

Low-Code/No-Code im Kurzportrait – Chancen, Herausforderungen und Anwendungsbeispiele

Eine Orientierungshilfe unter Berücksichtigung aktueller KI-Entwicklungen



Vorwort



In der Digitalisierung und Vernetzung von Maschinen liegt gewaltiges Potential. Nach einem Jahrzehnt, in dem von Industrie 4.0 gesprochen wurde, können wir aktuell den Mehrwert dieser Entwicklungen tatsächlich auch in der Breite erleben. Wir spüren, wie das Bewusstsein sowie auch die Offenheit für den Umgang mit neuen Technologien in den Unternehmen und bei ihren Mitarbeiterinnen und Mitarbeitern stetig wächst.

Software ist inzwischen auch aus dem Maschinen- und Anlagenbau nicht mehr weg zu denken. Sie spielt nicht nur im Rahmen der Steuerung, sondern auch für übergreifende Prozesse und ergänzende Produkte und Dienstleistungen eine zentrale Rolle. Deshalb setzen wir uns im Vorstand des Fachverbands Software und Digitalisierung des VDMA mit den daraus entstehenden zentralen Herausforderungen für den Maschinen- und Anlagenbau auseinander.

Gerade neue Technologien und deren Entwicklungen, wie Low-Code (LC) oder Künstliche Intelligenz (KI), werfen Fragen in ihrer Verwendung und für die Anwendungsgebiete im Maschinen- und Anlagenbau auf. Dieses Whitepaper setzt deshalb einen Fokus auf das Thema Low-Code-Entwicklung.

Die Low-Code-Entwicklung wird zukünftig in der produzierenden Industrie eine Schlüsselrolle dabei spielen, die digitale Transformation voranzutreiben, die Effizienz zu steigern und die Innovationsfähigkeit zu erhöhen. Unternehmen, die diese Technologie erfolgreich nutzen, werden in der Lage sein, sich schnell an veränderte Marktanforderungen anzupassen und Wettbewerbsvorteile zu erzielen. Darüber hinaus kann auch Künstliche Intelligenz in Verbindung mit Low-Code auf verschiedene Weisen eingesetzt werden, um die Entwicklung von Anwendungen zu automatisieren, zu beschleunigen und zu optimieren. Unterkategorien von KI, wie „Generative KI“, spielen hierbei eine wichtige Rolle.

Auch persönlich begeistern mich diese Technologien und die Möglichkeiten, die sich daraus für den Einsatz in der Softwareentwicklung bzw. besonders auch für die Entwicklung von neuen Produkten und Lösungen ergeben. Deshalb war es mir eine Freude, mich bei der Entstehung dieses Whitepapers zu engagieren und wünsche, dass das Lesen zu neuen Umsetzungsideen inspiriert. Dadurch wird für Unternehmen im Maschinen- und Anlagenbau sowie auch darüber hinaus Vernetzung gelebt und das Ziel dieses, zur Erarbeitung des Whitepapers gegründeten, Arbeitskreises erfüllt. Gemeinschaftlich können wir so einen Schritt auf dem Weg hin zum Zukunftsbild einer flexiblen und intelligenten Fabrik gehen.

Wolfram Schäfer,
Geschäftsführer und Gründer, iT Engineering Software Innovations GmbH
und Mitglied des Vorstands VDMA Software und Digitalisierung

Vorwort	1
Inhaltsverzeichnis	2
1. Einführung: Innovative Softwareentwicklung im Maschinen- und Anlagenbau	4
2. Aktueller Status und Ausblick der Anwendung von Low-Code und No-Code in der Wirtschaft	6
3. Low-Code und No-Code - Theoretische Aspekte	8
3.1. Begriffsdefinition	8
3.2. Abgrenzung Tools	9
3.3. Chancen und Grenzen von Low-Code / No-Code (LC/NC)	11
3.3.1. Chancen	11
3.3.2. Grenzen	14
3.4. Ebenen und Kategorien von Low-Code und No-Code	16
3.4.1. Ebenen	16
3.4.2. Kategorien von Low-Code- und No-Code-Systemen	17
3.4.2.1. Dimension 1 – Steigende Abstraktion auf der Lösungsebene	17
3.4.2.2. Dimension 2 – Verlagerung auf die Anforderungsebene	18
3.5. Organisation und Struktur	18
3.5.1. Organisation und Struktur im Speziellen	19
3.5.1.1. Zusammenarbeit im Team	17
3.5.1.2. Agile Organisationen	19
3.5.2. Rollen und Personen	20
3.6. Compliance und Governance	22
3.7. Aktuelle KI-Entwicklungen	24
4. Anwendungsbeispiele aus dem Maschinen- und Anlagenbau und bei Softwarefirmen	27
4.1. iT Engineering Software Innovations GmbH (intern) – Einsatzmöglichkeiten von LLMs in der eigenen Softwareentwicklung sowie in Kundenprojekten	27
4.1.1. Ausgangssituation / Zielsetzung	27
4.1.2. Umsetzung	27
4.1.3. Ergebnis	28
4.1.4. Stakeholder / Rollen im Prozess	29
4.2. CAS Software AG + ASYS Group – Effiziente Nutzung von No-Code / Low-Code im Maschinenbau	29
4.2.1. Ausgangssituation / Zielsetzung	29
4.2.2. Umsetzung	30
4.2.3. Ergebnis	31
4.2.4. Stakeholder / Rollen im Prozess	31
4.3. SAERTEX GmbH & Co. KG + Simplifier AG – Kompatibilität und Mobilfähigkeit durch Low-Code	32
4.3.1. Ausgangssituation / Zielsetzung	32
4.3.2. Umsetzung	32
4.3.3. Ergebnis	33
4.3.4. Stakeholder / Rollen im Prozess	33
4.4. The MathWorks GmbH – Low-Fault Detection Workflow	34
4.4.1. Ausgangssituation / Zielsetzung	34
4.4.2. Umsetzung	34
4.4.3. Ergebnis	35
4.4.4. Stakeholder / Rollen im Prozess	35
4.5. GAL Digital GmbH (intern) – Einsatz von generativer KI im eigenen Unternehmen	36
4.5.1. Ausgangssituation / Zielsetzung	36
4.5.2. Umsetzung	36
4.5.3. Ergebnis	36
4.5.4. Stakeholder / Rollen im Prozess	37
4.6. valantic GmbH + YAK MAT - Bewegliches Inventar immer im Blick: YAK MAT vereinfacht ERP-Prozesse mit Outsystems und valantic	38
4.6.1. Ausgangssituation / Zielsetzung	38
4.6.2. Umsetzung	38
4.6.3. Ergebnis	38
4.6.4. Stakeholder / Rollen im Prozess	38
4.7. Optimierung eines Montage-Prozesses mit SAP Build Apps	39
4.7.1. Ausgangssituation / Zielsetzung	39
4.7.2. Umsetzung	39
4.7.3. Ergebnis	40
4.7.4. Stakeholder / Rollen im Prozess	40
5. Fazit / Ausblick	40
6. Anhang A: Übersicht / „Entscheidungshilfe“ zu LC- und NC-Plattformen	43
7. Übersicht zu den Mitwirkenden	44
8. Literaturverzeichnis	45
Impressum	48

Abkürzungsverzeichnis

4GL	vierte Generationssprache
CRM	Customer Relationship Management
ERP	Enterprise Resource Planning
GPT	Generative Pre-Trained
HR	Human Resources
IoT	Internet of Things
IP	Intellectual Property
KI	Künstliche Intelligenz
LC	Low-Code
LC/NC	Low-Code und No-Code
LLMs	Large Language Models
NC	No-Code
O365	Microsoft 365 und teilweise Office 365
OEE	Overall Equipment Effectiveness

1. Einführung: Innovative Softwareentwicklung im Maschinen- und Anlagenbau

Autoren: Wolfram Schäfer, Judith Armbruster, Hien Nguyen, iT Engineering Software Innovations GmbH

Die heutigen Anforderungen an Unternehmen im Maschinenbau sind vielschichtig. Heterogene IT-Landschaften, die sich aus Produktspektrum, Fertigungsvielfalt, Betriebsgröße und Unternehmenshistorie ergeben, erfordern flexible Ansätze in der Softwareentwicklung. Die drängende Notwendigkeit, sich durch innovative Lösungen am Markt zu differenzieren und interne Prozesse zu optimieren, erfordert kontinuierliche Bemühungen im Bereich der Digitalisierung. Doch der Mangel an spezialisierten Fachkräften, die Tendenz zu Insellösungen und die oft langen "time-to-market"-Zeiträume stellen ernsthafte Hürden dar.

Nichtsdestotrotz ist der Maschinen- und Anlagenbau für Entwicklungen im Rahmen der Digitalisierung eine treibende Kraft in der produzierenden Industrie. Denn die Unternehmen liefern nicht nur immer neue Produkte, die durch die Digitalisierung möglich werden, sondern sie entwickeln auch selbst digitale Lösungen und sind Anwender innovativer Technologien. Daher sind Entwicklungen neuer Technologien, wie z. B. Low-Code, von besonderem Interesse, da sie die Beschleunigung und Demokratisierung der Softwareentwicklung ermöglichen.

Grundsätzlich kann die Softwarelandschaft im Maschinen- und Anlagenbau in drei grundlegende Bereiche unterteilt werden:

(1) Unternehmensanwendungen:

Hierunter fallen u.a. Enterprise Resource Planning (ERP), Customer Relationship Management (CRM) oder Human Resources (HR)-Systeme. Diese bilden das Rückgrat der internen Betriebsabläufe und ermöglichen die strukturierte Koordination sämtlicher unternehmerischer Prozesse.

(2) Produktionsnahe IT-Lösungen und Services:

Im Fokus stehen hier Anwendungen wie Manufacturing Execution Systems (MES), Internet of Things (IoT)-Anwendungen oder z. B. Overall Equipment Effectiveness (OEE)-Dashboards. Sie spielen eine Schlüsselrolle in der Optimierung von Produktion und Qualität und sorgen für nahtlose, transparente Prozessabläufe.

(3) Entwicklung eigener Softwareprodukte und Services als integraler Bestandteil der Maschinen oder als ergänzende Softwareprodukte für Betrieb, Wartung und Service der Maschinen sowie zur Differenzierung gegenüber dem Wettbewerb:

Um sich im globalen Wettbewerb zu behaupten, müssen Unternehmen im Maschinenbau ihre Produkte und Services kontinuierlich weiterentwickeln. Von der Steuerungstechnik über Cloud-Services und KI-Anwendungen erfordern diese Bereiche maßgeschneiderte Softwarelösungen.

Die aktuellen Entwicklungen zeigen, dass innerhalb all dieser drei Anwendungsbereiche Low-Code ein vielversprechender Ansatz ist, um dem steigenden Bedarf an tiefgehenden Programmierkenntnissen zur Realisierung von Softwareanwendungen zu begegnen. In Unternehmensanwendungen können diese Ansätze die Gestaltung und Anpassung von Workflows vereinfachen, während sie in der Produktionssteuerung die Entwicklung maßgeschneiderter Lösungen erleichtern. Zudem ermöglichen sie in der Entwicklung von differenzierenden Services eine schnelle Umsetzung von Ideen. Gleichzeitig ist durch die Entwicklung von Low-Code eine genauere Differenzierung von Problemstellungen notwendig – denn nicht für alle Herausforderungen kann damit eine passende Lösung erstellt werden. Kompetenzen und Technologien müssen genau mit gestellten Problem-

stellungen abgeglichen werden, dann kann der Einsatz von Low-Code aktuelle Entwicklungsansätze ergänzen und robuster werden lassen. Darüber hinaus ermöglicht auch der Einsatz von KI, besonders das Konzept der Large Language Models (LLMs), die traditionellen Ansätze der Softwareentwicklung zu bereichern und zu beschleunigen. LLMs dienen z. B. durch automatisierte Codegenerierung oder Datenanalyse dazu, innovative Lösungen zu entwickeln und den Entwicklungsprozess zu verkürzen.

Dieses Whitepaper leistet einen Beitrag, um einerseits Aufklärung zu betreiben, was hinter Low-Code als Technologie steckt und aufzuzeigen, welche Möglichkeiten es für die konkrete Anwendung im Kontext des Maschinen- und Anlagenbaus gibt. Das folgende Kapitel gibt einen Einblick über den aktuellen Status Quo im Maschinen- und Anlagenbau (Kapitel 2). Darauf aufbauend wird ausgehend von einer Definition und Beschreibung der Entwicklung von Low-Code (Kapitel 3.1) auch eine Einordnung gemacht, welche Rahmenbedingungen für den erfolgreichen Einsatz notwendig sind (Kapitel 3.2) und welche Chancen beziehungsweise Grenzen sich für Unternehmen daraus ergeben (Kapitel 3.3). Low-Code-Tools spielen in Bezug auf Architektur und Technik auf verschiedenen Ebenen eine Rolle – dies wird in einem separaten Kapitel diskutiert (Kap. 3.4). Nicht nur der Technologie-Stack, sondern auch bestehende Strukturen in den Unternehmen sind damit einer Veränderung ausgesetzt. Die Verantwortungsbereiche einzelner Personen verändern sich genauso wie auch die benötigten Kompetenzen im Team. Diese beeinflussen die Zusammenarbeit und Kultur im Miteinander maßgeblich (Kapitel 3.5). Im darauffolgenden Kapitel werden die Themen Compliance und Governance betrachtet, die bei der Nutzung von Low-Code-Plattformen ebenso

betrachtet werden müssen (Kapitel 3.6). Auch aktuelle und zukünftige Entwicklungen, die sich gerade aus dem Zusammenspiel mit KI ergeben, werden abschließend herausgestellt (Kapitel 3.7).

Das Kapitel 4 stellt mehrere Anwendungsbeispiele vor, die einen konkreten Einblick geben, wie der Einsatz von Low-Code/No-Code, auch in Verbindung mit generativer KI, gestaltet werden kann und welcher Mehrwert sich daraus ergibt. Damit geben die Unternehmen, die an der Erstellung dieses Whitepapers mitgewirkt haben, praktische Anregungen, um daraus Inspiration und Motivation für die Auseinandersetzung mit Low-Code zu ziehen.

Das Whitepaper „*Low-Code/No-Code im Kurzportrait – Chancen, Herausforderungen und Anwendungsbeispiele*“ wurde in verschiedene Kategorien unterteilt. Nutzen Sie daher die Navigation aus dem Inhaltsverzeichnis, um schnell zum zugehörigen Abschnitt zu springen.

2. Aktueller Status und Ausblick der Anwendung von Low-Code und No-Code in der Wirtschaft

Autoren: Carsten Rückriegel und Florian Klein, VDMA Software und Digitalisierung

Die fortschreitende Digitalisierung stellt Unternehmen aus der Industrie vor zahlreiche Herausforderungen. Gleichzeitig eröffnet sie jedoch auch die Möglichkeit, ihre Wettbewerbsfähigkeit weiter auszubauen. Um von der Digitalisierung profitieren zu können, müssen diese Unternehmen Änderungen herbeiführen, Innovationen fördern und neue Technologien einführen. Dadurch wird es ihnen ermöglicht, neue digitale Produkte und Dienstleistungen zu entwickeln und neue Geschäftsmodelle zu etablieren. Ein entscheidender Erfolgsfaktor hierbei ist die Entwicklung und Implementierung moderner und anpassungsfähiger Softwareprodukte als Grundlage für digitale Prozesse. Die Anforderungen von Seiten der Kunden nach schnellen Lösungen oder auch andere disruptive Faktoren, wie die Corona-Pandemie, zeigen auf, dass Low-Code und No-Code ein interessantes Werkzeug sind, um aktuelle und künftige Herausforderungen zu meistern (Gartner 2021).

Die Beschleunigung der Digitalisierung durch die Pandemie hat zur Folge, dass die Softwareunternehmen die Nachfrage nach Lösungen nicht mehr sofort bedienen können, so dass ein Großteil (80 %) dem Zeitplan hinterläuft (Mendix 2023). Die Umfrage von techconsult (2022) unter deutschen Unternehmen kam zu einem etwas positiveren Ergebnis, dennoch sind nur etwas mehr als die Hälfte (56 %) der Befragten mit ihren Release-Zyklen zufrieden. Die anderen Unternehmen können die angefragten digitalen Lösungen nicht in adäquater Zeit bereitstellen. Dies ist ein sich fortsetzender Trend, wie auch schon eine frühere Untersuchung von Forrester (2017, siehe auch Herzog 2018) zeigt, die die größten Herausforderungen bei der Erstellung individueller Unternehmensanwendungen mit-

tels traditioneller Entwicklungsmethoden analysierte. Als größte Herausforderungen nannten die Befragten „Es ist schwierig, Anforderungen zeitnah zu erfüllen“, gefolgt von „Mangelnde Flexibilität“ und „Es dauert zu lange, Anwendungen zu aktualisieren“. Der zunehmende Mangel an Programmiererinnen und Programmieren kann diesen Entwicklungen nicht entgegenwirken, denn in Deutschland fehlten im Jahr 2022 bereits ca. 137.000 IT-Fachkräfte (Bitkom 2022).

Immer mehr Unternehmen setzen daher Low-Code- und No-Code-Technologien ein. Sie tun dies, um die Effizienz zu steigern, um dem Fachkräftemangel zu begegnen (Forrester 2022) oder um ganz einfach „am Ball“ zu bleiben und den Anschluss an neue Entwicklungen nicht zu verpassen. Gefragt nach den positiven Auswirkungen von Low-Code-Plattformen, gaben nahezu alle Befragten an, dass es entweder zu einer „Signifikanten Verbesserung“ (55 %) oder zu „Merklichen Verbesserung“ (43 %) kam.

Auch der Ausblick ist zuversichtlich, denn laut einer Gartner-Umfrage aus dem Jahr 2021 gaben die befragten Unternehmen an, dass im Jahr 2025 rund 70 % der zukünftig entwickelten Anwendungen auf Low-Code oder No-Code-Technologien beruhen werden. KI-unterstütztes Softwareengineering ist zunehmend massentauglich geworden. So sieht es auch die Mehrheit der dazu befragten IT-Experten (69 %): innerhalb ihrer Organisation werden LC-/NC-Lösungen bereits für die Softwareentwicklung eingesetzt (Mendix 2021). Etwas mehr als die Hälfte gab sogar an, dass Low-Code auch für die Generierung komplexerer und unternehmenskritischer Programme eingesetzt wird (Mendix 2021).

Zu einem ähnlichen Ergebnis kommt die Untersuchung deutscher Unternehmen der techconsult GmbH – eine große Mehrheit der gut 250 Befragten (78 %) beschäftigt sich bereits mit Low-Code/No-Code-Lösungen: LC/NC ist entweder bereits vorab in die Softwareentwicklungsprozesse integriert oder die Integration wird zumindest innerhalb der nächsten fünf Jahre anvisiert. Nur ein geringer Anteil sieht aktuell und in naher Zukunft keine Bedeutung für Low-Code/No-Code-Lösungen für die eigenen Produkte.

Im Zuge dieses Whitepapers wird auch auf die Möglichkeiten eingegangen, die Künstliche Intelligenz (KI) für die Programmierung bietet. Die Lücke zwischen benötigten und verfügbaren Programmierern ist nicht kleiner geworden – damit verbunden ist ein Rückstau an nicht bearbeiteten Projekten (TechTarget 2023). Um dieses Dilemma zu beheben, kann die Kombination von KI und Low-Code- und No-Code-Ansätzen nützlich sein. Nach Einschätzung von Gartner soll bis 2027 rund die Hälfte der Softwareengineers auf KI-gestützte Werkzeuge zurückgreifen, um Programme zu schreiben (Vogel Communications 2023).

Die wichtigsten Einsatzbereiche sind laut einer weltweiten Umfrage von Mendix (2021), insbesondere aus der Sicht der Befragten aus Deutschland: Die „Anwendungen für komplexe, individuelle Unternehmenssoftware (37 %), industrielle IoT-Apps (35 %), für automatisierte, existierende Arbeitsprozesse (35 %), für Data Modeling und Visualisierung (34 %) sowie für automatisierte Anwendungen der Robotic Process Automation (31 %)“.

Alles in allem ist ein großes Potential zu erkennen, das nicht nur innerhalb der Unternehmen, sondern auch gesamtheitlich für die Wirtschaft und Forschung einen Mehrwert stiftet. Der weltweite Markt für die Anwendung von Low-Code-Technologien beläuft sich nach einer Einschätzung von Gartner auf rund 26,9 Milliarden US-Dollar (Gartner 2022a). Zudem zeigt sich auch in den vorgelagerten Prozessen der angewandten Forschung und Entwicklung ein stark steigendes Interesse am Thema seit 2018, wie sich am Anstieg der wissenschaftlichen Artikel und Patentdokumente erkennen lässt (Gottschick, Tiemann 2020).

