



# **Darstellung und Vergleich mehrerer Möglichkeiten zur Umsetzung eines sequentiellen HR-Prozesses im RESTful API-Umfeld**

## **Projektarbeit 1**

im Rahmen der Prüfung zum  
**Bachelor of Science (B.Sc.)**

des Studienganges Wirtschaftsinformatik  
an der Dualen Hochschule Baden-Württemberg Karlsruhe

von

**Tom Wolfrum**

- Sperrvermerk -

Abgabedatum:	4. September 2023
Bearbeitungszeitraum:	05.06.2023 - 03.09.2023
Kurs:	WWI22B5
Ausbildungsfirma:	SAP SE Dietmar-Hopp-Allee 16 69190 Walldorf, Deutschland
Betreuer der Ausbildungsfirma:	Steven Rösinger
Gutachter der Dualen Hochschule:	Paul Peitz

# Sperrvermerk

Die nachfolgende Arbeit enthält vertrauliche Daten der:

SAP SE  
Dietmar-Hopp-Allee 16  
69190 Walldorf, Deutschland

Der Inhalt dieser Arbeit darf weder als Ganzes noch in Auszügen Personen ausserhalb des Prüfungs- und Evaluationsverfahrens zugänglich gemacht werden, sofern keine anders lautende Genehmigung des Dualen Partners vorliegt.

# Selbstständigkeitserklärung

Ich versichere hiermit, dass ich die vorliegende Projektarbeit 1 mit dem Thema:

*Darstellung und Vergleich mehrerer Möglichkeiten zur Umsetzung eines sequentiellen HR-Prozesses im RESTful API-Umfeld*

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Karlsruhe, den 29. Juli 2023

---

Wolfrum, Tom

# Inhaltsverzeichnis

<b>Abkürzungsverzeichnis</b>	<b>V</b>
<b>Abbildungsverzeichnis</b>	<b>VI</b>
<b>1. Einleitung</b>	<b>1</b>
1.1. Unternehmensprofil und Anwendungsbezug . . . . .	1
1.2. Motivation und Problemstellung . . . . .	1
1.3. Aufbau und Ziel der Arbeit . . . . .	2
1.4. Abgrenzung . . . . .	3
1.5. Methodisches Vorgehen . . . . .	3
<b>2. Theoretische Grundlagen</b>	<b>5</b>
2.1. RESTful Application Programming Interface . . . . .	5
2.2. ABAP Restful Application Programming Model . . . . .	10
2.3. SAP Fiori Elements . . . . .	15
<b>3. Praktischer Teil</b>	<b>21</b>
3.1. Lösungsansätze . . . . .	21
3.1.1. Business Workflows . . . . .	21
3.1.2. Business Events . . . . .	24
3.1.3. Background Processing Framework . . . . .	25
3.2. Vergleich der Ansätze . . . . .	27
3.3. Entscheidungsmatrix . . . . .	35
<b>4. Schlussbetrachtungen</b>	<b>36</b>
4.1. Zusammenfassung . . . . .	36
4.2. Handlungsempfehlung . . . . .	36
4.3. Reflexion der Arbeit und Ausblick . . . . .	36
<b>Literaturverzeichnis</b>	<b>VII</b>
<b>A. Anhang</b>	<b>IX</b>
A.1. Fragebögen Experteninterviews . . . . .	IX
A.1.1. Workflows . . . . .	IX
A.1.2. Business Events . . . . .	X
A.1.3. bgPF . . . . .	XII
A.2. Transkripte Experteninterviews . . . . .	XIV
A.2.1. Workflows . . . . .	XIV
A.2.2. Business Events . . . . .	XIX

A.2.3. bgPF . . . . .	XXVI
-----------------------	------

# Abkürzungsverzeichnis

<b>SaaS</b>	Software-as-a-Service
<b>AIS</b>	Application Innovation Services
<b>HCM</b>	Human Capital Management
<b>API</b>	Application Programming Interface
<b>REST</b>	Representational State Transfer
<b>URI</b>	Uniform Resource Identifier
<b>URL</b>	Uniform Resource Locator
<b>RAP</b>	Restful Application Programming Model
<b>ABAP</b>	Advanced Business Application Programming
<b>CDS</b>	Core Data Service
<b>BO</b>	Business Object
<b>UI</b>	User Interface (Benutzeroberfläche)
<b>TCO</b>	Total Cost of Ownership (Gesamtkosten für Entwicklung und Betrieb)
<b>BW</b>	Business Workflow
<b>BE</b>	Business Events
<b>bgPF</b>	background Processing Framework
<b>LUW</b>	Logical unit of work
<b>bgRFC</b>	background Remote Function Call
<b>BTP</b>	Business Technology Platform

# Abbildungsverzeichnis

1.	REST Designprinzipien . . . . .	6
2.	RESTful Application Programming Model Architektur . . . . .	11
3.	Model-View-Controller Konzept . . . . .	17
4.	Fiori Elements List Report Floorplan . . . . .	19
5.	Fiori Elements Object Page Floorplan . . . . .	19
6.	Aufbau eines Business Workflows . . . . .	22
7.	Business Object als Event-Erzeuger . . . . .	24
8.	Funktionsweise des bgPF . . . . .	26

# 1. Einleitung

## 1.1. Unternehmensprofil und Anwendungsbezug

SAP SE ist ein börsennotierter Softwarekonzern mit Sitz in Walldorf. Das Hauptgeschäft des 1972 gegründeten Unternehmens ist die Entwicklung von Unternehmenssoftware zur Abwicklung von Geschäftsprozessen. Heute erwirtschaften 105.000 Mitarbeiter in 157 Ländern einen Umsatz von ca. 30 Mrd. €. Erfolgreich wurde das Unternehmen mit dem Verkauf von ERP Standardsoftware. In den letzten Jahren stand die Transformation des gesamten Produkt-Portfolios in Richtung Cloud-Services als Abo-Modell im Fokus der Unternehmensstrategie.<sup>1</sup>

Die Abteilung AIS HCM ist Teil des Unternehmensbereichs Product Engineering und zuständig für den Development-Support und Neuentwicklungen der SAP Personallösung HCM. Diese deckt Prozesse rund um das Personalwesen ab. Zudem stellt die Abteilung mehrere SAP Fiori Apps als Self-Service für Mitarbeiter bereit. Durch das hohe Nutzungsvolumen dieser Apps und der somit großen betriebswirtschaftlichen Relevanz, sind diese und auch der Untersuchungsgegenstand dieser Arbeit für das Produkt HCM von großer Bedeutung.

## 1.2. Motivation und Problemstellung

Die von der Abteilung betriebenen Fiori Apps, die schon im Zusammenhang der Einleitung angesprochen wurden, sind auf Basis des Frameworks SAP UI5 Freestyle für ein älteres Produkt - SAP ECC - entwickelt worden. Durch die strategische Entscheidung HCM im neuen S/4 HANA System ("S/4" abgekürzt) durch die neue cloudbasierte Personallösung SuccessFactors abzulösen war dieser Umstand ursprünglich kein Problem. Diese Entscheidung wurde aufgrund fehlender Funktionalitäten in SuccessFactors und hoher verlässlicher Einnahmen durch Wartungsverträge für HCM revidiert und HCM ist

---

<sup>1</sup>Vgl. SAP 2023a.



Bestandteil von S/4. Somit finden die S/4-Design-Guidelines darauf Anwendung, die z. B. Oberflächen-Design oder zu verwendende Technologien, festlegen. Fiori Apps müssen dadurch die Technologie Fiori Elements verwenden. Aus Praktikabilitätsgründen dürfen existierende Apps auf Basis der älteren Technologie in S/4 weiterbetrieben werden und nur neu entwickelte Apps müssen Fiori Elements verwenden.

Diese Situation sorgt für ein Problem in Geschäftsprozessen, die über solche Apps abgebildet werden sollen. Das Framework Fiori Elements generiert das gesamte Front-End der Anwendung selbstständig. Dadurch ist es erleichtert auf der einen Seite die Entwicklung der Apps, auf der anderen Seite kann dadurch keine eigene Programmlogik mehr im Front-End eingebaut werden. Zudem bietet das Programmiermodell RAP, das in Fiori Elements für eine konsistente Durchführung der Datenbankoperationen zuständig ist, nur einen transaktionalen Kontext für das Ausführen von eigener Logik. Jedoch ist es in bestimmten Geschäftsprozessen nötig, nachgelagert in einem weiteren transaktionalen Kontext noch Programmcode ausführen zu können. Da RAP das modellseitig nicht zulässt, soll in der vorliegenden Arbeit nun untersucht werden, wie sich solche asynchronen Prozesse, trotz den eben dargelegten Einschränkungen im S/4 Umfeld mit den neueren Technologien umsetzen lassen.

### **1.3. Aufbau und Ziel der Arbeit**

Als erstes wird im einleitenden Kapitel die SAP und die Abteilung AIS HCM, in der die Praxisphase absolviert wurde, kurz vorgestellt und somit der Anwendungsbezug der Arbeit hergestellt. Danach wird mit der Motivation und Problemstellung der Untersuchungsgegenstand und die Bedeutung der Arbeit für die Abteilung und die Kunden erläutert. Danach soll die Arbeit klar von verwandten Themen abgegrenzt werden, um einen klaren Rahmen für die Untersuchung zu schaffen. Im methodischen Vorgehen werden dann abschließend für die Einleitung noch auf die wissenschaftlichen Methoden, die verwendet wurden, um die Untersuchungsergebnisse zu erhalten, eingegangen. Der Hauptteil der Arbeit besteht aus zwei Teilen: Im ersten Teil werden die theoretischen Grundlagen der Arbeit gelegt. Hier werden die Designprinzipien einer RESTful API, wie diese im RESTful Application Programming Model der SAP eingesetzt werden und die Technologie Fiori Elements näher beleuchtet. Der praktische Hauptteil stellt drei Ansätze vor, wie das in der Problemstellung thematisierte Problem gelöst werden kann. Hierfür

werden die Technologien Business Workflows, Business Events und das Background Processing Framework vorgestellt und im Bezug auf Stärken und Schwächen sowie Effizienz und Robustheit verglichen. Diese Ergebnisse werden dann in einer Entscheidungsmatrix dargestellt. Das Ziel soll es sein, dass diese Entscheidungsmatrix klare Tendenzen gibt, welche der untersuchten Technologien sich in einem konkreten Anwendungsfall für das Abbilden von asynchronen Prozessen im RESTful API Umfeld anbietet. Im Schlussteil werden die Ergebnisse der Arbeit nochmals zusammengefasst, eine konkrete Handlungsempfehlung für die Lösung dieses Problems gegeben und die Ergebnisse abschließend kritisch reflektiert und ein Ausblick auf zukünftige Entwicklungen gegeben.

## **1.4. Abgrenzung**

Der Zweck der vorliegenden Arbeit ist es, die drei vorgestellten Technologien vergleichend zu bewerten und je nach Anwendungsfall eine Handlungsempfehlung im Bezug auf eine sich anbietende Technologie zu geben. Über diese drei Technologien hinaus werden keine anderen Möglichkeiten asynchrone Prozesse abzubilden, wie z. B. im Cloud Application Programming Model (CAP) behandelt. Außerdem findet aufgrund des beschränkten Umfangs der Arbeit lediglich ein Vergleich der Technologien statt und keine direkte Implementierung dieser in einem konkreten Anwendungsfall. Hierfür sei auf die offizielle Dokumentation der SAP mit Showcases für die respektiven Technologien verwiesen.

## **1.5. Methodisches Vorgehen**

Nachdem die drei Ansätze vorgestellt wurden, sollen diese im Bezug auf mehrere Kriterien betrachtet und anhand dieser miteinander verglichen werden. Diese Kriterien werden bei den einzelnen Ansätzen durch Experteninterviews bewertet und dann die betrachteten Ansätze anhand dieser Kriterien gegenübergestellt. So kommt dann die Entscheidungsmatrix zu Stande, welche Technologie sich bei welchen Anforderungen und Rahmenbedingungen anbietet.

Die Methode des Experteninterviews bietet sich an, weil sich das Thema der Arbeit auf spezifische SAP-Technologien bezieht. Diese Technologien können von Fachleuten auf diesem Gebiet gut umrissen und im Bezug auf die Vergleichskriterien eingeordnet werden.

Experteninterviews sind eine qualitative Forschungsmethode, durch die eine umfassende Einsicht und persönliche Perspektive von mit dem Thema vertrauten Personen erfasst. Die direkte Kommunikation erleichtert das Sammeln von Informationen, die in der Literatur zu unspezifisch oder nicht ausreichend vorkommen. Dies ist besonders bei praxisnahen Themen wichtig, da hier Best Practices und bekannte Herausforderungen ermittelt und daraus auf spezielle Anwendungsfälle passende Ansätze erarbeitet werden können.

Zur Auswahl stehen ein strukturiertes und unstrukturiertes Experteninterview. Für die hier durchgeführten Interviews wird eine Mischform aus beiden gewählt. In der Vorbereitung des Interviews wird strukturiert vorgegangen und es werden klare Fragen formuliert. Die Durchführung des Interviews ist eher unstrukturiert, damit die Reihenfolge der Fragen geändert werden kann und bei Bedarf auch noch flexibel Rückfragen gestellt werden können.

## 2. Theoretische Grundlagen

Im Folgenden sollen die theoretischen Grundlagen für die nachfolgende vergleichende Darstellung der Umsetzung sequentieller Prozesse im REST-Umfeld gelegt werden. Zuerst wird grundsätzlich erklärt, was eine RESTful-API ist und danach die Umsetzung von REST in ABAP näher erleutert. Abgeschlossen wird der theoretische Teil der Arbeit mit einer Darstellung von Fiori Elements.

### 2.1. RESTful Application Programming Interface

Eine API ist eine Schnittstelle, über die verschiedene Softwareanwendungen miteinander kommunizieren können. Die API definiert die Methoden, Protokolle und Tools, die für den Zugriff auf die Funktionen und Daten einer Softwareanwendung verwendet werden können. Somit standardisiert eine API die Kommunikation verschiedener Anwendungen und ermöglicht den Zugriff auf bereitgestellte Daten, ohne dass die zugreifende Anwendung die interne Logik oder Implementierung der anderen Anwendung kennen muss.<sup>1</sup>

Eine RESTful-API ist eine spezielle Schnittstelle, die den Designkonventionen nach REST folgt.

---

<sup>1</sup>Vgl. Biehl 2015, S. 15ff.

