Optionen für die Modernisierung von IBM i-Anwendungen, Teil 2

Der Königsweg führt über LANSA RAMP

Wenn es um die Modernisierung und die funktionale Aufwertung von IBM i-Anwendungen geht, bietet die LANSA-Familie viele Optionen. Im ersten Beitrag wurden bereits die Charakteristika von LANSA aXes und Visual LANSA aufgezeigt. In diesem Artikel steht der „Modernisierungs-Königsweg" mit LANSA RAMP im Mittelpunkt.

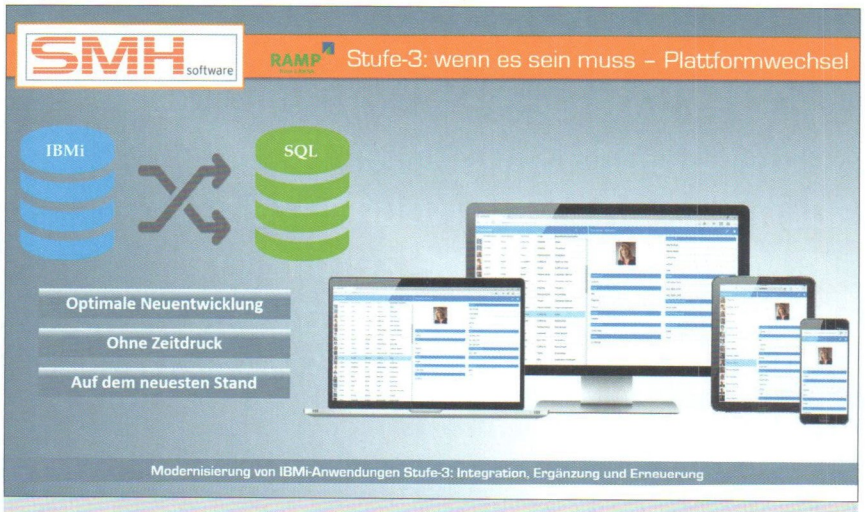
Der Digitalisierungsdruck in vielen mittelständischen Unternehmen wirkt sich auch auf die bestehenden und bestens bewährten IT-Lösungen aus: Es sind zeitgemäße Benutzerschnittstellen gefragt, die bei der Smartphone-Generation Akzeptanz finden. Neuartige Funktionalitäten müssen sich schnell einbauen lassen – und das alles vor dem Hintergrund, dass Softwarespezialisten im Entwicklungsbereich extrem gesucht sind. Mittlerweile geben Technologieunternehmen wie Google sogar die Devise heraus, dass sie ihre neuen Anwendungen zuerst für mobile Plattformen herausgeben – also „Mobile First".

Damit gewinnt das Thema Software-Modernisierung eine noch höhere Priorität. In den meisten Fällen macht es aber gar keinen Sinn, die erprobten und optimal an die derzeitigen Geschäftsprozesse angepassten Lösungen wegzuwerfen. Doch zusätzliche Schnittstellen sowie eine Überführung der bestehenden Anwendungen auf neue Einsatzbereiche wie die Web- oder Mobilplattform rangieren in den Anwenderunternehmen weit oben auf der Wunschliste.

Da aber die Nachfrage an Software-Entwicklern extrem angezogen hat, sind Konzepte gefragt, die durch einen möglichst geringen Entwicklungsaufwand gekennzeichnet sind. Und für diesen Ansatz gibt es auch schon das passende Schlagwort: „Low Code Technology". Wer Tools aus dieser Kategorie einsetzt, dem wird versprochen, dass er sich auf die Innovation selbst fokussieren kann und deutlich weniger Aufwand in deren Realisierung stecken muss. Es sollen sich mit diesem Ansatz Entwicklungszeiten erzielen lassen, die nur mehr ein Zehntel der Zeit betragen – verglichen mit traditionellen Software-Entwicklungsmethoden.

Dabei erweist sich die Plattformneutralität als ein wichtiger Erfolgsfaktor. Im Mobilbereich zum Beispiel sind mehrere Plattformen (Apple, Android und eventuell sogar noch die von Microsoft) zu unterstützen – und dabei auch verschiedene Versionsstände. Vor allem bei Android gibt es erstaunlich viele „Geschmacksrichtungen". Die Pflege des Codes und seine Weiterentwicklung bedeuten für die meisten der mittleren und kleineren Unternehmen einen viel zu großen Aufwand. Daher sind Tools nötig, die einem hier möglichst viel Arbeit abnehmen und die Modernisierung bestehender Applikationen optimal unterstützen – und dabei die technischen Details der Zielplattform weitestgehend verbergen.

Ein interessantes Toolset in diesem Kontext stellt die LANSA-Familie dar. Bereits im ersten Teil dieser Beitrags-



Mit den passenden Tools verliert auch ein Plattformwechsel seinen Schrecken.

Quelle: S.M.Hartmann Software

reihe (siehe Midrange Magazin Ausgabe 5/2018 Seite 30 und 31) wurden LANSA aXes und Visual LANSA skizziert. Durch einen integrativen und modularen Aufbau gibt es mit der LANSA-Familie für alle Wege des Modernisierungsprozesses die richtigen Werkzeuge und Lösungen, die in konsequenter Umsetzung des Low-Code-Konzepts die Entwicklungszeiten tatsächlich radikal verkürzen.

Als Königsweg für Modernisierungsprojekte hat sich dabei LANSA RAMP etabliert.