3. Low-Code und No-Code - theoretische Aspekte

Im nachfolgenden Kapitel werden einzelne Begrifflichkeiten einheitlich definiert und beschrieben. In der Literatur gibt es für diese Begriffe und deren Verständnis viele verschiedene Definitionen. Es ist daher von enormer Wichtigkeit, eine einheitliche Begriffsdefinition für dieses Whitepaper festzulegen, um verstehen zu können, was unter der Thematik „Low-Code/ No-Code“ gemeint ist.

Darüber hinaus gibt es Einblicke in den Ursprung von Low- und No-Code-Anwendungen bzw. Plattformen, eine Abgrenzung der genannten Tools, eine Gegenüberstellung von Chancen und Grenzen sowie einen Überblick in die verschiedenen Ebenen und Kategorien.

3.1. Begriffsdefinition

Autoren: Wolfram Schäfer und Hien Nguyen,
iT Engineering Software Innovations GmbH

Bevor in den folgenden Kapiteln tiefer in die Rahmenbedingungen und Anwendungsmöglichkeiten von Low-Code (LC) und No-Code (NC) im Zusammenhang mit KI eingetaucht wird, werden anhand dieses Kapitels zunächst die zentralen Begrifflichkeiten dieses Whitepapers definiert, um ein gemeinsames Verständnis zu schaffen.

Low-Code sowie auch No-Code sind Entwicklungsphilosophien, die sich von herkömmlichen Entwicklungstools dadurch unterscheiden, dass wenig beziehungsweise keine tiefen Programmierkenntnisse erforderlich sind, um ausführbare Anwendungen zu erstellen. Im nachfolgenden Infokasten sind die Eigenschaften der beiden Philosophien in aller Kürze dargestellt.

Low-Code

Bei Low-Code-Anwendungen kann durch die Verwendung visueller Werkzeuge und vorgefertigter Bausteine eine Anwendung einfach „zusammengeklickt“ werden. Durch grafische Oberflächen und einfache Interaktionen (z. B. Drag-and-Drop) entstehen so Anwendungen, ohne dass diese von Grund auf neu programmiert werden müssen. Ergänzungen oder Veränderungen, die selbst programmiert werden, sind an den Bausteinen möglich.

No-Code

No-Code geht noch einen Schritt weiter und eliminiert die Notwendigkeit von Programmierung nahezu vollständig. Bei diesem Ansatz werden Anwendungen mithilfe von intuitiven visuellen Werkzeugen und Eingaben zur Konfiguration erstellt, ohne auch nur eine Zeile Code schreiben zu müssen. No-Code ist besonders benutzerfreundlich und ermöglicht es Menschen, Softwareanwendungen zu erstellen, ohne Programmierkenntnisse zu besitzen.

Ursprung von Low- und No-Code-Anwendungen und -Plattformen

Low-Code- und No-Code-Anwendungen wurden entwickelt, um die Softwareentwicklung zu beschleunigen und zu demokratisieren, sodass dem stetig wachsenden Bedarf nach Softwareanwendungen entsprochen werden kann. Zudem haben sie ihre Wurzeln in verschiedenen Ideen und Konzepten zur Strukturierung von Quellcode. Vorläufer waren 4GL, die sogenannte vierte Generationssprache sowie Rapid Application Development (RAD), die durch eine höhere Abstraktionsebene ermöglichten, vorgefertigte Funktionen, wiederverwendbare Bausteine und visuelle Entwicklungswerkzeuge zu verwenden. Nicht zuletzt wurde es durch Cloud Computing und Software as a Service (SaaS) bzw. SaaS-Plattformen

einfacher, auf Low-Code-Anwendungen kollaborativ zuzugreifen sowie deren Skalierbarkeit zu fördern. Dies förderte deren Popularität zunehmend.

Auch in den kommenden Jahren wird die Low-Code-Technologie voraussichtlich einige bedeutende Veränderungen und Fortschritte erfahren. Denn besonders KI kann in Verbindung mit Low-Code auf verschiedene Weisen eingesetzt werden, um die Entwicklung von Anwendungen zu automatisieren, zu beschleunigen und zu optimieren. Besonders die Unterkategorie der „Generativen KI“, spielt hierbei eine wichtige Rolle. „Generative KI“ bezieht sich dabei auf Modelle, die in der Lage sind, neue Inhalte zu generieren, sei es Text, Bilder oder sogar (Quell-) Code. So kann die KI dazu dienen, Quellcode automatisiert zu generieren, Code zu vervollständigen oder Fehler zu erkennen. Auch in der Qualitätssicherung oder dem automatisierten Testing sowie in der Integration natürlicher Sprache bietet Generative KI zahlreiche Chancen.

Durch die fortschreitende Entwicklung wird voraussichtlich auch die Low-Code-Technologie künftig flexibler, leistungsfähiger und benutzerfreundlicher. Dies wird Unternehmen dabei unterstützen, schneller auf sich ändernde Anforderungen zu reagieren und maßgeschneiderte Anwendungen effizient zu erstellen.

3.2. Abgrenzung Tools

Autoren: Wolfram Schäfer und Hien Nguyen,
IT Engineering Software Innovations GmbH

Entwicklungsphilosophien, wie es Low- und No-Code Plattformen sind, erlauben, wie in Kapitel 3 ausführlicher beschrieben, einen schnellen und effizienten Ansatz zur Entwicklung von Softwareanwendungen. Diese benutzerfreundlichen Platt-

formen ermöglichen es, technisch versierten Mitarbeiterinnen und Mitarbeiter, den sogenannten Citizen Developers, also nicht professionellen Entwicklern Anwendungen zu schreiben. Ermöglicht wird dies durch ihre äußerst intuitive Bedienoberfläche. Bedienelemente, wie beispielsweise Textfelder und Schaltflächen, der zu entwickelnden Anwendungen werden durch Drag-und-Drop Interaktionen hinzugefügt. Darüber hinaus bestehen Schnittstellen zu weit verbreiteten Datenbanken, sowie zu ERP-Datenquellen, welche nicht mehr manuell programmiert werden müssen. Somit können sowohl einfache als auch komplexe Prozesse innerhalb der Anwendung abgebildet werden.

Bestehende Rollen und deren Tätigkeiten werden neu definiert, bei denen Citizen Developer, meist, fachliche Experten, die aktiv an der Entwicklung gemeinsam mit Softwareentwicklern zusammenarbeiten (siehe ausführlicher zu Rollen und Personen in Kap. 3.5.2). Die Arbeitsumgebung gestaltet sich dynamischer, ausgelöst durch das Einfließen von kreativen Ideen, ohne dass technische Hürden im Weg stehen. Dieser kollaborative Ansatz trägt dazu bei, innovative Lösungen zu entwickeln, die den Anforderungen des modernen Marktes gerecht werden und gleichzeitig die Entwicklungszeit erheblich verkürzen. Dadurch wird der Softwareentwicklungsprozess demokratisiert und Unternehmen können Projekte mit kürzerer Entwicklungszeit und mit minimalem Ressourcenaufwand umsetzen.

Doch es stellt sich die Frage, wie sich dieser neue Entwicklungsansatz vom konventionellen Softwareentwicklungsprozess unterscheidet. Die nachstehende Tabelle fasst die genannten Merkmale von Low- und No-Code, auf die im dritten Kapitel detaillierter eingegangen wird, zusammen und vergleicht diese mit der konventionellen Softwareentwicklung.

Tab. 1: Merkmale von Low- und No-Code im Vergleich zur konventionellen Programmierung

	No-Code	Low-Code	konventionell
Rollen & Zielgruppe	Citizen Developer	Citizen Developer und Softwareentwickler	Softwareentwickler
Anforderung	keine Programmierkenntnisse	Visuelles Interface, geringe Programmierkenntnisse	Programmierkenntnisse
Entwicklungskomplexität	mittel, intuitive Entwicklung durch Drag-& Drop Oberflächen	mittel, ergänzend zu NC-Ansatz besteht ein Angebot an vordefinierten mit Code anpassbaren Modulen	hoch, manuelle Programmierung von Anfang bis Ende
Kosten	niedrig, schnelle Entwicklung	niedrig, schnelle Entwicklung	hoch, lange Entwicklung
Skalierbarkeit	eher begrenzt, können sich schnell als unflexibel und ungeeignet erweisen. Gebunden an den Plattformanbieter	mittel, können größere Systeme unterstützen. Gebunden an den Plattformanbieter	hoch, am besten skalierbar. Hohe Lasten können verteilt und auf großen Systemen skaliert werden.
Flexibilität & Anpassungsfähigkeit	niedrig, gebunden an den vorgegebenen Workflow der NC-Plattform	mittel, Anpassungen sind per Code realisierbar	hoch, hohe Kontrolle und Anpassungsfähigkeit durch individuellen Code

Die Softwareentwicklung mit NC- und LC-Plattformen stellt einen neuen Ansatz dar, der die Erstellung von Software ohne die Notwendigkeit von Programmierkenntnissen ermöglicht. Dieser Ansatz ist nicht nur eine zusätzliche Methode oder Technologie, sondern eine völlig neue Denkweise, die die Möglichkeiten und den Zugang zur Softwareentwicklung erweitert. Durch den erweiterten Kreis der potenziellen Entwickler erfordert dies auch eine andere und neue Art des Problemlösens und der Zusammenarbeit.

Die Softwareentwicklung ist ein kreativer und komplexer Prozess, der verschiedene Phasen wie Planung, Analyse, Design, Implementierung, Test und Wartung umfasst. Die klassische Softwareentwicklung erfordert in der Regel ein hohes Maß an Programmierkenntnissen, um den Anforderungen der Kunden gerecht zu werden. Besonders wichtig sind die fortlaufende Wartung und Aktualisierung der Software, um sicherzustellen, dass sie den aktuellen Anforderungen entspricht. NC- und LC-Plattformen beeinflussen nicht nur

funktionale Aspekte, sondern auch nicht-funktionale Aspekte wie beispielsweise die Wartbarkeit, Erweiterbarkeit und Evolvierbarkeit der Software (gemäß ISO 9126). Daher spielen sie eine entscheidende Rolle für die Nachhaltigkeit. Diese Aspekte sind wichtig, um eine hohe Qualität, Sicherheit und Lebensdauer der Software sicherzustellen, und erfordern nach wie vor Expertise in der klassischen Softwareentwicklung. NC- und LC-Plattformen können diese Aspekte sowohl verbessern als auch verschlechtern, je nach ihrer Anwendung. Zum Beispiel können sie die Wartbarkeit erleichtern, indem sie die Software verständlicher und in kleinere Module unterteilt. Andererseits können sie die Software durch Abhängigkeit von externen Plattformen und deren Intransparenz erschweren. Deshalb ist es essenziell, dass Citizen Developer, die Qualitätsstandards der Software kennen und eng mit professionellen Entwicklern, genannt Pro Codern, zusammenarbeiten.

Um eine erfolgreiche Softwareentwicklung zu gewährleisten, müssen sowohl die Citizen Developer als auch die Pro Coder über bestimmte Kompetenzen verfügen, die sich teilweise überschneiden, aber auch unterscheiden. Zu den gemeinsamen Kompetenzen gehören z. B. die Fähigkeit, die Anforderungen und Bedürfnisse der Nutzer zu verstehen, die Logik und Funktionalität der Software zu entwerfen, die Software zu testen und zu verbessern als auch diese zu dokumentieren und zu kommunizieren. Verschiedene Kompetenzen umfassen zum Beispiel die Fähigkeit abstrakt, analytisch und umfassend zu denken, was für Pro Coder wichtiger ist, um die Software auf einer tieferen Ebene zu verstehen und zu optimieren. Für Citizen Developer ist hingegen die Fähigkeit kreativ und innovativ zu denken, wichtiger, um die Software auf einer höheren Ebene zu gestalten und zu erweitern. Insofern muss in einem Projekt bzw. Team auch hierfür entsprechende Kompetenz vorhanden sein, um die Vorteile der NC- und LC-Plattformen optimal zu nutzen.

3.3. Chancen und Grenzen von Low-Code / No-Code (LC/NC)

Autoren: Kay Jeschke, Georg Braun, SAP Deutschland SE & Co. KG und Karl Haag, Simplifier AG

Obwohl sich mit Low-Code und No-Code zahlreiche Chancen ergeben, geforderte neue Anwendungen und Funktionen schneller und kostengünstiger zu realisieren, sind auch Grenzen zu beachten. Zum einen gibt es einen weltweiten Trend hin zu Cloud-Anwendungen, auch im ERP- Umfeld, was den Druck zu einer gewissen Standardisierung erhöht, um die Releasefähigkeit der Kernsysteme zu gewährleisten. Zum anderen werden durch die vereinfachten Möglichkeiten von Low-Code und No-Code-Lösungen Begehrlichkeiten geweckt, Zusatzfunktionen komplett außerhalb der IT zu erstellen, was dem Trend einer Standardisierung entgegenwirkt. Hier muss

jedes Unternehmen seinen eigenen Weg finden, um eine planbare, sichere und zukunftsfähige Administration seiner unternehmensweiten Software-Architektur zu gewährleisten.

Eine interessante Umfrage von Gartner aus dem Jahr 2021 hat ermittelt, dass im Schnitt ca. 41 % der Belegschaft eines Unternehmens sogenannte „Business Technologists“ ausmachen, die außerhalb der IT-Abteilungen berichten (zentralisierte oder Geschäftsbereichs-IT) und Technologie- oder Analysefunktionen für den internen oder externen Geschäftsgebrauch erstellen. Davon sind ca. 80 % sogenannte „Citizen Technologists“, die im Rahmen (oder zusätzlich zu) ihrer Vollzeitbeschäftigung technologische Ergebnisse erbringen bzw. Lösungen erarbeiten. Diese Nicht-ITler können unabhängig voneinander Low-Code- und No-Code-Technologien nutzen, um die Effizienz, Wirksamkeit und Agilität des Unternehmens zu verbessern.

3.3.1. Chancen

Low-Code und No-Code-(LC/NC) Plattformen haben bereits heute einen erheblichen Einfluss auf die Softwareentwicklung und Digitalisierung in Unternehmen der Fertigungsbranche. Es ist zu erwarten, dass ihre Bedeutung in Zukunft noch weiter zunehmen wird.

Zu den wesentlichen Vorteilen gehören zum einen die höhere Geschwindigkeit sowie eine größere Anzahl potenzieller Entwickler, mit der neue Anforderungen realisiert und einer damit einhergehenden Kostenersparnis erzielt werden. Dies zeigt die Bedeutung von Low-Code und No-Code.

Die nachfolgende Grafik zeigt, dass mit der einhergehenden Demokratisierung der Softwareentwicklung durch die Einbindung von Citizen Developern zusätzliche Ressourcen geschaffen werden. Der häufig existierende Rückstand (Backlog) an Anpassungsentwicklungen,

Automatisierungsanforderungen und App-Entwicklung der IT-Abteilung kann so reduziert werden. Gleichzeitig lässt sich auch die limitierte Kapazität der IT-Abteilung klar erkennen, die sich überwiegend mit den kritischen Themen und Systemen auseinandersetzen muss und dadurch die übrigen Anforderungen liegen bleiben oder spät bis gar nicht fertiggestellt werden können – die aber, sofern vertretbar, an Citizen Developer abgegeben werden können.

aller realisierten Anwendungen sowie Use-Cases und kann sich weiter auf die geschäftskritischen Themen fokussieren.

Eine weitere Umfrage von IDC (IDC 2021: The Low-Code Development Imperative) verdeutlicht, dass LC/NC-Tools bei vielen Unternehmen schon angekommen sind, wie die folgende Abbildung zeigt.

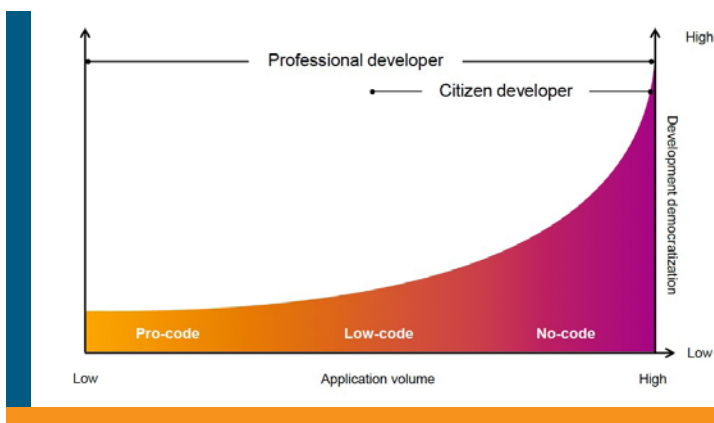


Abb. 1: Durch LC bzw. NC ist eine Skalierung von Applikationsentwicklung durch „Citizen Developer“ möglich (Quelle: SAP Build Apps Overview, 2022)

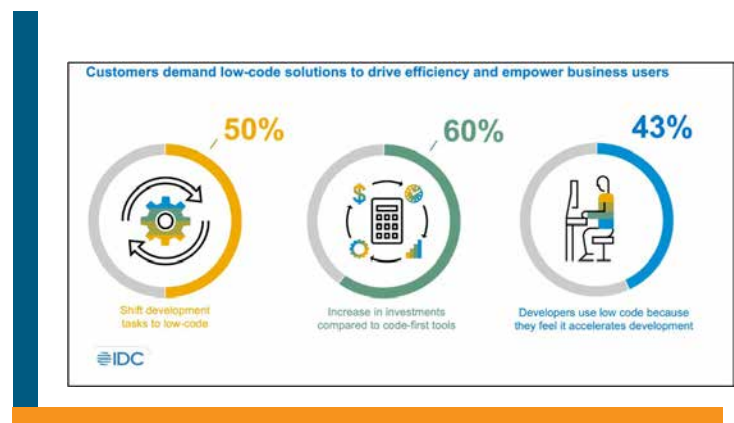


Abb.2: Warum wählen IT-Abteilungen LC/NC Tools (Quelle: IDC 2021 – The Low-Code Development Imperative)

Hier bietet sich für Unternehmen die Chance anzusetzen, um den Fach-, Geschäftsexperten oder Citizen-Developern die Werkzeuge an die Hand zu geben, die einfacheren Anforderungen und Use-Cases umzusetzen. Durch ihr Expertenwissen und etwas Schulungsaufwand sowie mit Hilfe von LC/NC-Tools können sie selbstständig, in einem bestimmten Rahmen, agieren. Dies entlastet die IT-Abteilung und beschleunigt die Digitalisierung und Automatisierung in den Fachbereichen sowie des Unternehmens insgesamt. Die IT-Abteilung bzw. eine dedizierte Abteilung kümmert sich dann um die Bereitstellung und Administration dieser Tools sowie das Monitoring

Im Folgenden werden die, vorher bereits erwähnten, drei wichtigsten Vorteile nochmal im Detail betrachtet. Anschließend erfolgt eine detaillierte Auflistung der genannten Unterpunkte:

(1) Erhöhung der Geschwindigkeit der Anwendungsentwicklung:

Mit LC/NC-Plattformen können Unternehmen Anwendungen viel schneller erstellen und bereitstellen als mit traditionellen Methoden. Dies ist besonders nützlich in einem wettbewerbsorientierten Geschäftsumfeld, in dem die Geschwindigkeit der Markteinführung entscheidend sein kann.

- **Beschleunigung der Digitalisierung:**

Diese Tools können Unternehmen dabei helfen, ihre Geschäftsprozesse noch schneller zu digitalisieren und zu automatisieren, was zu einer höheren Effizienz und Produktivität führen kann. In der produzierenden Industrie ist die Digitalisierung entscheidend, um wettbewerbsfähig zu bleiben. Low-Code-Plattformen ermöglichen es, maßgeschneiderte Anwendungen schnell zu entwickeln, um Prozesse zu automatisieren, Daten zu analysieren und Effizienzsteigerungen vorzunehmen. Dies beschleunigt den digitalen Wandel und ermöglicht es Unternehmen, sich an veränderte Marktbedingungen anzupassen.

- **Integration von IoT und Datenanalyse:**

Da IoT (Internet of Things) in der produzierenden Industrie immer wichtiger wird, können Low-Code-Plattformen leicht Schnittstellen zu IoT-Geräten und -Datenquellen nutzen oder gar erstellen. Das ermöglicht eine bessere Überwachung, Analyse und Steuerung von Produktionsprozessen und trägt somit ebenso zu einem Geschwindigkeitsgewinn bei.

- **Skalierbarkeit:**

Unternehmen der produzierenden Industrie können mit Low-Code einfacher neue Anwendungen hinzufügen oder bestehende Anwendungen skalieren, wenn sie wachsen oder sich verändern. Dies fördert die Agilität und ermöglicht es, auf neue Geschäftschancen schnell zu reagieren.