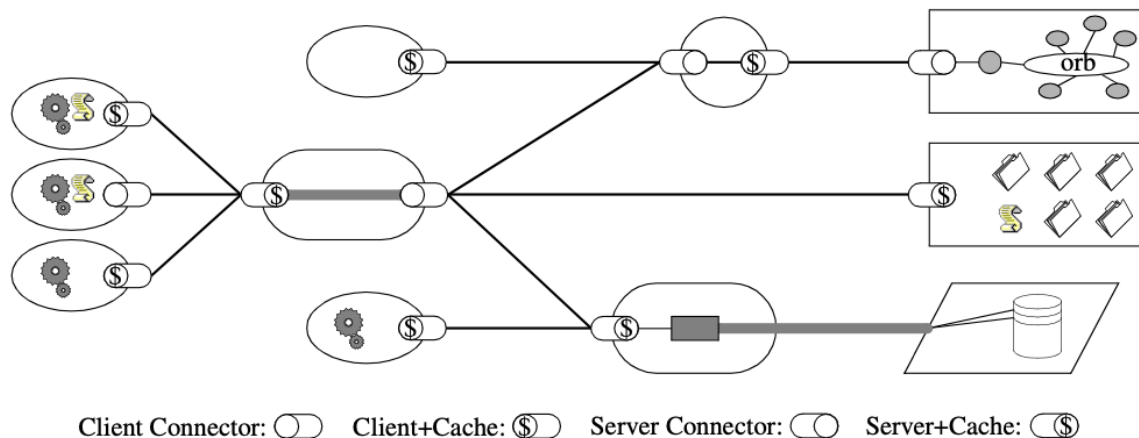


Abbildung 1.: REST Designprinzipien, Abgerufen von Fielding 2000 am 05.07.2023.

## Client-Server-Architektur

Das erste Prinzip ist die Client-Server-Architektur. Das bedeutet, dass die Benutzeroberfläche von den gespeicherten Daten getrennt wird. Die Benutzeroberfläche und Sitzung existiert nur auf dem Client und die gespeicherten Daten oder zur verfügung gestellten Funktionen existieren nur auf dem Server. Somit wird die Portierbarkeit und Skalierbarkeit des Gesamtsystems verbessert. Zudem wird die Möglichkeit einer unabhängigen Weiterentwicklung der verschiedenen Komponenten sichergestellt.<sup>2</sup>

## Zustandslosigkeit

Zudem soll eine RESTful-API zustandslos angelegt sein. Das heißt im Genaueren, dass die Kommunikation der verschiedenen Parteien zustandslos sein muss. Es muss für den Server somit möglich sein, die Anfrage des Clients vollständig zu verstehen bzw. zu verarbeiten, ohne zusätzlich auf vergangene Anfragen zugreifen zu müssen. Auf der anderen Seite bedeutet das auch, dass der Client jede Antwort des Servers ohne zusätzliche Informationen, die eventuell zu einem früheren Zeitpunkt angefordert wurden, verstehen können muss. Das heißt, dass in jeder Anfrage immer alle notwendigen Informationen mitgeschickt werden müssen und von keinem "Vorwissen" ausgegangen werden darf. Das hat wiederum zur Folge, dass Sitzungsinformationen ausschließlich auf dem Client

<sup>2</sup>Vgl. Fielding 2000, S. 78.

gespeichert werden. Durch diese Bedingung verbessert sich die Skalierbarkeit weiter, da der Server Ressourcen, die ansonsten für die Speicherung der Stati der Requests benötigt würden, nicht freihalten muss. Zudem steigt die Zuverlässigkeit der Schnittstelle, da bei einem Fehler immer nur ein Request betrachtet werden muss. Somit ist ein Fehler einfacher behebbar und hat keine Auswirkungen auf andere Anfragen. Damit einher geht auch ein vereinfachtes Monitoring, da immer nur eine Request betrachtet werden muss und nicht erst eine Kette zusammenhängender Anfragen nachvollzogen werden muss.<sup>3</sup>

## **Caching**

Die dritte Designkonvention besagt, dass auf der Client Seite ein Cache vorhanden sein muss. Durch das implizite oder explizite Markieren von Daten als cache-fähig dürfen die Anfrage-Daten vom Client für spätere identische Requests wiederverwendet werden. Durch dieses Caching von Daten ist es möglich manche Client-Server Interaktionen teilweise oder ganz zu vermeiden, wodurch die Netzwerkauslastung und Skalierbarkeit verbessert wird. Jedoch birgt die Verwendung eines Caches das Risiko, dass die Daten im Cache im Vergleich zu den auf dem Server gespeicherten Daten schon veraltet sind, was gegebenenfalls zu Fehlern in der weiteren Verarbeitung führen könnte.<sup>4</sup>

## **Einheitliche Schnittstelle**

Das vierte Prinzip und zentrales Unterscheidungsmerkmal von REST ist das einheitliche Interface zwischen den verschiedenen Komponenten. Hierdurch wird die Systemarchitektur durch das Prinzip der Generalität vereinfacht. Die Schnittstelle ist einfacher benutzbar. Zudem wird eine unabhängige Weiterentwicklung der verschiedenen kommunizierenden Komponenten gewährleistet, da die Implementierung der einzelnen Komponenten von den angebotenen Services getrennt wird. Jedoch entsteht durch die einheitliche Schnittstelle auch ein Effizienzverlust, da diese nicht an die Bedürfnisse einer speziellen Anwendung angepasst werden kann. Um ein einheitliches Interface zu erreichen, finden mehrere Beschränkungen auf die Schnittstelle Anwendung: Die Ressourcen der Schnittstelle sollen eindeutig identifizierbar sein. Eine Ressource ist eine vom Interface bereitgestellte Information, die eindeutig über einen URI identifizierbar ist. Die Informationen, die durch

---

<sup>3</sup>Vgl. Fielding 2000, S. 78f.

<sup>4</sup>Vgl. Fielding 2000, S. 79ff.

die Ressourcen repräsentiert werden können statisch festgelegt sein, oder sich auch im Zeitablauf verändern. Zudem kann eine Ressource auch existieren, ohne dass die Information schon existiert. Das erleichtert die Verarbeitung verschiedener Informationsarten, da auf abstrakter Ressourcenebene nicht zwischen bestimmten Typen unterschieden wird. Außerdem kann so die benötigte Information auch noch zu einem späten Zeitpunkt, je nach Inhalt der Anfrage, festgelegt werden. Zudem hat jeder Service, der nach den REST Prinzipien entworfen ist eine URL, also eine eindeutige Adresse. Durch diese URL ist der Zugriffsweg zum Webservice standardisiert. Durch diese eindeutig identifizierbaren Ressourcen und Services wird zudem die Kombinierbarkeit verschiedener Ressourcen eines Services bzw. von verschiedenen Services in einem größerem System erleichtert. Eine weitere Beschränkung für die Schnittstelle ist die Verwendung von Repräsentationen zur Veränderung von Ressourcen. Eine Repräsentation ist eine Folge von Bytes, die eine Ressource in einer bestimmten Darstellung und zugehörige Metadaten abbildet. Somit kann eine Ressource vom Server in verschiedenen Repräsentationen, je nach Anfrage, zurückgegeben werden. Zudem werden mit der Repräsentation alle Informationen, wie die Ressource verändert werden kann, mitgeschickt. Veränderungen der Ressource finden nur über die Repräsentation statt. Des Weiteren sollen Antworten des Servers auf Anfragen selbsterklärend sein. Das heißt, dass Standard-Methoden und -Datentypen verwendet werden, um die Ressource zu verändern oder Informationen auszutauschen. Diese Standard-Methoden sind zwar in REST selbst nicht festgelegt, werden aber normalerweise durch die Verwendung des Protokolls auf der Anwendungsschicht definiert. Für das meistens im Internet verwendete HTTP-Protokoll sind diese z. B. : GET (gewünschte Ressource vom Server anfordern), POST (neue Ressource unterhalb angegebener Ressource einfügen) oder PUT (angegebene Ressource anlegen bzw. ändern).<sup>5</sup> Die letzte Beschränkung wird als "Hypermedia as the Engine of Application State" bezeichnet. Hiermit ist gemeint, dass die Interaktion mit einer API dynamisch über Hypermedien abläuft. Somit ist auf der Client-Seite nur Basiswissen über Hypertext und fast kein Wissen über die Interaktion mit der spezifischen Schnittstelle nötig. Somit können Client und Server voneinander entkoppelt werden, da der Server dem Client neben den angeforderten Informationen dynamisch mögliche Interaktionen zurückgibt.<sup>6</sup>

---

<sup>5</sup>Vgl. Fielding 2008.

<sup>6</sup>Vgl. Fielding 2000, S. 81f.

## **Schichtenarchitektur**

Die Systemarchitektur soll zudem in Schichten aufgebaut sein. Das heißt, dass eine Schicht jeweils nur die nächste darunter- und darüberliegende Schicht sehen und mit ihr interagieren kann. Durch diese Architektur wird die Komplexität des Gesamtsystems reduziert und die unabhängige Weiterentwicklung der einzelnen Schichten gefördert. Zudem können veraltete Dienste abgekapselt werden und neue somit von diesen getrennt werden. Durch die Aufteilung der Architektur in Schichten kann zudem redundante oder selten benutzte Funktionalität in eine "Zwischenschicht" ausgelagert werden. Durch diese Aulagerung verbessert sich zudem die Skalierbarkeit des Systems, da load-balancing, also die Lastverteilung eines Services auf mehrere Netzwerke oder Prozessoren, ermöglicht wird. Dennoch bringt die eine Schichtenarchitektur auch Nachteile mit sich. Durch die Kapselung der Dienste und Funktionalitäten in Schichten steigt der Verwaltungs- und Wartungsaufwand des Gesamtsystems. Zudem sinkt auch die Geschwindigkeit, mit der Daten verarbeitet werden, da die Anfrage im Verarbeitungsprozess wesentlich mehr Schnittstellen passieren muss. Dieser Nachteil kann jedoch durch die Verwendung von geteilten Caches in den Zwischenschichten kompensiert werden, da durch diese Caches die Anzahl der Schnittstellen, die die Anfrage passieren muss, reduziert werden kann. Ein weiterer Vorteil der Schichtenarchitektur ist, dass die Anfragen selektiv von den einzelnen Schichten verändert werden können, da der Inhalt dieser selbst-beschreibend und die Bedeutung der Nachricht für die Zwischenschichten sichtbar ist.<sup>7</sup>

## **Code on Demand**

Die sechste (optionale) Designkonvention von REST besagt, dass wenn nötig Code in Form von Skripten oder Apps über die Schnittstelle vom Client heruntergeladen und ausgeführt werden kann. Dies vereinfacht die Programmlogik des Clients, da weniger Programme schon im Voraus vorhanden sein müssen. Zudem wird dadurch die Erweiterbarkeit eines Systems verbessert, da auch nach dem initialen Installieren eines Systems, dieses noch durch das Bereitstellen von Code über die Schnittstelle erweitert werden kann.<sup>8</sup>

---

<sup>7</sup>Vgl. Fielding 2000, S. 82f.

<sup>8</sup>Vgl. Fielding 2000, S. 84f.



## **2.2. ABAP Restful Application Programming Model**

ABAP RAP ist ein Programmiermodell der SAP auf Basis von ABAP, das die Architektur für die Entwicklung von OData-Services definiert. Es basiert auf den Designprinzipien nach REST. Mit diesem Modell können sowohl Web APIs veröffentlicht und Business Events erzeugt als auch Fiori Apps entwickelt werden. RAP ist nach dem "Classic ABAP Programming" und dem "ABAP Programming Model for SAP Fiori" die dritte Evolutionsstufe des ABAP Programming Model. RAP kann sowohl in Cloud-, als auch in on-premise Systemen eingesetzt werden.<sup>9</sup>

RAP baut im Allgemeinen auf drei Säulen auf: Anders als in den vorhergehenden Programmiermodellen, wo mehrere Tools zum entwickeln von z. B. einer Fiori App nötig waren, sind sollen in RAP Implementierungsaufgaben in einer Entwicklungsumgebung integriert werden um einen standardisierteren Entwicklungsprozess zu gewährleisten und diesen für den Entwickler zu vereinfachen. Die zweite Säule ist die Programmiersprache ABAP: Durch Erweiterungen und Anpassungen ist es möglich diese für die Entwicklung mit RAP zu verwenden. Hierbei kommen Technologien wie CDS-Views zum Einsatz um Daten-Modelle zu definieren. Zudem können vorgefertigte APIs für generische Entwicklungsaufgaben verwendet werden. Die dritte Säule sind umfassende Frameworks, die dem Entwickler helfen, effizient und in kurzer Zeit eine Anwendung zu entwickeln, da einzelne Bausteine automatisch generiert werden und an bestimmten Stellen noch anwendungsspezifische Logik eingefügt werden kann.<sup>10</sup>

---

<sup>9</sup>Vgl. SAP 2023b.

<sup>10</sup>Vgl. SAP 2023b.

## Architektur eines Business Services in RAP

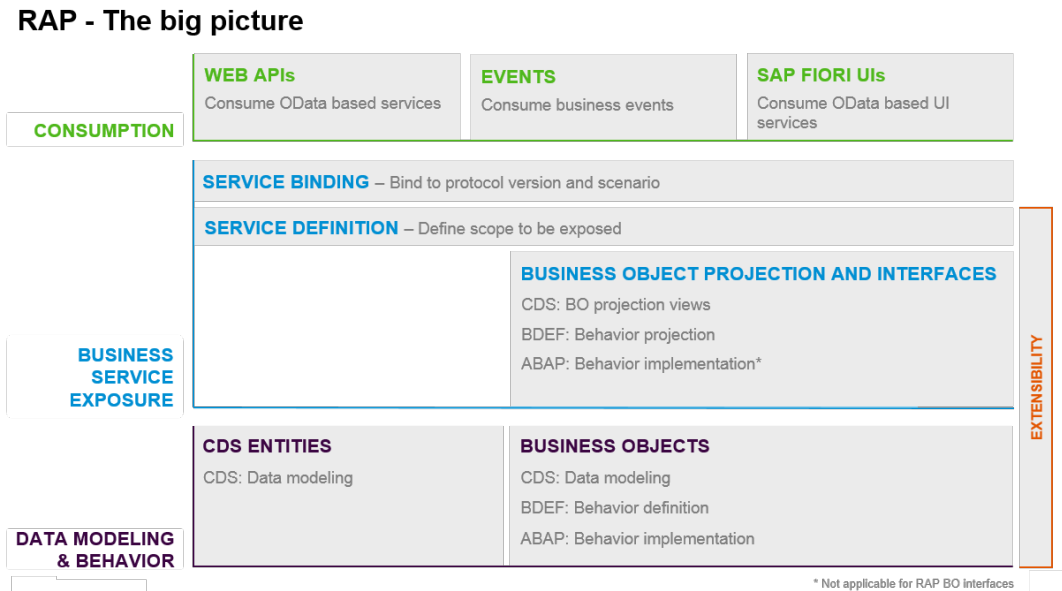


Abbildung 2.: RESTful Application Programming Model Architektur, Abgerufen von SAP 2023b am 09.07.2023.

## Modellierung von Daten und Verhalten

Auf der untersten Ebene werden die benötigten Daten und das beabsichtigte Verhalten modelliert. Dies kann entweder durch CDS-Views oder durch Business Objects geschehen. Core Data Services sind ein Framework um ein Datenmodell, basierend auf der HANA-Datenbank, zu definieren und zu organisieren. Es können selektiv die benötigten Daten aus den Datenbanktabellen ausgewählt werden. Genauer werden alle benötigten Spalten aus einer oder mehreren Tabellen ausgewählt und diese bei Bedarf mit Annotationen versehen, um speziellere Anforderungen zu erfüllen. Es kann bei Bedarf auch noch nach Datensätzen, die gewisse Bedingungen erfüllen gefiltert werden. Somit strukturiert und gruppiert ein CDS-View die benötigten Daten. Die SQL-Abfrage, um diese Daten von der Datenbank abzurufen ist in dem CDS-View integriert. Der Zweck eines CDS-Views ist jedoch lediglich das Lesen und Strukturieren der Daten; es können hiermit keine Daten verändert werden.<sup>11</sup>

<sup>11</sup>Vgl. SAP 2023b.

