Engineer

Software

Programme und Daten, mit denen ein Computer arbeitet

Was es ist

Software umfasst alle Teile eines Computersystems, die sich durch Schreiben statt durch Löten verändern lassen: die Programme, die darin enthaltenen Daten und die Einstellungen, die ihr Verhalten bestimmen. Im Englischen ist das Wort ein nicht zählbares Substantiv. Ein einzelnes Exemplar heißt deshalb „a piece of software“ („ein Stück Software“) oder „a program“ („ein Programm“). Was ein Mensch öffnet, um eine Aufgabe zu erledigen, heißt meist Anwendung. „Code“ meint üblicherweise den Quelltext, den Entwickler bearbeiten. Spezifikationen kürzen das Ganze manchmal mit SW ab.

Der übliche Gegenbegriff ist Hardware, die physische Maschine, mit Firmware im Zwischenraum. Das NIST-Glossar zieht die Grenze anhand der Veränderbarkeit: Software bezeichnet „Computerprogramme und zugehörige Daten, die während der Ausführung dynamisch geschrieben oder verändert werden können“, während Firmware während ihrer Ausführung nicht verändert werden kann. Das Betriebssystem eines Laptops ist Software, der Code in dessen Tastaturcontroller Firmware und die Tastatur Hardware.

Innerhalb der Software gibt es Schichten. Systemsoftware, etwa ein Betriebssystem oder ein Gerätetreiber, verwaltet die Maschine. Anwendungssoftware erledigt etwas für einen Menschen: eine Tabellenkalkulation, eine Buchungswebsite oder ein Nachrichtenclient. Dazwischen liegen Bibliotheken und Pakete, fertige Bausteine, aus denen beide Arten zusammengesetzt werden. In der Praxis stammt der größte Teil des Codes eines modernen Produkts aus solchen Bausteinen und wurde nicht eigens dafür geschrieben.

Eine weitere Unterscheidung entscheidet später über vieles. Quellcode ist das, was Menschen schreiben und lesen. Ein Compiler oder Interpreter macht daraus Anweisungen, die ein Prozessor ausführt. Sowohl Quellcode als auch kompilierter Code sind Software, beim Kauf aber nicht austauschbar. Werden nur kompilierte Dateien geliefert, kann ausschließlich der Lieferant sie ändern.

Das Konzept für eine Entscheidung nutzen

Nutzen Sie „Was ist Software?“ für eine konkrete Entscheidung. Wir können helfen, das Konzept mit dem System in Verbindung zu bringen, das Sie entwickeln oder einkaufen. Die Struktur eines Systems wird zuerst auf dem Papier durchdacht, damit die Teile zusammenpassen und die Kosten feststehen, bevor die erste Zeile Code entsteht.

Unsere Leistungen entdecken Kontakt aufnehmen →

Die Annahmen hinter dem Begriff prüfen

Nutzen Sie die Erklärung und die verlinkten Arbeiten, um Fragen für Ihr Projekt zu formulieren. Wir können die Annahmen gemeinsam mit Ihnen prüfen, bevor ein Werkzeug ausgewählt wird. Ein Infrastrukturplan für 14 Abteilungen — 14 Forschungsabteilungen mit einem gemeinsamen Infrastrukturplan

Ein Infrastrukturplan für 14 Abteilungen Kontakt aufnehmen →

Warum es wichtig ist

Wer Software in Auftrag gibt, kauft oder mietet, entscheidet damit, wer für ihre Pflege verantwortlich ist. Was Software attraktiv macht, nämlich dass Kopieren nichts kostet und Änderungen nach der Lieferung gegenüber dem Umbau einer Maschine günstig sind, erklärt zugleich, warum sie nie fertig ist. Die verwendeten Komponenten veröffentlichen Korrekturen. Betriebssysteme und Browser stellen Schnittstellen ein. In einer Bibliothek wird eine Schwachstelle bekannt, und niemand im Team erinnert sich daran, sie ausgewählt zu haben. Daraus entsteht jeweils Arbeit für diejenigen, die für den Code verantwortlich sind.

