systems und Agentic Process Automation

Agentic Process Automation beschreibt den Einsatz KI-gestützter Agenten, um komplexe Arbeitsabläufe autonom zu steuern. Dabei übernehmen die Agenten Entscheidungs- und Koordinationsaufgaben, auch von anderen Agenten, ohne dass menschliches Eingreifen nötig ist. So können Prozesse intelligenter, flexibler und effizienter gestaltet werden. Mit Workflow-basierten Agents können Arbeitsabläufe automatisiert und orchestriert werden. Sie koordinieren Schritte und Rollen, indem sie vordefinierten Prozessen folgen oder auf Ereignisse reagieren. Dies steigert Effizienz und Qualität, indem redundante manuelle Schritte reduziert werden.

27 Democratizing Programming ▲

Lösungen wie No-Code/Low-Code befähigen Mitarbeitende, unabhängig von der IT-Abteilung Lösungen zu erstellen. Wichtig ist dabei ein klares Regelwerk aufzustellen, denn diese eher kleineren Lösungen werden oft unstrukturiert erstellt und es kann schnell ein Wildwuchs entstehen. Um eine effektive Umgebung aufzubauen, sollte nebst gewissen Richtlinien auch eine gute Datenstrategie und ein entsprechendes Datenmanagement als Fundament dienen.

RETHINK

28 Data Strategy/Governance

Daten sind das Fundament moderner Unternehmen – ohne eine klare Strategie, Governance, Integration und Harmonisierung bleiben sie jedoch oft ungenutzt. Eine effektive Datenstrategie sorgt dafür, dass Daten sinnvoll strukturiert und als wertvolles Asset genutzt werden. Dazu gehören definierte Verantwortlichkeiten, hohe Datenqualitätsstandards und die Schaffung einer konsistenten Architektur für Speicherung, Verarbeitung und Zugriff. Nur durch die Integration und Harmonisierung von Daten aus verschiedenen Quellen entstehen einheitliche, verwertbare Informationen. Dies bildet die Basis für zukünftige Innovationen wie Generative AI-Agents und ermöglicht datenbasierte Entscheidungen, die Wettbewerbsvorteile sichern.

Tools

USE

31 Dependency Vulnerability Scanning ▲

Die Software Supply Chain und externe Abhängigkeiten sind ein bevorzugtes Einfallstor für Angreifer und können erhebliche Sicherheits- und Compliance-Risiken mit sich bringen. Dependency Scanning ermöglicht die automatische Erkennung unsicherer Versionen und erleichtert es, Schwachstellen frühzeitig zu adressieren. Diverse Tools unterstützen die Ingenieure im Scannen dieser Abhängigkeiten, jedoch ist ein reines Scannen nicht genug – ohne klare Prozesse bleibt technische Schuld bestehen. Unternehmen sollten automatisierte Sicherheitsprüfungen in ihre CI/CD-Pipelines integrieren, veraltete Abhängigkeiten aktiv vermeiden und sich nicht blind auf «neuste Version = sicherste Version» verlassen. Ein bewusster Umgang mit Dependencies, inklusive regelmässiger Audits und einer Minimal-Abhängigkeitsstrategie, reduziert das Risiko und verbessert langfristig die Wartbarkeit der Software.

EVALUATE

43 Programming Support with AI-Technologies ▲

AI-gestützte Entwicklungstools wie GitHub Copilot, Cursor, Cody u.w. verändern die Art und Weise, wie Software geschrieben wird. Diese Tools können Unterstützung in vielen Bereichen bieten. Nebst den reinen Tools können Generative AI-Tools auch in anderen Bereichen wie automatisierten PR-Reviews eingesetzt werden. Ein durchdachter Einsatz kann die Developer Experience verbessern, und Probleme können effizienter gelöst werden. Ein Ersatz für ein tiefgehendes technisches Verständnis sind sie in keinem Fall. Unternehmen sollten evaluieren, in welchen Bereichen diese Tools echte Produktivitätsgewinne bringen und wo sie möglicherweise mehr Risiko als Nutzen erzeugen.

RETHINK

44 Kubernetes

Kubernetes wurde zum De-facto-Standard für Container-Orchestrierung und bietet unvergleichliche Flexibilität, Skalierbarkeit und Automatisierungsmöglichkeiten. Diese Vorteile kommen jedoch mit einer hochkomplexen Infrastruktur, welche tiefgehendes Fachwissen erfordert. Viele unserer Umfelder erfordern nicht ansatzweise die Mächtigkeit von Kubernetes und könnten mit simpleren Umgebungen wie PaaS- oder Serverless-Ansätzen effektiver betrieben und betreut werden. Es sollte kritisch geprüft werden, ob der Mehrwert die Komplexität rechtfertigt.

Method or technique

USE

51 Domain Storytelling

Nichts bleibt so im Gedächtnis hängen wie eine gute Geschichte. Beim Erzählen und Visualisieren von Geschichten im Kontext einer Domäne werden Prozesse und Domänenwissen für das Entwicklungsteam fassbar. Es entsteht eine gemeinsame Sprache und ein gemeinsames Verständnis für die Abläufe und Rahmenbedingungen der Softwareentwicklung in der Domäne.