Eine andere Möglichkeit ein Datenmodell zu erzeugen sind Business Objects. Diese bieten zudem die Implementierung von Behaviors (Verhalten) und einer Laufzeit. Ein BO ist aus struktureller Sicht ein hierarchisch aufgebauter Baum aus mehreren Knoten, die Daten enthalten und durch Eltern-Kind-Beziehungen (sogenannte Kompositionen) miteinander verknüpft sind. Diese Knoten werden durch CDS Entitäten dargestellt. Der hierarchisch oberste Wurzelknoten stellt dabei die Repräsentation des BO an sich dar. Um das Verhalten eines BO zu spezifizieren, muss eine "Business Object Behavior definition" ("Behavior definition" abgekürzt) angelegt werden. Dieses ABAP Objekt beschreibt das gewünschte Verhalten des BO in RAP. Die Behavior definition bezieht sich immer auf die Wurzel-CDS-Entität eines BO und wird als ABAP Klasse implementiert. Ein Verhalten beschreibt welche Operationen für ein BO verfügbar sein sollen. Es besteht außerdem noch aus der "Behavior characteristic", die zusätzliche Eigenschaften, wie z. B. Autorisierungen für Operationen festlegt. Operationen sind z. B. create() für das Erstellen, update() für das Aktualisieren und delete() für das Löschen eines Datensatzes. Diese modify-Operationen verändern im Gegensatz zu den read-Operationen auch die tatsächlichen Daten auf der Datenbank. Die Laufzeit eines BO besteht aus 2 Teilen: In der Interaktionsphase werden durch das Ausführen von Operationen Daten gelesen und/oder verändert. Diese Veränderungen werden zunächst in einem "transactional buffer" (Pufferspeicher) gespeichert und nachdem alle Änderungen durchgeführt wurden in der sog. "save sequence" auf der Datenbank persistiert.<sup>12</sup>

## Veröffentlichung des Business Service

Die zweite Ebene sorgt für das Projizieren von BOs und die Erstellung sowie Veröffentlichung von Business Services. Ein Business Service ist in RAP ein RESTful Service, der Repräsentationen von Ressourcen veröffentlicht, die dann von Konsumenten abgerufen werden können. Die Bestandteile eines Business Services werden später noch genauer erläutert. Die Projektion eines BO ist notwendig, um es flexibel konsumieren zu können, da dieses an sich komplett unabhängig vom OData-Service ist. Das BO an sich stellt die maximal möglichen Funktionen und Daten bereit, die service-unabhängig implementiert und ggf. durch die Projektion auf die für den Service relevanten Aktionen und Daten eingeschränkt werden. Zudem können genauere Anpassungen z. B. im Bezug auf die Darstellung auf einer Benutzeroberfläche über UI-Annotationen erfolgen, die

---

<sup>12</sup>Vgl. SAP 2023b.

aber nicht Teil des Datenmodells sein sollen. Eine zusätzliche Projektions-Schicht hat mehrere Vorteile: Zum einen kann das zugrundeliegende BO angepasst und erweitert werden, ohne dass der darauf aufbauende Service davon betroffen ist. Zum anderen können verschiedene Projektions-Views für verschiedene Anforderungen erstellt werden, die alle dasselbe BO wiederverwenden. Zudem können die Daten und Funktionen eines Services für eine Fiori App oder Web API veröffentlicht werden. Des Weiteren können Services somit auch rollenbasiert veröffentlicht werden, sodass unterschiedliche Daten und Funktionen für unterschiedliche Anwender bereitgestellt werden können. Um eine solche Projektionsschicht zu erstellen, muss zusätzlich eine CDS Projection View erstellt werden, um die speziellen Daten einer Projektion darzustellen. Dieser basiert auf der CDS View des BO und erzeugt selbst keine neue SQL-View, sondern nur eine Repräsentation der dargestellten Entitäten. Um eine CDS-Entität eines BO zu projizieren müssen die Wurzel-Entität sowie alle Eltern-Entitäten ebenfalls projiziert sein. Zudem wird auch eine Projection Behavior Definition benötigt, die alle Verhalten, die für einen speziellen Service veröffentlicht werden sollen projiziert.<sup>13</sup>

Nachdem Teile des BO projiziert wurden und die zugehörigen Artefakte erstellt wurden, muss in der zweiten Ebene ein Service definiert werden. In einer "business service definition" (abgekürzt "Service Definition") wird festgelegt, welche CDS Entitäten eines Datenmodells, also welche Daten, in einen bestimmten Service veröffentlicht werden sollen. Die Service Definition stellt eine protokoll-unabhängige und Konsumenten-spezifische Sichtweise auf das Datenmodell dar. Es können auch mehrere CDS-Entitäten oder eine komplette BO Struktur in einem Service veröffentlicht werden. Dafür muss in der Service Definition die hierarchisch höchste Entität des BO, die veröffentlicht werden soll, markiert werden. Diese dient dann als Einstiegspunkt für den Service.<sup>14</sup>

Als letzter Schritt in der zweiten Ebene muss noch das Kommunikationsprotokoll des Service im Service Binding definiert werden. Ein häufiges Beispiel wäre hierbei OData, für das Bereitstellen von Daten in einer Fiori App. Ein Service Binding bezieht sich immer direkt auf eine oder mehrere Service Definitions. Es können auch mehrere Service Bindings basierend auf einer Service Definition erstellt werden. Das ist z.B. hilfreich, wenn derselbe Service mit unterschiedlichen Kommunikationsprotokollen veröffentlicht werden soll, da durch die Trennung von Service Definition und Binding das Protokoll von

---

<sup>13</sup>Vgl. SAP 2023b.

<sup>14</sup>Vgl. SAP 2023b.

der Geschäftslogik getrennt wird. Somit kann der Entwicklungsaufwand für einen Service erheblich reduziert werden. Ein Service kann für die Protokolle OData Version 2 und 4 veröffentlicht werden, wenn das Ziel ist, die Daten in einer Fiori App darzustellen. OData ermöglicht außerdem das Erstellen von HTTP-basierten Services, deren Ressourcen über URIs identifizierbar sind und über HTTP-Nachrichten abgerufen und modifiziert werden können. Dies korrespondiert wiederum mit den Designkonventionen nach REST aus dem vorhergehenden Kapitel. Zudem stehen noch die Protokolle Information Access für Analysezwecke und SQL zur Verfügung. Ein Service kann grundlegend auf zwei Arten veröffentlicht werden: Entweder als UI Service mit den Protokollen OData oder Information Access, indem man dem durch UI-Annotationen eine Fiori Elements oder andere Benutzeroberfläche hinzufügt. Für alle anderen Anwendungsfälle wird der Service als Web API veröffentlicht, die von Clients über das Web mit OData konsumiert werden kann. Wenn ein Service veröffentlicht ist, kann er jedoch im Standard nur innerhalb der Systemlandschaft abgerufen werden. Ein Service kann zudem in mehreren Versionen existieren. Dies geschieht durch das Hinzufügen oder Entfernen von zusätzlichen Service Definitions zu einem Service Binding. Damit kann ein Service geändert oder erweitert werden.<sup>15</sup>

## **Konsumieren des Services**

Die dritte und oberste Ebene der RAP Architektur ist für das Konsumieren der Daten eines Business Services verantwortlich. Es gibt zwei Möglichkeiten, diese Daten durch einen Service zu konsumieren. Die erste Möglichkeit ist durch eine Web API. Die Metadaten eines so veröffentlichten Services enthalten keine Informationen über eine Benutzeroberfläche für die Darstellung der Informationen. Der Zugriff auf die bereitgestellten Daten erfolgt über eine öffentliche Schnittstelle des OData Services. Eine weitere Möglichkeit einen Service zu konsumieren ist innerhalb einer Fiori Elements Anwendung als UI Service. Hier werden die Konfigurationen für die Benutzeroberfläche und das Front-End der Anwendung, die im Back-End als Annotationen in den CDS Entitäten festgelegt wurden, über die Metadaten mitgegeben. Somit kann das Fiori Elements Framework aus diesen Metadaten direkt eine fertige UI generieren. Als letzte Möglichkeit ein BO zu konsumieren sind noch

---

<sup>15</sup>Vgl. SAP 2023b.

Business Events zu nennen. Diese werden hier nur kurz genannt und in einem späteren Kapitel detaillierter beschreiben.<sup>16</sup>

## **2.3. SAP Fiori Elements**

Fiori Elements ist ein Framework zum Entwickeln benutzerfreundlicher und ansprechender Anwendungen. Apps werden auf Basis von OData-Services und UI-Annotationen in den CDS Entitäten durch ein umfassendes Framework fast automatisch generiert. Somit ist im Gegensatz zur älteren SAP UI5 Freestyle Technologie kein JavaScript Coding mehr nötig um das Front-End zu programmieren. Elements benutzt vordefinierte Layouts und Controller für Aktionen der App.<sup>17</sup>

Fiori Elements bietet drei zentrale Vorteile: Es soll dabei helfen, dass sich Entwickler auf die spezifische Geschäftsprozess-Logik und die Back-End Entwicklung fokussieren und somit weniger Zeit für die Programmierung der Benutzeroberfläche benötigen, was insgesamt zu einer verkürzten Entwicklungszeit von Apps und somit auch für niedrigere Entwicklungskosten sorgt. Zudem wird die Kontinuität des UI über alle Fiori Apps hinweg und die Übereinstimmung der Apps mit den SAP Designkonventionen sichergestellt. Die Benutzer der Apps haben so eine einheitliche Benutzungserfahrung (Layout, Navigation, Suche, ...) über alle Apps hinweg. Zudem stellt das zentrale Framework auch sicher, dass der Programmcode für die Benutzeroberflächen der Apps immer sofort funktioniert und bietet außerdem weitere Funktionen wie Übersetzungen und Unterstützung für mobile Endgeräte. Diese Vorteile tragen wiederum zu einer reduzierten Entwicklungszeit und somit einem Kostenersparnis bei.<sup>18</sup>

### **Vergleich Fiori Elements - SAP UI5 Freestyle**

Verglichen mit SAP UI5 Freestyle (Freestyle abgekürzt), der anderen Technologie, um Fiori Apps zu entwickeln, lassen sich folgende Unterschiede feststellen: Die generelle Herangehensweise bei Fiori Elements zielt eher auf das effiziente und schnelle Entwickeln und Veröffentlichen einer App ab. Im Gegensatz dazu, liegt der Fokus bei Freestyle

---

<sup>16</sup>Vgl. SAP 2023b.

<sup>17</sup>Vgl. SAP 2022a.

<sup>18</sup>Vgl. SAP 2022a.

eher auf der Flexibilität, spezielle Anforderungen, die ggf. auch nicht mit dem Elements Standard übereinstimmen abbilden zu können. Das wirkt sich auch auf die Möglichkeiten im UI-Design aus: In Fiori Elements ist der Entwickler an die vordefinierten SAP Vorlagen gebunden, während es bei den Freestyle Designs auch möglich ist, mit einer leeren Vorlage zu beginnen. Das hat aber auch zur Folge, dass bei Freestyle wesentlich mehr Webentwicklungs Kenntnisse nötig sind, da die Oberfläche mithilfe von JavaScript Coding selbst programmiert werden muss. In Fiori Elements hingegen wird die gesamte Oberfläche vom Elements Framework generiert und muss nur durch UI-Annotationen in den CDS-Entitäten an die Wünsche des Kunden angepasst werden. Was die Wartbarkeit und entwicklungstechnischen Freiheiten der beiden Technologien angeht, sind auch bei Freestyle größere Spielräume vorhanden: Dadurch, dass die Oberfläche selbst programmiert wird, ist es möglich eine Logik in das Front-End einzubauen. Diese Tatsache ist für die Problemstellung der Arbeit sehr relevant, da sequentielle Prozesse, die asynchrone Kommunikation benötigen, sich durch eben diese Logik leicht abbilden lassen. Da in Fiori Elements die das UI generiert wird und die Logik von SAP kommt, fehlt diese Gestaltungsmöglichkeit, was die Umsetzung der angesprochenen Geschäftsprozesse erschwert. Dennoch muss man sagen, dass in allen anderen Fällen Fiori Elements erhebliche Vorteile bietet, da bei der Entwicklung einer App sehr viel Zeit und somit auch Geld gespart werden kann, was somit auch die TCO reduziert.<sup>19</sup>

### **Model-View-Controller Konzept**

Beide Fiori Technologien, Elements und Freestyle verwenden das Model-View-Controller Konzept, auf welches im Folgenden genauer eingegangen wird.

---

<sup>19</sup>Vgl. SAP 2023c.

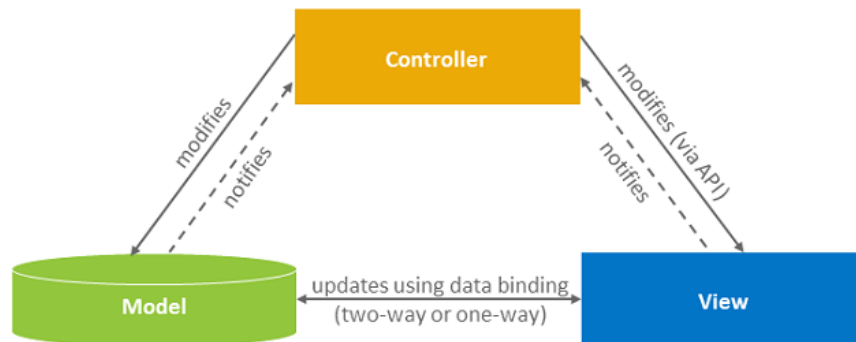


Abbildung 3.: Model-View-Controller Konzept, Abgerufen von SAP 2023d am 13.07.2023.