- **Resilienz und Agilität:**

LC/NC-Tools ermöglichen es Unternehmen schnell auf sich ändernde Anforderungen zu reagieren und die Lösungen in einem schnelleren Tempo an die Bedürfnisse der Fachbereiche anzupassen. Das macht Unternehmen widerstandsfähiger gegenüber ungeplanten Veränderungen auf dem Markt.

(2) Erhöhung der Anzahl potenzieller Entwickler:

Es ist weniger technisches Know-how erforderlich, da LC/NC-Tools eine visuelle Entwicklungsumgebung bieten. Sie sind zugänglicher für Benutzer ohne umfangreiche Programmierkenntnisse. Dies bedeutet, dass mehr Mitarbeitende in der Lage sind, Anwendungen zu erstellen und zu warten. Das schließt die Lücke zwischen den Megatrends Fachkräftemangel, teilweise verursacht durch den demografischen Wandel, und dem Druck einer schnelleren Digitalisierung. Im Folgenden sind die genannten Punkte im Detail aufgeführt:

- **Demokratisierung der Entwicklung:** Low-Code-Plattformen machen die Softwareentwicklung für ein breiteres Spektrum von Mitarbeitenden zugänglich. Fachkräfte ohne tiefgehende Programmierkenntnisse können Anwendungen erstellen und somit die Innovationsfähigkeit des gesamten Unternehmens steigern.

- **Steigerung der Zusammenarbeit zwischen den Fachbereichen und der zentralen IT:**

Die zentrale IT kann Teillaufgaben an die Fachbereiche auslagern, um die Kapazität zu erhöhen, ohne dabei auf eine zentrale Governance der neu erstellten Anwendungen verzichten zu müssen.

(3) Kostenersparnis:

Durch die Verwendung von LC/NC-Tools können Unternehmen Kosten sparen, da weniger Zeit und Ressourcen für die Entwicklung von Anwendungen im Vergleich zu einer Standard Pro-Code-Entwicklung aufgewendet werden müssen. Darüber hinaus können diese Tools dazu beitragen, die Abhängigkeit von externen Entwicklern oder Beratern zu verringern. Auf die einzelnen Aspekte der Kostenersparnis wird nachstehend genauer eingegangen:

- **Kosteneffizienz:**

Die Reduzierung der Entwicklungszeit und -kosten ist ein entscheidender Faktor für Unternehmen. Low-Code-Plattformen ermöglichen es, Anwendungen schneller und kostengünstiger zu erstellen, da sie weniger Entwicklungsressourcen und technische Fachkenntnisse erfordern.

- **Flexibilität und Anpassungsfähigkeit:**

Änderungen und Aktualisierungen von mit LC/NC-Tools erstellten Anwendungen können relativ leicht vorgenommen werden, was zu geringeren Wartungskosten und zu einer größeren Agilität führen kann.

- **Reduzierung der Abhängigkeit von externen Entwicklern:**

Die Einbindung von Fachbereichsexperten in die Anwendungsentwicklung mit LC/NC-Tools kann dazu führen, dass das Einkaufsvolumen externer Pro-Code-Entwickler deutlich sinkt und dass das Know-how im Unternehmen bleibt – dies führt ebenso zu einer Reduktion der Kosten.

Zusammengefasst kann davon ausgegangen werden, dass alle beteiligten Parteien von der Nutzung der LC/NC-Tools profitieren – sowohl das Unternehmen als Ganzes durch Steigerung der Effizienz und Produktivität der Mitarbeitenden sowie die Fachbereiche durch die digitale Unterstützung im Tagesgeschäft. Nicht zuletzt gewinnt auch die IT-Abteilung durch die Reduzierung des Rückstandes an Entwicklungsaufgaben und Konzentration auf ihre Spezialkompetenzen und Fähigkeiten für die komplexeren Anforderungen.

3.3.2. Grenzen

Nachdem im vorherigen Kapitel die Chancen von Low-Code und No-Code aufgezeigt wurden, werden in diesem Kapitel die Schwächen dieses Werkzeuges genauer betrachtet. Low-Code-/No-

Code-Tools können viel, aber sie sind nicht das neue Allheilmittel für alle zukünftigen Anforderungen in der Anwendungsentwicklung in einem Unternehmen. Grenzen in Komplexität, Integration, Security, Governance und IP-Rechten sowie bei der KI-Nutzung sind zu beachten. Auf diese aufgeführten Aspekte soll im Folgenden genauer eingegangen werden:

- **Komplexität:**

Für komplexe, maßgeschneiderte Anwendungen oder Anwendungen, die für die vorgedachten Elemente von Low-Code/No-Code nicht vorgesehen wurden, sind traditionelle Entwicklungsansätze möglicherweise besser geeignet. Das Gleiche gilt für große Entwicklungsprojekte, bei denen viele Entwickler gleichzeitig im Code arbeiten.

- **Integration:**

Neue Anwendungen kommen selten ohne die Integration in andere Anwendungen aus. Für diese Integration ist oft ein spezifisches Know-how der Quellanwendungen erforderlich, um die Integration in dem Format zur Verfügung zu stellen, damit ein LC/NC-Tool die zu integrierenden Daten konsumieren kann. Ein zentrales Monitoring aller Integrationen ist hier zu empfehlen.

- **Security, Data Security:**

Es dürfen keine Sicherheitslücken entstehen, wenn LC/NC-Tools auf sensible Daten anderer Anwendungen zugreifen. Rollenkonzepte und Berechtigungen müssen sowohl bei der Entwicklung mit LNCN-Tools als auch bei der Entwicklung von Integrationen berücksichtigt werden. Unternehmensweite Authentifizierungsmethoden sollten unbedingt eingebunden werden. LC/NC-Tools sind oft Cloudanwendungen, es ist deshalb darauf zu achten, in welchem Land und in welchem Datacenter Daten gespeichert werden, denn für bestimmte Länder gibt es hier Einschränkungen.

- **Standardisierung:**

Viele ERP-Systeme werden zukünftig in der Cloud betrieben, was einen gewissen Grad an Standardisierung mit sich führt, um release-fähig zu bleiben. LC/NC-Tools erlauben oftmals eine Individualisierung des Standards ohne die Releasefähigkeit zu verlieren. Eine zu weit gefasste Nutzung dieser Möglichkeiten treibt oft Kosten und Einführungsdauer einer ERP-Transformation in die Höhe.

- **Governance-Konzept:**

Die Grenze, was erlaubt ist und was nicht, muss von jedem Unternehmen definiert werden. Es muss festgelegt werden, welche LC/NC-Tools erlaubt sind und welche Integrationen in bestehende Anwendungen sowie Daten verwendet werden dürfen. Dabei müssen Fragen, wie die folgenden beantwortet werden: Wer ist für die Wartung, Anpassung und Leistungsfähigkeit der neu erstellten Anwendungen verantwortlich und wer stellt den Support sicher? Wer sorgt dafür das Sicherheits- und Compliance-Richtlinien eingehalten werden? Ein durchgehendes Governance-Konzept ist umso wichtiger, je größer und stärker einzelne Fachbereiche und Organisationen in einem Unternehmen sind. Dies ist notwendig, um die Vorteile von LC/NC-Tools zu heben und sich nicht wieder in der Unwartbarkeit diverser Schatten-IT Organisationen zu verlieren.

- **IP-Rechte:**

Werden Open Source-Komponenten genutzt, unterliegt oft die gesamte Anwendung, die mit Hilfe dieser Tools erstellt wird, den Lizenzbestimmungen der Open Source-Komponente. Das kann dazu führen, dass auch die neue Anwendung, rechtlich gesehen, eine Open Source-Anwendung wird.

- **KI:**

Künstliche Intelligenz wird auch in LC/NC-Tools zunehmend zur Code-Generierung verwendet. Sollten KI-Funktionen auch für die Erstellung von Inhalten verwendet werden, ist unbedingt auf die Quellen zu achten mit denen die KI gespeist wird.

- **Ethik:**

Dies ist ein Thema, dass mit der KI eng verbunden ist, deshalb verweisen wir in diesem Dokument nur auf die wesentlichen Richtlinien, die sehr gut in den EU-Leitlinien der KI definiert werden. (Europäische Kommission 2018: Ethik-Leitlinien für eine vertrauenswürdige KI).

Generell ist zu beachten, dass jede Low-Code-Plattform ihre eigenen Stärken und Schwächen hat. Daher sollte die Auswahl stets individuell erfolgen und in Übereinstimmung mit der Unternehmensstrategie getroffen werden. In vielen Fällen führt die Implementierung einer Low-Code/No-Code-Plattform nicht allein und automatisch zu einer Verbesserung eines ineffizienten Prozesses. Oft ist es ratsam, zuerst den Prozess selbst zu optimieren, bevor über den Einsatz einer solchen Plattform nachgedacht wird. Testen und evaluieren Sie daher verschiedene Low-Code-/No-Code-Plattformen, bevor eine Plattform übernommen wird. Und überlegen Sie, ob die jeweiligen Vor- und Nachteile mit ihren eigenen individuellen Projekt- oder Geschäftsanforderungen übereinstimmen. Zudem sollte mit zunehmender Dauer und Erfahrung darüber nachgedacht werden, ob mehrere Plattformen, angepasst an den jeweiligen Zweck, eingesetzt werden sollten (Stichwort: Multiplattformstrategie). Eine entscheidende Rolle für eine optimale Lösung spielt dabei das Mindset (siehe auch Kapitel 5).

3.4. Ebenen und Kategorien von Low-Code und No-Code

Im Folgenden werden zunächst die Ebenen von Low-Code und No-Code und im darauffolgenden Teilkapitel die unterschiedlichen Kategorien beziehungsweise Dimensionen von LC/NC betrachtet.

3.4.1. Ebenen

Autor: Dr. Rainer Mümmeler, The MathWorks GmbH

Low-Code-Tools spielen in Bezug auf Architektur und Technik auf verschiedenen Ebenen eine Rolle. Im Folgenden sind einige der Ebenen, auf denen Low-Code-Tools eine Bedeutung haben, aufgeführt. Dies beinhaltet die Anwendungsebene, die Integrations- und Schnittstellenebene, die Workflow- und Prozessebene sowie die Datenbank- und Modellierungsebene.

Auf der **Anwendungsebene** bieten Low-Code-Tools visuelle Drag-and-Drop-Tools, Apps und vorgefertigte Komponenten, um die Entwicklung von Anwendungen mit minimalem, manuellen Programmieraufwand zu beschleunigen. Sie helfen sowohl den professionellen Entwicklern, die kunden-spezifischen Programmiercode erstellen, als auch Business-Entwicklern, die dank Low-Code-Umgebungen ohne technische Kenntnisse eigene Anwendungen entwickeln können (Citizen Developer s. Kapitel 2 und 5.2). Die Beweggründe Low-Code-Werkzeuge zu verwenden, sind unter anderem das schnelle Erlernen und Anwenden neuer Techniken, wie z. B. der Signalanalyse oder des maschinellen Lernens. Zudem können sie bei allgemeinen oder iterativen Aufgaben wie der Datenvisualisierung, der Datenbereinigung, des Modelltrainings und der gemeinsamen Nutzung von Ergebnissen effizient eingesetzt werden. Auch Entwickler, die vor dem Programmieren zurückschrecken oder es als zeitaufwändig empfinden, finden in Low-Code-Tools eine Lösung (siehe auch Kapitel 5.2 Rollen und Personen). Sowohl für Domänenexperten als auch für erfahrene Programmierer bieten einige Low-

Code-Tools die Möglichkeit automatisch Code für verschiedene Deployment-Szenarios (Embedded, Edge, Cloud) zu generieren

In Bezug auf die **Integrations- und Schnittstellenebene** bieten Low-Code-Tools eine Reihe von Funktionen, die die Integration von Anwendungen und Systemen erheblich erleichtern. In der Regel bieten sie vorgefertigte Konnektoren zu gängigen Datenquellen, APIs und Services. Diese Konnektoren ermöglichen es Entwicklern, nahtlose Verbindungen zwischen verschiedenen Anwendungen und Systemen herzustellen, ohne dass sie tiefgreifende Kenntnisse über die zugrunde liegenden Technologien benötigen. Darüber hinaus können Low-Code-Tools oft auch benutzerdefinierte Integrationen unterstützen, indem sie Funktionen zur Erstellung von benutzerdefinierten APIs und Services bieten.

Mit Blick auf die **Workflow- und Prozessebene** bieten Low-Code-Tools Funktionen zur Modellierung und Automatisierung von Workflows und Geschäftsprozessen. Sie bieten eine visuelle Schnittstelle zur Erstellung von Workflow-Diagrammen, in denen die verschiedenen Schritte eines Prozesses und die Regeln, die bestimmen, wie und wann sie ausgeführt werden, definiert werden können. Darüber hinaus bieten sie oft auch Funktionen zur Automatisierung von Aufgaben, wie z. B. das Senden von Benachrichtigungen oder das Aktualisieren von Datenbanken. Dies ermöglicht es Entwicklern, komplexe Geschäftsprozesse zu modellieren und zu automatisieren, ohne dass sie Codes schreiben müssen.

Bezüglich der **Datenbank- und Modellierungsebene** bieten visuelle Schnittstellen zur Erstellung von Datenmodellen, die die Struktur der Daten, die in einer Anwendung verwendet werden, eine weitere Möglichkeit der Unterteilung. Darüber hinaus bieten sie oft auch Funktionen zur Datenvalidierung und Datenmanipulation, wie z. B. das Überprüfen der Gültigkeit von Eingabedaten oder das Durchführen von Datenbankabfragen. Dies ermöglicht es den Entwicklern, die Datenbanken und

die Datenstrukturen zu modellieren, ohne dass sie SQL oder andere Datenbank-Sprachen kennen müssen.

Obwohl die genaue Rolle von Low-Code-Tools auf diesen Ebenen je nach spezifischer Plattform und Tool variieren kann, ermöglichen sie insgesamt eine schnellere und effizientere Entwicklung von Anwendungen. Durch die Vereinfachung der Integrationen und die Automatisierung von Prozessen können Entwickler sich auf die Lösung von Geschäftsproblemen konzentrieren, anstatt sich mit technischen Details zu befassen (siehe Kapitel 3.3: Chancen und Grenzen).

3.4.2. Kategorien von Low-Code- und No-Code-Systemen

Autor: Dr. Thomas Genßler, CAS Software AG

Die Einordnung von Low-Code und No-Code-Ansätzen kann grundsätzlich entlang zweier Dimensionen erfolgen. Die erste Dimension betrifft die zunehmende Abstraktion und Ausdrucksstärke der Sprachmittel zur Formulierung einer technischen und/oder fachlichen Lösung eines gegebenen Problems und konzentriert sich damit auf die Lösungsebene. Die zweite Dimension, die insbesondere in der jüngeren Vergangenheit an Bedeutung gewonnen hat, betrifft die zunehmende Verlagerung der Betrachtungsebene weg von der Formulierung einer konkreten Lösung auf die exakte Beschreibung des zu lösenden Problems und der zu lösenden Anforderungen (Anforderungsebene).

3.4.2.1. Dimension 1 – Steigende Abstraktion auf der Lösungsebene

Die ersten Low-Code-Ansätze hatten hauptsächlich das Ziel, die Produktivität bei der Codierung einer Problemlösung zu erhöhen, sowie die Wiederverwendbarkeit einmal entwickelter Lösungen, auch auf neuen Betriebsumgebungen, sicherzustellen. Dies wurde durch die Abkehr von der direkten Codierung in Maschinensprache oder Assembler und die Einführung so genannter Hochsprachen erreicht. Die Hochsprachen ermöglichten es (Fach-)

Entwicklern, von der zugrundeliegenden Computerhardware und den benutzten Betriebssystemen zu abstrahieren und sich auf die kompakte Formulierung einer Problemlösung zu konzentrieren, ohne sich mit technischen Details beschäftigen zu müssen. Der Fokus lag dabei auf der Erhöhung der Ausdrucksstärke, Kompaktheit und Verständlichkeit des Programmcodes. Technisch wurde dies erreicht durch den Einsatz von Compilern, Codegeneratoren und schnittstellenstandardisierten Laufzeitumgebungen. In der Literatur werden die entsprechenden Programmiersprachen oft als 3GL (third generation languages, bspw. Java, C++), 4GL (Python, Ruby, SQL) und 5GL (OPS5, Prolog) bezeichnet.

Neuere Low-Code / No-Code-Ansätze haben darüber hinaus das Ziel, auch nicht IT-affine Domänenexperten (so genannte „Citizen Developers“) zu befähigen, bestimmte fachliche IT-Probleme selbst zu lösen, ohne dass ein Programmierer benötigt wird (s. auch Kap. 3.5.2). Dies wird erreicht durch den Übergang von General-purpose-Programmiersprachen hin zu so genannten Domänensprachen, die sich mit ihren (eingeschränkteren, vereinfachten) Sprachmitteln auf eine bestimmte Problem-domäne konzentrieren, damit aber nicht mehr gut zur Lösung allgemeiner Probleme geeignet sind. Charakteristisch für Domänensprachen ist der Einsatz grafischer Modellierungsumgebungen, sowie die Orientierung der Sprachmittel in Vokabular und Struktur an der adressierten Domäne. Beispiele dafür sind etwa Excel, WordPress, Workflow-Umgebungen oder grafische CNC-Programmiersysteme.

Dennoch ist bei all diesen Verfahren gemein, dass sie sich mit der Beschreibung einer konkreten Problemlösung, dem „Wie“ beschäftigen.

3.4.2.2. Dimension 2 – Verlagerung auf die Anforderungsebene

Insbesondere in der jüngeren Vergangenheit und als Ergebnis der steigenden Leistungsfähigkeit von KI-Systemen ist eine Verlagerung der Betrachtungsebene weg von der Beschreibung einer konkreten Lösung, hin zu der Beschreibung des eigentlichen Problems und der notwendigen Anforderungen an eine passende Problemlösung zu beobachten. Dies stellt einen fundamentalen Paradigmenwechsel dar, da nun nicht mehr der gewünschte Lösungsweg spezifiziert wird, sondern die von einer passenden Lösung gewünschten Eigenschaften. Die Entwicklung der Lösung selbst obliegt dabei dem No-Code-System. Aktuelle Beispiele für diesen Ansatz sind Systeme der generativen KI wie beispielsweise ChatGPT, wenngleich diese Systeme nicht ausschließlich als No-Code-Systeme konzipiert sind. Heute stehen diese Systeme, trotz ihrer bereits beeindruckenden Leistungsfähigkeit, vergleichsweise noch am Anfang und werden hauptsächlich als Assistenzsysteme innerhalb klassischer Entwicklungsumgebungen eingesetzt. Eine Weiterentwicklung hin zu vollwertigen No-Code-Systemen ist jedoch bereits absehbar.

3.5. Organisation und Struktur

Autoren: Judith Armbruster und Wolfram Schäfer, IT Engineering Software Innovations GmbH

Wie in den vorangegangenen Kapiteln aufgezeigt wurde, bringt die Beschäftigung und Integration von Low-Code zahlreiche Chancen sowie auch Herausforderungen mit sich, gleichzeitig wird deutlich, dass Low-Code nicht nur eine zusätzliche Technologie oder Methode ist, sondern ein gänzlich neuer Denkansatz in der Durchführung von Projekten. Dementsprechend ist es nicht nur für einzelne Personen und Teams wichtig, sich mit dem Thema zu beschäftigen, sondern es braucht auch ein ganzheitliches Verständnis im Unternehmen sowie ein passendes Vorgehen bei der Einführung im Sinne der Organisationsent-

wicklung. Im nachfolgenden Infokasten werden die Begriffe „Organisationsentwicklung“ und „Change Management“ kurz gegenübergestellt.

Organisationsentwicklung vs. Change Management

Change Management und Organisationsentwicklung sind zwei Begrifflichkeiten, die beide im Kontext für bewusste Veränderung in Unternehmen verwendet werden. Change Management steht dabei für jegliche Art von projektorientierten Veränderungsprozessen, somit zumeist eher kurz- bis mittelfristig angelegte Veränderungen in überschaubaren Projekten. Organisationsentwicklung hingegen setzt auf eine Langfristigkeit des Veränderungsprozesses sowie größtmögliche Partizipation, wobei das Verhalten der Mitarbeitenden im Mittelpunkt steht (vgl. Haufe 2023). Weiterführend ist agile Organisationsentwicklung durch eine iterative Vorgehensweise und eine erlebbare Veränderung gekennzeichnet und befähigt Menschen, mit gegebenen Modellen und Werkzeugen eigene passende Lösungen zu entwickeln (Armbruster 2023).

Veränderung wird durch die intensive Auseinandersetzung mit dem Thema Low-Code in den Unternehmen gefördert. Diese Bewegung und die Notwendigkeit sich anzupassen, ist für Unternehmen jedoch nicht neu. Im größeren Kontext kann Low-Code somit auch in die Bewegungen des agilen Wandels eingeordnet werden. Denn Agilität wird im Sinne der Erhöhung der Anpassungsfähigkeit und Sensitivität für Veränderung von Unternehmen als Antwort auf die sich immer schneller ändernden Rahmenbedingungen gesehen. Somit soll im Folgenden für die strukturellen Themen der Fokus auf zwei Ebenen gelegt werden, nämlich auf moderne Strukturen innerhalb der Teams, sowie die Ebene der agilen Organisation als Ganzes.

3.5.1. Organisation und Struktur im Speziellen

In diesem Kapitel wird zunächst auf die Zusammenarbeit im Team (Kap. 3.5.1.1.) eingegangen und anschließend auf die verschiedenen Rollen und Personen, die involviert sind (Kap. 3.5.1.2). Sowohl das Mindset als auch die Teamzusammensetzung ist hierbei von wesentlicher Bedeutung.

3.5.1.1. Zusammenarbeit im Team

Neue Aufgaben, Rollen und Verantwortungen bringen Veränderung in die Arbeit der Teams, daraus ergeben sich neue Prozesse und Handlungsweisen teilweise wie von selbst, müssen aber spätestens für eine unternehmensweite Einführung bewusst gemacht und vereinheitlicht werden. Agile Prozesse und Strukturen setzen dabei vor allem darauf, Rahmen und Räume vorzugeben, in denen die beteiligten Personen selbstbefähigt arbeiten können – diese quasi mit Leben füllen.

Ein besonders wichtiger Aspekt in der Teamarbeit ist das Zusammenbringen der passenden und für das Ergebnis notwendigen Personen. Dadurch kann ganzheitliche Verantwortung für das Ergebnis an das Team abgegeben werden. Verschiedene Teamkonstellationen, wie z. B. agile Teams oder Fusion-Teams, sind dabei möglich.

Was sind agile Teams?

Das agile Team hat ein gemeinsames Ziel, z. B. ein Produkt zu entwickeln. Dabei sind möglichst alle unterschiedlichen Kompetenzen, die es dazu benötigt, im Team vorhanden. Jedes Teammitglied bringt seine eigene Expertise und Stärken ein, um ein gemeinschaftliches Ergebnis zu erzielen. Agile Methoden, wie beispielsweise Scrum oder Kanban, unterstützen die Teams in ihrer Arbeit (Armbruster 2023: 39).

Was sind Fusion-Teams?

Fusion-Teams verknüpfen technische Expertise und Domänenwissen, um mit gemeinschaftlicher Verantwortung digitale Anwendungen zu realisieren. Durch die kollaborative Arbeitsweise ermöglichen Fusion-Teams, dass auf effiziente Art und Weise benutzerdefinierte Anwendungen erstellt werden, ohne tiefgreifende Programmierkenntnisse zu benötigen. Dies fördert die Agilität, beschleunigt die Entwicklung und ermöglicht die Anpassung von Software an spezifische Anforderungen (Gartner 2022b).

Gemeinsam stark – das ist das Credo moderner Teamarbeit. Es gilt die Stärken der einzelnen Teammitglieder zu nutzen, Ideen gemeinsam weiterzuentwickeln und damit im offenen Austausch passende Ergebnisse für gegebene Herausforderungen zu entwickeln. Dies zu erleben, fördert nicht nur die Motivation und Kompetenzen der einzelnen Mitarbeitenden, zudem lässt sich über Leuchtturm-Projekte einzelner Teams auch der Umgang mit neuen Themen, wie Low-Code im gesamten Unternehmen etablieren. Schrittweise wird somit die Anpassungs- und Handlungsfähigkeit des Unternehmens als Ganzes gestärkt.

3.5.1.2. Agile Organisationen

Es gibt, wie bereits beschrieben, verschiedene Herangehensweisen und Modelle, um Veränderung in Unternehmen zu gestalten. Das Modell der agilen Blüte (siehe Abb. 3) ist besonders auf die Rahmenbedingungen kleiner und mittlerer Unternehmen zugeschnitten, die in Deutschland die Mehrheit der Unternehmen ausmacht. Es verdeutlicht die Komplexität der Themen im Rahmen von Veränderungen in Unternehmen und zeigt gleichzeitig konkrete Ansatzpunkte auf, um gemeinsam mit kleinen Schritten in einer Organisation Veränderungen zu gestalten.

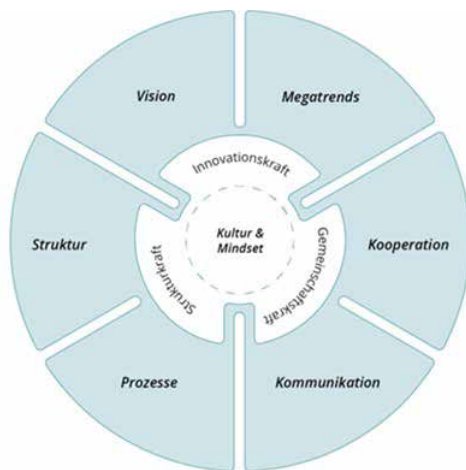


Abb.3: Die agile Blüte als agiles Organisationsentwicklungsmodell für KMU (Armbruster 2023)

Die agile Blüte basiert auf drei tragenden Wirkungskräften: der Strukturkraft, der Innovationskraft und der Gemeinschaftskraft. Dabei ist es die Strukturkraft, die wie ein Grundgerüst der Blüte und ebenso der Organisation Stabilität verleiht. Die Innovationskraft wirkt von innen nach außen, während die Gemeinschaftskraft von außen nach innen wirkt. Diese beiden Kräfte sorgen entsprechend für ein Wechselspiel zwischen Neuausrichtung und Verankerung. Sie fördern die Ambivalenz zwischen Stabilität und Instabilität, die eine agile Organisation benötigt. Dadurch stellt das Modell das Bewusstsein in den Mittelpunkt, dass es für die agile Organisationsentwicklung wichtig ist, regelmäßig und stetig neue Impulse zu setzen sowie gleichzeitig die Balance zu halten. In einem Team bzw. in einem Unternehmen sollte nicht alles auf einmal verändert werden, sondern in der Veränderung umso mehr auch auf Stabilität geachtet werden. Ebenso ist die Gemeinschaftskraft essenziell, die eine gute Kooperation und kontinuierliche Kommunikation beinhaltet.

Es sind schließlich viele Themen, die jede Mitarbeiterin und jeden Mitarbeiter betreffen. Aber es ist ein Weg, der gemeinsam gegangen wird und ein Ziel, das gemeinsam verfolgt wird. Die Auseinandersetzung mit Low-Code kann ein wichtiger Schritt auf diesem Weg sein. Dadurch bietet sich die Chance, nicht nur in einem Team neue Möglichkeiten der Zusammenarbeit zu erproben, sondern auch Offenheit für neue Technologien und Werkzeuge zu stärken.

Es gibt enge Wechselwirkungen zwischen den strukturellen Veränderungen und der Kultur im Miteinander, denn für diesen Weg „ins Neue und mit dem Neuen“ braucht es viele Mitwirkende mit unterschiedlichen Kompetenzen. Das Mitnehmen und das Einbeziehen der Mitarbeitenden sind ein Erfolgsfaktor und gleichermaßen eine Herausforderung. Deshalb ist der gemeinschaftliche Dialog unabdingbar, um den richtigen Weg in der Veränderung zu finden und die Organisation als Ganzes passend neu zu gestalten.

3.5.2. Rollen und Personen

Autoren: Kay Jeschke sowie Georg Braun, SAP Deutschland SE & Co. KG und Karl Haag, Simplifier AG

Das Rollenkonzept der Beteiligten im Softwareentwicklungsprozess wird sich im Low-Code- und No-Code-Umfeld nicht grundsätzlich ändern. Aber es wird um die neue Anwendergruppe der Citizen Developer deutlich erweitert. Dadurch wird auch der „Demokratisierung“ in der Entwicklung vermehrt genüge getan.

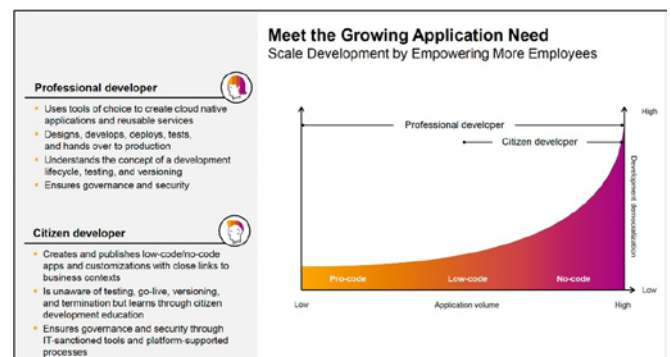


Abb. 4: Skalierung von Applikationsentwicklung durch "Citizen Developer" (Quelle: SAP Build Apps Overview, 2022)

Das Bild zeigt die Überschneidungen zwischen professionellen Entwicklern und der neuen „Klasse“ der Citizen Developer: Die Citizen Developer können den professionellen Entwicklern einen Teil der Aufgaben abnehmen.

Neben den beiden erwähnten Personengruppen, Pro-Coder und Citizen Developer, werden im folgenden Kapitel die wichtigsten Rollen und Personen beschrieben, die im Zusammenhang mit Low-Code und No-Code eine Rolle spielen.

- **Entscheider:**

Die Entscheiderin oder der Entscheider im C-Level-Bereich sind dafür verantwortlich, welche Tools eingesetzt und in welchem Umfang sie genutzt werden dürfen. Sie geben dabei den Rahmen und die Richtlinien vor, wie mit LC/NC-Tools umgegangen wird.

- **Führungskraft der IT-Abteilung:**

Er oder sie setzt die Richtlinien der Entscheiderinnen oder Entscheider um und achtet darauf, dass diese eingehalten werden. Sie sind verantwortlich, dass eine beschleunigte Digitalisierung realisiert wird. Er übernimmt die Aufgabe des Ansprechpartners, um die Akzeptanz innerhalb der Belegschaft zu fördern und dadurch mögliche Veränderungswiderstände zu überwinden – sowohl bei IT-Abteilungen als auch bei den Fachbereichen.

- **Mitarbeitende der IT-Abteilung:**

Die Mitarbeitenden der IT-Abteilung regeln den Zugang zu den Tools, verwalten die Berechtigungen und administrieren die Routineumgebungen für die neu entwickelten Lösungen. Sie sind verantwortlich für alle Integrationen und deren Integrationslösungen, d.h. sie kümmern sich um die gesamte Governance. Die Abteilung organisiert den Zugang zu benötigten Schulungen für die Citizen Developer als auch für Pro-Code-Entwickler und orchestriert die Zusammenarbeit zwischen Pro-Code- und Citizen-Entwicklern.

- **Professionelle Softwareentwickler und -entwicklerinnen:**

Bei einem Softwareentwickler und einer Softwareentwicklerin handelt es sich um einen Spezialisten, der komplexe und technisch anspruchsvolle Anwendungen und Softwarelösungen von Grund auf entwirft, entwickelt und implementiert. Im Gegensatz dazu nutzt ein Citizen Developer Low- bzw. No-Code-Plattformen, um einfache Anwendungen zu erstellen, ohne tiefgehende technische Fähigkeiten. Pro Coder hingegen sind Entwicklerinnen und Entwickler, die Low- bzw. No-Code-Tools als Ergänzung nutzen, um schnellere Lösungen zu erstellen.

- **Pro-Code-Entwickler der IT-Abteilung:**

Ein Pro Coder ist ein Spezialist, der in diversen Programmiersprachen Full-Code-Applikationen erarbeitet. Dabei verwendet er häufig auch LC/NC-Tools und Plattformen, um Anwendungen schneller erstellen zu können. Im Gegensatz zu traditionellem Programmieren setzt der Pro Coder auf visuelle Schnittstellen sowie auf vorgefertigte Bausteine und beschleunigt damit die Entwicklung und Implementierung. Sein Low-Code- bzw. No-Code-Tool dient der schnelleren Erzeugung von Pro-Code-Apps. Dies bedeutet, dass mit dem Tool ein Code erzeugt und anschließend in eine gängige Programmiersprache eingefügt wird. Pro Coder unterstützen auch Fachleute ohne tiefgehende Programmierkenntnisse, d.h., die im folgenden beschriebenen Citizen Developern, bei der Erstellung von Softwarelösungen.

- **Citizen Developer der Fachbereiche:**

Ein Citizen Developer ist ein „nicht-professioneller“ Softwareentwickler innerhalb eines Unternehmens, der mit Low- oder No-Code-Plattformen Anwendungen erstellt oder mithilft diese zu erstellen. Diese Plattformen erfordern nur geringe bis gar keine Programmierkenntnisse, ermöglichen jedoch die Entwicklung von Softwarelösungen für spezifische

geschäftliche Anforderungen. Citizen Developer tragen zur beschleunigten Entwicklung bei, sollten jedoch unternehmensweite Standards und IT-Richtlinien beachten, um potenzielle Risiken zu minimieren. Um Lösungen mit Integrationen zu externen Anwendungen zu realisieren, arbeiten sie meist mit den Pro-Code-Entwicklern der IT-Abteilung zusammen.

- **Business Developer:**

Hierbei handelt es sich um einen professioneller Citizen Developer der Fachabteilung, der nahezu ohne Hilfe der IT-Abteilung neue Anwendungen erstellen kann. Darüber hinaus übernimmt er zusätzlich auch noch Standardaufgaben des Fachbereiches.

- **Professioneller Low-Code Entwickler:**

Ein professioneller Low-Code-Entwickler nutzt LC/NC-Tools als wesentliche Entwicklungsumgebung. Er arbeitet mit Low-Code bzw. No-Code-Tools erzeugt selbst ein ausführbares Programm, welches direkt deployed werden kann. Er ist in der Lage die ihm zu Verfügung gestellten Integrationen selbstständig einzubinden. Er ist in der IT-Abteilung oder im Fachbereich angesiedelt.

- **Basisbetreuer der IT-Abteilung:**

Diese Personengruppe übernimmt das Monitoren der Systemverfügbarkeit der Tools, der entwickelten Apps und der erstellten Integrationen. Sie verwalten die Systeme und die Datenbanken.

- **Anwender bzw. Anwenderinnen (intern und extern):**

Für interne und externe Anwender macht es keinen Unterschied, wie die Software erstellt wurde, da sie nur das Endprodukt nutzen. Dies können zum Beispiel Ingenieurinnen und Ingenieure sein, die Daten analysieren oder Anwendungen schreiben, und dabei Low-Code-Tools verwenden, um diese Arbeitsschritte zu beschleunigen. Im analytischen Umfeld

bekommt diese Anwendergruppe einen höheren Freiheitsgrad, da moderne Daten-Analyse-Tools einen umfangreichen Self-Service Bereich haben, in dem die Anwender und Anwenderinnen ihre eigenen Reports in einem fest definierten Rahmen erstellen können. Trotz dieser Fähigkeiten würde man diese Anwendergruppe noch nicht zu den Citizen Developern zählen.

3.6. Compliance und Governance

Autor: Manuel Möller, gbo datacomp GmbH

Bei dem Einsatz von Low-Code- und No-Code-Plattformen sind zudem die Compliance und Governance der beteiligten Akteure zu beachten. Darunter sind zum einen die Einhaltung von Gesetzen, Richtlinien sowie freiwilligen Kodizes zu verstehen und zum anderen, die in diesem Zusammenhang relevanten Aspekte der Unternehmensführung, d. h. der übergeordneten Steuerung oder Regelung. In diesem Kapitel wird dabei insbesondere auf die Thematik „Daten“ eingegangen.

Die Nutzung von Low-Code-Plattformen kann einem Unternehmen große Möglichkeiten bieten, da Daten meist schnell integriert, ausgewertet und graphisch ansprechend dargestellt werden können. Die Daten, auf die zugegriffen werden sollen, ergeben sich aus dem jeweiligen Anwendungsfall und bieten meist den Anreiz weitere Datenquellen anzubinden. Die Nutzung der Daten und Anreicherung mit anderen Daten ist für den Anwender in erster Linie ein Erfolg und ermöglicht schnelle Ergebnisse. Die Problematik ergibt sich jedoch in genau diesen Datenquellen und deren Nutzung. Je nach Anwendungsfall müssen unterschiedlichste Daten genutzt werden. Besteht die Anforderung z. B. darin, die aktuellen Produktionsdaten (Mengen, Zeiten, Takt etc.) in einem produzierenden Unternehmen darzustellen, ist dies sicherlich für jeden richtig und nachvollziehbar. Aber bei scheinbar einfachen

Anwendungsfällen, bei denen Stakeholder mit unterschiedlichen Interessen beteiligt sind, kann der Entwicklungsprozess mühsam und aufwändig sein. Beteiligte Personengruppen sind beispielsweise: Produktionsmitarbeiter, Schichtleiter, Prozess- und Produktionsverantwortliche, Abteilungsleiter, Bereichsleiter und Geschäftsführer – zusätzlich bedarf es für die technische Umsetzung noch der Einbindung der IT-Abteilung.

Werden diese Stakeholder betrachtet, ist für den Bereich der Produktion die Transparenz der Daten, eine schnelle Darstellung und womöglich der Bezug zu einer einzelnen Maschine das vorrangige Ziel. Aus Sicht des Managements sind jedoch verdichtete und aggregierte Daten pro Abteilung oder Bereich für eine schnelle Analyse interessant. Die IT-Abteilung muss dahingehend die Integration der Daten ermöglichen – setzt jedoch den Fokus eher auf die Sicherheit des Datenzugriffs. Bei der Betrachtung der oben genannten Stakeholder ist nur ein Teil der internen Gruppen berücksichtigt worden. Hinzu kommt die Abstimmung mit dem Betriebsrat, den externen Stakeholdern wie z. B. den Kunden sowie mit dem Staat, der den gesetzlichen Rahmen festlegt. All diese Interessen müssen berücksichtigt und in Einklang gebracht werden, da Sie einen Einfluss auf die Verwendung der Daten haben. Dies führt unweigerlich zu der Frage, welche Rahmenbedingungen existieren und wie diese eingehalten werden können.

Der einem Unternehmen zugrundeliegende Rahmen wird als Corporate Governance bezeichnet. Er umfasst eben jene Rahmenbedingungen. Diese setzen sich aus den gesetzlichen und den unternehmensinternen Anforderungen und dessen Selbstverständnis zusammen. Diese Grundlage ist dokumentiert und wird durch interne Prüfungen oder auch durch eine externe Auditierung überwacht. Die im Rahmen der Corporate Governance festgelegten Regeln zur Überwachung und Leitung des Unternehmens

müssen ebenfalls bei der Einführung und Nutzung einer Low-Code Plattform berücksichtigt werden. Das betrifft zum einen den Umgang mit den Daten selbst sowie die dahintersteckenden Ziele.

Wird der Fokus auf die Corporate Governance gelegt, sollten die Ziele hinter der Einführung damit übereinstimmen. Das bedeutet auch, dass die Einführung einer Low-Code-Lösung keinen Selbstzweck erfüllen soll. Bei der Nutzung einer solchen Lösung sollte ein Mehrwert für das Unternehmen entstehen. Dieser Mehrwert kann unterschiedlich ausgestaltet sein und unterliegt dem jeweiligen Anwendungsfall. In einem produzierenden Unternehmen kann es vorteilhaft sein, eine leichte und übersichtliche Darstellung von aktuellen Daten zu haben bzw. diese mit weiteren Daten anreichern zu können. Diese Anreicherung kann zum Beispiel ein digitales Maschinenlogbuch sein. Für ein Softwareunternehmen kann durch die Integration einer Low-Code-Plattform ein weiterer Markt eröffnet werden, um den Kunden eine einfache Möglichkeit der Integration anzubieten. Die verfolgten Ziele sowie deren Motivation können unterschiedlicher nicht sein, sollten aber immer mit der Corporate Governance in einem angemessenen Verhältnis stehen, um ein gemeinsames globales Ziel zu erreichen.

Im Hinblick auf die Einführung einer Low-Code Lösung muss vor allem im Blick auf die Corporate Governance und Compliance darauf geachtet werden, dass sich diese in das Gesamtkonzept eines Unternehmens einbetten lässt. Lösungen, die durch engagierte Mitarbeitende eruiert werden, sollten nicht automatisch als Produktivsystem übernommen werden. Unterschiedliche Low-Code-Lösungen zu vergleichen und diese auf deren Einsatzmöglichkeit hin im eigenen Unternehmen zu bewerten, ist ein sinnvolles und rationales Vorgehen. Diese sollten aber in Zusammenarbeit mit der IT-Abteilung auf gesonderten Testumgebungen erfolgen und als solche wahrgenommen bzw. genutzt werden. Das bedeutet:

Zugriff nur für einen gesonderten Personenkreis und Zugriff nur auf solche Daten, die im Rahmen des Tests unbedingt notwendig sind. Andernfalls besteht die Gefahr, dass ein ursprüngliches Testsystem immer mehr Funktionen erhält und sich immer mehr Mitarbeiter an die Funktionen gewöhnen und diese im Unternehmensalltag nutzen. Das kann schnell zu einer Schatten-IT führen, in der ein als Testsystem aufgestelltes System von Mitarbeitern gepflegt und erweitert wird, ohne dass die IT-Abteilung darauf einwirkt. Im Hinblick auf die Einhaltung der definierten Anforderungen im Zuge der Corporate Governance hat man an dieser Stelle ein großes Problem – die Einhaltung ist hier nicht gewährleistet. Aus diesem Grund sollte immer das große Ganze betrachtet werden und eine Strategie zu Grunde gelegt werden, welche den sauberen Übergang von einem Testsystem hin zu einem Produktivsystem regelt.

Mit Blick auf den Umgang mit den Daten selbst, muss bei der Verwendung einer Low-Code-Lösung auf die Einhaltung des Datenschutzes sowie der im Rahmen der Corporate Governance festgelegten Rahmenbedingungen geachtet werden. Dieser Punkt ist zwar nicht ausschließlich auf Low-Code-Lösungen ausgerichtet, betrifft aber generell den Umgang mit Daten im Unternehmen. In diesem Zusammenhang hat es aber einen großen Stellenwert, da die Möglichkeit der Verknüpfung von Daten sehr einfach gegeben ist.

Der Zugriff auf Daten richtet sich nach deren Klassifizierung, die unterschiedlich vorgenommen werden kann – je nachdem aus welchem Blickwinkel diese betrachtet werden. Die Daten können beispielsweise gemäß ISO/IEC 27001 klassifiziert werden, jedoch müssen hierbei die zu verwendeten Kategorien durch das Unternehmen selbst definiert werden.

Wird der Zugriff auf Daten für die Verwendung in einer Low-Code-Lösung im Zusammenspiel mit der Klassifizierung der Daten betrachtet, muss

sichergestellt sein, dass nur berechtigte Personen einen Zugriff auf die Daten haben. Je nach Implementierung des Zugriffs sollte nach dem Minimalitätsprinzip vorgegangen werden. Wird eine Datenbank genutzt, sollte der Benutzende zum Beispiel nur auf die benötigten Tabellen und Spalten zugriffsberechtigt sein. Wird eine REST-Schnittstelle verwendet, kann die Nutzung eines API-Gateways helfen, um an dieser Stelle die Anforderungen zentral durchzusetzen.

Neben dem sicheren und authentifizierten Zugriff auf die Daten, muss bei der Nutzung einer Low-Code-Lösung darauf geachtet werden, wer die Datenhoheit besitzt. Sind die Datenzugriffe eingerichtet, können die Daten aus den konfigurierten Zugriffspunkten in einer Anwendung genutzt und unter Umständen auch bearbeitet werden. Dies ist nicht immer beabsichtigt und kann zu unerwünschten Nebeneffekten führen. So kann eine Anwendung ursprünglich zwar zur Anzeige von Daten gedacht und konzipiert worden sein, aber durch eine Anpassung können sich auch Datenpunkte verändern, weshalb die Nachvollziehbarkeit verloren gehen kann.

3.7. Aktuelle KI-Entwicklungen

Autoren: Wolfram Schäfer und Hien Nguyen, iT Engineering Software Innovations GmbH

Die neuesten Entwicklungen im Bereich der generativen KI-Technologien haben erhebliche Auswirkungen auf die Softwareentwicklung. Sie ermöglichen die Automatisierung verschiedener komplexer Aufgaben in der Softwareentwicklung. Besonders bemerkenswert sind die großen Sprachmodelle, bekannt als Large Language Models (LLMs). Diese werden mit umfangreichen Quellcode- und natürlichsprachlichen Daten trainiert und zeigen überzeugende Fähigkeiten beim Verstehen und Generieren von Code. LLMs erlauben jeder Person, Programmieraufgaben unabhängig von ihrer Erfahrung als Entwickler durchzuführen. Alles, was die Person tun muss, ist dem

Sprachmodell eine textbasierte Anweisung („Prompt“) zu geben. Besonders im Fokus steht dabei die Verbindung zu LC/NC-Plattformen, die bisher das Entwickeln von Anwendungen ohne tiefergehende Programmierkenntnisse ermöglichten. LLMs fungieren gewissermaßen als eine Art Weiterentwicklung dieser Idee. Sie können in LC/NC-Plattformen integriert werden. Dadurch werden diese Plattformen intelligenter.

Die Anfänge der LLM in diesem Zusammenhang sind BERT und GPT. BERT kann den semantischen Inhalt von Texten extrahieren, d.h. es besitzt die Fähigkeit Text zu verstehen. Aus diesem Grund wird BERT oft als Embedding Model (eine Art Encoder) für viele Sprachmodelle verwendet. Im Gegensatz dazu steht GPT, was für Generative Pre-Trained steht. Pre-Trained, oder vortrainiert bedeutet in diesem Kontext, dass das Modell auf einer großen Textmenge trainiert wurde, um die natürliche Sprache zu modellieren (<https://openai.com/research/language-unsupervised>). Dieses Modell kann dann für weitere Problemstellungen weiter verfeinert werden. Dieses Training wird auch als Feintuning (fine-tuning) bezeichnet (<https://openai.com/research/instruction-following>). Es folgten drei weitere GPT-Modelle (GPT-2, GPT-3, GPT-4), die mit noch größeren Datenmengen trainiert wurden (<https://openai.com/research/better-language-models>, <https://openai.com/research/gpt-4>). Weitere Entwicklungen ergaben sich aus dem Training von Programmiersprachen und deren geschriebenem Code, beispielsweise durch die Verwendung von öffentlich zugänglicher Git-Repositories, um sogenannte Code LLMs zu entwickeln. Die Einsatzmöglichkeiten von LLMs, insbesondere in der Softwareentwicklung sind daher vielfältig.

Codegeneration

Die Codegenerierung umfasst die automatische Erstellung von Programmcode oder die Vervollständigung von Code Snippets anhand des Prompts.

Codefixing

Durch die Fähigkeit, Programmcode zu verstehen, können LLMs für das Beheben von Code (Bugfixing) und die Verbesserung von Code (Refactoring) eingesetzt werden.

Zusammenfassen von Code

Ein weiterer Anwendungsfall von LLMs für code-relevante Aufgaben ist das Erstellen klarer und prägnanter Zusammenfassungen oder Erklärungen für einen Teil des Codes oder der Funktionalität. Diese Mechanismen können dann zur automatischen Code-Dokumentation genutzt werden.

Des Weiteren lassen sich LLMs in die von Entwicklerinnen und Entwickler schon bekannten Entwicklerwerkzeugen (IDEs und Texteditoren) als Plugin oder Add-On integrieren. Aktuelle Forschung beschäftigt sich mit der Entwicklung generativer Agenten, die LLMs verwenden, um menschliches Verhalten zu simulieren. Diese Agenten besitzen drei wesentliche Fertigkeiten: ein Gedächtnismodul, das Erinnerungen und Gespräche in natürlicher Sprache aufbewahrt; die Fertigkeit, diese Erinnerungen zusammenzufassen, um Schlussfolgerungen zu ziehen und ihr Verhalten anzupassen; und die Umsetzung der Schlussfolgerungen in Handlungen, die wiederum an das Gedächtnismodul zurückgeführt werden (<https://arxiv.org/pdf/2304.03442.pdf>).

Darüber hinaus wird an der Simulation von virtuellen Softwareunternehmen geforscht, in denen generative Agenten Schlüsselpositionen wie Projektmanager, Entwickler, Designer und QA-Engineers übernehmen. Erste Ergebnisse zeigen, dass allein eine textuelle Softwarespezifikation ausreicht, um erfolgreich einen funktionsfähigen Softwareprototypen mit bedienbarer Oberfläche (GUI) zu entwickeln. Dies wurde durch die Integration externer Schnittstellen und Werkzeuge wie Internetzugriff, Dateiverwaltung und statistische Codeanalysetools erreicht (<https://arxiv.org/pdf/2308.00352.pdf>, <https://arxiv.org/pdf/2307.07924.pdf>). Dennoch haben LLMs auch ihre Grenzen und Herausforderungen. Ein bedeutendes technisches Problem von LLMs ist das Halluzinieren oder Fabrizieren von Informationen. Dies kann dazu führen, dass die Ausgabe des Sprachmodells der Eingabe widerspricht, in sich widersprüchlich ist, Unwahrheiten enthält oder in keinem Zusammenhang mit der Eingabe steht (<https://doi.org/10.1186/s13054-023-04393-x>). Bei der Codegenerierung äußert sich dies in unvollständigen Implementierungen von Funktionen, fehlenden und fehlerhaften Projekt- oder Bibliotheksabhängigkeiten, sowie potenziellen Software-Bugs. Neben den technischen Herausforderungen besteht ein wesentliches Problem von LLMs darin, schnell auf neue Anforderungen reagieren und sich anpassen zu können. Während wissenschaftliche Untersuchungen oft von einem klaren Anforderungsset ausgehen, sind Softwareprojekte in der Realität selten so klar definiert. Daher sind flexible und agile Ansätze, wie z. B. das Scrum-Framework erforderlich, welches iterativ an die Entwicklung herangeht, um flexibel auf sich ändernde und unklare Anforderungen reagieren zu können.

Generative KI und LLMs besitzen das Potenzial, die Arbeitsweise von Menschen zu verändern, indem sie Teile einzelner Aufgaben automatisieren und die Fähigkeiten der einzelnen Mitarbeiterinnen und Mitarbeiter im Unternehmen erweitern. Für Softwareunternehmen bedeutet das, dass

die Programmierung einfacher erlernbar ist und potenziell viele neue Entwicklerinnen und Entwickler hervorgebracht werden können. Dies wirft die Frage auf, ob der Beruf des Entwicklers durch LLMs gefährdet ist, da LLMs die Codegenerierung automatisiert und somit die Effizienz und Fehlerfreiheit der Entwicklung erhöhen (<https://arxiv.org/pdf/2302.06590.pdf>). Allerdings ermöglicht dies auch eine Verschiebung der Entwicklerrolle in andere Bereiche, wie UI und UX-Design sowie die Anforderungsmanagement und Qualitätskontrolle, um eine ganzheitlichere Entwicklungsperspektive zu gewährleisten und die Softwarequalität insgesamt zu verbessern. Des Weiteren werden gute Entwicklerinnen und Entwickler nur begrenzten Nutzen aus LLMs ziehen können. Der Flaschenhals für Entwicklerinnen und Entwickler ist nicht das Tippen und Programmieren, sondern das abstrakte Denken und gute Problemlösungsfähigkeiten, wofür sie keine Vorschläge von LLMs brauchen. Die größte Herausforderung bei der Implementierung neuer Funktionen und der Behebung von Fehlern in einer großen Codebasis ist die Entscheidungsfindung, die daraus besteht, aus einer Vielzahl von Optionen die optimale Vorgehensweise unter Berücksichtigung aller Randbedingungen auszuwählen. Suboptimale Lösungen mögen zunächst unschädlich sein, können aber mit der Zeit zu technischen Schulden und letztendlich zu schlechter Codequalität, langsamer Entwicklung und weiteren Fehlern führen. Erfahrene Entwicklerinnen und Entwickler sind in der Lage, dieses Problem so weit wie möglich zu verzögern oder in Gänze zu vermeiden. Selbst LLMs können diese Herausforderungen über einen längeren Zeitraum nicht wesentlich verringern, da diese Entscheidungen oft auf Kompromissen beruhen und tiefes Domänenverständnis, sowie Kenntnis über mögliche zukünftige Entwicklungsrichtungen der Software erfordern.

4. Anwendungsbeispiele aus dem Maschinen- und Anlagenbau und bei Softwarefirmen

Nach der Erläuterung der theoretischen Aspekte im vorherigen Kapitel 3, bei dem eine Begriffsdefinition und -abgrenzung vorgenommen, die Chancen und Risiken und verschiedenen Ebenen der LC-/NC-Systeme aufgezeigt sowie auf die organisatorischen und personellen Aspekte als auch die neuesten KI-Entwicklungen eingegangen worden ist, werden im folgenden Kapitel 4 Anwendungsbeispiele aus dem Maschinen- und Anlagenbau und bei Softwarefirmen aufgeführt. Diese sollen der Leserin und dem Leser dienen, sich mit der Thematik zu beschäftigen und Ideen beziehungsweise Anregungen für das eigene Unternehmen aufzeigen. Jedes Beispiel ist dabei in die Abschnitte „Ausgangssituation / Zielsetzung“, „Umsetzung“, „Ergebnis“ und „Stakeholder / Rollen im Prozess“ unterteilt, so dass eine bessere Vergleichbarkeit erreicht wird.

4.1. iT Engineering Software Innovations GmbH (intern) – Einsatzmöglichkeiten von LLMs in der eigenen Softwareentwicklung sowie in Kundenprojekten

Autoren: Wolfram Schäfer und Hien Nguyen, iT Engineering Software Innovations GmbH

Sprachmodelle wie ChatGPT oder Bing AI sind Werkzeuge, die transformative Veränderungen in der Arbeitsweise von Menschen ermöglichen. Daher untersucht unsere Firma iT Engineering Software Innovations GmbH intern in sogenannten „Community of Practice für KI“ unter anderem Möglichkeiten, wie solche LLMs effizienzfördernd im Arbeitsalltag von Softwareentwicklerinnen und Softwareentwickler eingesetzt werden können oder deren Einsatzmöglichkeiten in verschiedenen Kundenprojekten. Im Folgenden geben wir einen Einblick über die Verwendung von LLMs zur effizienten Gestaltung der Arbeitsweise unserer Softwareentwicklerinnen und Softwareentwickler sowie zur Schaffung innovativer Lösungen.

4.1.1. Ausgangssituation / Zielsetzung

Softwareentwicklerinnen und Softwareentwickler streben stets nach Einhaltung der Clean-Code-Prinzipien. Daher legen sie großen Wert auf die Lesbarkeit, Wartbarkeit, Effizienz und Konsistenz der Codebasis. Ein Beispiel aus der objektorientierten Softwareentwicklung dient zur Verdeutlichung. Eine Klasse namens „Foo“ enthält eine Member-Variable namens „m_bar“, die berechnet werden soll. Vorab hat die Softwareentwicklung erkannt, dass es zwei Möglichkeiten gibt, die Member-Variable zu berechnen. Eine Möglichkeit besteht darin, den Wert mithilfe einer konstanten Funktion zu setzen, die den entsprechenden berechneten Rückgabewert verwendet. Eine andere Option ist der Einsatz einer nicht-konstanten Funktion, die intern das Feld „m_bar“ setzt und keinen Rückgabewert liefert. Die Frage, die sich hier stellt, ist, welche der beiden Alternativen besser geeignet ist, um den Code leichter verständlich, wartbar und weniger fehleranfällig zu gestalten. Die Entscheidung kann erhebliche Auswirkungen auf die Verständlichkeit des Codes und die zukünftige Wartbarkeit haben. Ein weiterer Anreiz für die Verwendung von LLMs oder Chatbots liegt in der Schwierigkeit, diese Problembeschreibung in eine Suchmaschine einzugeben.

4.1.2. Umsetzung

Zur Beantwortung dieser Frage haben wir Microsofts Bing AI Chat befragt. Bing AI Chat wurde im März 2023 veröffentlicht und ist ohne Benutzeranmeldung zur Benutzung verfügbar. Alternativ gibt es von OpenAI mit GPT3.5 und mit Googles Bard ähnliche Angebote, jedoch nur nach kostenloser Registrierung. Als gute Praxis zur Eingabe von LLMs sind, wie in Kapitel 3.7 beschrieben, textbasierte Anweisungen, die sogenannten Prompts. Prompts sind präzise Anfragen oder Instruktionen, welche zur effizienten Nutzung großer Sprachmodelle eingesetzt werden. Sie tragen entscheidend dazu bei, dass das Modell die gewünschte Aufgabe oder das gewünschte Verhalten präzise umsetzt. Unser Prompt lautete:

I'm asking a question about object-oriented software development. Answer like an expert software developer. Give good reasons for your answer and explain it thoroughly. Assume I have a class called Foo which has a member variable `m_bar`. There are two possibilities: 1) `m_bar` can be computed via a constant function `ComputeBar()` that returns the computed value, i.e. `m_bar = ComputeBar();`. 2) `m_bar` can be computed via a non-constant function that internally sets `m_bar` and `ComputeBar()` has no return value, i.e. `ComputeBar()` sets `m_bar`. Which of the two alternatives is better for code readability and maintainability?

Die Sprache des Prompts definiert die Sprache des Resultats und spielt in den oben genannten LLMs keine Rolle, d.h. Deutsch oder Englisch kann verwendet werden, ohne dass das Ergebnis beeinflusst wird. Eine genauere Analyse unseres Prompts ermöglicht die Identifizierung verschiedener Merkmale.

Zunächst wird die Frage auf den Anwendungsfall einer objektorientierten Entwicklung eingeschränkt. Diese Spezifizierung ermöglicht es dem Modell, gezielt auf die Anforderungen dieses Kontexts einzugehen und präzise Antworten zu generieren.

Anschließend weisen wir dem Modell an, sich wie ein erfahrener Softwareentwickler zu verhalten. Das bedeutet, dass es nicht nur oberflächliche Informationen berücksichtigt, sondern auch fundierte Kenntnisse und Erfahrungen in die Antwort einbringt. Durch diese Maßnahme stellen wir sicher, dass die erzielte Antwort nicht nur korrekt, sondern auch von hoher fachlicher Qualität ist. Zudem sollte die Antwort ausführlich begründet werden können. Im dritten Satz wird daher präzise erklärt, wie das Modell zu der Antwort kommen soll. Dadurch wird Transparenz geschaffen und der Anwender kann den Denkprozess des Modells nachvollziehen. Zuletzt wird dann die Frage mit vorhandenem Wissen und Kontext präzisiert.

Zusammenfassend enthält unser Prompt vier Elemente: die Anweisung (instruction) – eine bestimmte Aufgabe oder Anweisung, die das Modell ausführen soll, Kontext (context) – zusätzliche Informationen, um bessere Antworten zu bekommen, Eingabe (input) – die Fragestellung, Ausgabeindikator – die Art oder das Format der Ausgabe.

4.1.3. Ergebnis

Die Ergebnisse von Bing AI Chat sind äußerst positiv. Das Modell liefert eine detaillierte Antwort auf die gestellte Frage und zeigt durch das Wiederholen der Frage „in eigenen Worten“ ein Verständnis für das Problem. Die Antwort betont die Bedeutung von Lesbarkeit und Wartbarkeit in der Codebasis und schließt daraus, dass die Verwendung einer konstanten Funktion die bessere Option sei. Das Modell lieferte nicht nur eine klare Empfehlung, sondern erklärte auch die Gründe dahinter, was dazu beiträgt, dass die Entscheidung besser nachvollzogen werden kann. Dies trägt erheblich dazu bei, dass der Code leichter verständlich, wartbarer und weniger fehleranfällig ist, was letztendlich das Ziel von Softwareentwicklern ist. LLMs bieten einen besonderen Nutzen, da sie komplexe Probleme verstehen und Antworten liefern können, ohne dass aufwendige Suchanfragen in Suchmaschinen notwendig sind.

Abgesehen davon, dass es sich bei „`ComputeBar()`“ um eine Memberfunktion handelt und die Aufrufsyntax inkorrekt ist, ist es empfehlenswert, Antworten von LLMs kritisch zu hinterfragen. In der Programmiersprache C++ werden konstante Funktionen korrekterweise als „const member functions“ bezeichnet. In der Antwort sollte klar gestellt werden, dass solche Funktionen hauptsächlich dazu dienen, Unveränderlichkeit von Klassenmitgliedern anzuzeigen und nicht nur, um berechnete Werte zurückzugeben. Die Unterscheidung zwischen konstanten und nicht-konstanten Funktionen wird vereinfacht, indem impliziert wird, dass konstante Funktionen Werte zurückgeben, während dies bei nicht-konstanten Funktionen nicht der Fall ist. Es ist jedoch zu betonen, dass beide Arten von Funktionen Werte zurückgeben oder interne Zustände verändern können. Bei const member functions kann dies durch das Schlüsselwort „mutable“ erreicht werden.

