ERWEITERUNG DES WORKFLOW-MODULS DES ERP-SYSTEMS FACTWORK UM MÖGLICHKEITEN ZUR BENUTZERINTERAKTION

M.Sc. Benjamin Weikmann
Dipl.-Inf. Franz Laubmeier
Prof. Dr.-Ing. Frank Herrmann
Labor für Informationstechnik und Produktionslogistik (LIP)
Ostbayerische Technische Hochschule Regensburg (OTH Regensburg)
benjamin.weikmann@fee.de und frank.herrmann@oth-regensburg.de

SCHLÜSSELWÖRTER

ERP-Systeme, Geschäftsprozesse, Workflows, UI-Design.

ABSTRACT

Durch Workflow-Systeme computergestützt abgewickelte Geschäftsprozesse benötigen neben den Aktivitäten, die sie komplett automatisiert ausführen können, von Zeit zu Zeit Feedback durch einen oder mehrere Benutzer. Dazu muss es geeignete Formen der Kommunikation und Interaktion der Workflow-Engine mit dem Benutzer geben. Der folgende Artikel beschreibt die Erweiterung des bestehenden Workflow-Moduls eines Enterprise Resource Planning (ERP)-Systems, welches um solche Fähigkeiten ergänzt wird.

EINLEITUNG

Neben Stammdatenpflege und informationstechnischer Unterstützung aller Bereiche eines Unternehmens gehört auch die Ausführung und Überwachung von Geschäftsprozessen zu den typischen Aufgaben eines ERP-Systems und dessen Umfeld. Insbesondere die Abbildung firmenspezifischer gelebter Prozesse stellt dabei oftmals eine besondere Herausforderung dar, da die hierfür benötigten Abläufe in Standardsoftware meist nicht implementiert sind, weil diese schlichtweg zu individuell sind. Eine übliche Vorgehensweise ist häufig der Einsatz von externen Prozessmodellierungswerkzeugen, deren Laufzeitumgebungen später über geeignete Schnittstellen mit dem ERP-System kommunizieren bzw. auf Datenbankebene relevante Daten austauschen – oder die ERP-Umgebung bietet ein solches Tooling als integralen Bestandteil selbst an. Das von der Firma F.EE GmbH entwickelte und vertriebene ERP-System FactWork besitzt mit dem Workflow-Modul ein solches Werkzeug, das die grafische Modellierung von Geschäftsprozessen im direkten Kontext des ERP-Systems ermöglicht (siehe Abbildung 1). Neben üblichen Standardelementen zur Ablaufsteuerung sind zum Workflow-Design im Wesentlichen eine Vielzahl an Aktivitäten verfügbar, die direkt Dienste der ERP-Software ansprechen können. Des Weiteren ist der Aufruf von Prozeduren der FactWork-eigenen Skriptsprache fe.script möglich, wodurch sämtliche weitere Funktionalität des Systems zugänglich wird. Lediglich im Bereich der Benutzerinteraktion ist das Workflow-Modul aktuell etwas eingeschränkt. Es fehlen Funktionen zur direkten aktiven Nutzerteilnahme an Prozessen sowie die Zurverfügungstellung von Eingabedaten seitens des Nutzers, die der Prozess für dessen weitere Ausführung benötigt. Im Rahmen dieses Projekts sollen Möglichkeiten evaluiert und anschließend implementiert werden, die das Workflow-Modul um die Leistungsfähigkeit im Bereich der Benutzerinteraktion entscheidend erweitern, indem genannte fehlende Fähigkeiten hinzugefügt werden. Die anschließende Realisierung eines Beispielprozesses soll zum Test und zur Demonstration der neu geschaffenen Funktionen dienen.

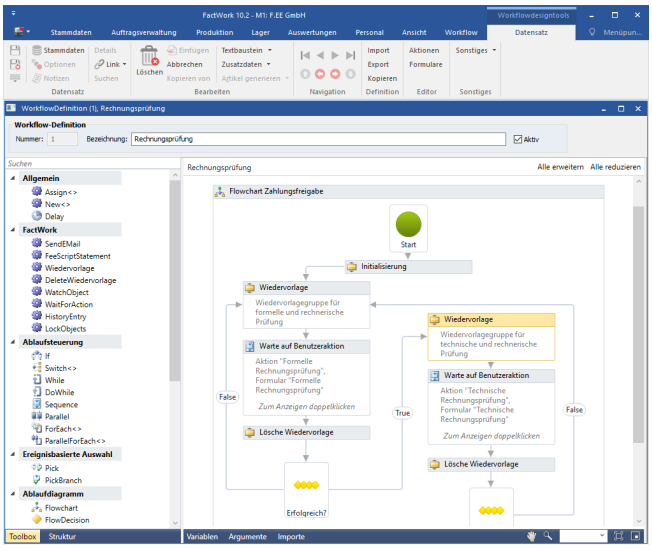

Abbildung 1: FactWork-Client mit geöffnetem Workflow-Designer

F.EE UNTERNEHMENSGRUPPE

Die 1982 von Hans Fleischmann gegründete F.EE-Unternehmensgruppe mit Stammsitz in Neunburg v. Wald beschäftigt mittlerweile rund 1.000 Mitarbeiter weltweit, davon 75 in ausländischen Niederlassungen, und kann einen jährlichen Umsatz von 175 Mio. Euro verbuchen (Stand 1/2018). Sie

besteht aus den vier Geschäftsfeldern Elektrotech Engineering, Automation Robotik, Informatik + Systeme und Energietechnik und bietet Leistungen und Produkte in den Bereichen Fertigungsautomatisierung, Systemprogrammierung, Energieerzeugung und IT an (FEE18). Die vom Geschäftsbereich Informatik + Systeme entwickelte und vertriebene Unternehmenssoftware FactWork ist dabei sowohl bei F.EE selbst als auch bei einer wachsenden Anzahl von Kunden im Einsatz.

EINORDNUNG

Im Zuge des Geschäftsprozessmanagements wird versucht, Erkenntnisse aus der Informationstechnik und Managementwissenschaften auf betriebliche Prozesse abzubilden, um daraus Vorteile wie z.B. eine Steigerung der Produktivität und Einsparungen von Kosten zu ziehen. Heutzutage existiert eine Fülle an Systemen, die auf unterschiedlichste Art und Weise die Abwicklung von Geschäftsprozessen unterstützen. Zur Modellierung von Prozessen existieren diverse gängige Notationssprachen. Nennenswert sind u.a. Petrinetze, Business Process Model and Notation (BPMN), Aktivitätsdiagramme nach dem Unified Modeling Language (UML) Standard und Event-driven Process Chains (EPCs). Diese unterscheiden sich teilweise erheblich in ihren Möglichkeiten, Ausdrucksfähigkeiten und Eignung für verschiedene Einsatzzwecke (Aalst 2013).

Als Quasi-Industriestandard gilt häufig die grafische Notationssprache BPMN, die von der Object Management Group (OMG) gepflegt wird (Penicina 2010). BPMN wird mittlerweile von einer Vielzahl auf dem Markt befindlicher Tools unterstützt, u.a. gibt es die kommerzielle ARIS Platform, aber auch bekannte Open-Source-Lösungen wie Activiti oder Camunda sind zu nennen. Diese unterstützten umfangreich die Modellierung, Analyse, Dokumentation und Ausführung bis hin zur Simulation und Optimierung von Prozessen (Neumann et al. 2015).

Eine wesentliche Designentscheidung bei der Erstellung des Workflow-Moduls des ERP-Systems FactWork war jedoch die Verwendung einer leichtgewichtigen einfach zu integrierenden Workflow-Engine sowie eines intuitiven, simplen Workflow-Designers. Daher fiel die Wahl auf den Einsatz der sog. Windows Workflow Foundation (WWF), die ein Teil der von Microsoft vertriebenen .NET-Entwicklungsplattform ist (siehe Microsoft Corporation 2016). Auch wenn diese in der Industrie eher ein Nischendasein führt, wird das Framework durchaus erfolgreich zur Modellierung und Ausführung länger andauernder Prozesse eingesetzt (siehe z.B. Ramasamy, Chua und Haw 2015). Da diese Arbeit auf der Infrastruktur des bestehenden Workflow-Moduls aufbaut, muss dies als gegebene Randbedingung berücksichtigt werden.

PROBLEMSTELLUNG

Eine typische Eigenschaft von Geschäftsprozessmodellierungswerkzeugen und deren zugehöriger Laufzeitumgebungen, die oftmals als sog. Workflow-Engines bezeichnet werden, ist deren Konzentration auf Aspekte der Ablaufsteuerung. Interaktion und Kommunikation mit Benutzern ist dabei zwar als notwendige Eigenschaft von Geschäftsprozessen vorgesehen, reicht aber meist nicht über das Abstraktionslevel von sog. *User Tasks* hinaus. Zur Anbindung und Realisierung dieser sind zwar Schnittstellen vorgesehen, die Implementierung wird aber meist dem Anwender überlassen, was u.a. die Integration mit bestehenden Systemlandschaften erheblich erschwert (Kossak et al. 2016).