Das Konzept unterteilt eine Anwendung in ein Modell, einen View und einen Controller. Das Modell verwaltet die Daten der Anwendung und stellt Methoden bereit, um die Daten aus der Datenbank abzurufen und bei Bedarf zu bearbeiten. Fiori unterstützt Client- und serverseitige Modelle. Das heißt, dass das Modell entweder komplett vom Client heruntergeladen wird und dort dann lokal existiert, oder bei serverseitigen Modellen nur die jeweils angefragten Daten vom Client beim Server angefordert werden. Daraus resultiert, dass clientseitige Modelle sich aus Performancegründen nur bei kleinen Datenmodellen empfehlen lassen und sich für große Modelle eher serverseitige Datenmodelle anbieten. Wenn das Modell auf der Client Seite existieren soll, kann entweder ein JSON-, XML- oder Ressourcen-Modell verwendet werden. Serverseitig kann ein OData-Service in den Versionen zwei und vier verwendet werden. Es ist auch möglich für verschiedene Bereiche einer Anwendung jeweils verschiedene Modelle zu definieren und diesen auch modellspezifisch über den Controller verschiedene Interaktionsmöglichkeiten mit dem View zu geben. Der View definiert und rendert die Benutzeroberfläche und legt somit das Aussehen der App für den Anwender fest und visualisiert die Daten des Modells. Standardmäßig werden XML- und JSON-Views unterstützt. Zudem kann man einen View als eigene Klasse selbst programmieren. Der Controller ist für die Interaktionen des Benutzers mit der App zuständig und enthält Methoden, die die Interaktion zwischen Modell und View regeln. Er enthält den Programmcode für die Geschäftsprozesslogik hinter der Anwendung. Zudem kann der Controller in Methoden oder zu bestimmten Zeitpunkten, wie z. B. bei Start oder Beenden der Anwendung Events auslösen. Diese Events können dann von anderen Controllern empfangen und verarbeitet werden. Somit können eventgesteuert bestimmte Aktionen in der Anwendung ausgelöst werden. "Data-



binding” wird verwendet, um die Daten des Modells und den View zu synchronisieren und das Bearbeiten dieser Daten direkt auf der Benutzeroberfläche zu ermöglichen. Das Data binding definiert, wie das Model und der View miteinander kommunizieren. Eine Art, die Daten des Models an die UI zu binden, ist das One-way Binding, welches nur in eine Richtung Daten bindet. Beispielsweise werden hier nur werden Veränderungen der Daten im Modell in die UI synchronisiert und nicht in die Gegenrichtung. Beim Two-way Binding ist diese Bindung in beide Richtungen vorhanden. Wenn sich also Daten entweder im View oder im Model ändern werden diese Änderungen dirket mit der jeweils anderen Komponente synchronisiert. Das Ziel des Konzepts ist es, die Daten der App logisch von den Interaktionen des Benutzers zu trennen. Eine Anwendung in die genannten drei Teile aufzuteilen hat folgende Vorteile: Der Programmcode der Anwendung ist leichter lesbar, wartbar und erweiterbar, da er in logisch zusammenhängende Teile aufgeteilt ist. Somit ist es möglich die Darstellung der Daten in der View zu verändern, ohne die darunterliegende Logik oder das Datenmodell zu ändern und mit mehreren Views mehrere Darstellungsweisen, je nach Anwendungsfall zu erstellen.<sup>20</sup>

### **Floorplans in Fiori Elements**

Apps sind in Fiori Elements immer auf vordefinierten Layouts aufgebaut. Diese können am Anfang der Entwicklung einer App ausgewählt werden und geben das generelle Aussehen und Funktionen vor.

---

<sup>20</sup>Vgl. SAP 2023d.

Standard

Search [ ] Editing Status: All Category: Supplier: Preferred Product: Adapt Filters (1) Go

Products (205)

Preview Image	Product	Name	Category	Supplier	Star Ratings	Price	Assign Supplier
	AR-FB-1013	13-Pocket Expanding File - Blue	Files & Binders	ChinaChain	★★★★★	6.90 USD	Assign Supplier
	WD-ERS-1005	2-hole Canister Pencil Sharpener - Blue	Eraser, Ruler & Sharpener	OffPOR	★★★★★	1.70 USD	Assign Supplier
	MC-B-1001	AA Alkaline Batteries - Pack of 6	Batteries	Office-Guru AG	★★★★★	3.50 USD	Assign Supplier
	MC-B-1000	AAA Alkaline Batteries - Pack of 8	Batteries	Office-Guru AG	★★★★★	3.50 USD	Assign Supplier
	MC-CA-1002	All-Purpose Cloth - Pack of 50	Cleaning Aids	ITel-Office	★★★★★	4.95 USD	Assign Supplier
	EP-BT-1001	Assorted Colours Rubber Band - Bag of 870	Bands & Twine	Office Line Prag	★★★★★	3.49 USD	Assign Supplier

Abbildung 4.: Fiori Elements List Report Floorplan. Abgerufen von SAP 2022a am 13.07.2023.

Auf dem Bild ist beispielhaft das Layout List Report zu sehen. Dieser kann verwendet werden, wenn der Zweck der Anwendung das Arbeiten mit einer Liste von Dingen z. B. Produkten ist. Diese werden dann in einer sortierten Tabelle dargestellt, in der der Benutzer nach Einträgen sortieren, filtern und suchen kann. Hier müssen nur die Produktdaten über einen OData Service bereitgestellt werden und die Darstellung der Daten über UI-Annotationen nach Bedarf angepasst werden. Die restliche Benutzeroberfläche mit Navigation, Suchleisten und Aktionen wird vollständig durch das Framework generiert. Der List Report wird häufig als Einstiegspunkt in einer App verwendet um nach bestimmten Objekten zu suchen und mit diesen dann in einer Object Page näher zu interagieren.

Lead Pencils - Pack of 6  
WD-PC-1001

Category: Pencils & Colouring Availability: 84 Price: 1.80 USD Star Ratings: ★★★★★

Basic Data

Product	Supplier	Price	Star Ratings
WD-PC-1001	ChinaChain	1.80 USD	★★★★★

Technical Data

Length	Width	Height	Weight
-	-	-	-

Admin Data

Created By	Created On	Modified By	Modified On
Date Generator	Jan 1, 2018, 1:00:00 AM	Date Generator	Jan 1, 2018, 1:00:00 AM

Reviews

Reviews (6)

Star Rating	User	Title	Text
★★★★★	Claire Walker Sep 26, 2019, 11:34:44 PM	Ok, but has a few issues	I am hoping that the product will continue to improve. Pretty looking product. Not expensive for the... <a href="#">More</a>
★★★★★	Sid Lu Mar 30, 2019, 2:05:35 PM	Function and fun all in one!	Excellent engineering and the feeling of a very high quality product. They hold up and last a very long... <a href="#">More</a>
★★★★★	Leo Alexander Apr 5, 2019, 6:42:42 AM	Horrible Quality in this package	I'm a dummy for not throwing all in the trash. Too expensive for what I received. DON'T receive what... <a href="#">More</a>

Abbildung 5.: Fiori Elements Object Page Floorplan. Abgerufen von SAP 2022a am 13.07.2023.

Das List Report Layout wird häufig mit dem Object Page Floorplan in einer App kombiniert, wenn es darum geht, mit den einzelnen Objekten in der Liste zu arbeiten. Die Objekt-Ansicht wird durch einen Klick auf das gewünschte Objekt in der Listen-Ansicht aufgerufen. Hier können dann weitere Informationen zu z. B. einem Produkt angezeigt und bei Bedarf auch bearbeitet werden. Des Weiteren gibt es noch Floorplans für eine Worklist, eine Liste mit Unterstützung für diverse Analyseverfahren, wie z. B. Drill-Down und ein Dashboard.

## 3. Praktischer Teil

Im Folgenden werden die verschiedenen praktische Lösungsansätze vorgestellt und anhand verschiedener Kriterien gegeneinander abgewogen, sodass am Ende eine Handlungsmatrix erstellt werden kann.

### 3.1. Lösungsansätze

Zunächst sollen drei verschiedene Technologien vorgestellt werden, mit denen sich asynchrone Prozesse mit sequentieller Kommunikation, trotz den Einschränkungen durch RAP und Fiori Elements, umsetzen lassen.

#### 3.1.1. Business Workflows

Der erste mögliche Ansatz sind Business Workflows (BW abgekürzt). BWs können benutzt werden um jeglichen Geschäftsprozess im SAP-System abzubilden. Sie decken das Spektrum von einfachen Genehmigungsprozessen bis hin zu komplexen Abläufen ab. Sie eignen sich vor allem für repetitive Prozesse mit mehreren Bearbeitern. BWs können zudem zur Fehlerbehandlung in anderen Prozessen oder eventgesteuert eingesetzt werden. Mit Workflows können durch die Benutzung der bereits bestehenden Funktionen und Transaktionen des SAP-Systems neue Geschäftsprozesse abgebildet werden. Die Funktionen und Transaktionen an sich werden dabei durch den Workflow nicht verändert. In Kombination mit Organisationsmanagement können die einzelnen Schritte des BW durch bestimmte Akteure ausgeführt werden. Das kann auch auf bestimmte Stellen abstrahiert werden, um von personellen Veränderungen innerhalb des Unternehmens unabhängig zu sein. Workflows können auch untereinander durch das Versenden und Konsumieren von Nachrichten kommunizieren. Diese Kommunikation ist auch zwischen verschiedenen SAP-Systemen über das Internet mit XML-Dokumenten möglich.<sup>1</sup>

---

<sup>1</sup>Vgl. SAP 2022b.

## Aufbau eines Business Workflows

Zunächst wird die Definition der Aufbau eines Business Workflows beschrieben. Diese lässt sich in vier Bereiche unterteilen.

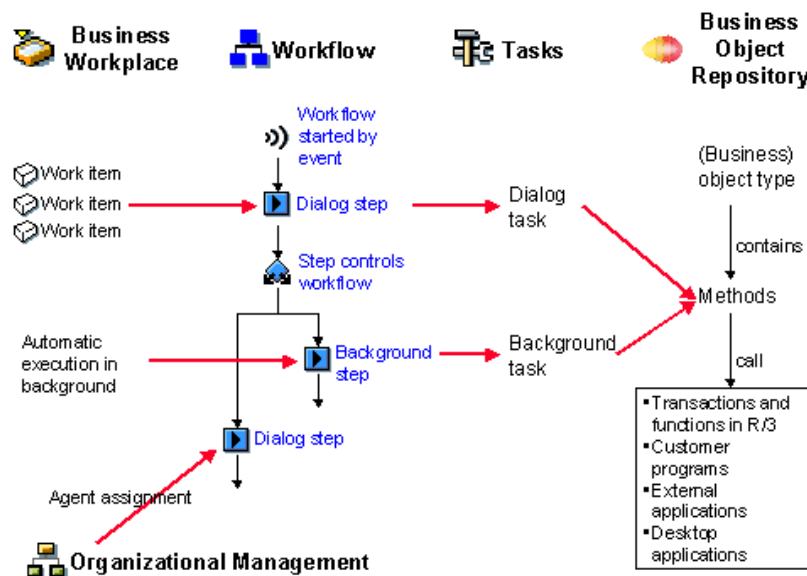


Abbildung 6.: Aufbau eines Business Workflows, Abgerufen von SAP 2022b am 17.07.2023.

Der Business Workplace ist der Ort in einem SAP-System, in dem der Endanwender "work items" (übersetzt aus dem Englischen: "Arbeitspakete" oder "Aufgaben") abhängig vom Zeitpunkt im Geschäftsprozess und den Berechtigungen des Users ausführen kann. Ein work item stellt zur Laufzeit des BW einen Schritt des Prozesses dar, der ausgeführt wird. Es werden hier jedoch nicht alle work items angezeigt. So werden z. B. solche, die einem Prozess-Schritt, der im Hintergrund ausgeführt werden soll, zugeordnet sind, hier nicht angezeigt.<sup>2</sup>

Ein Workflow muss vor Ausführung in der "workflow definition" angelegt werden. Diese Definition legt die Reihenfolge der auszuführenden Schritte des Prozesses fest und enthält zudem Kontrollschritte. Zusätzlich können noch Bearbeiter und Fristen für bestimmte Schritte festgelegt werden, die dann zur Laufzeit des Workflows vom "Work item manager" verwaltet werden. Es gibt viele Arten von Schritten, die gängigen Konzepten in der Programmierung ähneln, wie z. B. normale Aktivitäten, Fallunterscheidungen,

<sup>2</sup>Vgl. SAP 2022b.

Schleifen. Zudem gibt es Schritte zum Versenden von Nachrichten, Auslösen von Events, Benutzerentscheidungen, usw. Diese Schritte können entweder im Dialog mit einem Benutzer ausgeführt werden, wenn z. B. die Eingabe bestimmter Werte erforderlich ist, oder automatisch vom System im Hintergrund ausgeführt werden. Ein Workflow kann nicht nur manuell von einem Benutzer gestartet werden, sondern auch systemseitig von einem bestimmten Event ausgelöst werden. Hierfür muss in der Definition des Workflows das gewünschte Event als Auslöser angegeben werden. Wenn dann das Event auftritt, wird der Workflow automatisch gestartet. Im betrieblichen Kontext könnte hier z. B. ein Mitarbeiter einen Urlaubsantrag stellen, der dann den als Workflow abgebildeten Genehmigungsprozess auslöst. Das wäre ein Beispiel für einen asynchronen Prozess mit sequentieller Kommunikation.<sup>3</sup>

Die einzelnen Schritte, die innerhalb des Workflows ausgeführt werden, heißen Tasks und stellen grundlegende betriebliche Tätigkeiten dar. Die Dialog- und Hintergrund-Schritte in der Workflow Definition korrespondieren hier mit Dialog- oder Hintergrund-Tasks. Im Workflow bezieht sich ein Task immer auf eine Methode eines Objekttyps. Diese Methoden können automatisch ausführbar sein oder müssen aktiv von einem Benutzer gestartet werden. Eine Methode kann einerseits Transaktionen oder Funktionen innerhalb des ERP-Systems aufrufen. Spezielle Anforderungen können durch kundeneigene Logik, oder Schnittstellen zu anderen Systemen umgesetzt werden.<sup>4</sup>

Methoden, die innerhalb eines Workflows aufgerufen werden, sind immer Teil von Objekten. Diese Objekte können auch BOs sein. Im Allgemeinen ist ein Objekt ein konkreter Datensatz eines Objekttyps. Die Daten des Objekts werden durch seine Attribute definiert und die Aktionen, wie das Erstellen, Aktualisieren oder Löschen von Daten wird durch die Methoden des Objekts beschrieben. Einen weiteren wichtigen Teil von Objekten stellen Events dar. Diese werden ausgelöst, wenn bei einem Objekt seinen Status verändert. Das kann z. B. durch das Erstellen, Verändern oder Löschen von Daten passieren. Diese Events können dann unter anderem Workflows starten. Das "Business Object Repository" bietet eine Übersicht über alle in einem SAP-System verfügbaren Objekttypen. Man kann die bereits vorhandenen Objekttypen bei Bedarf anpassen oder neue erstellen.<sup>5</sup>

---

<sup>3</sup>Vgl. SAP 2022b.

<sup>4</sup>Vgl. SAP 2022b.

<sup>5</sup>Vgl. SAP 2022b.

### 3.1.2. Business Events

Der zweite Ansatz ist die Umsetzung mit Business Events (BE abgekürzt). Business Events sind Events die von BOs erzeugt und konsumiert werden können. Dieser event-gesteuerte Kommunikationsansatz, der die asynchrone Kommunikation zwischen dem Event-erzeugenden und -konsumierenden BO ermöglicht, wird von RAP im Standard unterstützt. Hier ist keine direkte Antwort des Empfängers nötig, ein BO erzeugt ein Event und dieses wird von anderen BOs dann weiterverarbeitet, ohne dass der Erzeuger weiterhin in diesen Prozess involviert ist. Somit spricht man hier von einer einseitigen, also asynchronen Kommunikation. Ein BE stellt eine signifikante Veränderung eines BO dar, die im Zuge eines Behaviours erzeugt wird. Dem Event werden dann über dessen Metadaten alle nötigen Informationen, anhand derer die Weiterverarbeitung, je nach speziellem Anwendungsfall, stattfindet, mitgegeben.<sup>6</sup>

Ein BO kann als Event-Erzeuger oder Event-Konsument auftreten. Zuerst wird die Erzeuger-Seite betrachtet.