Die üblichen Wege verteilen diese Arbeit unterschiedlich:

- **Als Dienst gemietet.** Der Anbieter betreibt die Server und führt die Aktualisierungen durch. Sie übernehmen seinen Veröffentlichungsrhythmus, seine Preise und den Aufwand eines Wechsels. Ihre Daten liegen bei ihm.
- **Lizenziert und selbst betrieben.** Sie entscheiden, wann aktualisiert wird, und übernehmen auch die Aktualisierungen, Sicherungen und Sicherheitspatches.
- **Für Sie entwickelt.** Die Software passt zum Ablauf statt umgekehrt. Für das Wartungsbudget sind Sie zuständig, solange die Software läuft. Die Kosten, an denen sich zeigt, ob das sinnvoll war, entstehen nach dem Start.

- **Open Source.** Eine Lizenz nach der Open Source Definition erlaubt Nutzung, Untersuchung, Änderung und Weitergabe. Trotzdem muss jemand die Aktualisierungen einspielen. Kostenlos zu beziehen und kostenlos zu betreiben sind unterschiedliche Eigenschaften.

Unabhängig vom gewählten Weg lohnen sich vor der Unterschrift zwei Fragen zum Ergebnis. Was erhalten Sie am Ende: den Quellcode, die Bauanleitung und eine Liste der Bestandteile? Und wie hält der Lieferant die Software sicher? NISTs Secure Software Development Framework dient zum Teil diesem zweiten Gespräch: Es gibt Käufern und Lieferanten ein gemeinsames Vokabular für Entwicklungspraktiken. Zur ersten Frage definiert ein NTIA-Bericht eine Softwarestückliste als „formale Aufzeichnung mit Einzelheiten und Lieferkettenbeziehungen der verschiedenen bei der Erstellung von Software verwendeten Komponenten“. Eine solche Liste anzufordern kostet wenig und zeigt, wie viel von dem, wofür Sie bezahlen, jemand anderes geschrieben hat.

Wie es in der Praxis aussieht

Zu wissen, was ausgeliefert wurde, ist der Teil, den Teams auslassen. Auf einer Plattform wurde die Sicherheitsarbeit im Build verankert, statt auf Einschätzungen zu beruhen: In einem kontinuierlichen Sicherheitsprogramm entsteht bei jedem Build mit Anchore und Syft eine Softwarestückliste. Dependabot überwacht 6 Ökosysteme, CodeQL und ZAP untersuchen den Code und die laufende Instanz. Selbst die externen Build-Schritte sind festgeschrieben: Jede Workflow-Action eines Dritten wird über ihren Commit-Hash referenziert. Den kann niemand wie ein Tag verschieben. Neben der Automatisierung stehen 21 schriftliche Sicherheitsaudits, darunter ein Bedrohungsmodell. Eine von den konfigurierten Prüfungen erkannte Schwachstelle kann den Build scheitern lassen. Der Preis ist die Menge der Befunde, die eine nie endende Bewertung und Priorisierung erfordert.

Software von einer Änderung in die Produktion zu bringen, ist die andere Hälfte. Wo Veröffentlichungen davon abhingen, dass jemand eine Folge manueller Schritte im Gedächtnis behielt und jedes Mal etwas anders ausführte, gewann die Automatisierung von Build, Tests und Deployment geschätzte 4.000 Stunden zurück und machte Veröffentlichungen schneller und verlässlicher. Der Gewinn entstand durch den jedes Mal gleichen Weg, nicht durch mehr Sorgfalt einzelner Menschen.

Die Grenze zwischen Software und Hardware ist auch auf der Reparaturwerkbank praktisch relevant. Laptop-Reparaturen umfassten Bildschirme, Scharniere, Tastaturen und Hauptplatinen sowie Softwareprobleme unter Linux, Windows und Mac. Bei einem Laptop, der sich nicht einschalten lässt, können Netzteil, Hauptplatine oder Akku die Ursache sein. Deshalb begann die Arbeit damit, den tatsächlichen Fehler zu finden, bevor etwas angefasst wurde.