Ähnlich verhält es sich mit dem bestehenden Workflow-Modul des ERP-Systems FactWork. In der Grundfunktionalität des eingesetzten Frameworks sind hauptsächlich Aktivitäten zur Kontrolle der Ablauflogik und zur Anbindung externer Dienste vorgesehen. Weitere Fähigkeiten müssen durch die Entwicklung benutzerdefinierter Aktivitäten ergänzt werden. Hierbei existieren inzwischen diverse Aktivitäten, die Funktionen der ERP-Software direkt ansprechen.

Die Möglichkeiten zur Benutzerinteraktion beschränken sich derzeit auf die Benachrichtigung eines Benutzers über ausstehende Arbeiten an Stammdatensätzen durch das in FactWork integrierte Wiedervorlagesystem (siehe Abbildung 2) sowie den Versand von E-Mails. Eine eingebaute Funktionalität zum Überwachen von Datenbankänderungen an einzelnen Datensätzen und der direkt mögliche Zugriff auf Datenbankobjekte aus dem Workflow heraus macht eine sofortige Reaktion auf geänderte fachliche Bedingungen möglich. Dies befähigt eine Workflow-Instanz ausgehend davon den weiteren Fortgang des Geschäftsprozesses zu bestimmen.

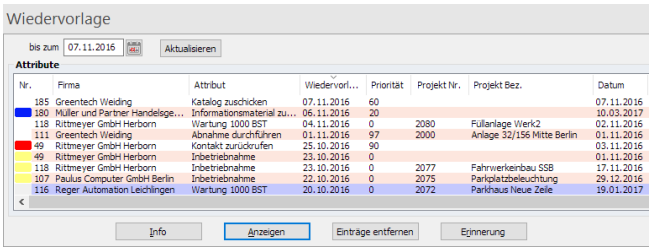

Abbildung 2: FactWork-Wiedervorlage zeigt ausstehende Arbeiten an mehreren Datensätzen

Die bisher vorhandenen Features des Workflow-Moduls erlauben bereits die Modellierung einer Vielzahl an denkbaren Einsatzszenarien. Trotzdem sind diese aktuell nicht ausreichend, wenn abzubildende Prozesse den direkten Einbezug des Nutzers erfordern oder weitere kontextspezifische Informationen vom Benutzer benötigt werden. Solche Prozesse sind mit diesem System im Moment nicht realisierbar, was die universelle Verwendbarkeit des Moduls

für Geschäftsprozesse im Kontext des ERP-Systems beeinträchtigt und somit einen erheblichen Nachteil darstellt. Dieser soll durch Implementierung geeigneter Funktionalitäten ausgeräumt werden, die im Folgenden erörtert werden.

ANFORDERUNGEN

Für die zusätzlichen notwendigen Fähigkeiten werden folgende funktionale Anforderungen ausgemacht:

- Workflow-Instanzen sollen auf eine Bestätigung durch den Nutzer warten können, bevor in den nächsten Zustand übergegangen wird, wenn dies fachlich notwendig wird. Damit kann z.B. das Warten auf die Sichtung eines Datensatzes realisiert werden: Erst wenn eine Sichtung und ggf. eine Korrektur der Daten erfolgt ist, kann eine Freigabe an die nächste Station erfolgen.
- Des Weiteren sollen Nutzer aktiv aus mehreren möglichen Entscheidungen wählen können, die in dem aktuellen Zustand des Prozesses fachlich möglich sind und den weiteren Fortgang des Prozesses beeinflussen.
- Geschäftsprozesse benötigen für deren weitere Ausführung oftmals zusätzliche Daten, die vom Benutzer eingefordert werden müssen. Dies kann z.B. durch das Ausfüllen eines Formulars geschehen, dessen eingegebene Daten an die Workflow-Instanz übermittelt und für dessen weitere Abarbeitung verwendet wird. Das Absenden eines solchen Formulars und die anschließende Weiterverarbeitung durch den Workflow soll nur möglich sein, wenn die vom Benutzer zur Verfügung gestellten Daten erfolgreich validiert worden sind.

Zu den dargelegten funktionalen Anforderungen für neue Benutzerinteraktionsmöglichkeiten gibt es zusätzliche gemeinsame weitere Anforderungen:

- Die aufgezählten Interaktionselemente sollen in Abhängigkeit des aktuellen Zustands des Workflows nur bestimmten Nutzern oder Nutzergruppen mit bestimmten Rollen zur Verfügung stehen, d.h. es muss ein Berechtigungskonzept geben.
- Da getroffene Entscheidungen und eingegebene Daten in Geschäftsprozessen kritischer Natur sein können und damit verbindlich sein sollten, ist eine Archivierung dieser zur Nachverfolgung in einer Historie notwendig. Diese soll alle wichtigen Informationen über getätigte Interaktionen sowie eingegebene Daten beinhalten und permanent gespeichert werden.
- Dem Benutzer müssen ausstehende Bestätigungen / zu treffende Entscheidungen sowie auszufüllende Formulare in einer geeigneten Art und Weise präsentiert und dafür geeignete Benutzeroberflächen geschaffen werden. Hierfür sollen verschiedene mögliche User Interface (UI)-Konzepte evaluiert und anschließend daraus ausgewählt werden.
- Die neuen Möglichkeiten sollen möglichst intuitiv und nahtlos im bestehenden FactWork-Workflow-Designer integriert sein. Der Großteil der Funktionalitäten soll dabei deklarativ, also ohne Programmierung seitens des Workflow-Erstellers abgedeckt sein. Für weiterführende komplexere Einsatzszenarien kann jedoch auch eine zusätzliche Programmierung mit der integrierten Skriptsprache fe.script notwendig sein.

Abbildung 3 zeigt die Anforderungen zusammengefasst in einem UML-Use-Case-Diagramm, welches zur Modellierung von Systemanforderungen üblich ist (Rupp, Queins und SOPHISTen 2012).

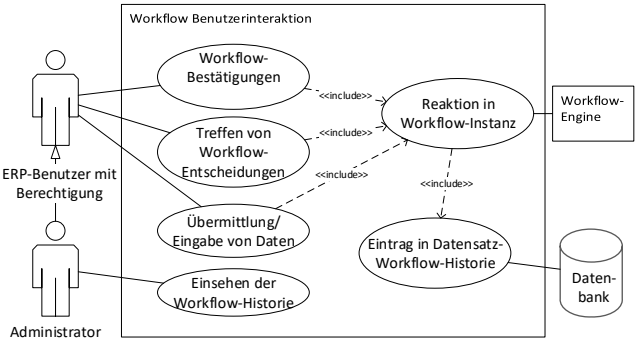

Abbildung 3: Use-Case-Diagramm zu den funktionalen Anforderungen

KONZEPT

In der Konzeptionsphase werden die genannten funktionalen Anforderungen in informationstechnisch realisierbare Konzepte überführt.

Wartende Aktionen an Datensätzen

Für das Warten einer Workflow-Instanz auf Bestätigung oder das aktive Treffen von Entscheidungen im Geschäftsprozess durch einen Benutzer werden als zunächst abstraktes Konzept sog. Aktionen an Datensätzen gewählt. Einem Stammdatensatz des ERP-Systems können durch den Workflow ein oder mehrere ausstehende Aktionen zugewiesen werden, die dann ein oder mehreren Nutzern im Kontext des Datensatzes präsentiert und von einem Nutzer bestätigt werden müssen bzw. aus denen er aktiv durch Auswahl eine Entscheidung treffen kann.

Im verwendeten Framework sind sog. Aktivitäten die kleinste Ausführungseinheit, die beim grafischen Design von Workflows als platzierbare Elemente im Designer zur Verfügung stehen. Jede einzelne Aktivität besitzt dabei eine bestimmte Funktion und erst die Gesamtheit aller Aktivitäten und deren strukturelle Anordnung bestimmt die Logik und den Ablauf des abgebildeten Geschäftsprozesses (White 2012). Aus diesem Grund ist auch für dieses Konzept die Erstellung einer neuen benutzerdefinierten Aktivität nötig, welche die technische Bezeichnung *WaitForAction* erhält.

Die Definition und Speicherung der zur Verfügung stehenden Aktionen soll dabei zunächst separat von der eigentlichen Struktur des Workflows stattfinden. Erst bei der Platzierung einer *WaitForAction*-Aktivität in der Workflow-Struktur soll eine Zuordnung durch Auswahl aus den verfügbaren Aktionen stattfinden. Dies ist sinnvoll, damit definierte Aktionen so an unterschiedlichen Stellen im Prozess wiederverwendet werden können.