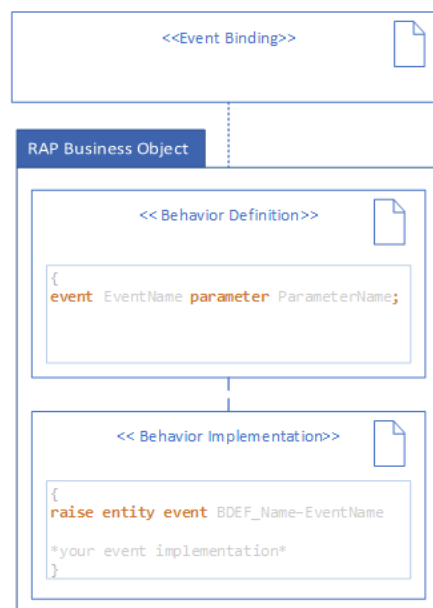


Abbildung 7.: Business Object als Event-Erzeuger, Abgerufen von SAP 2023e am 17.07.2023.

<sup>6</sup>Vgl. SAP 2023e.

Ein Event wird in der Behaviour Definition eines BO erstmals definiert. Hier können dann Parameter für die weitere Verarbeitung mitgegeben werden, die dann in den Metadaten des Events mitgeschickt werden. Danach kann es in der dazugehörigen Behaviour Implementation innerhalb einer speziellen Operation des BO erzeugt werden. Ein Event wird zu dem Zeitpunkt in RAP Laufzeit erzeugt, nachdem die Veränderungen des BO, die das Event auslösen, auf der Datenbank persistiert wurden. Durch das Event Binding wird das Event noch einem speziellen Namensraum, BO und zugehöriger Operation zugeordnet.<sup>7</sup>

BOs können BEs nicht nur erzeugen, sondern auch verarbeiten. Dies geschieht über das Event Consumption Model. Ein Event Consumption Model besteht aus einer Reihe von ABAP-Artefakten, mit denen man Events im SAP-System konsumieren kann. Diese Events müssen jedoch dem Standard eines Cloud Events entsprechen, da der Austausch über das SAP Event Mesh abgewickelt wird, was ein Dienst der SAP Business Technology Platform und somit komplett cloud-basiert ist. Hierfür muss ein inbound bzw. outbound Event-Binding für ein- bzw. ausgehende Events erstellt werden. Diese Bindings sind Teil des SAP Enterprise Event Enablement Frameworks, das den Austausch von Events zwischen dem S/4 System und der BTP regelt.<sup>8</sup>

### **3.1.3. Background Processing Framework**

Die letzte Möglichkeit, die betrachtet wird, um asynchrone Prozesse abzubilden, stellt das "background Processing Framework" (bgPF abgekürzt) dar.

Das bgPF ist ein Framework, das die Möglichkeit schafft asynchron zum Hauptprozess Logik auszulagern und auszuführen. Zuerst soll auf die grundlegende Funktionsweise des bgPF eingegangen werden.

---

<sup>7</sup>Vgl. SAP 2023e.

<sup>8</sup>Vgl. SAP 2022c.



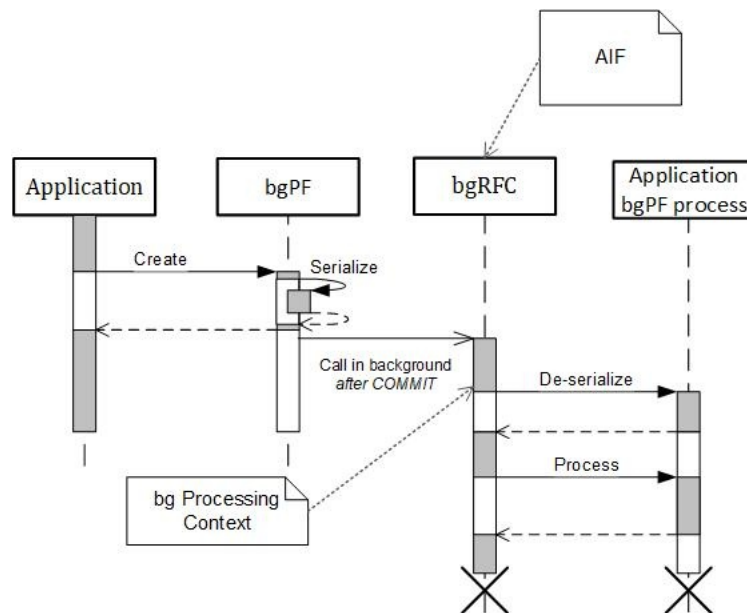


Abbildung 8.: Funktionsweise des bgPF, Abgerufen von SAP 2023f am 17.07.2023.

Nachdem das Interface für das bgPF in der Anwendung als eigene Klasse mit der jeweiligen Logik, die ausgelagert werden soll, implementiert wurde, kann es verwendet werden. Die Anwendung ruft dann zur Laufzeit diese Klasse auf und erstellt synchron ein bgPF-Objekt. Innerhalb dieses Objekts wird die Klasse dann serialisiert und mit einem Methodenaufruf für die spätere Ausführung gespeichert. In der Endphase des Hauptprozesses wird dann mit einem bgRFC, also einem Funktionsaufruf, der im Hintergrund ausgeführt wird, die Ausführung ausgelöst. Der bgRFC wird dann in einer neuen ABAP Laufzeit als neuer Prozess gestartet. Hier wird die serialisierte Klasse wieder deserialisiert und die implementierte Logik ausgeführt. Nach dem Ausführen wird der Prozess wieder beendet. Falls nötig, können durch das Bereitstellen eines eigenen Verarbeitungskontexts von der aufrufenden Anwendung spezifische Verarbeitungsprioritäten festgelegt werden. Dies ist zwingend notwendig, wenn die Verarbeitungsreihenfolge genau eingehalten werden muss. Der voreingestellte Kontext wird standardmäßig von anderen Anwendungen gemeinsam verwendet.<sup>9</sup>

Das Entkoppeln und asynchrone Ausführen gewisser Logik vom Hauptprozess bringt einige Vorteile mit sich: Zuerst wird die Robustheit und transaktionale Konsistenz des Hauptprozesses und somit auch die Konsistenz der Daten sichergestellt. Das bgPF bietet

<sup>9</sup>Vgl. SAP 2023f.

die Möglichkeit, diese Konsistenz zu gewährleisten, aber auch gleichzeitig relevantes Coding aus dem Hauptprozess auszulagern, was dessen Robustheit verbessert. Des Weiteren erscheint die Hauptanwendung für den Benutzer performanter, da gewisse Verarbeitungslogik in einer anderen Sitzung asynchron ausgeführt werden kann. Ein weiterer Vorteil ist die bessere Skalierbarkeit, wenn größere Prozesse in kleinere Teile unterteilt werden, die verteilt über mehrere Laufzeitumgebungen asynchron ausgeführt werden können. Die Skalierung einer Anwendung kann mit dieser Technologie dann komplett vom SAP-System automatisiert werden.<sup>10</sup>

### **bgPF innerhalb von RAP**

Das bgPF kann auch innerhalb von RAP verwendet werden. Grundlegend besteht die Laufzeit eines RAP BOs aus einer Interaktions- und einer Speicherungsphase. In der Interaktionsphase werden über die Behaviours des BO dessen Daten ausgelesen oder verändert. Falls Änderungen an den Daten stattgefunden haben, werden diese jedoch nicht direkt auf der Datenbank persistiert, sondern zuerst in einem Puffer gespeichert. Nachdem alle Änderungen erfolgreich durchgeführt wurden, werden diese auf einmal in der Speicherungsphase auf die Datenbank geschrieben. Konkret wird die Methode des bgPF-Objekts, die das Speichern für die spätere asynchrone Ausführung übernimmt, in der zweiten Phase der Laufzeit des BOs aufgerufen und das Objekt somit auf der Datenbank gespeichert.<sup>11</sup>

## **3.2. Vergleich der Ansätze**

Im Folgenden sollen die in den vorherigen Kapiteln vorgestellten Ansätze miteinander verglichen werden. Hierfür werden die Kriterien Komplexität der Implementierung, also welchen Aufwand das Umsetzen der Technologie im Anwendungsfall verursacht, die Auswirkungen auf die Systemlandschaft, also ob zusätzliche Systemkomponenten benötigt werden und die Performance der einzelnen Ansätze im Bezug auf Ausführungsdauer und Rechenlast verglichen werden. Weitere Kriterien sind zusätzliche Kosten die eine Lösung eventuell verursacht, die Abwärtskompatibilität zu älteren Systemen und die Flexibilität

---

<sup>10</sup>Vgl. SAP 2023f.

<sup>11</sup>Vgl. SAP 2023f.

der Technologien im Bezug auf Gestaltungsmöglichkeiten von Anwendungsszenarien und Integration mit anderen Technologien/ Frameworks. Abgeschlossen wird der Vergleich mit einer Betrachtung der Skalierbarkeit und Wartbarkeit der drei Umsetzungsmöglichkeiten.

### **Komplexität der Implementierung**

Als erstes Kriterium wird die Komplexität der Implementierung der drei Technologien herangezogen. Genauer wird hier auf die Anzahl der zu erstellenden Artefakte, eine Beschreibung der Implementierungstätigkeiten und eine ungefähre Lines-of-Code Angabe betrachtet.

Die Komplexität von Workflows hängt von dem Geschäftsprozess ab, der abgebildet werden soll. Somit ist das Spektrum von einem sehr einfachen Workflow, der beispielsweise bei einem gewissen Ereignis einen Mitarbeiter per E-Mail benachrichtigt bis hin zu einem Workflow für einen komplexen Geschäftsprozess, wie einer Abwesenheitsmeldung, bei dem viele Schritte mit Genehmigungsprozessen nötig sind und der in viele Stellen im System hineingreift. Durch die freien Gestaltungsmöglichkeiten eines Workflows, einen Prozess aus beliebig vielen Schritten in beliebiger Konstellation zu modellieren sind Workflows in Produktivsystemen meistens sehr komplex. Hinzu kommt die weitgehenden Konfigurationsmöglichkeiten eines jeden Schrittes in einem Workflow und die Möglichkeit eigenes ABAP-Coding in einem Schritt auszuführen. Hierdurch kann die Komplexität nochmals stark steigen. Zudem können Workflows, wie oben angesprochen ereignisgesteuert gestartet werden und miteinander interagieren, wodurch komplexe Wechselwirkungen zusätzlich beachtet werden müssen. Somit lässt sich insgesamt sagen, dass Workflows zwar potenziell simpel sein können, jedoch in der Realität eher sehr komplex sind. (vgl. Anhang A.2.1, Z. 22-32)

Bei Business Events hängt die Komplexität vom der verwendeten Ansatz ab. Beispielsweise sind Standard Events von SAP sehr einfach, diese sind bereits im System vorhanden und müssen nur über eine Einstellung eingeschaltet werden. Dadurch ist das Event exponiert. Normalerweise werden die Events bereits von der SAP entwickelt und an die Kunden ausgeliefert. Wenn jedoch für eine spezielle Anforderung ein eigenes Event benötigt wird, kann auch ein Custom Event erstellt werden. Diese werden entweder RAP-basiert entwickelt, was einen nicht unerheblichen Aufwand darstellt und Erfahrung in der Entwicklung mit RAP voraussetzt. Die andere Option ist das Netweaver Event

Enablement Add-On, womit man durch einen Low-Code-Ansatz ein Event einfach in kurzer Zeit konfigurieren kann. Insgesamt ist festzustellen, dass der Aufwand von wenigen Sekunden mit dem Aktivieren eines Standard Events bis hin zu mehreren Tagen mit einem selbst programmierten RAP-basiertem Event reicht. Auch die zu erstellenden Artefakte sind je nach verwendeter Technologie unterschiedlich. (vgl. Anhang A.2.2, Z. 28-63)

bgPF

### **Auswirkungen auf die Systemlandschaft**

Die Auswirkungen auf die Systemlandschaft werden daran bemessen, ob für die Implementierung der jeweiligen Technologie zusätzliche Systemkomponenten, im Spezielleren das Einbinden von Cloud-Komponenten, notwendig sind.

Workflows sind in der sogenannten "SAP-Basis" enthalten. Die Systemkomponente "SAP-Basis" bildet die technische Grundlage eines SAP-Systems, indem sie Systemadministration, Datenbankmanagement, Kommunikation, Sicherheit und Performance-Optimierung gewährleistet und somit eine stabile und effiziente Plattform für die Ausführung der SAP-Anwendungen bereitstellt. Das heißt, dass sie in jedem SAP-System als Grundfunktionalität verfügbar sind und somit mit dem initialen Lizenzkauf der Software benutzbar sind. Es müssten keine zusätzlichen Systemkomponenten integriert werden und Workflow sind sowohl in on-premise als auch in Cloud-Systemen verfügbar. Somit lässt sich zusammenfassen, dass Workflows keine Auswirkungen auf die Systemlandschaft haben. (vgl. Anhang A.2.1, Z. 33-40)

Das Verwenden von Business Events hat auf die Systemlandschaft definitiv mehrere Auswirkungen. Zum einen müssen die Prozesse des Unternehmens selbst angepasst werden, da man von synchronen batchorientierten Prozessen zu asynchronen Prozessen umsteigt. Zudem wird ein Event Broker oder Event Mesh (Netzwerk mehrerer Event Broker) als zusätzliche Systemkomponente benötigt, das den Austausch der Events regelt. Das SAP Event Mesh ist zudem eine Cloud-Komponente, was eine relevante Änderungen für reine on-premise Systemlandschaften darstellt. Es ist aber auch möglich einen Event Broker lokal zu betreiben, wenngleich dann die Vorteile der globalen Vernetzung mit Events verloren gehen. Zudem können mit SAP-Technologie auch Events anderer Hersteller, die dem Cloud Event Standard entsprechen verarbeitet werden, was für heterogene

Systemlandschaften mit Systemen Lösungen mehrerer Hersteller ein großer Vorteil ist. (vgl. Anhang A.2.2, Z. 64-108)

bgPF

## **Performance**

Die Performance der drei Lösungsansätze wird anhand der Rechenlast, die auf dem System durch das Ausführen des Prozesses erzeugt wird, der Ausführungsdauer und der Netzwerklast die dadurch entsteht gemessen.

