Agile

Agile Softwareentwicklung hat sich als Stand der Technik weitgehend durchgesetzt – kann aber auch schiefgehen. Eine dem jeweiligen Zweck angepasste Auswahl an Methoden, die richtigen Werkzeuge für verteilte Retrospektiven und solides Schätzen des Aufwands weisen den Weg zum Erfolg.

ab Seite 7

Qualitätssicherung

Künstliche Intelligenz unterstützt Entwicklerinnen und Entwickler in immer mehr Bereichen vom Schreiben des Codes bis zu dessen Analyse. Mit KI aufgerüstete Entwicklungsumgebungen und Unit-Testing-Tools erleichtern den Programmieralltag. Automatisierte Qualitätskontrolle gelingt aber auch ohne KI.

ab Seite 31

Agile

Qualitätssicherung

Programmiersprachen

GitOps und Containerisierung

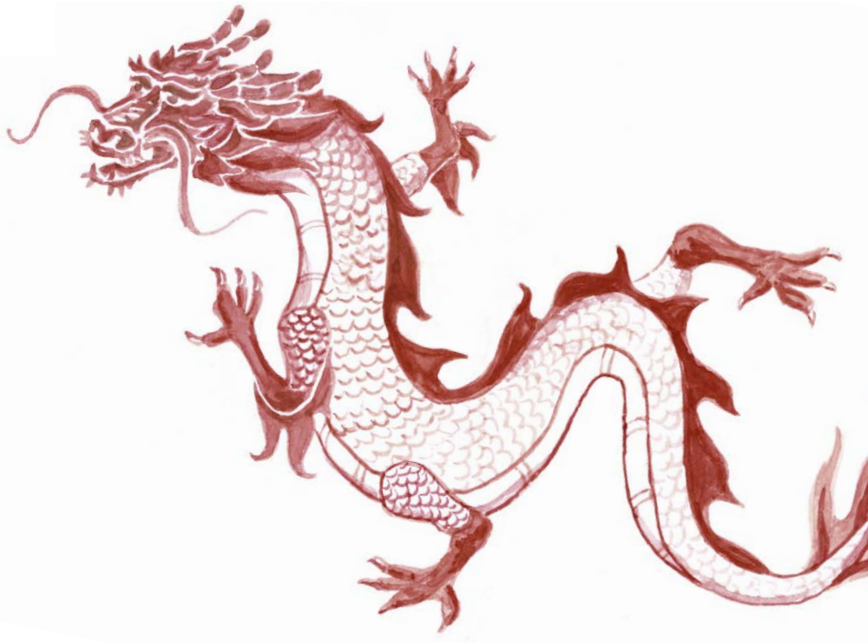
Extras

Sonstiges

Programmiersprachen

Im schier unüberschaubaren Programmiersprachenwald tritt Rust an, einiges besser zu machen als etablierte Sprachen, beispielsweise durch sicherere Speicherverwaltung und Codeprüfung zur Kompilierzeit. Mit den Prinzipien des Clean Code lässt sich aber auch C++20 optimieren und auf das Wesentliche reduzieren, um den Code lesbarer zu gestalten und Fehler zu vermeiden.

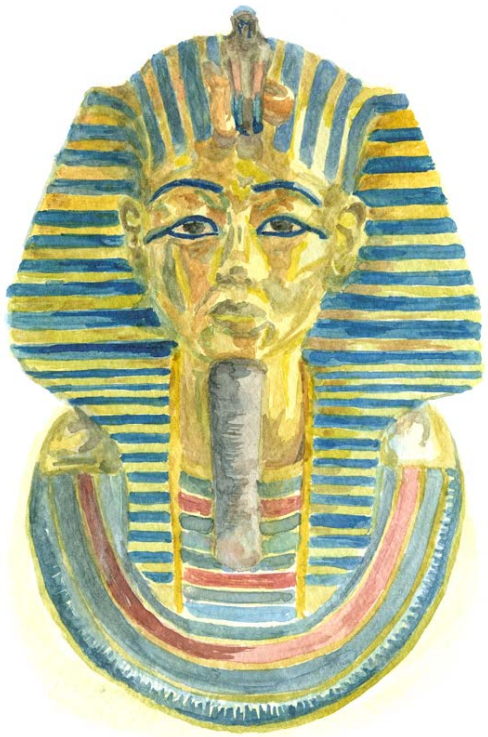
ab Seite 85



GitOps und Containerisierung

Als neues Paradigma will GitOps die Ära des automatisierten IT-Betriebs einläuten. Von der Infrastructure as Code bis zum vollständigen Lebenszyklus einer Anwendung soll mehr Automatisierung helfen, typische Fehler zu vermeiden, und so zu höherer Qualität beitragen. Das macht ein Kubernetes-Praxisbeispiel aus der Versicherungsbranche ebenso deutlich wie ein Tutorial zur Optimierung containerisierter Java-Anwendungen.

ab Seite 105



Extras

Mit Value Design lassen sich moderne Softwareentwicklung und Ethik in Einklang bringen. In Cloud-nativen Architekturen helfen die aus dem Reactive Manifesto abgeleiteten Prinzipien, die hohen Anforderungen an Elastizität und Resilienz zu erfüllen. Für das Arbeiten mit RESTful APIs existieren viele Ansätze. Wie es mit der Hypertext Application Language geht, zeigt ein Erfahrungsbericht.

ab Seite 129