Wo es schiefgeht

Aus einer Sicherheitsmeldung wird eine Woche Archäologie

In einer weit verbreiteten Komponente wird eine Schwachstelle bekannt, und niemand kann sagen, welche Systeme der Organisation sie enthalten. Das Inventar existierte nur in den Köpfen der Menschen und in Lockdateien, die niemand liest. Nun muss es unter Druck System für System rekonstruiert werden. Erzeugen Sie bei jedem Build eine Stückliste und bewahren Sie sie mit der Veröffentlichung auf, die sie beschreibt. Die Frage „Sind wir betroffen?“ wird dann zur Suche.

Veröffentlichungen hängen von der ausführenden Person ab

Das Deployment funktioniert bei einer Person und scheitert bei einer anderen. Als Unterschied erweist sich ein Schritt, der nur in jemandes Gedächtnis existiert. Jede Veröffentlichung ist eine kleine Improvisation. Überführen Sie den ganzen Weg in einen automatisierten Job, der jedes Mal gleich baut, testet und ausliefert. Betrachten Sie jeden verbleibenden manuellen Schritt als einen zu beseitigenden Defekt.

Das Budget endet mit dem Start

Nach zwei oder drei Jahren wirkt jede Aktualisierung riskant. Das Framework liegt mehrere Hauptversionen zurück, und der einzige Vorschlag ist eine Neuentwicklung. Die Software wurde wie ein Gebäude bezahlt und dann sich selbst überlassen, während sich ihre ganze Umgebung weiterveränderte. Planen Sie ein laufendes Wartungsbudget ein, aktualisieren Sie Abhängigkeiten regelmäßig in kleinen Schritten, solange jeder Schritt noch klein ist, und rechnen Sie damit, dass sich die darunterliegenden Plattformen nach eigenem Zeitplan ändern.

Der Name bleibt gleich, während sich der Code dahinter ändert

Derselbe Build liefert an zwei Tagen verschiedene Ergebnisse, ohne dass dazwischen ein Commit liegt. Irgendwo verweist eine Referenz auf ein bewegliches Ziel: einen Versionsbereich, ein `latest` -Tag oder eine CI-Action, deren Tag der Eigentümer neu setzen kann. Schreiben Sie fremden Code auf eine exakte Version oder, wenn sich Tags verschieben lassen, einen Commit-Hash fest. Checken Sie die Lockdatei ein und lassen Sie einen Bot Aktualisierungen vorschlagen. So kommt jede Änderung als prüfbarer Diff an.

Weiterführende Lektüre

- NIST-CSRC-Glossar: Software — Die Definitionen der NIST-Publikationen, darunter die Abgrenzung von Software und Firmware.
- NIST SP 800-218, Secure Software Development Framework — Ein Kernbestand an Praktiken für Software mit weniger Schwachstellen, so formuliert, dass Käufer ihn im Gespräch mit Lieferanten nutzen können.
- NTIA: Die Mindestelemente einer Softwarestückliste — Was eine Liste der Bestandteile einer Software festhalten muss, um nützlich zu sein.
- Die Open Source Definition — Die zehn Kriterien, die eine Lizenz erfüllen muss, damit darunter veröffentlichte Software als Open Source gilt.

Glossar

Für Ihre tatsächlichen Rahmenbedingungen auswählen

Beschreiben Sie den Arbeitsablauf, die vorhandenen Schnittstellen und die Nutzer, die die Veränderung benötigen. Wir können eine prüfbare Lieferung definieren und die Übergabe so gestalten, dass Ihr Team die gelieferten Bestandteile warten kann.

Preise ansehen Kontakt aufnehmen →

Die Begriffe praktisch nutzbar machen

Fragen Sie, wie „Was ist Software?“ auf Ihre Organisation zutrifft. Ein konkretes Beispiel aus Ihrem bestehenden System reicht aus, um das Gespräch zu beginnen.

Unsere Leistungen entdecken

welcome@engineer.company +45 50 10 61 84 Kopenhagen, Dänemark

Nehmen Sie diese Seite mit

Kontaktkarte

Mit Sorgfalt und Liebe gestaltet von D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – heute · Engineer ApS · Alle Rechte vorbehalten.

Gedruckt von https://engineer.company/de/glossary/software/