Allgemein lässt sich sagen, dass der Performance-Aspekt bei Workflows in einer typischen Systemlandschaft kein Problem darstellt und auch bei sehr häufig stattfindenden Prozessen keine Leistungsgrenzen erreicht werden. Zudem werden Workflows, bedingt durch die Prozesse, die mit ihnen abgebildet werden, meist nicht auf einmal ausgeführt, sondern immer nur einzelne oder wenige Schritte. Diese Schritte erzeugen für sich keine nennenswerte Last auf dem System und sind meist sehr unterschiedlich. Der limitierende Faktor hingegen ist eher der Anwender, da in vielen Prozessen manuell Entscheidungen getroffen werden müssen oder der Benutzer aktiv Dinge tun muss, wodurch die Ausführung blockiert ist. Auch in dem Fall, dass viele Schritte eines Workflows hintereinander ausgeführt werden, bearbeitet das System alle Tätigkeiten im Hintergrund ohne den Anwender zu beeinflussen. Kommunikation über Systemgrenzen hinaus, die die Performance verschlechtern würde, ist zwar theoretisch möglich, aber in der Realität meist nicht notwendig und die Ausführung von Workflows findet beim Großteil der Geschäftsprozesse rein lokal statt. Somit sind Workflows insgesamt eine performante Technologie. (vgl. Anhang A.2.1, Z. 41-66)

Der Einsatz von Business Events ist für das Quellsystem der Events von erheblichen Vorteil, da durch den eventgesteuerten Ansatz die nachgelagerte Code-Ausführung gezielt ausgelöst werden kann und keine Last durch permanente API-Calls, um zu prüfen, ob das relevante Ereignis eingetreten ist, entsteht. Das ausschlaggebende Kriterium bei Business Events im Bezug auf die Performance sind die mitgeschickten Daten: Je mehr Daten bei einem Event mitgeschickt werden, desto größer wird es und desto mehr Last entsteht bei der Erzeugung und Weiterleitung. Hier gilt es, den Mittelweg zwischen der Menge an mitgeschickten Informationen und der Menge an API-Calls zu finden. Insgesamt dürfte das für den konkreten Anwendungsfall jedoch kein Problem darstellen,

da kleine Benachrichtigungs-Events ausreichen um eine asynchrone Code-Ausführung im Hintergrund anzustoßen. (vgl. Anhang A.2.2, Z. 109-135)

bgPF

## **Kosten**

Zunächst wird die monetäre Seite der drei Technologien betrachtet. Es soll die Frage beantwortet werden, ob durch die drei Technologien zusätzliche Lizenzkosten, laufende Kosten durch Abonnements und Kosten für Netzwerk-Traffic entstehen.

Da die Workflows in der SAP-Basis mit jedem System enthalten sind, entstehen für den Kunden keine weiteren Kosten durch die Implementierung dieser Technologie. (vgl. Anhang A.2.1, Z. 67-72)

Die Implementierung von Business Events verursacht Kosten. Zum einen durch das Betreiben eines Brokers, wie z. B. dem SAP Event Mesh. Kleinere Broker wie dieser, die vor Allem auf kleinere Notification-Events ausgelegt sind, werden gemietet und nach dem Benutzungsvolumen bezahlt. Hier liegen die Kosten, wenn nur einzelne Anwendungsfälle oder Bereiche abgebildet werden im dreistelligen Bereich pro Monat. Soll jedoch die gesamte Unternehmensarchitektur mit Events gesteuert werden, müssen umfangreichere Technologien zum Einsatz kommen, bei denen die Kosten schnell im sechsstelligen Bereich liegen. Hier ist der relevante Faktor die Betrachtung des Break-even-Punkts, ab dem man Geld spart, da die Infrastruktur vorhanden ist und Skaleneffekte eintreten. Ab welcher Anzahl an Events diese Technologie günstiger ist, muss im Einzelfall betrachtet werden. Im konkreten Anwendungsfall sollte das nicht gegen Business Events als Lösungsansatz sprechen, da hier nur ein spezieller Anwendungsfall mit kleinen Events betrachtet wird, der auch mit dem SAP Event Mesh abgebildet werden kann. (vgl. Anhang A.2.2, Z. 136-163)

bgPF

**Flexibilität**

Beim Kriterium der Flexibilität soll die spätere Anpassbarkeit der Lösungsansätze, die Gestaltungsmöglichkeiten die sie bei spezielleren Anforderungen bieten und die Integrationsmöglichkeiten zu anderen Technologien betrachtet werden.

Bei der Flexibilität von Workflows muss zwischen klassischen und flexiblen Workflows unterschieden werden. Bei der Betrachtung der Gestaltungsmöglichkeiten bei speziellen Anforderungen verhalten sich beide Varianten gleich, die Vorlagen, die hier von SAP an Kunden ausgeliefert werden, sind nochmals an die konkreten Bedürfnisse und Geschäftsprozesse anpassbar. Im Bezug auf die spätere Anpassbarkeit sind flexible Workflows im Vorteil, da hier Fehlerkorrekturen im Gegensatz zu klassischen Workflows automatisch im Kundensystem übernommen werden, sobald diese von der SAP veröffentlicht wurden. Bei klassischen Workflows müssen diese noch aktiv vom Kunden umgesetzt werden. Jedoch sind flexible Workflows nur für Cloud-Produkte verfügbar. Die Integrationsmöglichkeiten mit anderen Technologien sind für Workflows auch sehr umfangreich, da auf diese als Teil der SAP-Basis von allen anderen Softwarekomponenten aus zugegriffen werden kann. Insgesamt sind Workflows sehr flexibel, da der Kunde nicht an die Vorlagen der SAP gebunden ist und durch das Ausführen von eigenem ABAP-Coding weitgehende Freiheiten hat. (vgl. Anhang A.2.1, Z. 73-115)

Die Flexibilität von Business Events hängt stark von der verwendeten Technologie ab. Auf der einen Seite gibt es Standard-Events, die sehr unflexibel sind, auf der anderen Seite sind Custom Events mit dem Netweaver Event-Enablement Add-On sehr flexibel und schnell anpassbar. Durch die Möglichkeit ein eigenes Event mit RAP zu programmieren erhält der Entwickler fast unbegrenzte Freiheiten, solange man sich im Cloud Events Standard bewegt. Events lassen sich auch untereinander mit anderen Events kombinieren. Diese können z. B. beim Passieren des Event Brokers verändert werden oder in anderen Systemkomponenten gewisse Logik auslösen. Somit sind Business Events insgesamt im Bezug auf ihre spätere Anpassbarkeit und ihre Integrationsmöglichkeiten mit anderen Technologien sehr flexibel. (vgl. Anhang A.2.2, Z. 164-186)

bgPF

## **Skalierbarkeit**

Die Skalierbarkeit der Komponenten beschreibt ihre Fähigkeit bei wachendem Nutzungsvolumen und Datenlast effizient zu wachsen und die Leistung beibehalten. Genauer wird hier die Aufteilbarkeit der Technologien auf mehrere Prozesse und das Vorhandensein von Frameworks, die die Skalierung automatisch abwickeln herangezogen.

Workflows sind an sich große zusammenhängende Prozesse, die aufgrund der sequentiellen Reihenfolge der einzelnen Schritte nicht unterteilt und z. B. parallel ausgeführt werden können. Somit skalieren Workflows nur linear, die Menge an beanspruchten Ressourcen steigt proportional zur Anzahl der laufenden Prozesse. (vgl. Anhang A.2.1, Z. 116-127)

Das Thema Skalierbarkeit ist eine der großen Stärken von Business Events. Event Broker sind darauf ausgelegt bis hin zu mehreren Milliarden von Events pro Tag zu verarbeiten und weiterzuleiten. Bei dieser Technologie erreichen als erstes das Back-End-System, von dem aus die Events losgeschickt werden oder das System, das die Events letztendlich empfängt ein Limit. Aber auch dieses Limit wird erst sehr spät erreicht, da die Technologie allgemein auf starke Skalierung ausgelegt ist. (vgl. Anhang A.2.2, Z. 187-193)

bgPF

## **Wartbarkeit**

Software wird als leicht wartbar bezeichnet, wenn sie leicht verändert, erweitert, repariert und aktualisiert zu werden, um Fehler zu beheben oder neue Anforderungen zu erfüllen. Dieses Kriterium wird anhand der Analysemöglichkeiten, die für eine Technologie zur Verfügung stehen und ob notwendige Anpassungen an einer Stelle oder über mehrere Systeme verteilt erfolgen müssen, bewertet.

Workflows werden im SAP-System zentral an einer Stelle erstellt, konfiguriert und verwaltet. Von dieser Stelle kann auch zentral das Debugging stattfinden, auch wenn andere Systemkomponenten beeinflusst werden. Somit kann die Wartung als relativ einfach beschrieben werden.

Die Wartbarkeit von Business Events wird maßgeblich davon beeinflusst, inwieweit die Grundsätze der eventgesteuerten Architektur bei der Implementierung eingehalten wurden. Dann kann die gesamte Wartung nur an der Eventquelle geschehen, da die Events



unabhängig von ihrem Konsumenten konstruiert sind. Realistisch ist der Wartungsaufwand und die Komplexität jedoch etwas höher, da oft Konsumenten schon im Voraus aus Effizienzgründen Annahmen über die Gestalt der Events treffen. Somit muss die Wartung dann neben dem Quell-System auch beim Konsumenten und dem Event Broker erfolgen. Auch die Art der Events ist entscheidend, da kleinere Notification Events grundsätzlich einfacher zu warten sind als große Daten Events. Insgesamt sind Business Events jedoch gut wartbar. (vgl. Anhang A.2.2, Z. 194-212)

bgPF

### **Abwärtskompatibilität**

Zuletzt wird noch die Abwärtskompatibilität der Ansätze betrachtet.

Klassische Workflows sind keine neue Technologie mehr und funktionieren in allen gängigen (auch älteren) SAP-Systemen. Auf flexible Workflows trifft das jedoch nicht zu, da diese für die Cloud entwickelt wurden und eine neuere Technologie sind. Bei klassischen Workflows wurde in neueren Versionen lediglich die Benutzeroberfläche angepasst, die Funktionalität ist gleich geblieben. Somit können Workflows, die in neueren Versionen entwickelt wurden problemlos in ältere Systeme kopiert werden. Zusammenfassend eignen sich Workflows sehr gut, wenn Abwärtskompatibilität ein wichtiges Kriterium ist.

Die Abwärtskompatibilität wird auch im Wesentlichen von der verwendeten Technologie bestimmt. Sollen Standard Events zum Einsatz kommen, muss das S/4 Back-End-System immer auf der aktuellsten Version sein, da ältere Versionen die neuen Features nicht unterstützen. Genau für ältere Systeme gibt es das Event Enablement Add-On, was dafür ausgelegt ist, auch noch mit alten SAP ECC Systemen zu arbeiten. Somit sind Business Events zumindest teilweise abwärtskompatibel. (vgl. Anhang A.2.2, Z. 213-223)

bgPF

### **3.3. Entscheidungsmatrix**

Hier soll eine Entscheidungsmatrix entwickelt werden, welchen Lösungsansatz man in Abhängigkeit von mehreren Faktoren am besten verwenden soll.

Experteninterviews:

## **4. Schlussbetrachtungen**

Im folgenden sollen die wichtigsten Ergebnisse noch einmal zusammengefasst werden. Zudem soll eine Handlungsempfehlung gegeben und die Arbeit einmal kritisch reflektiert werden. Abgeschlossen wird mit einem Ausblick auf weitere Entwicklungen.

### **4.1. Zusammenfassung**

### **4.2. Handlungsempfehlung**

### **4.3. Reflexion der Arbeit und Ausblick**

# Literaturverzeichnis

- [1] SAP. *Geschichte der SAP / Über SAP SE*. German. 2023. URL: <https://www.sap.com/germany/about/company/history.html> (Einsichtnahme: 13.07.2023).
- [2] Biehl, M. *API Architecture*. API-University Series. CreateSpace Independent Publishing Platform, 2015. URL: <https://books.google.de/books?id=6D64DwAAQBAJ> (Einsichtnahme: 18.07.2023).
- [3] Fielding, R. „Architectural Styles and the Design of Network-based Software Architectures“. Diss. Irvine: University of California, 2000.
- [4] Fielding, R. *REST APIs must be hypertext-driven & Untangled*. 2008. URL: <https://roy.gbiv.com/untangled/2008/rest-apis-must-be-hypertext-driven> (Einsichtnahme: 13.07.2023).
- [5] SAP. *ABAP RESTful Application Programming Model / SAP Help Portal*. English. 2023. URL: <https://help.sap.com/docs/btp/sap-abap-restful-application-programming-model/abap-restful-application-programming-model?locale=en-US> (Einsichtnahme: 13.07.2023).
- [6] SAP. *SAP UI5 Documentation, Developing Apps with Fiori Elements*. English. 2022. URL: <https://ui5.sap.com/#/topic/03265b0408e2432c9571d6b3feb6b1fd> (Einsichtnahme: 13.07.2023).
- [7] SAP. *Developing SAP Fiori Applications with SAP Fiori Tools / SAP Help Portal*. English. 2023. URL: [https://help.sap.com/docs/SAP\\_FIORI\\_tools/17d50220bcd848aa854c9c182d65b699/f09752ebcf63473e9194ea29ca232e56.html](https://help.sap.com/docs/SAP_FIORI_tools/17d50220bcd848aa854c9c182d65b699/f09752ebcf63473e9194ea29ca232e56.html) (Einsichtnahme: 13.07.2023).
- [8] SAP. *SAPUI5: UI Development Toolkit for HTML5 / SAP Help Portal*. English. 2023. URL: [https://help.sap.com/docs/SAPUI5/9c39e0e47bde4ed993d07d56b3ce3e08/95d113be50ae40d5b0b562b84d715227.html?locale=en-US&state=DRAFT&version=1.98\\_SAPUI5](https://help.sap.com/docs/SAPUI5/9c39e0e47bde4ed993d07d56b3ce3e08/95d113be50ae40d5b0b562b84d715227.html?locale=en-US&state=DRAFT&version=1.98_SAPUI5) (Einsichtnahme: 13.07.2023).

- [9] SAP. *SAP Business Workflow / SAP Help Portal*. English. 2022. URL: [https://help.sap.com/docs/SAP\\_NETWEAVER\\_700/109b51f56c531014b4e7c143c4b731a9/4f41e8a0dd89535fe10000000a421937.html?locale=en-US](https://help.sap.com/docs/SAP_NETWEAVER_700/109b51f56c531014b4e7c143c4b731a9/4f41e8a0dd89535fe10000000a421937.html?locale=en-US) (Einsichtnahme: 17.07.2023).
- [10] SAP. *Business Events / SAP Help Portal*. English. 2023. URL: <https://help.sap.com/docs/btp/sap-abap-restful-application-programming-model/business-events> (Einsichtnahme: 17.07.2023).
- [11] SAP. *Creating an Event Consumption Model / SAP Help Portal*. English. 2022. URL: <https://help.sap.com/docs/btp/sap-abap-development-user-guide/creating-event-consumption-model?locale=en-US> (Einsichtnahme: 17.07.2023).
- [12] SAP. *bgPF / SAP Wiki*. English. 2023. URL: <https://wiki.one.int.sap/wiki/x/FhNsxg> (Einsichtnahme: 17.07.2023).