LANSA RAMP als der Favorit

Eine typische 5250-basierte Anwendung besteht aus verschiedenen Bildschirmdarstellungen, die über Tastenkombination oder Schaltflächen miteinander logisch verbunden sind. Wenn es darum geht, in eine solche Anwendung neuartige Funktionalitäten einzubauen, sollten möglichst alle technischen Implementierungsdetails vom eingesetzten Entwicklungswerkzeug abgedeckt werden. Damit ergeben sich zwei große Vorteile: Zum einen „kümmert" sich das Entwicklungs-Tool um sämtliche technischen Neuerungen, und zum anderen muss der Ausgangscode – bei „AS/400-Programmen" meist auf RPG oder Cobol-Basis – nicht modifiziert werden. Speziell beim Einsatz von LANSA RAMP ist auch noch darauf hinzuweisen, dass alle Modernisierungen, die mit Hilfe von LANSA aXes und Visual LANSA erstellt wurden, mit übernommen werden können.

Die Arbeitsweise von RAMP ist mit einem Routenplaner gut vergleichbar: Man muss dazu die einzelnen „5250-Bildschirmdarstellungen" einer bestehenden Anwendung erfassen, sie dann katalogisieren und anschließend einander zuordnen. Als erstes ist zu definieren, wohin man kommen will, also um welchen Zielbildschirm es sich handelt – die sogenannten „Destinations". Hierbei handelt es sich um die Bildschirme, die dem Anwender in der fertigen Applikation zur Verfügung gestellt werden sollen (also z. B. die Auftragserfassung oder die Seite mit der Darstellung von „Offenen Posten" im Buchhaltungsprogramm, deren Inhalte aus der Datenbank geholt werden).

Im nächsten Schritt geht es um die „Kreuzungen" in der Anwendung – die „Junctions". Hier wird festgelegt, wie der Applikationsfluss geleitet werden soll. Das sind in der ursprünglichen Anwendung zum Beispiel Menüs, die ihrerseits verschiedene Auswahlmöglichkeiten bieten, oder Steuerungsbildschirme, in denen der Benutzer verschiedene Optionen eingeben kann. Diese Kreuzungen sind später nach außen hin nicht mehr sichtbar, sie werden in der modernisierten Applikation je nach Ziel-Destination quasi „selbstfahrend" angesteuert und überquert.

Als dritte Bildschirm-Kategorie gibt es die „unerwarteten Ereignisse" (die „Special Screens"). Special Screens sind für den Ablauf und die Steuerung der Anwendung nicht relevant (dies kann z. B. der Hinweis sein, dass man als Nutzer bereits in einer anderen Sitzung angemeldet ist). Über diese drei Kategorien – Destinations, Junctions und Special Screens – lässt sich die komplette bestehende Anwendung katalogisieren und anschließend auch die Steuerung der Applikation umsetzen: Jeder Ort – sprich jede Maske und jeder Bildschirm – erhält seinen eigenen, eindeutigen Namen. Somit lässt sich zum Beispiel dem Namen „Kundenumsatz" die Stelle zuordnen, an der die Kundenumsatz-Maske in der fertigen Anwendung erscheinen soll. Jede Destination, die man anzeigen möchte, bekommt eine Ziel-Zuweisung.

Im letzten Schritt sind dann sozusagen alle „Orte" miteinander zu verbinden, damit die effizientesten Verbindungen von Bildschirm zu Bildschirm gefunden werden. Das ist eine Art Tracking-Information, die automatisch erzeugt wird, wenn man sozusagen durch die Anwendung blättert. So kann RAMP später intelligent entscheiden, welches z. B. der schnellste Weg vom „Kundenumsatz" zur „Offenen-Posten-Anzeige" ist.

Tracking-Infos genutzt

Somit werden prinzipiell nur die ursprünglichen 5250-Masken in dem Tool abgebildet und aus dieser Tracking-Information erzeugt das Tool automatisch eine Javascript-Funktionalität, die im Hintergrund immer dann abgerufen wird, wenn auf dem Bildschirm irgendwo eine andere Destination angeklickt wird. Auf diese Art lassen sich sehr schnell neue Anwendungen auf der Basis eines bestehenden Systems entwickeln. Dabei werden Anpassungen, die zuvor bereits mit LANSA aXes vorgenommen wurden, übernommen.

Doch das stellt noch nicht das Ende der Möglichkeiten dar, die sich mit LANSA RAMP realisieren lassen. Entwickler können auch Programme hinzufügen, die bereits mit Visual LANSA erstellt wurden. Besonders flexibel erweist sich dieser Ansatz in der Praxis, weil man natürlich auch ganz neue Funktionalitäten erstellen und integrieren kann, wie etwa den Zugriff auf andere Datenbanken, die auf der IBM i nicht zur Verfügung stehen. Das alles lässt sich in der Entwicklungsumgebung machen, dazu ist nur eine passende Einbindung nötig.

Daher gibt einem dieses Tool eine enorme Freiheit, wenn es um die Frage geht, wo der komplette Modernisierungsprozess enden soll. Falls es im extremsten Fall sogar zu einer Abkehr von der IBM i kommen sollte, kann man die IBMi-Datenbanken als Basis für den Umstieg verwenden, weil sie ja bereits in SQL übertragen wurden. So kann man Schritt für Schritt neue Funktionen schreiben, diese einbauen und – wenn es denn tatsächlich sein muss – sogar auf die IBM i verzichten.

Siegfried Hartmann und Jörg Hamacher ■