Trifft die Workflow-Engine auf eine *WaitForAction*-Aktivität, so soll der Workflow an dieser Stelle im aktuellen Ausführungszweig pausiert werden, bis die Aktion von einem berechtigten Benutzer bestätigt wurde. Die Verwendung mehrerer *WaitForAction*-Aktivitäten mit ggf. unterschiedlichen zugeordneten Aktionsdefinitionen in parallelen Workflow-Ausführungszweigen kann das gleichzeitige Warten mehrerer Aktionen erwirken, was eine Entscheidungsmöglichkeit durch Nutzer bedingen kann (siehe Abbildung 4). Bei Betätigen einer Aktion wird der Workflow an der zugehörigen Stelle weiter ausgeführt. Weitere Aktionen in anderen Ausführungszweigen bleiben dann je nach modellierter Logik weiter im wartenden Zustand oder werden abgebrochen.

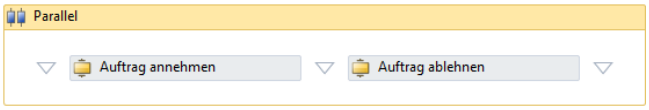

Abbildung 4: Zwei wartende Aktionen innerhalb einer *Parallel*-Aktivität

Daten werden mit Aktivitäten über sog. Ein- und Ausgabeargumente ausgetauscht. Als Minimum muss der neuen Aktivität dabei das Datensatz-Objekt übergeben werden, dem die wartende Aktion zugeordnet werden soll. In dieser Konstellation ist das Betätigen der Aktion dann allen Nutzern möglich, die auch für den entsprechenden Datensatz die Berechtigung besitzen. Dies kann weiter eingeschränkt werden, indem der Aktivität zusätzlich eine Liste von berechtigten Mitarbeiter-Objekten und/oder eine Liste von Wiedervorlagegruppen-Objekten übergeben wird, deren Mitglieder die Berechtigung erhalten. Damit wird das bestehende Berechtigungs- und Rollenkonzept des Wiedervorlagesystems wiederverwendet.

Das hier vorgestellte Konzept stellt eine gut geeignete Möglichkeit dar, eine kontextspezifische Benutzerinteraktion mit Prozessen in dieser Umgebung zu ermöglichen, da es die gegebenen Randbedingungen hervorragend berücksichtigt. Hierzu zählen u.a. die Eigenschaften und technischen Möglichkeiten des eingesetzten Frameworks sowie die Datensatzorientierte Arbeitsweise und bereits vorhandene Funktionen des ERP-Systems.

Formulare zur Datenübermittlung an Workflows

Formulare sollen dem Benutzer die Eingabe von Daten erlauben, die für die weitere Abarbeitung des Geschäftsprozesses von Nöten sind. Die Definition von Formularen soll dabei auf geeignete Art und Weise in einem zu erstellenden Formular-Designer stattfinden, der eine frei konfigurierbare Gestaltung von Eingabeformularen ermöglicht. Dies soll im Wesentlichen durch eine freie Anordnung von Formularfeldern aus mehreren möglichen üblichen Formularfeldtypen wie Textfeldern, Checkboxen, Radio-Buttons und Dropdown-Feldern geschehen. Jedes Formularfeld hat als konfigurierbare Eigenschaft mindestens einen Namen, eine Beschriftung und einen zugrundeliegenden Datentyp (wie z.B. Logisch, Text, Ganzzahl, Dezimalzahl, Datum/Uhrzeit). Je nach Feldtyp können weitere Konfigurationsmöglichkeiten wie Nachkommastellen, Mehrzeiligkeit, Vorgabewerte und Validierungsangaben sinnvoll sein.

Die Integration der so gestalteten Formulare in den Workflow-Ablauf erfolgt dabei durch die Verknüpfung mit dem bereits vorgestellten Konzept der Aktionen an Datensätzen. Einer Aktion kann dabei optional ein Formular zugeordnet werden. Ist dies der Fall, ist bei einer wartenden Workflow-Aktion zunächst das Ausfüllen des Formulars durch den Benutzer inkl. erfolgreicher Eingabevalidierung nötig. Das Abschicken des Formulars entspricht dem Auslösen der zugeordneten Aktion. Die eingegebenen Daten stehen dem Workflow direkt nach der *WaitForAction*-Aktivität zur Verfügung, indem diese um entsprechende Ein- und Ausgabeargumente erweitert wird.

Die Nutzung von Formularen stellt eine für den Benutzer einfache und intuitive Methode dar, zusätzliche für die Fortsetzung notwendige Prozess-Eingabedaten zu übermitteln. Sie ist mit klassischen Formularen in Papierform zu vergleichen, die bei der manuellen Abwicklung von Geschäftsprozessen anfallen und weitergereicht werden, um so den nächsten Bearbeitungsschritt anzustoßen. Die Knüpfung an das vorhandene Konzept der Datensatz-Aktionen erfolgt zunächst aus technischen Gründen, sorgt aber auch für eine Vereinheitlichung des Ansatzes, welche sowohl für den Workflow-Ersteller als auch dem Endnutzer Vorteile bringt.

Workflow-Historie

Alle wichtigen Nachverfolgungsinformationen wie Zeitstempel, durchführender Benutzer und Informationen über ausgeführte Aktionen sowie ggf. vom Benutzer zur Verfügung gestellte Daten in Formularen sollen permanent in der FactWork-Datenbank gespeichert werden, sobald diese entstehen. Diese sind dabei mit dem Datensatz zu verknüpfen, dem die jeweiligen Aktionen zugeordnet waren. Die Anzeige der Workflow-Historie zu einem Datensatz erfolgt in einem neu dafür zu erstellenden Dialog. Damit ist die Verbindlichkeit aller getätigten Prozessinteraktionen gewährleistet, über welche der neue Dialog eine Übersicht schafft und so auch als Kontrollinstrument dient.

Evaluierung geeigneter UI-Techniken

Die genannten Konzepte müssen auf geeignete Art und Weise in die FactWork-Umgebung integriert und hierfür geeignete Benutzeroberflächen geschaffen werden, wobei es jeweils verschiedene Alternativen gibt, aus denen eine sinnvolle Auswahl zu treffen ist.

Darstellung und Bestätigung der ausstehenden Aktionen: Ein oder mehrere Aktionen können auf das Auslösen durch einen berechtigten Nutzer im Kontext eines Stammdatensatzes warten. Die Signalisierung über ausstehende Arbeiten an einem Datensatz kann dabei weiterhin über das für diesen Zweck bestehende Wiedervorlagesystem erfolgen. Jedoch muss der Benutzer nach dem Öffnen eines Datensatzes über wartende Aktionen aufmerksam gemacht werden und es dafür eine geeignete Betätigungsmöglichkeit geben. Hierfür werden verschiedene Möglichkeiten ausgemacht:

- Aufploppen eines nicht-modalen Popup-Dialogs bei Öffnen des Datensatzes: Dies wäre eine prominente aber gleichzeitig sehr aufdringliche Möglichkeit zum Anbieten ausstehender Workflow-Aktionen. Der Dialog könnte zwar seitlich platziert werden, würde aber hierbei ggf. einen Teil der Datensatzmaske überdecken. Nicht immer wird ein Datensatz mit der Intension des Auslösens einer Workflow-Aktion geöffnet, sodass ein solches Vorgehen eventuell als störend empfunden werden könnte.
- Integration als zusätzlicher Reiter in Masken: Diverse Datensatzmasken in FactWork besitzen bereits mehrere Reiter (siehe Abbildung 5). Hier wäre die Einblendung eines zusätzlichen Reiters möglich, der bei Bedarf ausstehende Workflow-Aktionen anzeigen kann. Allerdings sind eine Vielzahl der einfacheren Masken reiterlos, des Weiteren würde die Integration in sämtliche Maskenklassen einen enormen Aufwand provozieren.

Abbildung 5: FactWork-Datensatzmaske mit mehreren Reitern

- Nutzung der neuen FactWork-Ribbon-Oberfläche: Seit der Version 10 besitzt FactWork eine moderne Ribbon-Oberfläche im Stil der Microsoft Office Produkte, wodurch eine komplette strukturelle Neuanordnung und Modernisierung der Menüführung erreicht wurde. Ein grundlegendes Feature solcher Ribbon-Oberflächen ist das Anbieten von kontextspezifischer Funktionalität (siehe Abbildung 6). Da ausstehende Workflow-Aktionen immer kontextspezifisch und vom geöffneten Datensatz abhängig sind, kann bei deren Vorhandensein eine zusätzliche Kontext-Tools-Kategorie eingeblendet werden, die über das Anbieten von Schaltflächen das Auslösen der Aktionen ermöglicht.

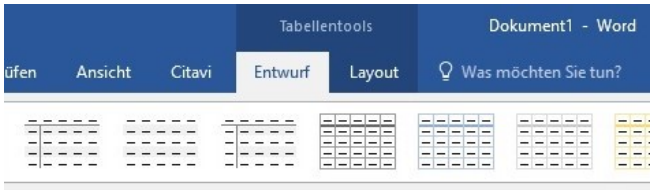

Abbildung 6: Kontextspezifisches Ribbon-Toolsmenü am Beispiel Microsoft Word

Nach Abwägung der Vor- und Nachteile der verschiedenen Methoden wird sich für die Verwendung der letzteren Ribbon-Methode entschieden. Diese ist wenig aufdringlich und nicht störend, aber gleichzeitig informativ und effektiv. Zusätzlich folgt sie der modernen kontextorientierten Bedienerführung der neuen Oberfläche des ERP-Systems.