# A. Anhang

## A.1. Fragebögen Experteninterviews

### A.1.1. Workflows

#### Allgemein

Wer bist du und was ist deine Aufgabe in der AIS? Was machst du in deinem täglichen Alltag und wo kommst du mit Workflows in Kontakt? Darf ich das Interview aufzeichnen, transkribieren und in meiner Praxisarbeit verwenden?

#### Komplexität der Implementierung

Wie komplex ist eine Implementierung von Workflows? (genauere Beschreibung der Implementierung, welche/ wie viele Artefakte müssen erstellt werden, ca. Angabe Lines of Code)

#### Auswirkungen auf die Systemlandschaft

Hat die Implementierung von Workflows Auswirkungen auf die Systemlandschaft? (Zusätzliche Systemkomponenten, Cloud (weitere Implikationen: Datenschutz, rechtliches, ) nötig, )

#### Performance

Wie sind Workflows performance-technisch einzuordnen? (Rechenlast, Ausführungszeit, Netzwerklast)

Ist für Workflows eine Kommunikation über Systemgrenzen hinaus notwendig?

## **Kosten**

Entstehen durch die Implementierung von Workflows zusätzliche Kosten? (Lizenzkosten, Cloud-Abonnement (laufende Kosten), Netzwerk-Traffic)

## **Flexibilität**

Wie flexibel sind Workflows im Bezug auf ihre spätere Anpassbarkeit, Gestaltungsmöglichkeiten bei speziellen Anforderungen, Integrationsmöglichkeiten mit anderen Technologien?

## **Skalierbarkeit**

Wie skalierbar sind Workflows? (Aufteilbar (Load-Balancing) auf mehrere Systeme, Frameworks die Skalierung übernehmen) Bezug auf Fiori-Apps mit sehr hohem Nutzungsvolumen

## **Wartbarkeit**

Beschreiben Sie die Wartbarkeit von Workflows (Wartung zentral über mehrere Instanzen verteilt, Analysemöglichkeiten).

## **Abwärtskompatibilität**

Sind Workflows abwärtskompatibel zu älteren Releases bzw. älteren SAP-Systemen?

## **A.1.2. Business Events**

### **Allgemein**

Wer bist du und was machst du bei SAP?

Wo kommst du mit Business Events in Kontakt?

Darf ich das Interview aufzeichnen, transkribieren und in meiner Praxisarbeit verwenden?

## **Komplexität der Implementierung**

Wie komplex ist eine Implementierung von Business Events? (genauere Beschreibung der Implementierung, welche/ wie viele Artefakte müssen erstellt werden)

## **Auswirkungen auf die Systemlandschaft**

Hat die Implementierung von Business Events Auswirkungen auf die Systemlandschaft? (Zusätzliche Systemkomponenten, Umstrukturierung von Prozessen, Migrationsaufwand on-premise Landschaft zu eventgesteuerter Architektur, Cloud nötig, )

## **Performance**

Wie sind Business Events performance-technisch einzuordnen? (Rechenlast, Ausführungszeit, Netzwerklast)

Ist für Business Events eine Kommunikation über Systemgrenzen hinaus notwendig?

## **Kosten**

Entstehen durch die Implementierung von Business Events zusätzliche Kosten? (Lizenzkosten, Cloud-Abonnement, Netzwerk-Traffic)

In welchem Verhältnis stehen diese zum Mehrwert, der durch eine eventgesteuerte Architektur entsteht?

## **Flexibilität**

Wie flexibel sind Business Events im Bezug auf ihre spätere Anpassbarkeit, Gestaltungsmöglichkeiten bei speziellen Anforderungen, Integrationsmöglichkeiten mit anderen Technologien?



## **Skalierbarkeit**

Wie skalierbar sind Business Events? (Aufteilbar (Load-Balancing) auf mehrere Systeme, Frameworks die Skalierung übernehmen) -> Bezug auf Fiori-Apps mit sehr hohem Nutzungsvolumen

## **Wartbarkeit**

Beschreiben Sie die Wartbarkeit von Business Events (Wartung zentral über mehrere Instanzen verteilt, Analysemöglichkeiten).

## **Abwärtskompatibilität**

Sind Business Events abwärtskompatibel zu älteren Releases bzw. älteren SAP-Systemen?

## **A.1.3. bgPF**

### **Allgemein**

Who are you and what do you do at SAP?

Where do you encounter bgRFCs or the bgPF in your daily work?

May I record the interview, transcribe it and use it in my practice work?

### **Komplexität der Implementierung**

How complex is an implementation of a bgRFC/ bgPF (more detailed description of the implementation, what/how many artifacts need to be created).

### **Auswirkungen auf die Systemlandschaft**

Does the implementation of bgRFC/ bgPF have any impact on the system landscape? (Additional system components, restructuring of processes, cloud necessary, ...)

## **Performance**

How should bgRFC/ bgPF be classified in terms of performance? (Computing load, execution time, network load) Is communication across system boundaries necessary for business events?

## **Kosten**

Does the implementation of bgRFC/ bgPF result in additional costs? (License costs, cloud subscription, network traffic)

## **Flexibilität**

How flexible are bgRFCs/ bgPF in terms of their subsequent adaptability, design options for special requirements, integration options with other technologies?

## **Skalierbarkeit**

How scalable are bgRFCs/ bgPF? (Divisible (load-balancing) to multiple systems, frameworks that handle scaling) Reference to Fiori apps with very high usage volume.

## **Wartbarkeit**

Describe the maintainability of bgRFC/ bgPF. (Maintenance centralized or distributed across multiple instances, analytics capabilities).

## **Abwärtskompatibilität**

Are bgRFC/ bgPF backward compatible with older releases or older SAP systems?

## A.2. Transkripte Experteninterviews

### A.2.1. Workflows

**Befragender:** Tom Wolfrum (Abkürzung: **T**)

**Befragter:** Eric Serie (Abkürzung: **E**)

**Datum:** 21.07.2023

1   **T:**   Hallo Eric, vielen Dank, dass du dir die Zeit für dieses Interview im Rahmen meiner  
2           Praxisarbeit genommen hast. Thematisch soll es heute um die SAP-Technologie  
3           Business Workflows gehen. Doch starten wir bei dir als Person. Wer bist du und  
4           was ist deine Aufgabe in unserer Abteilung AIS HCM? Du kannst auch darauf  
5           eingehen, was du in deinem Arbeitsalltag machst und wo du mit Workflows in  
6           Kontakt kommst.

7   **E:**   Hallo Tom, ich bin Eric Serie. Ich habe 1997 bei SAP angefangen. Ich war anfangs  
8           in einer französischen Abteilung und jetzt seit über zehn Jahren in der AIS HCM.  
9           In meinem Arbeitsalltag kümmere ich mich größtenteils um Kundenmeldungen  
10          und helfe Kollegen bei Fragen. Mein Bereich ist die Personaladministration und ich  
11          betreue die Workflows der SAP Basis. Hier komme ich mit Workflows in Kontakt.  
12          Ich betreue den Teil der Workflows, der mit Personalentwicklung und Objekten wie  
13          Planstellen, Organisationseinheiten und Personalnummern zu tun Hauptprozess.

14   **T:**   Danke für deine kurze Vorstellung. Noch vorneweg: Darf ich das Interview aufzeich-  
15          nen, transkribieren und - wenn du damit einverstanden bist - in meiner Praxisarbeit  
16          verwenden?

17   **E:**   Ja.

18   **T:**   Ok, super. Dann kommen wir zu den inhaltlichen Fragen. Zur Komplexität der  
19          Implementierung: Kannst du beschreiben, wie komplex eine Implementierung von  
20          Workflows ist, also was man genau machen muss und welche Artefakte erstellt  
21          werden müssen?

22   **E:**   Ein Workflow kann sehr einfach sein, also eine einfache Aufgabe erfüllen, wie z. B.  
23          eine E-Mail versenden oder sehr komplex, da ein Workflow viele Aufgaben oder

Schritte haben kann und diese können dann auch wieder sehr verschieden sein, wie zum Beispiel eine Entscheidung oder Genehmigung oder eine komplizierte Aufgabe, wie eine Abwesenheit anzulegen. Es gibt bei Workflows viele verschiedene Schritte und Schritttypen, deshalb können sie sehr komplex sein. Zudem kann in einem Schritt eines Workflows eigenes ABAP-Coding ausgeführt werden, was Workflows auch nochmal komplexer macht.

**T:** Okay, das heißt, dass die Komplexität eines Workflows sozusagen mit der Anzahl seiner Schritte steigt?

**E:** Ja, das ist auf jeden Fall so.

**T:** Kommen wir zur nächsten Frage: Hat die Implementierung von Workflows Auswirkungen auf die Systemlandschaft, also braucht man zusätzliche Systemkomponenten, sind Cloud-Komponenten nötig?

**E:** Nein, Workflows sind komplett in der SAP-Basis enthalten und jedes System hat diese Funktionalität.

**T:** Das heißt, dass wenn sich der Kunde ein SAP ERP-System kauft, ist das dort alles mit enthalten?

**E:** Genau.

**T:** Meine nächste Frage betrifft den Punkt Performance: Wie würdest du sagen, sind Workflows performance-technisch einzuordnen? Mögliche Kriterien wären hier Rechenlast, Ausführungszeit oder Netzwerklast.

**E:** Die Performance ist bei Workflows kein Problem, die Ausführung geht sehr schnell.

**T:** Ok, das heißt, dass das auch im Bezug auf Fiori-Apps, wie zum Beispiel der Prozess einer Krankmeldung, bei denen der Workflow sehr häufig gestartet würde, kein Problem wäre?

**E:** Ja, das wäre kein Problem, weil die Prozesse hinter Workflows meist so aufgebaut sind, dass immer nur kleine Schritte ausgeführt werden und der Workflow nicht auf einmal durchläuft. Zudem sind die hintereinander ausgeführten Schritte meist sehr verschieden und sind für sich sehr schnell in der Ausführung. Häufig sind zum Beispiel Benutzerentscheidungen notwendig, wo auf den Anwender gewartet werden muss und somit ist die Performance der Workflows an sich eigentlich nie

ein Problem. Der einzige Fall, in dem das nicht so ist, wäre, wenn ein Fehler in der Anwendung passiert, was natürlich nur ein Ausnahmefall ist.

**T:** Ok, das heißt, dass Workflows an sich performance-technisch sehr gut sind, weil meist nie der ganze Prozess bzw. Workflow auf einmal durchläuft, sondern eben nur ein kleiner Schritt und die auch nicht alle auf einmal?

**E:** Genau, aber selbst wenn mehrere Schritte auf einmal ausgeführt werden, ist das auch kein Problem, da das alles im Hintergrund passiert und für den Anwender nicht sichtbar ist.

**T:** Und findet die Ausführung von Workflows komplett lokal statt oder ist hier die Kommunikation über Systemgrenzen hinaus notwendig?

**E:** Es kann sein, dass die Kommunikation mit anderen Systemgrenzen durch den abgebildeten Prozess notwendig ist, aber das ist von der Performance her kein Problem. Zudem findet der Großteil der Ausführung von Workflows lokal statt.

**T:** Damit einhergehend der Kostenfaktor: Entstehen durch die Implementierung von Workflows irgendwelche zusätzlichen Kosten?

**E:** Nein, gar nicht.

**T:** Ok, dann auch hier alles mit dem einmaligen Kauf der SAP-Softwarelizenz abgedeckt?

**E:** Ja.

**T:** Dann kommen wir zur Flexibilität von Workflows: Wie flexibel sind Workflows im Bezug auf ihre spätere Anpassbarkeit, Gestaltungsmöglichkeiten bei speziellen Anforderungen und auch auf Integrationsmöglichkeiten mit anderen Technologien?

**E:** Workflows sind ziemlich flexibel, SAP liefert zwei Arten von Workflows aus: klassische und flexible Workflows. Beim klassischen Workflow liefert SAP große Muster-Workflows für bestimmte Geschäftsprozesse aus und der Kunde kopiert sich diese Muster und passt diese dann auf seine Bedürfnisse an. Klar, der Kunde muss etwas von der Technik verstehen, aber er muss die Muster nur an seine Gegebenheiten anpassen. Der Nachteil an diesem Konzept ist, dass wenn der Muster-Workflow fehlerhaft ist und wir eine Korrektur veröffentlichen, diese Korrektur nicht automatisch im System des Kunden übernommen wird, sondern der Kunde seinen Workflow

84 selbstständig anpassen muss. Für die Cloud gibt es ein neues Konzept, die flexiblen  
85 Workflows. Hier liefern wir nur kleinere einzelne Schritte bzw. Workflows, die der  
86 Kunde dann in seinen Workflow integriert. Sollte dann ein Fehler in diesen kleinen  
87 Schritten gefunden werden, ist nur dieser kleine Teil des Workflows betroffen und  
88 der Kunde bekommt zudem automatisch die Korrektur. Zudem sind diese kleineren  
89 Workflows flexibler, da sie sich modular zusammensetzen lassen.

90 **T:** Ok, ich fasse das nochmal zusammen: Vorher [Anmerkung: on-premise-Umfeld] war  
91 es so, dass SAP einen großen Workflow ausgeliefert, der große Prozesse abgedeckt  
92 hat und den der Kunde nur noch an sich anpassen musste. Wenn in diesem Workflow  
93 ein Fehler gefunden wurde, musste der Kunde, euere Korrektur trotzdem noch  
94 einmal selbst bei sich anpassen. Jetzt ist dann sozusagen das Ziel, dass nur noch  
95 kleinere Workflows ausgeliefert werden.

96 **E:** Ja genau, jetzt liefern wir nur noch kleinere Teile und aus diesem muss der Kunde  
97 nur noch seinen Workflow zusammensetzen. Wenn SAP einen Fehler in diesem  
98 kleinen Teilen findet, bekommt der Kunde die Korrektur automatisch.

99 **T:** Du hast ja gesagt, dass sich Workflows aus einzelnen Schritten zusammensetzen.  
100 Das heißt, dass der Kunde sich auch aus allen verfügbaren Einzelschritten bei einer  
101 speziellen Anforderung seinen komplett eigenen Prozess frei zusammenbauen, also  
102 er ist nicht an die SAP-Vorlagen gebunden?

103 **E:** Ja genau, da hast du recht.

104 **T:** Kannst du etwas zu Integrationsmöglichkeiten von Workflows zu anderen Technolo-  
105 gien sagen. Du hast beispielsweise schon das Versenden von E-Mails angesprochen.

106 **E:** Da Workflows Teil der SAP-Basis Softwarekomponente sind, können diese mit allen  
107 anderen Komponenten/ Schichten interagieren. Somit kann jeder Entwickler auf die  
108 Workflows in der SAP-Basis zugreifen und diese gegebenenfalls an ihre Bedürfnisse  
109 anpassen. Deshalb sind Workflows gut mit anderen Technologien integrierbar.

110 **T:** Hier spielt dann ja auch wahrscheinlich mit hinein, dass man in einem Workflow-  
111 Schritt sein eigenes ABAP-Coding ausführen kann und somit alles machen kann?