54 Incident response process ▲

Mit der Einführung von RED, NIS-2 und dem Cyber Resilience Act werden neue Anforderungen an einen Secure Software Development Lifecycle gestellt. Unter anderem wird durch NIS-2 ein Incident Response Process vorausgesetzt. Eigene Umgebungen sollten schnellstmöglich gegen die Anforderungen dieser Vorgaben der EU geprüft werden.

60 Technical debt management

Technische Schulden sind ein unvermeidbarer Bestandteil jeder Softwareentwicklung – sie entstehen durch bewusste Trade-offs oder unzureichende Qualitätssicherung. In vielen Systemen liegt der Fokus stark auf der Umsetzung neuer Features, während technische Schulden weder systematisch erfasst noch aktiv angegangen werden. Eine nachhaltige Lösung erfordert jedoch die strukturierte Erfassung, Priorisierung und gezielte Reduzierung technischer Schulden. Davon betroffen sind nicht nur die Architektur und der Code, sondern auch begleitende Aspekte wie die Dokumentation.

EVALUATE

68 Worldmodels

Worldmodels sind interne Repräsentationen, die eine AI von ihrer Umgebung aufbaut. Sie ermöglichen der AI, zukünftige Szenarien zu simulieren und Handlungen vorauszuplanen. So kann das System hypothetische Situationen und deren Konsequenzen durchspielen – beispielsweise im Unternehmen oder im Marktsegment – bevor echte Aktionen ausgeführt werden.

RETHINK

69 Continuous Delivery

Continuous Delivery ist heutzutage für viele Systeme essenziell, um schnell ausliefern zu können. Continuous Delivery ist gesamtheitlich zu denken: kontinuierlich Mehrwert an den Kunden ausrollen. Es reicht nicht, wenn nur ein Continuous Deployment eingerichtet wird – die Kontinuität sollte über die ganze Wertschöpfungskette gehen.

70 Scrum

Scrum ist tot, es lebe Scrum. In den vorigen Versionen des Technica Radars lag der Fokus «Beyond Scrum» für Teams mit sehr viel Erfahrung. Bei einem Team, das neu anfängt, ist es wichtig, die Diskussion zu führen, was von Scrum wie auf das Manifest für Agile Softwareentwicklung einzahlt und was nicht – statt einer Diskussion «Ist das Scrum oder ist das nicht Scrum?». Auch hier gilt: Die Regeln zuerst verstehen, um sie später brechen zu können. Ein Regelwerk ist Orientierung, kein Dogma.

Libraries, Frameworks, Programming Languages

USE

85 Rust ▲

Rust ist gekommen, um zu bleiben. Ob es C oder sogar C++ ablöst, ist noch fraglich – bei C unserer Meinung nach wahrscheinlich. Neben dem Einzug in den Linux-Kernel setzen auch grosse Unternehmen wie Microsoft bei der Entwicklung des Windows-Kernel oder Azure System Code auf Rust statt C.

EVALUATE

89 Model Context Protocol (MCP)

MCP standardisiert als öffentliches Protokoll, wie Applikationen Kontext an LLMs anbieten und auch empfangen. Das erlaubt Applikationen wie IDEs die Interaktion zwischen LLMs und Datenquellen oder Services – so können LLM-gestützte Automatisierungen erstellt werden. MCP wird heute, als der De-facto-Standard behandelt. Experimentieren und herausfinden, wie lange der Standard lebt. Die Herausforderung liegt in der Standardisierung und der sicheren Verwaltung des Kontexts: Welche Daten dürfen verwendet werden?

RETHINK

91 Angular ▼

Nach ein paar Jahren Unsicherheit in der Web-UI-Welt haben sich mehrere ausgereifte Frameworks/Bibliotheken etabliert. Angular ist eines davon. Die Maintenance-Kosten bei Angular sind jedoch nicht ausser Acht zu lassen, gerade die letzten Aktualisierungen führten zu aufwändigeren Migrationen, wenn die neuen Funktionen unterstützt oder abgekündigte Features umgebaut werden sollen. Zudem erfordert Angular spezifisches Wissen, das sich nur schwer auf andere Frameworks übertragen lässt. Somit ist ein möglicher impliziter Entscheid für ein nicht so leichtgewichtiges Angular als Standard zu hinterfragen.

Dieser Radar kann Spuren von AI generierten Texten enthalten.

Legal Disclaimer: While we have made every attempt to ensure that the information in this publication has been obtained from reliable sources, bbv Software Services AG (bbv) is not responsible for any errors or omissions, or for the results obtained from the use of this information. All information is provided with no guarantee of completeness or accuracy, and without warranty of any kind. In no event will bbv or its employees therefore be liable to you or anyone else for any decision made or action taken in reliance on the information in this publication. The information in this publication should not be used as a substitute for consultation with professional bbv advisors. Before making any decision or taking any action, you should consult a bbv professional. The names of actual companies and products mentioned in this publication may be the trademarks of their respective owners.

Vol. 5 – Published: April 2025
This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.