Formulardesign und -darstellung: Auch für die Gestaltung des Formulardesigners und die spätere Anzeige des Formulars sind verschiedene Oberflächentechniken denkbar:

- Schaffung einer webbasierten Oberfläche zur Bedienung aus dem Browser (sowohl für Formulardesign als auch Präsentation): Dies hätte zur Folge, dass prinzipiell auch externe Benutzer in Workflows mit einbezogen werden könnten, die sich nicht innerhalb der Firma befinden oder nicht über die Installation eines FactWork-Clients verfügen. Der Zugang zu Formularen würde durch das Versenden von Links stattfinden, die Formulare wären ggf. durch Zugriffsbeschränkungen geschützt.
  Gängige Workflow-Management-Systeme bieten meist nur unzureichende Unterstützung für unternehmensübergreifende (B2B) Geschäftsprozesse, da dies auch nicht deren primäres Designziel darstellt (Bussler 2002). Das Anbieten von B2B und B2C (Business-to-Consumer) Fähigkeiten in Workflows sind aber durchaus wünschenswerte Eigenschaften für automatisierte Geschäftsbeziehungen über Unternehmensgrenzen hinweg (Müller 2006). Die Bereitstellung webbasierter Formularoberflächen aus dem Workflow heraus wäre ein Schritt in diese Richtung, wenn auch nur im Sinne der direkten Einbeziehung von Endnutzern in Geschäftsprozesse (im Gegensatz zu hier nicht behandelter direkter System-zu-System-Kommunikation).
- Direkte Integration und Aufrufbarkeit des Formulardesigners aus dem bestehenden Workflow-Designer im FactWork-Client: Der integrierte Workflow-Designer der WWF verwendet die .NET-Oberflächentechnologie Windows Presentation Foundation (WPF) und deren Auszeichnungssprache Extensible Application Markup

Language (XAML). Für eine nahtlose Anbindung an die bestehende Design-Infrastruktur wäre die Erstellung des Formulardesigners als WPF-Dialog vorteilhaft. Dadurch könnten auch bereits vorhandene Komponenten wie fe.script-Ausdruckseditoren wiederverwendet werden. Das so gestaltete Formular würde direkt im Datenbankobjekt der Workflow-Definition als XAML-Dokument abgespeichert werden, wie auch die Struktur des Workflows selbst.
Die Präsentation der Formulare würde ebenfalls direkt im FactWork-Client auf Basis dieser Technologie stattfinden. Somit wäre nach dem Öffnen eines Datensatzes und dem Betätigen einer Workflow-Aktion sofort das Ausfüllen und Abschicken des Formulars in einem Dialog möglich, was für den Nutzer einen durchgängigen und intuitiven Arbeitsablauf darstellt.

Auch wenn eine webbasierte Formulartechnik die Reichweite und Einsatzmöglichkeiten erhöhen würde, so liegt der Haupteinsatzzweck des Workflow-Moduls doch in ERP-integrierten Geschäftsprozessen, wodurch der Fokus auf die Benutzerfreundlichkeit für unternehmensinterne ERP-Nutzer gelegt wird. Daher fällt die Wahl auf die direkte Integration aller Benutzerinteraktionsfähigkeiten in den FactWork-Client. Eine mögliche zukünftige Realisierung eines Web-Moduls wird aber nicht ausgeschlossen, wenn hierfür Bedarf besteht.

Da auch die Workflow-Historie das Anzeigen von Formularinhalten ermöglichen soll, wird für diese ebenfalls die WPF-Technik gewählt und so bestehende UI-Module wiederverwendet.

REALISIERUNG

Für die Entwicklung der ERP-Software FactWork wird derzeit die Entwicklungsumgebung Microsoft Visual Studio 2015 verwendet. Diese besteht aus einer Vielzahl an Modulen, die in einer Projektmappe zusammengefasst sind. Der Großteil der Module ist in der Programmiersprache C++ verfasst, es existieren aber auch diverse in C# programmierte .NET-Module. Der Workflow-Designer der WWF nutzt die .NET-Oberflächentechnologie WPF, jedoch basiert die FactWork-Oberfläche größtenteils auf C++-Techniken. Das Workflow-Modul ist daher mit Hilfe der Programmiersprache C++/CLI realisiert, welches Standard-C++ um diverse Syntaxerweiterungen ergänzt, damit Komponenten des .NET-Frameworks angesprochen werden bzw. selbst solche Komponenten verfasst werden können (Sivakumar 2007). Für die im Rahmen dieses Projekts entstehenden Erweiterungen wird deshalb hauptsächlich ebenfalls mit dieser Technik gearbeitet.

Datenbank-Schema

FactWork verwendet als Persistenzschicht die objektorientierte Datenbank FastObjects der Firma Actian Corporation (Actian Corporation 2017). Geschäftsobjekte werden auf diese Weise nicht in relationalen SQL-Tabellen gespeichert, sondern direkt in ihrer Objektform ohne notwendiges Objektrelationales Mapping (ORM). Das Datenbankschema wird direkt aus den in C++-Headerdateien angegebenen Klassendefinitionen über einen Präprozessor abgeleitet, der Austausch von Daten mit der Datenbank erfolgt zur Laufzeit auf Basis von C++-Objekten im Speicher.

Für die Persistenz der Aktions- und Formulardefinitionen wird die Klasse *PtWorkflowDefinition*, welche für die Speicherung einer Workflow-Definition zuständig ist, um zwei zugehörige Binary Large Object (BLOB)-Felder erweitert, in denen die Definitionen abgelegt werden. Eine Abbildung als reine native C++-Klassenstruktur, was dem üblichen Vorgehen im ERP-System bzw. der verwendeten Datenbank entspricht, wäre zwar möglich, hätte jedoch die Erstellung einer Vielzahl an C++-Klassen und ein kompliziertes Mapping zwischen den beiden Technologiewelten zur Folge. Stattdessen wird sich für eine direkte effiziente Speicherung entschieden, indem die .NET-Objektstrukturen mithilfe der XAML-Technik serialisiert werden und in den nativen Datenbankklassen nur als BLOB-Felder platziert werden.

```
persistent class PtWorkflowDefinition
{
  PtString Bezeichnung;
  PtBlob XamlDefinition;
  PtBool Disabled;

  PtBlob Actions;
  PtBlob Forms;
};
```

Listing 1: Klasse PtWorkflowDefinition (Auszug)

Die Klasse *PtWorkflowInstance* repräsentiert eine aktuell ausgeführte, sich im wartenden Zustand befindliche Workflow-Instanz. Die Klasse wird um das Set-Member *ActionObjects* ergänzt, in welches alle relevanten Datensätze eingetragen werden, für die die Workflow-Instanz ausstehende wartende Aktionen beinhaltet. Damit beim Öffnen eines Datensatzes über eine Abfrage schnell passende Workflow-Instanzen gefunden werden, wird für dieses Feld ein Datenbankindex gesetzt. Nach Identifizierung passender Workflow-Instanzen müssen weitere detaillierte Informationen zu den Aktionen aus den Instanzdaten und der Workflow-Definition nachgeladen werden.

```
persistent class PtWorkflowInstance
{
  PtString InstanceId;
  PtWorkflowDefinitionOndemand WorkflowDefinition;

  WorkflowStatus WorkflowStatus;
  PtBlob InstanceData;
  PtBlob InstanceMetaData;

  PtToolBaseOndemandSet ActionObjects;
  // ...
};
```

Listing 2: Klasse PtWorkflowInstance (Auszug)

Eine neue Klasse *PtWorkflowHistory* wird zur Speicherung der Workflow-Historien-Einträge verwendet. Diese enthält alle nötigen Nachverfolgungsinformationen wie das zugeordnete Datensatzobjekt, Datum/Uhrzeit, auslösender Mitarbeiter, Infotext, Zuordnung zu Workflow-Instanz und Workflow-Definition sowie ggf. vom Benutzer eingegebene serialisierte Formulardaten.

```
persistent class PtWorkflowHistory
{
  PtToolBaseOndemand TargetObject;
  PtDateTime DateTime;
  PtMitarbeiterOndemand User;
  PtString Info;

  PtWorkflowDefinitionOndemand WorkflowDefinition;
  PtWorkflowInstanceOndemand WorkflowInstance;
  PtString WorkflowInstanceId;

  PtString ActionId;
  PtString FormId;
  PtBlob FormData;
};
```

Listing 3: Klasse PtWorkflowHistory (Auszug)

Aktionseditor

Zur Definition der Workflow-Aktionen wird ein neuer Dialog erstellt, der als Aktionseditor das Anlegen und die Bearbeitung dieser ermöglicht (siehe Abbildung 7). Neben wichtigen Eigenschaften wie Bezeichnung und Beschreibung, welche später dem Benutzer bei der Präsentation ausstehender Aktionen angezeigt werden, ist optional die Zuordnung eines Formulars vorgesehen. Beabsichtigt ein Benutzer das Auslösen einer solchen Aktion, so ist zunächst das Formular erfolgreich auszufüllen. Des Weiteren können an dieser Stelle bereits als globale Vorgabe fe.script-Ausdrücke zur Bestimmung des Zieldatensatzes der Aktion sowie der Berechtigungsvorgaben hinterlegt werden.

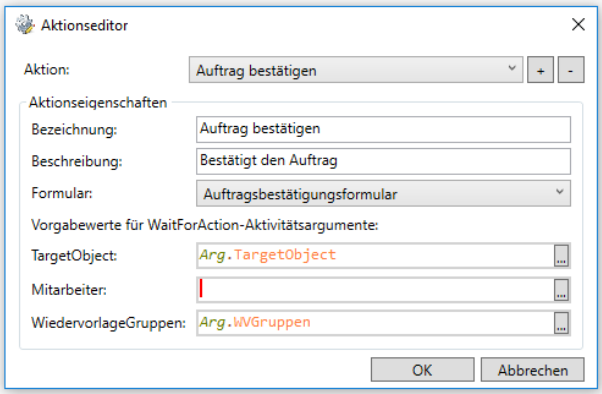

Abbildung 7: Aktionseditor zur Definition von Workflow-Aktionen

Die Abbildung von Workflow-Aktionen erfolgt durch eine neue verwaltete Klasse *WorkflowAction*, die alle zu treffenden Optionen als Eigenschaftsmitglieder (sog. *Properties*) enthält. Diese werden mit üblicher WPF-Technik über Datenbindung direkt an die Dialogelemente gebunden. Alle auf diese Art und Weise definierten Aktionen werden aus bereits erläuterten Gründen zur Persistenz als XAML-Dokumente serialisiert und im zugehörigen BLOB-Feld der *PtWorkflow-Definition*-Klasse abgelegt. Die gewählten Vorgehensweisen minimieren einerseits die nötige Programmierarbeit, zudem lehnen sie sich an die Arbeitsweise des Frameworks sowie bisherige Konzepte innerhalb des FactWork Workflow-Moduls an.

```
public ref class WorkflowAction
{
public:
  property Guid ID;
  property String^ Name;
  property String^ Description;

  property WorkflowForm^ Form;

  property InArgument<Object^>^ TargetObject;
  property InArgument<IEnumerable<Object^>^>^
      WiedervorlageGruppen;
  property InArgument<IEnumerable<Object^>^>^
      Mitarbeiter;
};
```

Listing 4: Klasse WorkflowAction (Auszug)

Formulareditor

Ein neuer Formulareditor-Dialog dient zur Gestaltung von Workflow-Formularen (siehe Abbildung 8). Dieser erlaubt zunächst die Spezifizierung allgemeiner Eigenschaften wie Bezeichnung und Beschreibung. Die Angabe einer fe.script-Validierungsroutine ermöglicht die zusammenhängende Prüfung aller eingegebenen Formularwerte auf Gültigkeit. Ohne erfolgreiche Validierung wird später das Absenden des Formulars verhindert.

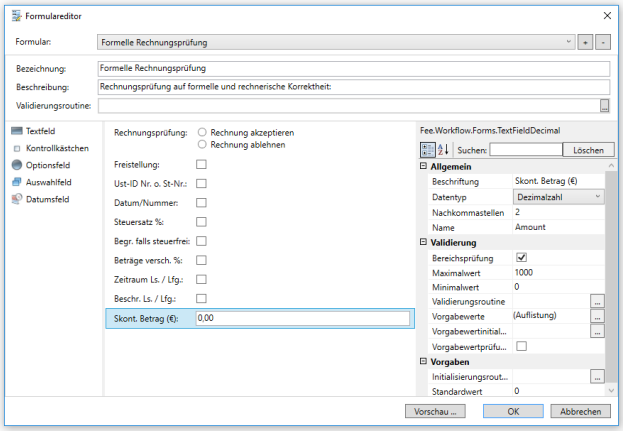

Abbildung 8: Formulareditor zur Gestaltung von Benutzer-Formularen

Die wichtigste Aufgabe des Formulardesigners ist die freie Gestaltungsmöglichkeit des Formularaufbaus. Dazu kann auf

der linken Seite aus verschiedenen verfügbaren Formularfeldtypen ausgewählt werden. Via Drag&Drop kann durch Ziehen eines ausgewählten Feldtyps in den Gestaltungsbereich in der Mitte ein neues Formularfeld hinzugefügt werden. Ebenso ist durch Ziehen mit der Maus die Anordnung der Felder auf einfache Weise zu ändern.

Auf der rechten Seite sind über einen Eigenschaftseditor die zahlreichen Einstellungsmöglichkeiten eines Formularfeldes zu ändern. Je nach Feldtyp stehen unterschiedliche Optionen zur Verfügung. Diese reichen beispielsweise von zugrundeliegendem Datentyp über Vorgabewerte, Mehrzeiligkeit, Bereichsangaben, Nachkommastellen bis hin zu weiteren zusätzlichen Validierungsmöglichkeiten.

Die Speicherung eines Formulars erfolgt in der Klasse *WorkflowForm*, welche alle genannten Eigenschaften abbildet:

```
public ref class WorkflowForm
{
public:
  property Guid ID;
  property String^ Name;
  property String^ Description;

  property FormFieldCollection^ Fields;
  property Activity<bool>^ ValidationRoutine;
};
```

Listing 5: Klasse WorkflowForm (Auszug)

Für jeden Formularfeldtyp existiert eine eigene Klasse, welche jeweils die verschiedenen Eigenschaften und Optionen eines Feldtyps abdeckt. Diese leiten alle von der Klasse *FormField* ab, welche alle gemeinsamen Charakteristiken von Formularfeldern beinhaltet. So hat jedes Formularfeld einen technischen Namen und eine Beschriftung, Einzel-Initialisierungs- und Validierungsroutinen sowie einen Standardwert und einen aktuellen Wert. Die Klasse *FormField* enthält einen generischen Typparameter *TValue*, welches den zugrundeliegenden Datentyp eines Feldes darstellt, der üblicherweise bei der Ableitung angegeben wird.

```
generic <typename TValue>
public ref class FormField
{
public:
  property String^ Name;
  property String^ Label;

  property Activity^ InitializationRoutine;
  property Activity<bool>^ ValidationRoutine;

  property TValue DefaultValue;
  property TValue Value;
};
```

Listing 6: Klasse FormField (Auszug)

Die Gestaltung der Formulare mit den gewählten Methoden erfolgt aus denselben Gründen wie bei den bereits vorgestellten Workflow-Aktionen, nämlich eine nahtlose Integration in und Adaption der bestehenden Infrastruktur. Weiterhin bietet die gewählte generisch arbeitende Klassenstruktur für Formularfelder eine solide Basis für Formularfeldtypen verschiedenster komplexer Art, die über die Erstellung weiterer Ableitungen beliebige Ergänzungen zulassen und daher eine gute Grundlage für zukünftige Erweiterungen bilden.

WaitForAction-Aktivität

Eine neue benutzerdefinierte Aktivität *WaitForAction* wird als gleichnamige Klasse erstellt, indem von der hierzu im WWF-Framework vorhandenen Klasse *NativeActivity* abgeleitet wird. Diese Aktivität ist später im Workflow-Designer zur Platzierung in einer Workflow-Struktur verfügbar. Trifft die Workflow-Engine in einem Ausführungszweig auf diese, so wird der Workflow an dieser Stelle pausiert, bis ein berechtigter Anwender die Aktion bestätigt.

Primär ist für die Aktivität die Spezifikation einer vorher definierten Workflow-Aktion möglich. Des Weiteren sind der Zieldatensatz und Berechtigungsoptionen anzugeben, falls diese von der Aktionsvorgabe abweichen sollen. Zusätzlich sind Ein- und Ausgabeparameter zum Austausch von Formularinhalten verfügbar, falls die Aktion mit einem Formular verknüpft wurde.

Damit eine benutzerdefinierte Aktivität einen Wartezustand des Workflows einleiten kann, muss dies durch passende Überschreibung der virtuellen Eigenschaft *CanInduceIdle* bekanntgegeben werden. Ansonsten wird das Verhalten der Aktivität hauptsächlich durch Implementierung zweier virtueller Methoden bestimmt (White 2012):

- In *CacheMetadata* werden Aktivitätseigenschaften und -argumente auf Gültigkeit validiert und Strukturdetails der Implementierung bekanntgegeben.
- *Execute* beinhaltet die eigentliche Ausführungslogik der Aktivität. Diese besteht im Wesentlichen aus der Bekanntgabe der Informationen über die ausstehende neue Workflow-Aktion und Hinterlegung dieser in der Datenbank sowie Erzeugung eines Workflow-Historien-Eintrags.

```
public ref class WaitForAction : public
    NativeActivity
{
public:
  property WorkflowAction^ Action;

  property InArgument<Object^>^ TargetObject;
  property InArgument<IEnumerable<Object^>^>^
      WiedervorlageGruppen;
  property InArgument<IEnumerable<Object^>^>^
      Mitarbeiter;

  property InArgument<IEnumerable<KeyValuePair<
      String^,Object^>>^>^ FormInput;
  property OutArgument<Dictionary<String^,Object
      ^>^>^ FormData;

protected:
  virtual void CacheMetadata(
      NativeActivityMetadata) override;
  virtual void Execute(NativeActivityContext^
      context) override;
```

```
  property bool CanInduceIdle {
    virtual bool get() override {
      return true;
  }}
};
```

Listing 7: Klasse WaitForAction (Auszug)

Insgesamt bringt die neue Aktivität alle bereits vorgestellten Benutzerinteraktionskonzepte unter einen Hut und erlaubt die einfache und intuitive Einbindung dieser in den Prozessablauf über den vom Framework vorgesehenen Weg.

Ribbon-Erweiterungen für ausstehende Aktionen

Damit dem Benutzer ausstehende Aktionen signalisiert werden können, wird die FactWork-Ribbon-Oberfläche um einen neuen kontextspezifischen Tools-Reiter „Workflow“ erweitert. Dieser wird nur angezeigt, wenn der Mitarbeiter für den aktuell geöffneten Datensatz ausstehende Workflow-Aktionen betätigen kann (siehe Abbildung 9). Prinzipiell können mehrere Workflows auf einen bestimmten Datensatz warten, diese werden in mehreren Spalten mit dem jeweiligen Workflow-Namen angezeigt. Zusätzlich ist bei entsprechender Berechtigung des Nutzers der Aufruf der Workflow-Historie des Datensatzes möglich, falls hierfür bereits Einträge bestehen.

Abbildung 9: Ribbon-Buttons für ausstehende Workflow-Aktionen

Technisch erfolgt die Bekanntgabe der einzelnen Elemente sowie die Ereignisbehandlung bei Betätigung dieser über die dafür vorgesehenen Schnittstellen der verwendeten Oberflächenbibliothek. Zusammenfassend werden die Möglichkeiten der neuen Ribbon-Oberfläche des FactWork-Clients optimal ausgenutzt. Das realisierte Bedienkonzept ist zudem für zukünftige ähnliche Ergänzungen des Clients als Beispiel zu sehen und richtungsweisend, von der ursprünglich flachen Menüführung immer mehr hin zu kontextspezifischen adaptiven Menüs.

Workflow-Historie

Die Anzeige der Workflow-Historie erfolgt in einem neu erstellten zweiteiligen Dialog (siehe Abbildung 10). Auf der einen Seite stellt eine mehrspaltige Liste alle Historieneinträge des Datensatzes mit allen relevanten Informationen wie Datum/Uhrzeit, ausführender Benutzer, Informationstext und erzeugender Workflow bereit. Daneben erscheint bei angeklicktem Listeneintrag ein Abbild des Formulars inkl. eingegebener Daten, sofern der ausgewählten Aktion ein Formular zugeordnet war.

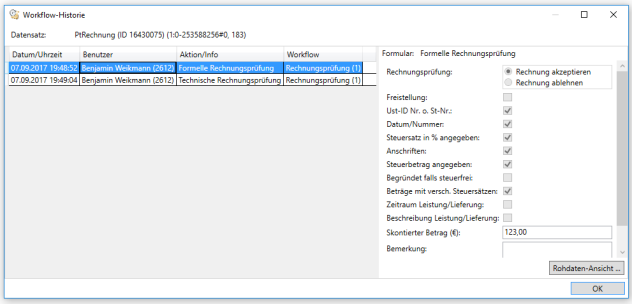

Abbildung 10: Workflow-Historie zeigt Details zu ausgeführten Workflow-Aktionen und eingegebene Formulardaten

Alles in allem ermöglicht die Workflow-Historie einen übersichtlichen Überblick über alle im Rahmen von Workflow-Benutzerinteraktion entstandener Geschäftsvorfälle an einem Datensatz inkl. sämtlicher relevanter Details hierzu.

ENTWICKLUNG EINES BEISPIEL-WORKFLOWS

Durch die Entwicklung eines Beispiel-Workflows sollen die Funktionsfähigkeit und die Einsatzmöglichkeiten der neu gewonnenen Fähigkeiten aufgezeigt werden.

Ausgangsbasis

Zur Demonstration der bisherigen Fähigkeiten des Workflow-Moduls existiert bislang ein rudimentärer Beispiel-Workflow, der einen einfachen möglichen Prozess der Rechnungsprüfung abbildet. Dieser nutzt die vormals ausschließlich vorhandenen Funktionen zur Benutzereinbindung, also das FactWork-Wiedervorlagesystem sowie den Versand von E-Mails.

Der Prozess wird im Hintergrund gestartet, sobald eine neue Rechnung im ERP-System angelegt wird. Er besteht zum einen aus der Station Zahlungsfreigabe, in der entweder ein Projektleiter, sofern dies für die Rechnung zutrifft, oder ein Mitarbeiter der Einkaufsabteilung die Aufgabe des Prüfens der Rechnung zugeteilt bekommt und sein Einverständnis zur Zahlung der Rechnung erteilen muss. Dazu wird für den Mitarbeiter bzw. die Wiedervorlagegruppe Einkauf (≡ Rolle) eine Wiedervorlage angelegt, welche die betreffenden Benutzer an prominenter Stelle im FactWork-Client über die ausstehende Arbeit informiert. Öffnet dieser den zugehörigen Datensatz (z.B. direkt aus der Wiedervorlageliste heraus), so kann dieser nach erfolgreicher Prüfung das in der

Rechnungsmaske standardmäßig vorhandene Feld Freigabestatus (siehe Abbildung 11) auf den entsprechenden Wert setzen und die Rechnung anschließend abspeichern.

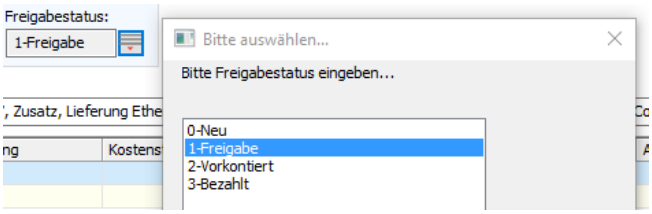

Abbildung 11: Rechnungsfeld Freigabestatus

Vom Workflow wird eine solche Änderung an der Rechnung erkannt, da dieser eine Datensatzüberwachung für das Rechnungsobjekt initiiert hat und vom Datenbankserver aktiv über die Speicherung des Objekts informiert wird. Nach einer Prüfung des Freigabestatusfeldes kann so der geänderte fachliche Zustand festgestellt und in den nächsten Zustand gewechselt werden. Erfolgt die Zahlungsfreigabe nicht innerhalb einer bestimmten Zeit, werden die Nutzer zusätzlich per E-Mail über die zu erledigende Tätigkeit informiert. In einem weiteren Schritt erfolgt die sog. Vorkontierung durch die Finanzbuchaltung, deren Ablauf aber identisch mit dem hier vorgestellten Ablauf der Zahlungsfreigabe ist.

Dieses Beispiel zeigt zwar auf, dass mit den zunächst vorhandenen Möglichkeiten durchaus sinnvolle Geschäftsprozesse zu modellieren waren, macht aber auch die Grenzen des Systems deutlich:

- Die Übermittlung von Daten an den Workflow war nur möglich, da es in der Stammdatensatzmaske für Rechnungen ein passendes Datenfeld gibt, das die gewünschte Information abbildet. Die Bereitstellung weiterer Daten seitens des Nutzers, v.a. exklusiver im Kontext des Workflows relevanter Daten, war nicht möglich.
- Die Ereignisse der erfolgten Zahlungsfreigabe und Vorkontierung konnten vom Workflow auch nur passiv über eine Datensatzüberwachung und durch Berücksichtigung regulärer Datensatzfelder erkannt werden. Für eine aktive Einflussnahme oder sogar das Treffen von kontextspezifischen Workflow-Entscheidungen durch den Nutzer gab es keine Möglichkeiten.

Geschäftsprozess-Analyse

Anhand eines neuen Beispiel-Workflows, der alle nun zur Verfügung stehenden Möglichkeiten der Benutzerinteraktion berücksichtigt, sollen die Vorteile dieser gegenüber dem bisherigem System aufgezeigt werden. Dies soll wieder am Beispiel der Rechnungsprüfung stattfinden, dabei aber wesentlich vertieft und weiter ausgebaut werden. Hierzu wird der in der Firma F.EE tatsächlich im Einsatz befindliche gelebte Prozess der Rechnungsprüfung als Vorlage herangezogen, der aktuell über das Dokumentenmanagementsystem DocuWare abgewickelt wird.

In DocuWare werden alle schriftlich oder digital eingehenden Rechnungen elektronisch archiviert. Die Rechnungsprüfung findet auf Basis dieser erfassten Rechnungen statt und erfolgt in zwei Schritten:

- Die Rechnung wird anhand einer Liste mit diversen Kriterien auf formelle und rechnerische Korrektheit geprüft.
- In der anschließenden technischen Rechnungsprüfung wird die Rechnungsstellung auf deren Berechtigung und sachliche Korrektheit überprüft.

Einem für eine Station zuständigen Sachbearbeiter werden in einer Liste alle ausstehenden Rechnungen präsentiert. Dieser nimmt für eine Rechnung die entsprechenden Prüfungen vor und dokumentiert bei Erfolg die Korrektheit durch die Platzierung eines sog. Stempels auf der elektronischen Rechnung (siehe Abbildung 12). Anschließend erfolgt die Zuteilung der Rechnung an den zugehörigen Bearbeiter für den nächsten Schritt. Bei der Feststellung von Unstimmigkeiten muss dagegen Kontakt mit dem Lieferanten aufgenommen werden, sodass die Rechnung ggf. korrigiert werden kann oder u.U. ganz zurückgezogen wird.

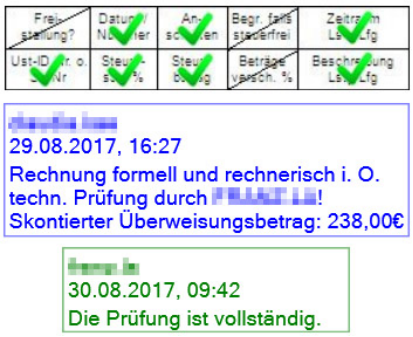

Abbildung 12: Stempel auf elektronisch archivierter Rechnung dokumentieren erfolgte Rechnungsprüfung

Implementierung

Der bisher extern durchgeführte Rechnungsprüfungsprozess soll unter Nutzung der nun zur Verfügung stehenden Möglichkeiten als Workflow realisiert werden, der komplett innerhalb des FactWork-ERP-Clients abgewickelt wird. Für die Interaktion mit den Benutzern wird insbesondere das Konzept der wartenden Aktionen an Datensätzen (in diesem Fall die betreffende Rechnung) und zur Angabe nötiger zusätzlicher Informationen im Laufe des Prozesses die Formularfunktion verwendet. Die Dokumentation und Nachverfolgungsmöglichkeit getätigter Rechnungsprüfungen wird durch die Workflow-Historie abgedeckt.

Für die beiden Teilstationen der formellen/rechnerischen und der technischen Rechnungsprüfung wird jeweils eine

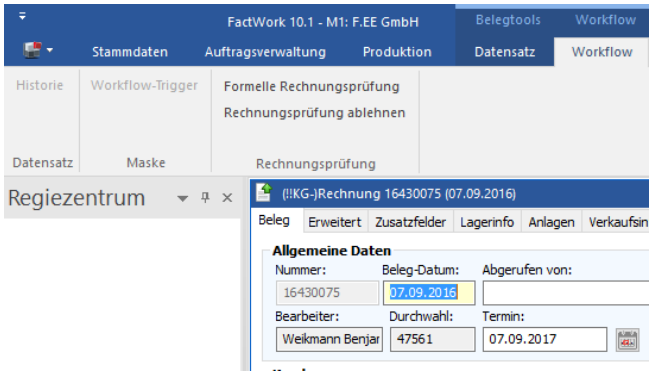

Abbildung 13: Wartende Aktionen bei geöffneter Rechnung

Workflow-Aktion angelegt, die später einem zuständigen Sachbearbeiter beim Öffnen der Rechnung zur Bestätigung angezeigt wird (siehe Abbildung 13). Die Zuordnung zu Benutzern wird über zwei Wiedervorlagegruppen realisiert, die die entsprechenden Rollen widerspiegeln. Eine dritte Aktion „Rechnungsprüfung ablehnen“ nimmt als Ausnahme eine Sonderrolle ein: Soll eine Rechnungsprüfung endgültig abgelehnt werden, da ihre Forderung unberechtigt ist oder die Rechnung zurückgezogen wurde, dann wird durch Auslösen dieser Aktion der Workflow abgebrochen und die Rechnung entsprechend gekennzeichnet.

Den beiden Aktionen zur Bestätigung einer Teil-Rechnungsprüfung wird jeweils ein Formular zugeordnet, in welchem der Nutzer zusätzliche Angaben zum getätigten Schritt machen muss. Abbildung 14 zeigt das Formular für die formelle/rechnerischen Rechnungsprüfung, welches folgende Felder beinhaltet:

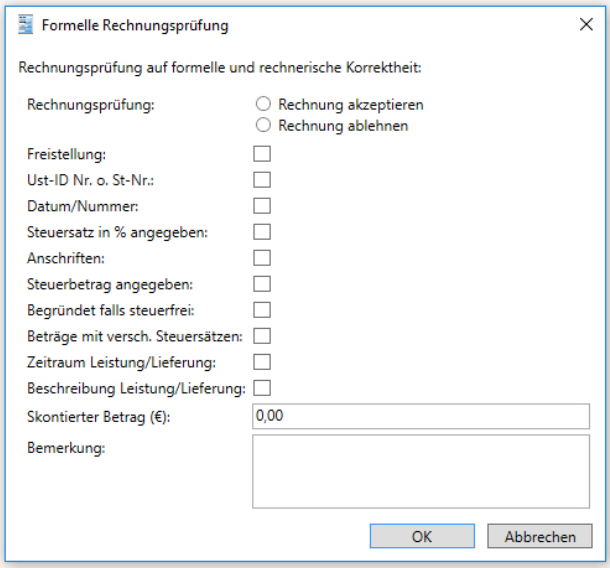

Abbildung 14: Formular für formelle Rechnungsprüfung

- Radio-Buttons „Rechnung akzeptieren“ und „Rechnung ablehnen“: Die hier zusätzlich zu treffende Auswahl soll die Intention des Bearbeiters explizit und verbindlich bestätigen. Eine an dieser Stelle abgelehnte Rechnungsprüfung führt nicht zur endgültigen Ablehnung, sondern nur zur Zurückstellung der Rechnungsprüfung zur späteren erneuten Vorlage (ggf. zwischenzeitliche Kontaktaufnahme mit Lieferanten).
- Diverse Kontrollhäkchen zur Angabe und Bestätigung verschiedener Merkmale einer geprüften Rechnung
- Textfeld „Skontierter Rechnungsbetrag“ zur Währungseingabe des auf der Rechnung angegebenen Gesamtbetrags
- Feld Bemerkung für zusätzliche Anmerkungen, hier eingegebener Text wird zudem weiteren Sachbearbeitern im Laufe des Workflows ebenfalls vorgelegt

In Abbildung 15 wird das erstellte Formular zur technischen Rechnungsprüfung dargestellt. Dieses enthält neben Bestätigungs-/Ablehnungsoptionen ebenso wieder das Bemerkungsfeld, das die Eingaben vorhergehender Sachbearbeiter zeigt bzw. welches um weitere Angaben ergänzt werden kann.

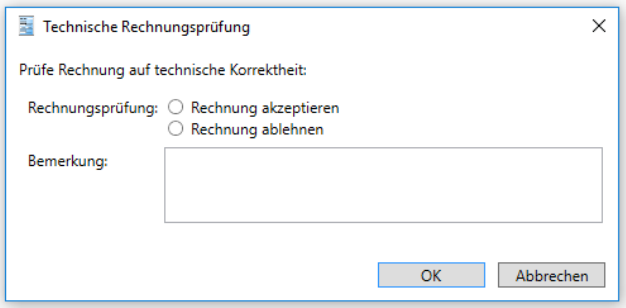

Abbildung 15: Formular für technische Rechnungsprüfung

Abbildung 16 beschreibt den im FactWork-Workflow-Designer erstellten finalen Ablauf der Zahlungsfreigabe. Die Prozesslogik wird innerhalb einer sog. *Flowchart*-Aktivität modelliert. Diese erlaubt mit Hilfe ihres zugehörigen Designers die Abbildung von Prozessteilen im Stil von Ablaufdiagrammen. Aktivitäten können innerhalb des Designerfelds frei platziert werden, deren logische Abfolge kann über das Setzen von Verbindungspfeilen bestimmt werden. Des weiteren werden Verzweigungen unterstützt, deren einzuschlagende Richtung zur Laufzeit über die Auswertung von angegebenen Bedingungen bestimmt wird (Collins 2010).