112 **E:** Ein Beispiel hierfür wären Berechtigungen: Ein Basis-Workflow kann keine Berech-  
113 tigungen, die eventuell in anderen Anwendungen benötigt werden, prüfen. Hier

114 kann man beispielsweise durch eigenes ABAP-Coding diese Berechtigungsprüfungen  
115 selbst implementieren und die Funktionalität des Workflows beliebig erweitern.

116 **T:** Ok, danke. Dann zur nächsten Frage: Wie skalierbar sind Workflows? Ist die  
117 Ausführung eines Workflows auf mehrere Prozesse aufteilbar und gibt es Frameworks,  
118 die die effiziente Skalierung übernehmen? Gerade wir in der AIS setzen Workflows  
119 ja auch in Fiori-Apps ein, die ja ein sehr hohes Nutzungsvolumen haben.

120 **E:** Das ist eine gute Frage. Ich würde sagen, dass Workflows an sich sehr flexibel  
121 sind. Die Workflows der Basis sind sehr performant und schnell in der Ausführung.  
122 Normalerweise gibt es hier keine Probleme bei Anwendungen, die viele Workflows  
123 ausführen müssen.

124 **T:** Das heißt der Workflow an sich ist ein großer zusammenhängender Prozess, der  
125 nicht nochmal unterteilt und über mehrere Applikationsserver verteilt ausgeführt  
126 werden kann.

127 **E:** Ja genau, da hast du recht.

128 **T:** Zum Thema Wartbarkeit: Wie wartbar sind Workflows? Also kann ich das zentral  
129 an einer Stelle machen oder muss ich Probleme über mehrere Komponenten verteilt  
130 suchen?

131 **E:** Nein, die Workflows werden an einer Stelle erstellt und können auch von dort aus  
132 zentral gewartet werden, auch wenn sie andere Komponenten beeinflussen. Die  
133 Wartung ist ziemlich einfach.

134 **T:** Dann kommen wir auch schon zu meiner letzten Frage, die Abwärtskompatibilität.  
135 Oft findet man die Situation vor, dass Kunden eine Systemlandschaft aus neueren  
136 und älteren Systemen haben. Dahingehend die Frage, wie abwärtskompatibel sind  
137 Workflows?

138 **E:** Klassische Workflows funktionieren in allen Systemen. Flexible Workflows jedoch  
139 nicht, da es diese noch nicht so lange gibt und nur für die Cloud gedacht sind.  
140 Klar gibt es im Bezug auf die UI einige Verbesserungen mit den neuen Versionen,  
141 aber abgesehen davon ist es in allen Systemen das Gleiche. Auch das Entwickeln  
142 eines Workflows in einem höheren Release und das Herunterziehen in einen älteren  
143 Release ist überhaupt kein Problem.

144 **T:** Ok gut, das wars dann auch schon mit meinen Fragen. Dann bedanke ich mich für  
145 das Interview und würde dir dann das Transkript nochmal zuschicken, damit das  
146 für dich passt und ich das dann in meiner Arbeit verwenden kann.

147 **E:** Ok, so machen wir das.

## A.2.2. Business Events

**Befragender:** Tom Wolfrum (Abkürzung: **T**)

**Befragter:** Karsten Strothmann (Abkürzung: **K**)

**Datum:** 24.07.2023

1 **T:** Hallo Karsten, vielen Dank, dass du dir die Zeit für dieses Interview im Rahmen mei-  
2 ner Praxisarbeit genommen hast. Thematisch soll es heute um die SAP-Technologie  
3 Business Events gehen. Doch starten wir bei dir als Person. Wer bist du und was  
4 machst du bei SAP? Du kannst auch darauf eingehen, wo du in deinem täglichen  
5 Alltag mit Business Events in Kontakt kommst.

6 **K:** Mein Name ist Karsten Strothmann, ich bin Lead Product Manager für das Event  
7 Mesh und SAP Integration Suite Advanced Event Mesh, das heißt eigentlich für die  
8 Infrastrukturkomponenten, was eventgetriebene Architekturen angeht. Dabei bleibt  
9 es aber üblicherweise nicht, weil letztendlich müssen die Events irgendwoher kom-  
10 men, das heißt Business Applications, Back-End-Systeme und die müssen natürlich  
11 irgendwo konsumiert werden, das heißt üblicherweise habe ich die End-to-End-  
12 Brille auf und schaue da so ein bisschen holistisch drauf, das heißt verschiedenste  
13 Themen und Perspektiven. Ich habe eine recht hohe Bandbreite und die muss ich  
14 auch haben, weil das Thema immernoch recht neu ist und es noch sehr viele Themen,  
15 wie Rechtliches, Coding, Roll-out/ -in bei Kunden und Definitionen von komplett  
16 neuen Prozessen zu klären gibt. Vieles ist einfach noch komplettes Neuland und ich  
17 arbeite sehr viel mit Kunden und auch bei den Kunden ist es so, dass sie gerade  
18 erst anfangen. Wie bin ich dazu gekommen? Ich habe vor einigen Jahren mal an  
19 einem Produkt namens SAP Gateway gearbeitet, was dazu dient synchron SAP  
20 Back-End-Systeme zu erweitern, das heißt, man hat einen OData-Call, kann den  
21 konfigurieren, kann teilweise entwickeln um die Back-End-Systeme zu öffnen und



ruft diese über APIs synchron auf. Das hat sich in Richtung asynchrone Kommunikation entwickelt. Ansonsten bin ich fast 25 Jahre bei der SAP und habe davor bei einem Beratungshaus gearbeitet.

**T:** Dann noch formal: Darf ich das Interview aufzeichnen, transkribieren und in meiner Praxisarbeit verwenden?

**K:** Darfst du sehr gerne machen.

**T:** Ok super, ich würde jetzt mit meinen inhaltlichen Fragen zu Kriterien, anhand denen ich Business Events mit anderen Technologien vergleiche, beginnen. Die erste Frage wäre, wie komplex es ist Business Events zu implementieren. Vielleicht kannst du hier etwas genauer beschreiben, wie die Implementierung abläuft und welche bzw. wie viele Artefakte man erstellen muss.

**K:** Ich fange mal mit der Kundenebene an. Für den Kunden soll es natürlich so einfach wie möglich sein. Idealerweise, wie es bei unseren Standard Events ist, braucht man 10 Sekunden, nachdem die Infrastruktur eingerichtet ist, um ein neues Event aktiv zu schalten, das heißt, die Events können durch das einfach Umlegen eines Schalters exponiert werden. Ganz wichtig ist es, dass die Events aktiv im Back-End-System eingeschaltet werden müssen und nicht standardmäßig eingeschaltet sind. Die Events werden erstellt bzw. entwickelt von unserer Entwicklung, das heißt die erstellen auch die zugehörigen Artefakte, auf die du von der Entwicklungsseite hinaus willst. Zudem können Kunden auch noch Custom Events erstellen. Hier hängt der Aufwand von den verwendeten Technologien ab. Eine Möglichkeit sind RAP-basierte Events. Diese sind aktuell noch aufwändiger und man muss RAP-Know-How haben. Es gibt noch einen zweiten Ansatz, das ist das sogenannte Netweaver Event Enablement Add-On. Das funktioniert mit unseren on-premise Back-End-Systemen, also ECC und S/4. Hier ist es so, dass ich verschiedenste Trigger wählen kann und dann in einem Low-/ No-Code-Ansatz mein Event zusammenklicken kann. So etwas geht normalerweise in 20 Minuten. Was man dafür braucht hängt normalerweise vom Ansatz ab, man kann auch das komplett high-level machen, also konfigurationsbasiert im SAP-UI oder man fängt an sich das Event selbst zu programmieren. Ich selbst habe das noch nicht gemacht, das wäre eher für sehr spezielle Fälle, in denen man Events anpassen will oder auf Back-End-Seite nochmal filtern will, dann kann man das hiermit umsetzen. Das heißt, letztenendes

ist der Aufwand für den Kunden komplett unterschiedlich. Von minimal in ca. einer Minute bei Standard-Events über 20 Minuten, um mit dem Event Add-On ein Custom Event zu erstellen bis hin zu drei Tagen für ein hochoptimiertes Event mit RAP. Auch die Artefakte sind je nach verwendeter Technologie und Trigger jedes Mal unterschiedlich. Bei dem RAP-basiertem Ansatz als vierte Variante werden die Standard Events im Herbst eine konfigurierbare Erweiterbarkeit bekommen, das Ganze nennt sich dann "Derived Events". Dadurch kann man die Standard Events dann konfigurationsbasiert erweitern. Die Kernaussage ist, dass das ganze Spektrum vom einfachen Umlegen eines Schalters, ohne das Erstellen von Artefakten bis hin zu einer Reihe von Artefakten bei selbst geschriebenen Events vorhanden ist.

**T:** Ok, danke. Ein weiteres Kriterium, was ich mir anschauen möchte sind die Auswirkungen auf die Systemlandschaft. Inwiefern hat die Implementierung von Business Events Auswirkungen auf die Systemlandschaft, also im Bezug auf zusätzliche Systemkomponenten, die Umstrukturierung von Prozessen und der Migrationsaufwand einer reinen on-premise Landschaft zu einer eventgesteuerten Architektur mit Business Events?

**K:** Also letztendlich muss man natürlich seine Prozesse anpassen, da man von synchronen batchorientierten Prozessen weg, hin zu komplett asynchronen, hoch parallelen Prozessen umsteigt. Ein passender Vergleich wäre der eines Nachrichtendienstes: Im Back-End passiert etwas, das Event wird abgeschickt und jeder der auf ein Event lauscht, wird informiert über die Änderung. Man hat sozusagen eine geschlossene Gruppe in die ein Event geschickt wird und über das dann jeder in dieser Gruppe informiert wird. Wenn man dann sozusagen vom Telefonieren als synchrone Kommunikation zu einem Nachrichtendienst als asynchrone Kommunikation wechselt, stellt man optimalerweise auch seinen Geschäftsprozess um. Das heißt, es wird definitiv Änderungen dabei geben. Sehr viele Kunden haben bei dieser Umstellung ein Problem, da man versucht mit alten Sichtweisen an die neuen Ansätze heranzugehen, was für Probleme sorgt. Wenn man von einem batch-basiertem Ansatz der einmal am Tag läuft zu einem Echtzeit-Ansatz wechselt, müssen die Geschäftsprozesse zwangsläufig angepasst werden.

**T:** Ganz allgemein: Wenn ich Business Events benutzen möchte, brauche ich aber schon eine Cloud-Komponente in meiner Systemlandschaft, wie eben das Event-Mesh?

86 **K:** Ja, es wird ein Event Broker oder Mesh benötigt. Der SAP Event Mesh ist  
87 eigentlich nur ein Event Broker, der irreführend benannt ist. Man braucht mindestens  
88 ein Broker, wie zum Beispiel das SAP Event Mesh als Cloud Komponente. Wir  
89 positionieren zusätzlich gerade den Advanced Event Mesh, womit man tatsächlich ein  
90 Netzwerk aus mehreren Brokern erstellen kann. Das ist zur Zeit die Marschrichtung.  
91 Das hat den Vorteil, dass unsere Kunden meistens global aufgestellt sind und ihre  
92 Back-End-Systeme weltweit verteilt sind. Meistens wird ein Event Broker neben  
93 das Back-End-System gestellt, dem das Event übergeben wird und müssen sich  
94 dann um nichts mehr kümmern. Der weitere Punkt ist, dass diese Event Broker  
95 nicht in der Cloud stehen. Das ist vor Allem für Kunden, die nicht mit ihren  
96 Daten in die Cloud gehen möchten, von Vorteil, da man diesen Event Broker auch  
97 lokal on-premise betreiben kann. Insgesamt ist die Cloud somit keine zwingende  
98 Voraussetzung, wenn sie auch enorme Vorteile, wie die globale Vernetzung der  
99 Systeme mit Events bietet. Um es ganz deutlich zu sagen: Es wird ein Event Broker  
100 als zusätzliche Systemkomponente benötigt. In den ABAP-Systemen gibt es schon  
101 sehr lange Events. Diese sind jedoch immer nur lokal genutzt worden. Jetzt ist der  
102 große Unterschied, dass diese Events jetzt extern und global verteilt werden können,  
103 teilweise über Herstellergrenzen hinweg.

104 **T:** Das heißt, dass ich mit dem SAP Event Mesh auch Events anderer Hersteller  
105 verarbeiten könnte?

106 **K:** Unsere Events sind normalerweise im Cloud Events Format, das von Microsoft,  
107 Google, SAP, Oracle und weiteren Tech-Konzernen vorangetrieben wird. Das heißt,  
108 wir können die Events auch unter den Herstellern austauschen.

109 **T:** Danke. Mein nächstes Kriterium wäre die Performance. Wie kann man generell  
110 Business Events performancetechnisch einordnen? Mögliche Messgrößen wären hier  
111 Rechenlast, Ausführungszeit und Netzwerklast.

112 **K:** Die Events können unterschiedlich groß sein, von extrem kleinen Notification-  
113 Events bis hin zu sehr großen Daten-Events. Lasst uns das mal versuchen zu  
114 strukturieren: Das Back-End-System, der Event Broker und die Konsumenten. Für  
115 das Back-End und die Konsumenten ist es von enormen Vorteil, da die Alternative  
116 zum eventgesteuerten Ansatz das permanente Rufen und Prüfen auf Änderungen  
117 des Back-End-Systems ist. Dadurch, dass dieses permanente Rufen entfällt wird

erheblich Performance gespart, da einfach nur ein Event losgeschickt wird, wenn tatsächlich etwas passiert. Der Broker ist darauf ausgelegt, enorm zu skalieren. Bei der Erstellung von diesen Events muss man sehr vorsichtig sein, dass man nur wirklich benötigte Events erstellt und dass die Events möglichst groß werden. Denn in diesem Moment erzeugt man Last auf dem Back-End-System. Man muss die Balance zwischen dem Einsparen von API-Calls und dem Erzeugen der Events in der richtigen Größe. Hier ist definitiv ein großes Optimierungspotenzial bei den S/4-Kollegen, gerade weil unsere Back-End-System etwas dahin tendieren, sehr monolithisch zu sein.

**T:** Das ging jetzt vielleicht schon etwas aus den vorherigen Fragen hervor, aber die Kommunikation über Systemgrenzen hinaus liegt ja in der Natur der Business Events, oder?

**K:** Richtig, Business Events sind darauf ausgelegt. Die Standardvorgehensweise ist meistens, dass die SAP-Events aus den Systemen nach außen geschickt werden, da die SAP-Back-End-Systeme als Eventquellen sehr gut sind und die SAP-Events sehr wertvoll sind. Dass andere Events von außen in die SAP-Systeme kommen ist etwas seltener, aber passiert definitiv auch, also es soll und muss definitiv in die Richtung herstellerübergreifende Kommunikation gehen.

**T:** Ok, mein nächster Punkt wäre die Kostenseite, also entstehen durch die Implementierung von Business Events zusätzliche Kosten, wie zum Beispiel durch Cloud Komponenten und in welchem Verhältnis stehen diese zum Mehrwert, den eine eventgesteuerte Architektur bietet?

**K:** