8 Tipps für die Low-Code-Praxis

Wie Sie Citizen Developer managen

IT-Entscheider, die sich auf Citizen Developer verlassen, müssen dennoch ihren Teil dazu beitragen, dass die Entwicklung produktiv und sicher abläuft.

› Entscheidend für den Erfolg von Citizen-Development-Initiativen ist unter anderem der Support durch die IT beziehungsweise Profi-Entwickler. (Foto: Gorodenkoff – shutterstock.com)

Die Grundidee hinter Low-Code- und No-Code-Tools – die „App-Lücke" mit fähigen Business-Anwendern zu schließen – findet bei Unternehmen immer größeren Anklang. Laut Gartner bauen bereits 41 Prozent der Nicht-IT-Mitarbeiter ihre eigenen Lösungen oder passen sie an. Bis 2023 soll die Zahl solcher Citizen Developer die der professionellen Softwareentwickler in großen Unternehmen um den Faktor 4 übertreffen, prognostizieren die Auguren.

Die meisten Unternehmen setzen bereits mindestens ein Low-Code-Tool ein, das auf Abteilungsebene oder von einzelnen Anwendern genutzt wird. „Eines Tages, vielleicht schon in naher Zukunft, wird die Entwicklung von Anwendungen ohne das Schreiben von Code so selbstverständlich sein wie E-Mail und Tabellenkalkulationen", prognostiziert Forrester.

Doch mit den Vorteilen einer schnelleren Entwicklung gehen auch größere Risiken einher. Laut Gartner ist die Governance im Hinblick auf die größere Autonomie, die Trends wie Self-Service und Low Code mit sich bringen, zu einem wichtigen Aspekt

für IT-Entscheider geworden. Obwohl formale Governance-Strukturen noch nicht weit verbreitet sind, beginnen IT-Führungskräfte, sich mit dem Thema Low Code Management auseinanderzusetzen. Wir haben fünf Tipps für Sie zusammengetragen, die Sie dabei unterstützen, Citizen Developern auf pragmatische Art und Weise zum Erfolg zu verhelfen – ohne sie dabei von Experimenten und Self-Service abzuschrecken.

1. Aus dem Schatten treten

Damit Low Code erfolgreich sein kann, darf es weder als Schatten-IT noch als potenzielle Belastung gesehen werden, meint Gartner-Analyst Jason Wong: „Der Zweck von Citizen Development besteht darin, eine Vereinbarung zwischen der IT und dem Business zu treffen. Es geht darum das Engagement der Citizen Developer zu fördern und ihnen die richtigen, für sie geeigneten Tools an die Hand zu geben."

Darüber hinaus bieten Low-Code- und No-Code-Plattformen nicht nur die Möglichkeit, visuell Anwendungen zu erstellen: Sie zentralisieren auch den Zugriff und die Ressourcen und erfassen deren Nutzungsstatistiken. Aus diesem Grund kann die IT-Abteilung Richtlinien erstellen, die festlegen, wer auf welche Datenquellen zugreifen darf und wie die Apps und Automatisierungsabläufe, die diese Daten beinhalten, gemeinsam genutzt werden dürfen.

Eine granulare, rollenbasierte Zugriffskontrolle ermöglicht es der IT-Abteilung, den Zugriff auf bestimmte Endpunkte und Datentabellen bis auf Feld- und Datensatzebene zu verwalten und verschiedenen Abteilungen in verschiedenen Entwicklungsumgebungen entsprechenden Zugriff zu gewähren.

2. Kontrolle und Autonomie ausbalancieren

Das richtige Gleichgewicht zwischen Kontrolle und Autonomie ist für den Citizen-Developer-Erfolg entscheidend. Natürlich haben auch sie kein Interesse an laxer Security – sind die Kontroll- und Zugangsprozesse jedoch zu schwerfällig, steigt die Gefahr, dass die Mitarbeiter Wege finden, diese zu umgehen.

„Das ist nicht übermäßig komplex, erfordert aber Planung. Sie müssen alles für Ihre spezifische Umgebung durchdenken. Daten sind der Schlüssel, den man für die Governance braucht. Die Grundlage dafür ist die Antwort auf die Frage: Welche Daten haben wir, welche sind sensibel und welche nicht?", meint John Bratincevic, Senior Analyst bei Forrester. „Achten Sie darauf, dass alle für extern freigegebenen Daten

vor Leaks geschützt sind und Richtlinien existieren, die Entwickler warnen, wenn sie Automatisierungen oder Workflows erstellen, die gegen die Compliance verstoßen könnten – etwa durch das Kopieren von E-Mails oder personenbezogenen Daten."

Citizen Developer sollten darüber hinaus auch autonom sein, wenn es um die Plattformwahl geht, warnt Wong: „Es ist ein Fehler, die Mitarbeiter zu zwingen, sich auf eine einzige Low-Code-Plattform festzulegen. Wenn Geschäftsanwender nicht die Befugnis haben, die für sie geeigneten Tools auszuwählen, wird aus Citizen Development nichts."

Stattdessen sollten sich Entscheider auf ihr Governance Framework verlassen und deutlich machen, welches Level an Support die Benutzer erwarten können, die sich für ein anderes Tool entscheiden, so Wong: „Machen Sie den betreffenden Mitarbeitern klar, dass es nicht möglich ist, eigene IT-Ressourcen für spezielle Tools bereitzustellen."

3. Richtigen Ansatz wählen

Für den Erfolg einer Citizen-Development-Initiative hängt auch davon ab, ob Sie das richtige, strategische Modell dafür gewählt haben. Forrester nennt für diesen Fall drei gängige Ansätze. Der erste geht von kleinen, autonomen Teams aus, die sich aus Mitarbeitern mit Erfahrung in Prozessoptimierung zusammensetzen. Diese Teams sind in eine Business Unit eingebettet und berichten an die Geschäftsleitung. Ein strategischer Ansatz, der sehr agil ist, sich aber nicht leicht skalieren lasse, wie Bratincevic anmerkt.

Der Self-Service-Ansatz, bei der jeder mit Low-Code-Tools auf der Grundlage von Richtlinien und Leitplanken in den Plattformen entwickeln kann, ist ein weiterer Weg: „Man kontrolliert das Was und nicht das Wer. Das ‚Wer' kann jeder sein, aber das ‚Was' wird durch die Beschaffenheit der Plattform und den gewährten Zugang eingeschränkt", weiß Bratincevic.

Der dritte und ausgereifteste Citizen-Development-Ansatz verbindet agile Teams und eine breite Demokratisierung zu einem föderalen Modell, bei dem ein Center of Excellence (CoE) Low-Code-Plattformen verwaltet, Leitplanken implementiert und Teams oder einzelne Champions in Abteilungen und Geschäftsbereichen sowie Self-Service-Low-Code-Entwickler unterstützt. In diesem Modell hängt die Art und Weise, wie eine bestimmte Anwendung entwickelt wird, von ihrem Anwendungsfall, den verwendeten Daten und der Erfahrung der beteiligten Entwickler ab, erklärt Bratincevic: „Dieses Modell ist komplexer, deckt aber am meisten ab: Sie kontrollieren Daten und Entwicklungsprozess, gehen dabei aber pragmatisch vor."

4. Adäquaten Support bieten

Neben Governance und Richtlinien müssen CIOs auch Ressourcen und Support für Citizen Developer bereitstellen. „Dass die IT den Citizen Developern gegenüber positiv eingestellt ist, ist unglaublich wichtig für deren Produktivität und Erfolg", weiß Gartner-Analyst Wong. Um das zu erreichen, schlägt Gartner vor, Citizen Development mit einem Governance-Framework zu formalisieren, das verschiedene Zonen umfasst:

- „grüne" Safe Zones, in denen Workflows und Automatisierungen erstellt werden können.
- „gelbe" Support-Zonen, in denen Citizen Developer und Profi-Entwickler zusammenarbeiten, um leistungsfähigere Anwendungen zu erstellen.
- „rote" Gefahrenzonen, die die Aufsicht und Genehmigung der IT-Abteilung erfordern, wobei einige Anwendungen als so komplex und geschäftskritisch gelten, dass sie unter der Kontrolle der IT-Abteilung bleiben.

Ein Kompetenzzentrum könnte beispielsweise APIs und benutzerdefinierte Komponenten erstellen oder Fusionsteams ermöglichen, in denen professionelle Entwickler sowohl mit Low-Code- als auch mit traditionellen Umgebungen arbeiten. Das CoE könnte auch Lernressourcen und Expertenhilfe für komplexere oder kritische Tasks bereitstellen.

Diese Art der Zusammenarbeit und Unterstützung unterscheide Low Code von Schatten-IT, erklärt Wong: „Bei der Schatten-IT sind Einzelpersonen auf sich allein gestellt und erledigen Dinge, die im Verborgenen liegen. Im Fall von Citizen Development heißt es: Wir machen das offen, damit die Benutzer keine Angst haben müssen, für die Verwendung dieser Tools gemaßregelt zu werden. Wir bieten ihnen einen Weg, um zu lernen, was sie brauchen und die Möglichkeit, im Support-Fall die Community um Hilfe zu bitten – entweder andere Power-User oder auch die IT-Abteilung."

5. APIs richtig einsetzen

Damit Citizen Development erfolgreich sein kann, muss die IT-Abteilung proaktiv für die Bereitstellung von Konnektoren sorgen und robuste APIs für den Zugriff auf interne Daten erstellen, wie Kin Lane, Chief Evangelist beim API-Plattformanbieter Postman, weiß: „Stellen Sie sicher, dass Ihre APIs gut definiert sind, über eine Verwaltungsebene oder einen Katalog verfügen und dann leicht mit diesen Low-Code/No-Code-Lösungen verbunden werden können."

Entscheider müssten darüber hinaus auch nachverfolgen, wo APIs in der Produktion verwendet werden – sowohl um die Kosten für externe Schnittstellen im Blick zu behalten als auch um sicherzustellen, dass die Systeme, die eine interne API bereitstellen, über die entsprechenden Ressourcen verfügen. „Nicht alles, was wie eine API funktioniert, wird von einem robusten Backend produziert. Auch wenn wir gerne glauben würden, dass eine gut konzipierte RESTful-API eine API ausmacht, ist das nicht der Fall. Ein FTP-Speicherort mit einer CSV-Datei wird als API betrachtet, und Tabellenkalkulationen sind das A und O", meint Lane und empfiehlt: „Sie sollten auch Robotic Process Automation (RPA) nicht vergessen – eine zunehmend beliebte Methode, um Informationen aus Altsystemen in Low-Code-Anwendungen und Automatisierungs-Workflows zu übertragen.

6. Reviews und Metriken nicht vergessen

Einzelne Business User, die ihr eigenes Problem lösen, denken wahrscheinlich nicht über Hochverfügbarkeit, Geschäftsmetriken oder formale Reviews nach. Zudem verfügen nur wenige Low-Code-Plattformen über entsprechende Tools. Allerdings sind Metriken, wie etwa die Zeit, die ein bestimmter Prozess in Anspruch nimmt, durchaus hilfreich. Gleiches gilt für regelmäßige Reviews, um die Performance zu tracken und Weiterentwicklungsmöglichkeiten zu analysieren.

Metriken und Reviews bieten auch die Möglichkeit, Unternehmensprozesse zu untersuchen, da die Automatisierung eines schlechten Prozesses nur noch schneller zu schlechten Ergebnissen führt. Nutzen Sie Process Mining Tools, um Ineffizienzen oder zusätzliche Arbeit aufzudecken, die einige Teams möglicherweise leisten, und geben Sie den Mitarbeitern, die tatsächlich an einem Prozess arbeiten, die Möglichkeit, ihn zu rationalisieren, anstatt nur Anwendungen zu entwickeln, die das Problem überdecken.

7. Operations bei Bedarf weiterentwickeln

Analyse- und Überwachungstools in Low-Code-Plattformen können nicht nur die API-Nutzung nachverfolgen, sondern auch auf Anwendungen hinweisen, die so beliebt oder geschäftskritisch geworden sind, dass sie in die gelben oder roten Support-Zonen (wie von Gartner beschrieben) der höheren Stufe verschoben werden sollten.

Laut Gartner-Mann Wong sind bahnbrechende Ideen, die sich zu Anwendungen entwickeln, die so beliebt sind, dass sie mehr IT-Support benötigen, ein Zeichen

für geschäftliche Innovation. Die Aufgabe der IT-Abteilung sei es, diese nachhaltig zu gestalten: „In der Praxis kann das zu Spannungen führen: Der ursprüngliche Low-Code-Entwickler hat möglicherweise Bedenken, dass die IT-Abteilung das Tool übernimmt – das IT-Team wiederum hat möglicherweise Bedenken, eine Anwendung zu unterstützen, die es nicht selbst entwickelt oder spezifiziert hat. Eine Kultur der Zusammenarbeit zwischen Unternehmen und IT sollte dazu beitragen, Misstrauen auf beiden Seiten zu vermeiden."

8. Innovationskultur schaffen

Eventuell gehören Sie zu den IT-Entscheidern, die sich Sorgen machen, dass durch Low-Code-Experimente zu viele Anwendungen entstehen könnten, die das IT-Team nicht bewältigen kann. Allerdings besteht laut Bratincevic das Problem häufiger darin, dass nicht genug Schwung in die Strategie kommt, um sie zum Erfolg zu führen: „Viele Geschäftsanwender, die vor Problemen stehen, die sie mit Low Code lösen könnten, sehen sich einfach nicht als Entwickler." Viele Unternehmen stellten nach den Worten des Experten fest, dass interne Hackathons – mit Zeit für Schulungen, Mentoring und Support – Interesse wecken und einen Anwendungskern hervorbringen könnten.

Low Code kann zudem den ein oder anderen Schritt nach oben auf der Karriereleiter ermöglichen – etwa wenn Business-Anwender damit Fachwissen erwerben, dass sie für technischere Jobrollen qualifiziert: „Betrachten Sie es als eine Möglichkeit, Ihr zukünftiges digitales Personal zu fördern – und seien Sie bereit, Ihre Mitarbeiter zu unterstützen und zu belohnen. Ein Grund für das Scheitern von Low-Code-Programmen ist die Erwartung, dass die Mitarbeiter solche Initiativen als Zusatzaufgabe neben ihrer eigentlichen Tätigkeit erledigen", so Bratincevic.

Low Code als Chance zu betrachten, sich dabei jedoch der Unwägbarkeiten bewusst zu sein, ist schließlich der ultimative Ratschlag des Experten: „Es wird nicht perfekt sein. Es wird Probleme geben. Sie werden viele Fehler machen. Aber das ist Ihre Chance, all die Leute, die Anwendungen im gesamten Unternehmen entwickeln, auf eine sinnvolle Art und Weise zu orchestrieren, die vernetzt und automatisiert, statt alles dem Zufall zu überlassen."

Mary Branscombe

Mary Branscombe beschäftigt sich als freiberufliche Journalistin seit über 20 Jahren mit der IT-Branche. Sie schreibt unter anderem für unsere US-Schwesterpublikation cio.com.