Zunächst erfolgt die Erstellung einer Wiedervorlage zur formellen und rechnerischen Prüfung. Anschließend wird auf das Auslösen der entsprechenden Aktion gewartet. Wird die Rechnungsprüfung an dieser Stelle abgelehnt, so wird diese zunächst zurückgestellt. Nach einer Kontaktaufnahme mit dem Lieferanten und einer ggf. erfolgten Korrektur erfolgt eine erneute Prüfung. Wird die Rechnung akzeptiert, so geht diese an den nächsten Bearbeiter zur technischen

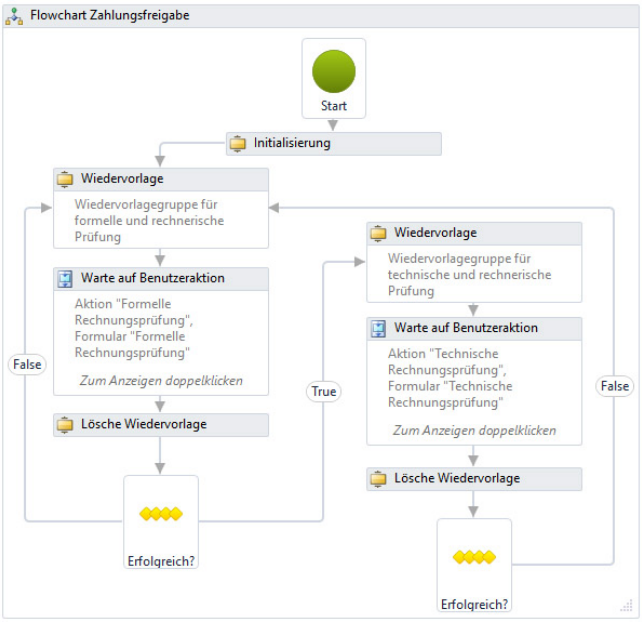

Abbildung 16: Flowchart Ablauf Zahlungsfreigabe

Prüfung. Ist diese nicht erfolgreich, so wird sie erneut zurück an den vorhergehenden Bearbeiter verwiesen. Etwaige Begründungen können über das Bemerkungsfeld der jeweiligen Formulare ausgetauscht werden. Nach Akzeptanz beider Prüfungen ist die Zahlungsfreigabe abgeschlossen.

In Abbildung 17 wird der detaillierte Ablauf des Wartens auf eine Benutzeraktion dargestellt. Neben der eigentlichen *WaitForAction*-Aktivität, welche das Warten auf die primäre Workflow-Aktion realisiert, gibt es innerhalb der umgebenden *Parallel*-Aktivität zwei weitere parallel ausgeführte Ausführungszweige: Zum einen wird auf eine weitere Workflow-Aktion zur endgültigen Rechnungsprüfungsablehnung gewartet. Wird diese betätigt, so wird an dieser Stelle der Workflow sofort beendet und die Rechnung als abgelehnt gekennzeichnet. Daneben findet eine E-Mail-

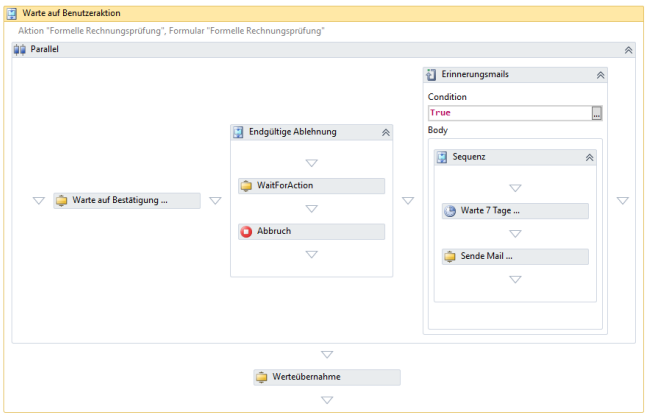

Abbildung 17: Ablauf des Wartens auf formelle Rechnungsprüfung

Erinnerungssequenz statt: Wird innerhalb von 7 Tagen (und danach alle weitere 7 Tage) keine Aktion betätigt, so werden die zuständigen Mitarbeiter über die ausstehende Tätigkeit per E-Mail informiert.

Test & Fazit

Nach erfolgter Modellierung wurde der so abgebildete Prozess erfolgreich in einer Testumgebung evaluiert. Dabei wurden alle beteiligten Nutzerrollen eingenommen und sämtliche Prozessschritte besonders im Hinblick auf die neu geschaffenen Funktionen zur Benutzerinteraktion überprüft. Der wesentliche Vorteil des auf diese Art und Weise realisierten Workflows im Vergleich zur bestehenden Lösung ist, dass dieser komplett innerhalb der ERP-Software abgewickelt wird. Der Einsatz externer Tools ist für solche Arten von Prozessen nicht mehr nötig, was einen erheblichen Zugewinn darstellt.

ZUSAMMENFASSUNG

Die Einbeziehung des Benutzers, wenn dies fachlich notwendig ist, ist eines der Kernaspekte von automatisierten Geschäftsprozessen. Umso wichtiger ist es, dass dies auf eine möglichst intuitive Art und Weise erfolgt und sich nahtlos in die bestehende Systemlandschaft eingliedert. Durch dieses Projekt konnte das Workflow-Modul der Unternehmenssoftware FactWork wesentlich um Fähigkeiten der Benutzerinteraktion erweitert werden, die genau diese Eigenschaften erfüllen. Dadurch konnte die Leistungsfähigkeit des Moduls und die breitere Verwendbarkeit für diverse Einsatzszenarien erheblich gesteigert werden. Insbesondere können nun Arten von Geschäftsprozessen innerhalb des FactWork-Clients abgebildet werden, welche vorher nicht möglich waren, was einen erheblichen Zugewinn für das Workflow-Modul und das ERP-System insgesamt darstellt. Die geschaffenen Erweiterungen sind dabei aber nicht als final anzusehen, gerade der Formulardesigner bietet eine gute Grundlage für die zukünftige Erstellung weitergehender und komplexerer Eingabemöglichkeiten in Workflows. Auch die Funktionalitäten des Workflows-Moduls an sich sind noch lange nicht als vollständig und als ausreichend für sämtliche jemals auftretende Anwendungsfälle zu bezeichnen. Die Abbildung von Geschäftsprozessen in Computersystemen ist ein komplexes und umfangreiches Thema, das für unterschiedlichste Anforderungen auch verschiedenste Ansprüche an die ausführende Workflow-Umgebung stellt, welche ggf. dahingehend erweitert werden muss.

LITERATUR

Aalst, Wil van der (2013). "Business process management: a comprehensive survey". In: *ISRN Software Engineering* 2013.

Actian Corporation (2017). *Actian NoSQL: Handle Complex Data Models with Ease - Actian*. URL: https://www.actian.com/data-management/versant-nosql-object-database/ (besucht am 20. 09. 2017).

Bussler, Christoph (Sep. 2002). "The Application of Workflow Technology in Semantic B2B Integration". In: *Distributed and Parallel Databases* 12.2, S. 163–191.

Collins, Mark (2010). *Beginning WF - Windows Workflow in .NET 4.0*. New York: Apress.

F.EE GmbH (2018). *Über uns - F.EE Partner für Automation, Schaltschrankbau, Hardwareplanung, Energietechnik*. URL: https://www.fee.de/unternehmen/ueber-uns.html (besucht am 22. 02. 2018).

Kossak, Felix et al. (2016). "A Typed Approach to User Interaction Modelling". In: *Hagenberg Business Process Modelling Method*. Springer, S. 85–116.

Microsoft Corporation (März 2016). *Windows Workflow Foundation*. URL: https://msdn.microsoft.com/de-de/library/dd489441(v=vs.140).aspx (besucht am 02. 09. 2017).

Müller, Joachim (2006). *Workflow-based Integration: Grundlagen, Technologien, Management*. Springer.

Neumann, Juliane et al. (2015). "Surgical workflow and process modeling - An evaluation of modeling languages and process modeling tools". In: *M2CAI Workshop, Medical Image Computing and Computer-Assisted Intervention–MICCAI*. Bd. 2015.

Penicina, Ludmila (2010). "Towards the mapping of multidimensional bpmn models to process definition standards". In: *Scientific Journal of Riga Technical University. Computer Sciences* 41.1, S. 76–83.

Ramasamy, R. Kanesaraj, Fang-Fang Chua und Su-Cheng Haw (2015). "Web Service Composition Using Windows Workflow for Cloud-Based Mobile Application". In: *Advanced Computer and Communication Engineering Technology: Proceedings of the 1st International Conference on Communication and Computer Engineering*. Hrsg. von Hamzah Asyrani Sulaiman et al. Cham: Springer International Publishing, S. 975–985.

Rupp, Chris, Stefan Queins und die SOPHISTen (2012). *UML 2 glasklar - Praxiswissen für die UML-Modellierung*. HANSER.

Sivakumar, Nishant (2007). "Introduction to C++/CLI". In: *C++/CLI in Action*. Birmingham: Manning, S. 3–45.

White, Bayer (2012). *Pro WF 4.5*. Apress.