In object-oriented software development, it is important to ensure that the code is **readable** and **maintainable**. In the given scenario, we have a class called `Foo` with a member variable `m_bar`. There are two possibilities to compute `m_bar`:

1. `m_bar` can be computed via a **constant function** `ComputeBar()` that returns the computed value, i.e., `m_bar = ComputeBar();`.
2. `m_bar` can be computed via a **non-constant function** that internally sets `m_bar`, and `ComputeBar()` has no return value, i.e., `ComputeBar()` sets `m_bar`.

In terms of code readability and maintainability, it is better to use the **constant function** approach to compute `m_bar`. Here's why:

- A **constant function** is a member function that promises not to modify any data members of the class. This promise is enforced by the compiler, which will generate an error if the function attempts to modify any data members ¹.
- Using a **constant function** to compute `m_bar` makes it clear that the value of `m_bar` is computed from some other data members of the class and not modified by any other functions.
- This approach also makes it easier to reason about the behavior of the class since we know that calling `ComputeBar()` will not modify any data members of the class.
- On the other hand, using a **non-constant function** to compute `m_bar` can lead to confusion since it is not immediately clear whether calling `ComputeBar()` will modify any other data members of the class.
- Additionally, using a **non-constant function** can make it harder to reason about the behavior of the class since we need to consider all functions that might modify `m_bar`.

Therefore, using a **constant function** to compute `m_bar` is better for code readability and maintainability.

I hope this helps!

Zusammenfassend lässt sich sagen, dass die Antwort vorschlägt, aus Gründen der Klarheit und Wartbarkeit, konstante Funktionen zu bevorzugen. Allerdings ist das Konzept zu einfach und berücksichtigt nicht alle möglichen Szenarien, in denen nichtkonstante Funktionen geeignet sein könnten. Die Wahl zwischen den beiden Ansätzen hängt von den spezifischen Anforderungen der Klasse und den Entwurfszielen ab und kann nicht vom Sprachmodell übernommen werden. Die Entscheidung erfordert die Expertise des Softwareentwicklers, wie im letzten Absatz von Kapitel 3.7 erläutert wird.

4.1.4. Stakeholder / Rollen im Prozess

Die Verwendung von Sprachmodellen erfordert keine speziellen Vorkenntnisse und ermöglicht es allen Personen, auf unkomplizierte Weise auf Informationen in der Softwareentwicklung zuzugreifen.

Um allerdings die weiter oben erwähnten Missverständnisse zu identifizieren, sind erfahrene Entwickler oder Personen erforderlich, die sich bestens mit der verwendeten Programmiersprache oder den Entwicklungszielen der Software auskennen.

4.2. CAS Software AG + ASYS Group – Effiziente Nutzung von No-Code / Low-Code im Maschinenbau

Autoren: Dr. Thomas Genßler und Lukas Schweizer, CAS Software AG

4.2.1. Ausgangssituation / Zielsetzung

Die Firma ASYS ist ein deutsches Maschinenbauunternehmen mit ca. 1.500 Mitarbeitern an 27 Standorten mit Sitz in Dornstadt. ASYS produziert u.a. Systeme für die Elektronik- und Leiterplattenfertigung, sowie Automatisierungstechnik im Bereich Life Science. Weltweit sind über 100.000 Maschinen von ASYS im Einsatz.

Das Unternehmen sah sich aufgrund ihres umfangreichen Produktportfolios vor Herausforderungen im Vertriebsprozess. Die Vielzahl der Produkte und ihre Komplexität erschwerte die Angebotserstellung durch die Vertriebsmitarbeiter, da diese sehr wenig auf einzelne Produkte spezialisiert sind, sondern das gesamte Produktportfolio vertreiben. Das Wissen zu den Produkten lag zum großen Teil in den Köpfen der Produktmanager und war nicht digitalisiert. Dies führte zu sehr langen Angebotszeiten und einem hohen Kommunikationsaufwand zwischen Vertrieb, Innendienst, Produktmanagement und Engineering. Des Weiteren war die Fehlerquote im

Angebotsprozess insgesamt sehr hoch. Darüber hinaus traten auch nach dem Angebotsprozess viele Fehler auf, was zu sehr hohen Änderungskosten führte.

Die von der Firma ASYS verfolgten Ziele waren daher die deutliche Reduktion der Fehlerhäufigkeit und des Abstimmungsaufwandes im Angebotsprozess und damit eine signifikante Steigerung der Skalierfähigkeit und Effizienz im Produktmanagement sowie im Vertrieb. Dabei wurden die folgenden Teilziele und Maßnahmen identifiziert:

- Digitalisierung des Produktwissens (Produktvarianten, Ausstattungsmerkmale, Baubarkeitsregeln, Plausibilitäten/Kardinalitäten, Preise, Vertriebsargumente etc.) durch die Produktmanager mittels eines grafischen No-Code-Ansatzes.
- Nahtlose Nutzung des digitalisierten Produktwissens in den Vertriebssystemen durch die Vertriebsmitarbeiter bzw. perspektivisch durch Endkunden selbst.
- Und damit gleichzeitig die Vermeidung regelmäßiger kosten- und zeitintensiver IT-Anpassungen bei Veränderungen in der Produktstruktur (bspw. neue Produkte, neue Ausstattungsmerkmale, zeitlich befristete Aktionen, ...).

4.2.2. Umsetzung

ASYS entschied sich für den Einsatz des No-Code-basierten Produktkonfigurators „Merlin CPQ“ der CAS Software AG. Merlin CPQ ist ein umfassendes Produktkonfigurationssystem, bestehend aus einer grafischen No-Code-Modellierungsumgebung, adaptive, sich automatisch an die Modellierung anpassende Angebotskonfiguratoren für Vertriebsmitarbeiter und Endkunden inkl. umfassender Werkzeuge zur Angebotskalkulation und Dokumentenerzeugung für Angebote, Stücklisten und Ähnliches. Des Weiteren bietet Merlin CPQ ein integriertes Kundenportal zur kollaborativen Angebotserstellung, sowie eine Reihe von

Unterstützungswerkzeugen wie eine Testumgebung für die Modellierung, einen grafischen Variantenbrowser, Empfehlungsassistenten für die Nutzer oder Schnittstellensystemen zur Integration in bestehende IT-Landschaften.



Abb. 5: Sales Journey mit einem Produktkonfigurator

ASYS integrierte den Produktkonfigurator in das vom Vertrieb bereits genutzte CRM-System. Zunächst wurden die verkaufsfähigen Artikel nach Produktlinien und -gruppen klassifiziert. Die ERP-Datenbasis wurde dazu erweitert, um diese Klassifizierung zu unterstützen. Die vertriebliche Regelwelt, sowie weitere vertriebsrelevante Aspekte (wie Produktinformationen, Preise, Aktionen) wurden anschließend über die grafische No-Code-Modellierungsumgebung des Konfigurators abgebildet.

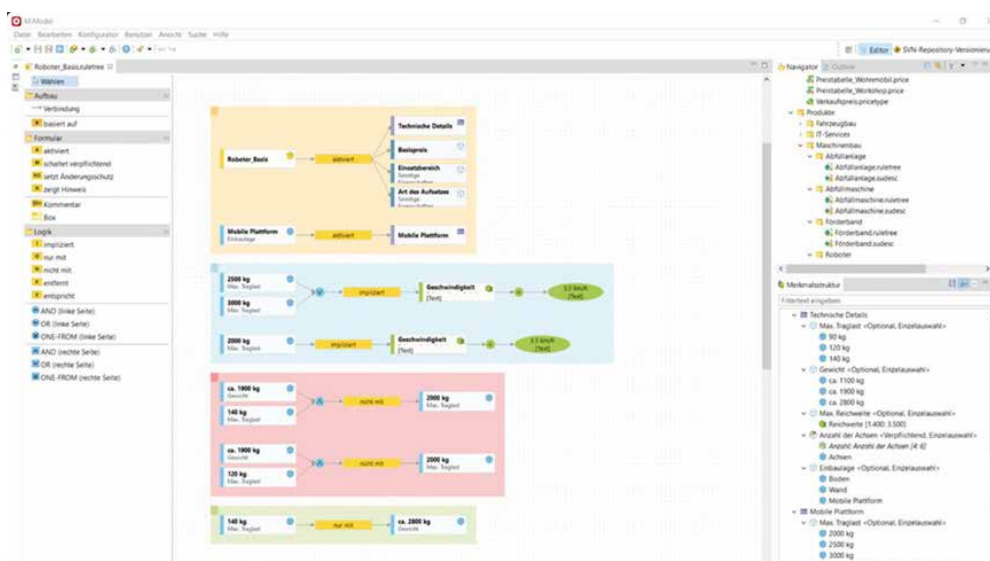


Abb. 6: Die grafische Modellierungsumgebung auf No-Code Basis

Über die No-Code-Modellierungsumgebung lassen sich praktisch alle produktfachlichen, vertriebslichen und prozessualen Aspekte des Konfigurators grafisch anpassen. Dies umfasst neben Produktstruktur und Baubarkeitsregeln auch Preise, Verkaufsargumente, Produktinformationen, Dokumententemplates für Angebote etc. und sogar den eigentlichen Configurationsworkflow inklusive der dazu gehörigen Nutzeroberfläche.

4.2.3. Ergebnis

Merlin CPQ wird heute in der kompletten ASYS Organisation weltweit zur Konfiguration von Produkten im Vertriebsprozess benutzt. Dabei sind im Konfigurator alle relevanten Produkte abgebildet, was in etwa 90 % des gesamten Produktportfolios entspricht. Mit 120 Nutzern und etwa 1.400 Angeboten pro Monat stellt die Einführung des Konfigurators eine erhebliche Effizienzsteigerung im Vertriebsprozess dar. Des Weiteren wurde die Anzahl an durchgeführten Angeboten erheblich gesteigert und das bei einer deutlich reduzierten Fehlerquote im Angebotsprozess und auch danach. Gleichzeitig konnten durch den No-Code-Ansatz eine hohe Flexibilität und Adaptivität erreicht werden und die Folgekosten durch sonst notwendige IT-Anpassungen bei fachlichen Änderungen in der Produktwelt deutlich begrenzt werden.

4.2.4. Stakeholder / Rollen im Prozess

Verschiedene Stakeholder und Rollen sind in den Prozess eingebunden. Das Produktmanagement

liefert die erforderlichen Daten und pflegt das vertriebsliche Beziehungswissen im Produktkonfigurator per No-Code-Ansatz in der grafischen Modellierungsumgebung. Die Mitarbeitenden des Produktmanagements schlüpfen damit in die Rolle der „Citizen Developer“. Der Vertrieb und der Innendienst sind die eigentlichen Hauptanwender des Konfigurators. Der Vertrieb nutzt den Konfigurator zur Angebotserstellung im Dialog mit dem Kunden. Der Innendienst ändert bei Bedarf vorher erstellte Konfigurationen in Abstimmung mit dem Engineering. Das Engineering gibt Feedback zum Configurationsergebnis und unterstützt das Produktmanagement bei der fachlichen Weiterentwicklung der Modellierung. Die IT kümmert sich schließlich um den reibungslosen Betrieb der Gesamtlösung, ist aber bei fachlichen Anpassungen nicht involviert.

Zusammenfassung

Die ASYS Group realisierte durch die Einführung eines Low-Code-Konfigurators signifikante Verbesserungen in ihrem Vertriebsprozess. Die Digitalisierung des Wissens, die Unterstützung des Vertriebs und die Reduktion von Angebotszeiten führten zu einer erfolgreichen Transformation. Die enge Zusammenarbeit der verschiedenen Stakeholder und die agile Weiterentwicklung trugen zum Gesamterfolg des Projekts bei. Dieses Fallbeispiel verdeutlicht, wie der Einsatz von No-Code / Low-Code im Maschinenbau zu effizienteren Abläufen und besseren Ergebnissen führen kann.

4.3. SAERTEX GmbH & Co. KG + Simplifier AG – Kompatibilität und Mobilfähigkeit durch Low-Code

Autor: Karl Haag, Simplifier AG

4.3.1. Ausgangssituation / Zielsetzung

SAERTEX, ein Unternehmen in der Verbundwerkstoffbranche, stand vor einer Reihe an Herausforderungen in seinen Produktionsprozessen. Diese umfassten papierbasierte Prozesse und veraltete Legacy-Systeme, die dringend ersetzt werden mussten. Ein besonderes Problem war der Mangel an qualifizierten SAP-Fachkräften, was die Implementierung von Anwendungen und den Übergang zum Clean-Core-Ansatz behinderte, der aufgrund einer Paradigmenänderung von SAP erforderlich war. Das hybride Team von Front-End- und Back-End-Entwicklern bei SAERTEX hatte bereits Erfahrungen mit Low-Code-Technologien gesammelt. Allerdings erfüllten die bisher getesteten Plattformen nicht die Anforderungen an mobile Funktionalität und nahtlose Integration mit SAP, SAP Sales Cloud und O365. Diese Anforderungen, gepaart mit dem Wunsch nach moderner Technologie und effizienter Zusammenarbeit, führten dazu, dass SAERTEX nach einer neuen Low-Code-Plattform suchte.

Zielsetzung:

Das Ziel war es, die internen Logistikprozesse mithilfe der Simplifier-Plattform zu optimieren, um Fehler zu reduzieren, die Effizienz zu steigern und die Zusammenarbeit zu verbessern.

Im Folgenden wird detailliert der optimierte Prozess "digitale Verladekontrolle" betrachtet. Die Herausforderungen bestanden darin, fehlerhafte manuelle Zuordnungen von Komponenten zu Transporten, unvollständige Verladungen von Transporten und die fehlerhafte oder unvollständige Erfassung von Details durch Medienbrüche zu überwinden.

4.3.2. Umsetzung

Um die Herausforderungen zu bewältigen, wurde eine digitale Anwendung entwickelt, die die folgenden Hauptfunktionen umfasst: Verplanen, Verladen, Gate und Info. Diese Funktionen zielen darauf ab, den Verladeprozess zu optimieren und Echtzeitinformationen bereitzustellen. Die Anwendung integrierte nahtlos mit SAP, SAP Sales Cloud und O365, um die bestehenden Systeme zu unterstützen.

Beschreibung der Applikation:

Die Verladekontroll-App von SAERTEX ermöglicht den Mitarbeitern, den Verlade- und Versandprozess effizient und transparent abzuwickeln. Jede Funktion dient einem spezifischen Zweck:

- (1) Verplanen: Ermöglicht das Verfolgen und Planen der anstehenden Transporte in Echtzeit basierend auf SAP-Transporten.
- (2) Verladen: Unterstützt den Verladeprozess durch Bereitstellung relevanter Transportinformationen und ermöglicht die Dokumentation mit Fotos.
- (3) Gate: Dient der Fehlerprävention bei der Verladung und verwendet QR-Codes zur Freigabe der Lagereinheiten.
- (4) Info: Zeigt alle Transporte mit Status und relevanten Details an und ermöglicht den Abschluss des Vorgangs.

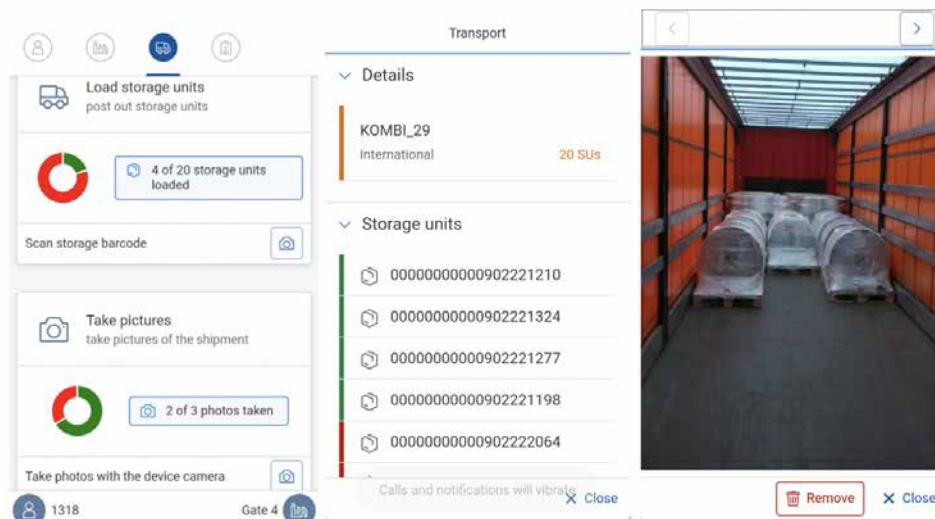


Abb. 7: App-Ausschnitt digitale Verladekontrolle (Quelle: <https://simplifier.io/applications/verladekontrolle/>)

4.3.3. Ergebnis

Durch die produktive Partnerschaft mit Simplifier konnte SAERTEX erfolgreich Herausforderungen wie Side-by-Side Development bewältigen. Es wurden Konzepte für Cybersicherheit, Applikationsentwicklung und Clean Core entwickelt. SAERTEX konnte durch nahtlose Integration mit SAP und den bestehenden Systemen auf dem Shop Floor und dem Top Floor optimieren. Die Zusammenarbeit resultierte in nachhaltigen Lösungen und wiederverwendbaren Schnittstellen, was zu erheblichen Ressourceneinsparungen und beschleunigten Projektabschlüssen führte. Die Anwendung reduzierte Fehler bei der Abwicklung um 97 %, was zu einer jährlichen Einsparung von einem höheren fünfstelligen Betrag und einer Zeitersparnis von neun Tagen pro Monat durch die digitale Abwicklung führte.

4.3.4. Stakeholder / Rollen im Prozess

Die am Prozess beteiligten Stakeholder und ihre Rollen waren vielfältig. Das SAERTEX-Team bestehend aus Front- und Backend-Entwicklern war für die Identifizierung der Anforderungen verantwortlich und führte die Umsetzung der Low-Code-Lösungen durch. Die Experten von Simplifier unterstützten bei Auswahl, Konfiguration und Implementierung der Low-Code-Lösung und brachten ihre technische Expertise ein. Die IT- und SAP-Experten gewährleisteten die Integration der neuen Lösungen in das bestehende SAP-System und sorgten für eine reibungslose Zusammenarbeit. Die Produktionsmitarbeiter waren Hauptnutzer der implementierten Lösungen und spielten eine zentrale Rolle bei der Validierung der neuen Prozesse. Die Führungsebene von SAERTEX trug die Verantwortung für die strategische Ausrichtung des Projekts, die Freigabe von Ressourcen und die Projektsteuerung. Diese Fallstudie verdeutlicht, wie eine effektive Partnerschaft und die Nutzung moderner Technologie zu erheblichen Verbesserungen in den Produktionsprozessen führen können.

4.4. The MathWorks GmbH – Low-Fault Detection Workflow

Autor: Dr. Rainer Mümmeler, The MathWorks GmbH

4.4.1. Ausgangssituation / Zielsetzung

Die Ausgangssituation beinhaltete die Fehlererkennung in einem Drehstrom-Asynchronmotor, eine Aufgabe, die normalerweise detaillierte Kenntnisse über den Betrieb des Motors und ein hohes Maß an technischer Fachkenntnis erfordert. Das Ziel war es, diesen Prozess durch die Verwendung von Low-Code-Tools zu vereinfachen, um Merkmale aus den elektrischen und Schwingungssignalen des Motors zu extrahieren. Diese Merkmale könnten dann verwendet werden, um die Anzahl der gebrochenen Rotorstäbe im Motor zu identifizieren, um den Fehlererkennungsprozess effektiv zu automatisieren.

4.4.2. Umsetzung

Der Umsetzungsprozess wurde durch die Verwendung von Low-Code-Tools erheblich vereinfacht. Zunächst wurde der Versuchsaufbau eines Drehstrom-Asynchronmotor eingerichtet. Elektrische und Schwingungssignale wurden unter verschiedenen Bedingungen vom System erfasst. Der Datenvorverarbeitungsschritt, der oft ein komplexer und zeitaufwändiger Teil jeder Datenanalyseaufgabe ist, wurde auch durch die Verwendung von Low-Code-Tools optimiert. Der Datensatz wurde in einzelne Ensemble-Mitgliedsdatendateien für alle Kombinationen von Gesundheitszustand, Lastdrehmoment und Experimentindex umgewandelt.

Eine weitere Low-Code-App zur Extraktion von Merkmalen wurde dann verwendet, um diese vorverarbeiteten Daten zu nutzen. Dieses Low-Code-Tool ermöglichte eine komplexe Datenverarbeitung und Merkmalsextraktion ohne die Notwendigkeit umfangreichen Codierens (siehe Abb. 8).

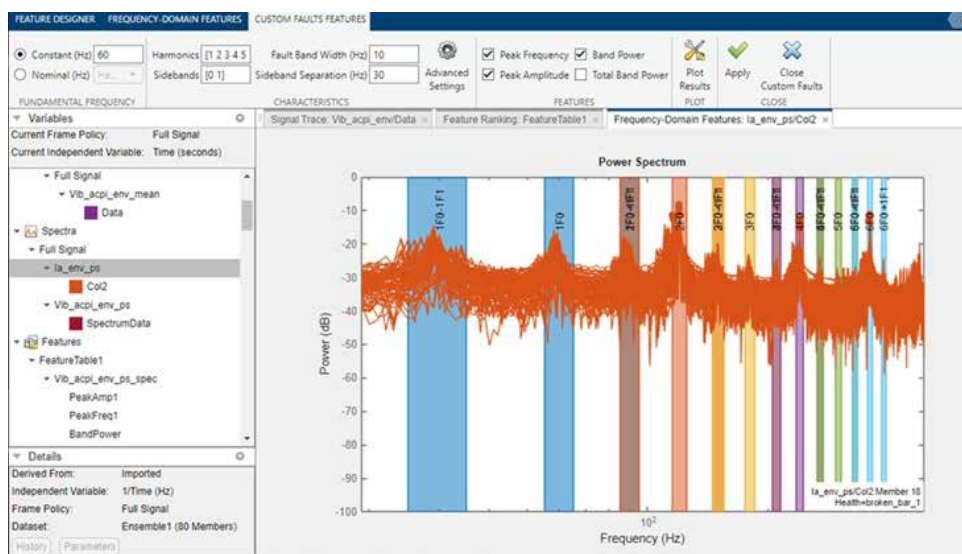


Abb. 8: Low-Code-App zur automatischen Erstellung von Merkmalen

4.4.3. Ergebnis

Das Ergebnis der Umsetzung war eine große Anzahl von Merkmalen, die zur Fehlererkennung im Drehstrom-Asynchronmotor verwendet werden konnten. Diese Merkmale wurden nach ihrer Effektivität bewertet und die ersten 10-15 Merkmale erwiesen sich als sehr effektiv bei der Klassifizierung der Motoren nach der Anzahl der gebrochenen Rotorstäbe.

Darüber hinaus wurde die Low-Code-App für die Klassifizierung der Zustände verwendet, um ein Vorhersagemodell auf Basis dieser Merkmale zu erstellen. Dieses Modell erreichte eine Genauigkeit von etwa 94 % bei der Klassifizierung der Motorfehler. Es wurde auch ein gewichtetes KNN-Modell erstellt, das gebrochene Rotorfehler mit etwa 98 % Genauigkeit klassifizieren konnte (siehe Abb. 9).

Wichtig ist, dass das Vorhersagemodell verwendet wurde, um automatisch Code zu generieren, der auf einem Embedded Device oder Edge-Gerät verwendet werden konnte. Dies ermöglicht eine Echtzeit-Fehlererkennung im Motor, was die Effizienz und Wirksamkeit des Fehlererkennungsprozesses erheblich verbessert.

4.4.4. Stakeholder / Rollen im Prozess

Die Hauptakteure in diesem Prozess sind die Ingenieure und Datenwissenschaftler, die die experimentelle Werkbank einrichteten, die Daten sammelten, die Low-Code-Tools zur Vorverarbeitung der Daten verwendeten und die Vorhersagemodelle erstellten. Die Eigentümer und Betreiber der Drehstrom-Asynchronmotoren profitieren auch von dieser Arbeit, da die Low-Code-Tools eine vereinfachte und effiziente Methode zur Fehlererkennung und -diagnose bieten.

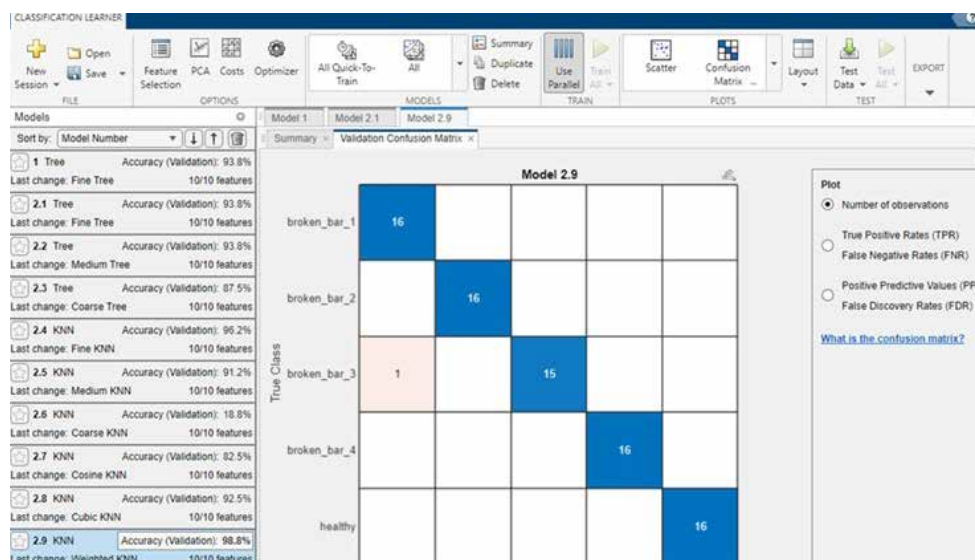


Abb. 9: Low-Code-App zu Klassifizierung

Zusammenfassung

Zusammenfassend hat dieses Projekt die Wirksamkeit von Low-Code-Tools bei der Vereinfachung und Automatisierung des Fehlererkennungsprozesses in Drehstrom-Asynchronmotoren demonstriert. Durch die Verwendung dieser Tools für die Datenvorverarbeitung, Merkmalsextraktion und Modellerstellung wurde eine hohe Genauigkeit bei der Fehlererkennung erreicht. Darüber hinaus wurde das Vorhersagemodell verwendet, um automatisch Code für die Echtzeit-Fehlererkennung auf einem Embedded Device oder Edge-Gerät zu generieren, was einen bedeutenden Fortschritt bei der Fehlererkennung und -diagnose darstellt und diese zugänglicher und effizienter macht.

4.5. GAL Digital GmbH (intern) – Einsatz von generativer KI im eigenen Unternehmen

Autor: Daniel Gal, GAL Digital GmbH

4.5.1. Ausgangssituation / Zielsetzung

Bei GAL Digital haben wir bereits im Jahr 2022 einen Schwerpunkt auf die Anwendung von generativer künstlicher Intelligenz (KI) in unserer Arbeit gelegt. Dabei haben wir auch den Bedarf an datenschutzkonformen Lösungen erkannt und versucht diese in unsere tägliche Arbeit zu integrieren. Unsere bisherigen Erfahrungen haben gezeigt, dass der Einsatz von KI-Systemen, insbesondere in der Softwareentwicklung, weit davon entfernt ist, Entwicklerinnen und Entwickler überflüssig zu machen. Im Gegenteil, wir haben erkannt, dass KI ein wertvolles Werkzeug sein kann, das die Effizienz von Entwicklungsprozessen steigert und gleichzeitig die Qualität der Ergebnisse verbessert – wie der Taschenrechner in der Schule.

4.5.2. Umsetzung

Zur Umsetzung haben wir ein eigenes Produkt, *nele.ai*, erstellt und in unsere tägliche Arbeit integriert. Wir haben in der Umsetzung darauf geachtet, dass wir schnell und genau auf Änderung im sehr volatilen KI-Market reagieren können, indem wir eine sehr generische Lösung für alle Abteilungen schaffen und die einfache Integration neuer und verschiedener KI-Modelle berücksichtigen.

Bei der Umsetzung haben wir von Beginn an alle Planungen und Strategien mit unserem Datenschutzbeauftragten besprochen, so dass wir durch entsprechende Verträge und technische Maßnahmen die Datensicherheit sowie den Datenschutz gewährleisten können.

Partnerschaften mit Technologieanbietern, kontinuierliche Forschung und Entwicklung, und eine nachhaltige Anpassung unserer Services an die sich verändernden Kundenbedürfnisse und den Markt sind dabei integraler Bestandteil unsere Umsetzung. KI hilft uns bei der Dokumentation von Quellcode, bei der Analyse und Aufbereitung von Briefings, bei der Ticket-Erstellung und im Bereich Requirements Engineering. Der automatisierte Prozess spart uns wertvolle Zeit und erhöht die Konsistenz und Genauigkeit unserer Arbeit. Darüber hinaus unterstützt die KI auch die Prüfung auf grobe Sicherheitslücken und erstellt wertvolle Berichte für die Qualitätssicherung, z. B. Bug Reports.

4.5.3. Ergebnis

Durch die Integration von KI in unsere Prozesse konnten wir wertvolle Zeit sparen und die Konsistenz und Genauigkeit unserer Arbeit erhöhen. Die Verwendung von KI zur Prüfung auf Sicherheitslücken, teilautomatisierte Beantwortung von E-Mails und Erstellung von Berichten für die Qualitätssicherung hat sich als besonders nütz-

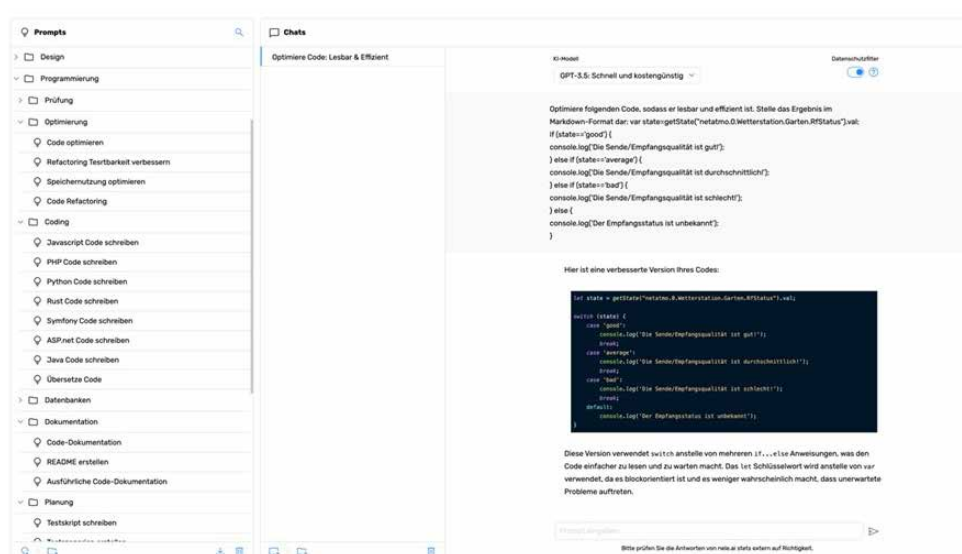


Abb. 10: Beispiel für Optimierung von Quellcode

lich erwiesen. Indem unsere Junior-Entwickler mit der KI arbeiten, verbessern sie ihre Fähigkeiten und lernen effektiver. Es geht nicht darum, Entwickler zu ersetzen, sondern ihre Fähigkeiten und ihr Wissen zu ergänzen und zu erweitern. Ein guter Entwickler wird durch den Einsatz von KI zu einem noch besseren Entwickler, während ein weniger erfahrener oder weniger guter Entwickler nicht automatisch zu einem guten Entwickler wird. KI ist ein Werkzeug, keine Zauberformel.

4.5.4. Stakeholder / Rollen im Prozess

Die Geschäftsführung hat zu Beginn allen Mitarbeitenden in regelmäßigen Meetings angeboten, an der Nutzung und Ideenfindung für den Einsatz von generativer KI im Unternehmen teilzunehmen. Dabei kamen 30 Akteure zusammen, die Ideen und Szenarien durchdachten. Die Hauptakteure in diesem Prozess sind unsere Entwickler, die die generative KI konfigurieren, anpassen und weiterentwickeln und dabei wertvolle Fähigkeiten und Kenntnisse erwerben.

Zusammenfassung

Zusammenfassend können wir feststellen, dass die generative KI ein echter Game-Changer in der Softwareentwicklung ist. Sie ermöglicht es uns, effizienter zu arbeiten, unsere Prozesse zu verbessern und die Qualität unserer Arbeit zu steigern. Aber es gilt auch: KI ist kein Ersatz für gute Entwickler. Im Gegenteil: Sie braucht gute Entwickler, um ihr volles Potenzial entfalten zu können. Entwickler werden also keineswegs überflüssig, sondern in ihrer Rolle aufgewertet und weiterentwickelt.

4.6. valantic GmbH + YAK MAT - Bewegliches Inventar immer im Blick: YAK MAT vereinfacht ERP-Prozesse mit OutSystems und valantic

Autor: Malte Limbrock, valantic GmbH

YAK MAT wurde 1976 gegründet und ist Nord-amerikas größter Lieferant sofort versandfertiger Schutzmatte für Bauprojekte. Diese sind überall in den USA und stets in unmittelbarer Nähe zu jedem Einsatzort verfügbar. Um weiterhin erfolgreich zu bleiben, benötigte YAK MAT ein neues ERP-System, welches es erlaubt, das umfangreiche Inventar zu koordinieren, da dieses sich über viele Lagerhallen erstreckt und ständig am richtigen Einsatzort verfügbar sein muss. In Zusammenarbeit mit valantic und auf Basis der Low-Code-Plattform OutSystems wurde ein maßgeschneidertes ERP mit allen notwendigen Integrations- und Automatisierungsfunktionen entwickelt.

Die Folge: Mehr Transparenz und eine gesteigerte Effizienz für YAK MAT sowie für seine Kunden und Lieferanten.

4.6.1. Ausgangssituation / Zielsetzung

Um einen Bestand aus Hunderttausenden von Schutzmatte effizient zu verwalten, die sich ständig in ganz Nordamerika bewegen, braucht es ein hochspezialisiertes ERP-System. Die Altsysteme von YAK MAT waren diesbezüglich nicht mehr zeitgemäß und mussten ersetzt werden. Das Problem: Alle relevanten Prozesse rund um das mobile Inventar von YAK MAT in einem System abzubilden, bedeutet umfangreiche Anpassungen für jede ERP-Lösung. Das Unternehmen benötigte eine Automatisierung, um den Prozess zu optimieren, sowie vereinfachte Suchfunktionen, um sicherzustellen, dass Mitarbeitende sowohl im Büro als auch im Außendienst effizient arbeiten konnten. Hierfür musste das neue ERP-System erweitert werden, so dass es mobile Inventar-Apps über APIs unterstützt und spezielle Funktionen für die Verwaltung des Versands und der Verteilung der Produkte bereitstellt.

4.6.2. Umsetzung

YAK MAT beauftragte den OutSystems-Partner valantic mit der Entwicklung einer Tracking-Lösung und mobiler APIs, die ein individuelles Front-End für das Infor M3 ERP liefern. Das neue Front-End optimiert die Verwaltung des Bestands und ermöglicht es, neue Aufträge zwischen mehreren Lagerhallen zu koordinieren, um allen Kundenanfragen gerecht zu werden. Das Projekt wurde von drei Entwicklern durchgeführt und dauerte von der Idee bis zum Go-Live insgesamt knapp eineinhalb Jahre. Infolge des erfolgreichen Initialprojekts wurde die Lösung bald durch ein Lieferantenportal ergänzt.

4.6.3. Ergebnis

Dank der Vereinfachung komplexer Prozesse mithilfe der Tracking-Applikation konnte das Unternehmen weiterwachsen. Der YAK MAT Tracker vereinfacht heute die Verfolgung des Lagerbestands, die Erstellung von Lieferungen und die Information der Kunden über den Auftragsstatus. Letzteres ist besonders wichtig, wenn Kunden große Mengen von Matte bestellen, welche in Tausenden von Lkw-Ladungen geliefert werden. Die Lösung vereinfacht auch die Bedienung des Systems durch den End-User, so dass dieser sich auf seine Aufgaben im Außendienst konzentrieren kann, während er mehr Daten erfasst. Das System hat heute 250 aktive Nutzer, die sowohl vom Büro als auch im Außendienst auf die Applikation zugreifen.

4.6.4. Stakeholder / Rollen im Prozess

Innerhalb des Low-Code-Projekts waren die Rollen in ein Engagement Team sowie ein Delivery Team unterteilt. Diese setzten sich wie folgt zusammen:

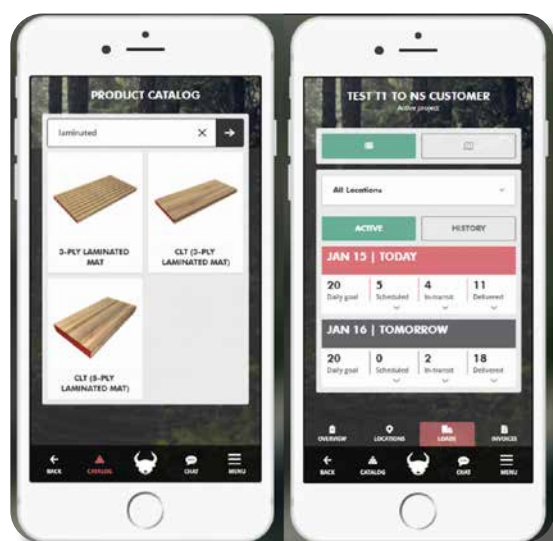


Abb. 11: Screenshots von YAK MAT zur Illustration

Engagement-Team:

- Business Sponsor (YAK MAT)
- Project Manager (YAK MAT) Business Users/ Testers (YAK MAT)
- Business Analysts (YAK MAT / valantic)
- Engagement Manager (valantic)

Delivery Team:

- Technical Lead (valantic)
- 1-3 Developers (valantic)
- UX-Designer (valantic)
- UI-Front End Developer (valantic)
- Quality Assurance Engineer (valantic)

Zusammenfassung

Das Ergebnis des Projektes mit valantic und die Nutzung der Low-Code-Plattform von OutSystems haben die Erwartungen von YAK MAT übertroffen. Das weitere Wachstum des Unternehmens wird heute erfolgreich unterstützt durch eine Applikation, welche sowohl die Prozesse im gesamten Unternehmen als auch die Zusammenarbeit mit Lieferanten und Endkunden effizienter macht (weiterführende Informationen unter: <https://www.valantic.com/de/digital-now/win-win-low-code-plus-ai-bringt-bis-zu-80-prozent-zeitgewinn/> und <https://www.valantic.com/de/low-code/>)

4.7. Optimierung eines Montage-Prozesses mit SAP Build Apps

Autor: Kay Jeschke, SAP Deutschland SE & Co. KG

4.7.1. Ausgangssituation / Zielsetzung

Produktionsmitarbeiter in mehreren Werken eines Maschinenbauunternehmens müssen komplexe Montageanleitungen für moderne variantenreiche Produkte befolgen und dokumentieren. Die Produktivität der Mitarbeiter wurde dadurch ausgebremst, da Informationen von mehreren Standorten und Systemen abrufen werden mussten. Ein geführtes Montagesystem stand nicht zur Verfügung, so dass wichtige Prozessschritte übersehen wurden, was zu Fehlern führte.

Um die Produktionseffizienz und -qualität zu verbessern, wollte das Unternehmen die Zeit, die die Mitarbeitenden in der Fertigung mit der Suche nach Informationen verbringen, reduzieren und die Fehlerrate durch nicht vollständige Informationen ebenfalls verringern. Deshalb entschied sich das Unternehmen für die Einführung einer Mitarbeiterführungs-App, die den Personen in der Produktion alle notwendigen Schritte und Informationen zur Erledigung ihrer offenen Arbeitsaufträge anzeigt.

4.7.2. Umsetzung

Das Unternehmen erkundete mehrere Möglichkeiten zur Erstellung einer Mitarbeiterführungs-App. Zunächst erwog das Unternehmen, eine eigene Inhouse-Lösung zu entwickeln, entschied sich jedoch aufgrund der Kritikalität der App für eine risikoärmere Option. Das Unternehmen betrachtete auch eigenständige Lösungen von verschiedenen Anbietern. Eine der wesentlichen Anforderungen war jedoch eine Lösung, die sich eng mit der bestehenden IT-Landschaft, einschließlich der SAP ERP-Anwendung, integrieren lässt. Nach einem gründlichen Proof-of-Concept-Pro-

zess entschied sich das Unternehmen, seine eigene App mit SAP Build Apps, einem Teil der SAP Business Technology Plattform (SAP BTP), zu erstellen.

Die erste Version der neuen Mitarbeiterführungs-App wurde in nur zehn Tagen, ein Zehntel der Zeit, die es gebraucht hätte, um die App von Grund auf neu zu codieren, erstellt. Das Team konnte die App an die genauen Anforderungen des Unternehmens anpassen, einschließlich in mehreren Sprachen, um den Bedürfnissen der Produktionsmitarbeiter in den verschiedenen Ländern gerecht zu werden. Die Integration ins SAP ERP ermöglichte den Zugriff auf und die Aktualisierung von Arbeitsauftragsinformationen in Echtzeit.

4.7.3. Ergebnis

Nun können Produktionsmitarbeiter, anstatt mehrere Systeme zu nutzen, Produktionsaufträge auf einem einzigen Bildschirm lesen und bestätigen sowie zusätzliche Montageinformationen abrufen. Außerdem steuert die Mitarbeiterführungs-App den Betrieb von Werkzeugen, wie Schraubendrehern und anderen Zusatzgeräten, was eine effiziente und genaue Ausführung von Montageaufgaben gewährleistet. Die Zeit für die Suche nach Informationen und für die Rückmeldung der Montageschritte hat sich auf ein Drittel reduziert. Gleichzeitig hat sich der Prozess für die Erstellung von Montageanleitungen verbessert und der Aufwand diese zu erstellen hat sich halbiert.

4.7.4. Stakeholder / Rollen im Prozess

Berater von SAP haben in enger Zusammenarbeit mit den nicht-technischen Benutzern die neue App erstellt.

Zusammenfassung

Die Nutzung von LC/NC-Tools hat gezeigt, dass mit hoher Geschwindigkeit eine App entwickelt und in einem Pilotprojekt getestet sowie für den weiteren Roll-out verfeinert werden kann. Die Offenheit den Prozess anzupassen und auch vorgelagerte Prozesse wie z. B. das Erstellen von Montageanleitungen, waren für eine erfolgreiche Umsetzung maßgeblich. In Zukunft wird das Unternehmen weitere Anwendung mit der LC/NC-Plattform von SAP realisieren.

5. Fazit / Ausblick

Autoren: Florian Klein und Carsten Rückriegel,
VDMA Software und Digitalisierung

Dieses Whitepaper zeigt auf: Low-Code bewegt sich am Puls der Zeit. Die Technologie vereinfacht bzw. beschleunigt die Umsetzung und Bereitstellung von Anwendungen stark und bietet damit nicht nur den Unternehmen, sondern auch deren einzelnen Kunden einen Mehrwert. Ideen und Konzepte können durch den Einsatz von Low-Code schnell und agil umgesetzt werden. Das bedeutet, Low-Code kann gewisse Prozesse in ihrer Individualität unmissverständlich abbilden und muss sich dadurch nicht umgekehrt an das jeweilige System anpassen. Low-Code beschreibt demnach sehr gut, wie Software in naher Zukunft in Unternehmen genutzt werden kann und zeigt auf, dass weniger (manueller) Programmieraufwand auch gleichbedeutend mit weniger Arbeitsstunden ist, die für die klassische Entwicklung aufgewendet werden müssen. Erhöhte Effizienz und weniger notwendige Ressourcen für die Softwareentwicklung sind demnach durch Low-Code möglich.

Low-Code ist mehr als nur ein Trendthema. Die beschriebenen und skizzierten Anwendungsfälle aus der Praxis machen deutlich, dass Low-Code-/No-Code-Plattformen unter Berücksichtigung aktueller KI-Entwicklungen bereits in den Unternehmen angekommen sind. So kann die IT-Abteilung mithilfe von Low-Code schneller und proaktiv auf Anforderungen der Fachbereiche reagieren und die Zusammenarbeit zwischen den involvierten Bereichen kann immens gefördert werden. Alle Protagonisten werden im Entstehungs- und

Umsetzungsprozess direkt von Beginn an einbezogen. Zudem können Ideen im Sinne der agilen Arbeitsweise anhand eines direkten Feedbacks validiert und ausgestaltet werden, was zu einer Beschleunigung von Innovationsprozessen führt und das Risiko einer Fehlinvestition deutlich reduzieren kann. In der Folge können MVPs (Minimal Viable Products) schneller umgesetzt werden. Jedoch muss sich durch diese Arbeitsweise auch die Denkweise bzw. das Mindset anpassen, denn Veränderung ist oftmals nicht nur zu bestehenden Prozessen notwendig, sondern in dieser neuen Art der Zusammenarbeit inhärent. Offenheit für Veränderungen braucht es deshalb bei allen Akteuren.

Low-Code kommt aber auch technologisch an seine Grenzen, je nachdem wie komplex der Aufbau und Umfang einer Anwendung geplant ist oder je tiefer diese im Unternehmen verankert ist. Das heißt, desto komplizierter die Logik, umso unübersichtlicher werden die Abhängigkeiten zwischen den einzelnen Komponenten, weshalb die innere Softwarequalität bei einer Umsetzung mit Low-Code/No-Code leiden kann. Entsprechend wird das Programm in diesem Fall semantisch nicht lesbar und lässt nur noch eine geringe Anzahl von Anpassungen zu. Auch sollte berücksichtigt werden, dass die gewonnene Zeiteinsparung durch LC-Plattformen ggf. wieder in die Qualitätssicherung gesteckt werden muss. Deshalb gilt es die jeweiligen Vor- und Nachteile der einzelnen Plattformanbieter im Vorfeld genau zu beäugen und frühzeitig zu testen, inwiefern diese in die eigene Umgebung integriert werden können. Nicht jede Plattform ist für alle Zwecke geeignet.

Um diese Lücke zu schließen kann generative KI eine wertvolle Unterstützungshilfe sein. Sie kann dazu beitragen, die Integration von Lösungen innerhalb des Softwareentwicklungszyklus zu beschleunigen – aber Entwicklerinnen und Entwickler werden dadurch keinesfalls obsolet. Strukturgebende und kontextsensitive Aufgaben werden auch zukünftig weiterhin in der Hand von erfahrenen Programmierern bleiben, jedoch können einfache und automatisierbare Aufgaben auch mit KI-Tools adäquat gelöst werden. Low-Code leistet somit in Verbindung mit generativen KI-Funktionen einen wertvollen Beitrag zur Schließung der Lücke des Fachkräftebedarfs und trägt zur Demokratisierung der Softwareentwicklung bei. Es werden mehr Personen aus der Belegschaft befähigt, bei der Entstehung von Softwareanwendungen aktiv mitzuhelfen. Jedoch gilt es zu bedenken, dass heutzutage generative KI zwar schon Softwarecode erstellen, testen und debuggen kann, das Ergebnis in der Regel jedoch nicht fehlerfrei ist. Deshalb muss der Code nach wie vor von einem professionellen Entwickler auf Qualität und Zuverlässigkeit geprüft werden, um Softwareanwendungen nachhaltig, skalierbar und wartbar zu bauen.

Alles in allem wird deutlich, Low Code-Plattformen werden in den kommenden Jahren aufgrund der möglichen Effizienzgewinne, eine immer stärkere Rolle einnehmen. Zwar können sie die traditionelle Softwareentwicklung (noch) nicht 1:1 ersetzen, da die Komplexität aktuell nicht in angemessener Art und Weise abgebildet werden kann. Allerdings können LC-/NC-Plattformen die Softwareunternehmen bzw. Entwicklungsabteilungen der Maschinenbauunternehmen stark entlasten, da deren erfahrenen Programmierer sich auf schwierige Aufgaben fokussieren können.

Mit Blick in die Zukunft verändert sich damit nicht nur die Zusammenarbeit innerhalb von Unternehmen, sondern auch über Unternehmensgrenzen hinweg und individuelle Stärken müssen neu positioniert werden. Beispielsweise kann es für etablierte Softwarehersteller bedeutsamer werden, sich auf das vorgelagerte Softwareengineering zu spezialisieren, um in Verbindung mit dem Prozesswissen ihrer Kunden eine nachhaltige und stabile Zusammenarbeit, zur Erstellung von Softwareanwendungen, aufzubauen. Low-Code und No-Code in Kombination mit generativer KI schafft ein neues Werkzeug, dass es mehr Akteuren ermöglicht, kreative und innovative Lösungen in Form von Software zu erstellen und damit den steigenden Bedarf zu decken.

6. Anhang A: Übersicht / „Entscheidungshilfe“ zu LC- und NC-Plattformen

Autor: Georg Braun, SAP Deutschland SE & Co. KG

Die Wahl einer Low-Code- oder No-Code-Plattform erfordert mehrere Entscheidungskriterien.

Dabei sind die **Anwendungsszenarien** oft entscheidend für die Auswahl der richtigen LC/NC-Plattform bzw. der LC/NC-Tools.

Unter anderem können folgende Entscheidungskriterien bei der Auswahl einer LC/NC-Plattform in Betracht gezogen werden:

- **Benutzerfreundlichkeit:** Der Hauptzweck solcher Plattformen besteht darin, den Entwicklungsprozess, insbesondere für Nicht-Entwickler, zu vereinfachen. Die Plattform sollte benutzerfreundlich und intuitiv sein.
- **Flexibilität und Anpassbarkeit:** Auch wenn diese Plattformen der Vereinfachung der Entwicklung dienen, sollten sie dennoch genügend Flexibilität für die Anpassung an Ihre spezifischen Geschäftsanforderungen bieten.
- **Skalierbarkeit:** Die Plattform sollte in der Lage sein, wachsende Arbeitsmengen kompetent zu bewältigen. Dazu gehört, dass Sie mit dem Wachstum Ihres Unternehmens eine zunehmende Anzahl von Benutzern, Datenvolumen und Komplexität verwalten können.
- **Integrationsfähigkeiten:** Die Plattform sollte sich problemlos in andere Systeme und Software integrieren lassen, die Ihr Unternehmen verwendet.
- **Sicherheit:** Da diese Plattformen häufig für den Umgang mit sensiblen Geschäftsdaten verwendet werden, sollten sie über robuste Sicherheitsfunktionen zum Schutz vor Datenschutzverletzungen und anderen Bedrohungen verfügen.
- **Kosten:** Kostenbetrachtungen umfassen nicht nur die Vorlaufkosten der Plattform, sondern auch die langfristigen Kosten für die Wartung und Aktualisierung der Plattform.
- **Anbieter-Support und Community:** Der Plattformanbieter sollte guten technischen Support bieten. Eine große und aktive Community kann eine wertvolle Quelle für Hilfe und Ratschläge sein.
- **Schulungen und Ressourcen:** Überlegen Sie, welche Art von Schulungen und Ressourcen verfügbar sind, um Ihnen und Ihrem Team dabei zu helfen, die effektive Nutzung der Plattform zu erlernen.
- **Bereitstellungsoptionen:** Überprüfen Sie, ob die Plattform die von Ihnen bevorzugte Bereitstellungsoption unterstützt, z. B. Cloud, lokal oder hybrid.
- **Mobile Funktionen:** Wenn Ihre Anwendung auf mobilen Geräten gut funktionieren soll, prüfen Sie die mobilen Funktionen der Plattform. Dazu können responsives Design, native Entwicklung mobiler Apps oder für Mobilgeräte optimierte Arbeitsabläufe gehören.
- **Compliance:** Wenn Ihr Unternehmen in einer regulierten Branche tätig ist, müssen Sie sicherstellen, dass die Plattform alle relevanten Vorschriften einhält.
- **Zukünftige Roadmap:** Überprüfen Sie die Entwicklungs-Roadmap des Anbieters. Es vermittelt einen Eindruck davon, in welche Richtung sich das Produkt entwickelt und ob es Ihren zukünftigen Anforderungen entspricht.

Denken Sie daran, dass es sinnvoll ist, vor der Auswahl einer Plattform eine gründliche Anforderungsanalyse durchzuführen und einige Optionen mit Testversionen zu überprüfen, bevor Sie eine endgültige Entscheidung treffen.

7. Übersicht zu den Mitwirkenden

Judith Armbruster
iT Engineering Software Innovations GmbH, Pliezhausen

Georg Braun
SAP Deutschland SE & Co. KG, Walldorf

Christopher Bouveret
Simplifier AG, Würzburg

Daniel Gal
GAL Digital GmbH, Hungen

Dr. Thomas Genßler
CAS Software AG, Karlsruhe

Karl Haag
Simplifier AG, Würzburg

Kay Jeschke
SAP Deutschland SE & Co. KG, Hamburg

Florian Klein
VDMA Software und Digitalisierung, Frankfurt am Main

Malte Limbrock
valantic GmbH, München

Heiko Masloh
SAP Deutschland SE & Co. KG, Hamburg

Manuel Möller
gbo datacomp GmbH, Augsburg

Dr. Rainer Mümmler
The MathWorks GmbH, München

Hien Nguyen
iT Engineering Software Innovations GmbH, Pliezhausen

Carsten Rückriegel
VDMA Software und Digitalisierung, Frankfurt am Main

Wolfram Schäfer
iT Engineering Software Innovations GmbH, Pliezhausen

8. Literaturverzeichnis

Armbruster, J. (2023): Praxishandbuch agile Organisationsentwicklung. München: UVK Verlag.

Azamfirei, Razvan (2023): Large language models and the perils of their hallucinations. Abgerufen unter: <https://doi.org/10.1186/s13054-023-04393-x> [14.12.2023]

Bitkom e.V. (2022): Trotz Krieg und Krisen: In Deutschland fehlen 137.000 IT-Fachkräfte. Abgerufen unter: <https://www.bitkom.org/Presse/Presseinformation/2022-01-03-Trotz-Krieg-und-Krisen-In-Deutschland-fehlen-137-000-IT-Fachkraefte> | Presseinformation | Bitkom e. V. [03.01.2024]

Europäische Kommission (2018): Ethik-Leitlinien für eine vertrauenswürdige KI. Hochrangige Expertengruppe für künstliche Intelligenz. Abgerufen unter: https://elektro.at/wp-content/uploads/2019/10/EU_Ethikleitlinien-KI.pdf [14.12.2023]

Forrester (2017): Forrester Wave: Low-Code Development Platforms For AD&D. Abgerufen unter: <https://www.forrester.com/report/The-Forrester-Wave-LowCode-Development-Platforms-For-ADD-Pros-Q4-2017/RES137262> [03.01.2024]

Forrester (2022): Low-Code Is Helping Technology Firms Build A Bridge To Get Over Developer Hiring Challenges. Abgerufen unter: <https://www.forrester.com/blogs/low-code-is-helping-technology-firms-build-a-bridge-to-get-over-developer-hiring-challenges/> [03.01.2024]

Gartner (2021): Gartner Forecasts Worldwide Low-Code Development Technologies Market to Grow 23% in 2021. Press Releases, Newsroom, STAMFORD, Conn., February 16, 2021.

Gartner (2022a): Gartner Forecasts Worldwide Low-Code Development Technologies Market to Grow 20% in 2023. Press Release, Newsroom, STAMFORD, Conn., December 13, 2022.

Gartner (2022b): Why Fusion Teams Matter. Abgerufen unter: <https://www.gartner.com/en/articles/why-fusion-teams-matter> [8.1.2024]

Gottschick, Jan, Tiemann, Jens (2020): Low Code. Herausgeber: Fraunhofer FOKUS Kompetenzzentrum Öffentliche IT. Abgerufen unter: [Low Code - Low Code | Kompetenzzentrum Öffentliche IT \(ÖFIT\) \(oefentliche-it.de\)](https://www.oefentliche-it.de/low-code/) [03.01.2024]

Haufe (2023) Organisationsentwicklung als Konzept für den Wandel. Abgerufen unter: https://www.haufe.de/steuern/haufe-steuer-office-excellence/organisationsentwicklung-als-konzept-fuer-den-wandel-121-definition-und-merkmale-der-organisationsentwicklung_idesk_PI25844_HI14494952.html [19.09.2023]

Herzog, Christoph (2018): Digitale Transformation mit Low-Code Development - Die Zukunft der Entwicklung von Unternehmensanwendungen. United Planet. Abgerufen unter: https://ki-gmbh.de/sites/default/files/ftp/download/news/Intrex_lowcode_0.pdf [03.01.2024]

IDC (2021): The Low-Code Development Imperative, November 2021.

Lowe, Ryan; Leike, Jan (2023): Aligning language models to follow instructions. Abgerufen unter: <https://openai.com/research/instruction-following> [14.12.2023]

Mendix (2023): Abgerufen am 03.01.2024 Low-Code-Entwicklung wird als Chance bei hoher Digitalisierungs-Nachfrage erkannt und genutzt.

OpenAI (2023): GPT-4. Abgerufen unter: <https://openai.com/research/gpt-4> [14.12.2023]

Park, Joon Sung et al. (2023): Generative Agents: Interactive Simulacra of Human Behavior Abgerufen unter: <https://arxiv.org/pdf/2308.00352.pdf> [14.12.2023]

Peng, Sida (2023): The Impact of AI on Developer Productivity: Evidence from GitHub Copilot. Abgerufen unter: <https://arxiv.org/pdf/2302.06590.pdf> [14.12.2023]

Qian, Chen et al. (2023): Communicative Agents for Software Development. Abgerufen unter: <https://arxiv.org/pdf/2307.07924.pdf> [14.12.2023]

Radford, Alec (2023): Improving language understanding with unsupervised learning. Abgerufen unter: <https://openai.com/research/language-unsupervised> [14.12.2023]

Radford, Alec et al. (2023): Better language models and their implications. Abgerufen unter: <https://openai.com/research/better-language-models> [14.12.2023]

SAP SE (2023): LC/NC-Webseite. Abgerufen unter: https://www.sap.com/germany/products/technology-platform/low-code.html?dfa=1&gad_source=1&gclid=Cj0KCQiAnfmsBhDfARIsAM7MKi0zE1eEcJT6dmGKKbeStFRKQr80ip2ObB9DS-b6rMg0ykctxs-SfEaAifbEALw_wcB&gclsrc=aw.ds [10.01.2024]

SAP SE (2023): LC/NC SAP Discovery Center. Abgerufen unter: <https://discovery-center.cloud.sap/serviceCatalog/sap-build-apps?tab=feature®ion=all> [10.01.2024]

Simplifier AG (2023): Digitale Verladekontrolle – Optimale Prozessdigitalisierung für lückenlose Transparenz. Abgerufen unter: <https://simplifier.io/applications/verladekontrolle/> [14.12.2023]

techconsult (2022): Low-Code-/No-Code-Development Enabler der digitalen Transformation-Status quo und Planungen von Low-Code-/No-Code-Development-Plattformen in deutschen Unternehmen. In Zusammenarbeit mit mgm technology partners GmbH. Abgerufen unter: https://www.mgm-tp.com/documents/Low-Code-Studie_techconsult_DE.pdf

TechTarget (2023): Wie KI Low-Code/No-Code-Entwicklung für DevOps beeinflusst. Abgerufen unter: [Wie KI Low-Code/No-Code-Entwicklung für DevOps beeinflusst | Computer Weekly](#) [03.01.2024]

Vogel Communications (2023): Softwareentwicklung - Gartner Hype Cycle sieht KI und Plattform-Engineering auf dem Weg zum Masseneinsatz. Vogel Communications Group GmbH & Co. KG. Abgerufen unter: <https://www.industry-of-things.de/ki-unterstuetzte-software-engineering-zukunft-der-programmierung-a-5d9c6f11ed1a0db5c614ccee58d-61fa0/?cmp=nl-345&uuid=443a0efbd806fdfe84565de6048ba122> [03.01.2024]

VDMA Software und Digitalisierung

Über uns

Der VDMA Software und Digitalisierung vertritt die Interessen der Software-Hersteller und spiegelt digitale Technologien an den Maschinenbau. Die Abteilung Informatik und der VDMA Software und Digitalisierung arbeiten sehr eng zusammen und werden als eine Einheit im VDMA geführt.

Ziel der beiden Gruppierungen ist es, die Zusammenarbeit von Softwareindustrie und Maschinenbau zu fördern und damit die digitale Transformation voranzutreiben.

vdma.org/software-digitalisierung
vdma.org/digitalisierung-industrie-40

Unsere Publikationsübersicht – kompakt und online

Die Publikationen beschäftigen sich mit verschiedenen Aspekten der Digitalisierung in Maschinenbauunternehmen sowie Cybersecurity und Informationssicherheit und dienen als Handlungsempfehlungen. Alle Infos zu unserer Publikationsübersicht – kompakt und online - vdma.org - VDMA

VDMA-Industrie Podcast

Der Audio-Blog für den Maschinen- und Anlagenbau beleuchtet auch digitale Trendthemen wie Plattformökonomie, Digitale Souveränität, Künstliche Intelligenz, Smart Factory, Security und Blockchain.

<https://derindustriepodcast.podigee.io>

Impressum

VDMA
Software und Digitalisierung

Lyoner Straße 18
60528 Frankfurt
Telefon + 49 69 66 03-13 60
E-Mail software@vdma.org
Internet <https://www.vdma.org/software-digitalisierung/>

Layout und Satz

VDMA DesignStudio

Copyright 2024

Das Werk, einschließlich seiner Teile, ist urheberrechtlich geschützt

VDMA

Software und Digitalisierung

Lyoner Straße 18

60528 Frankfurt am Main

Telefon + 49 69 66 03-13 60

E-Mail software@vdma.org

Internet <https://www.vdma.org/software-digitalisierung/>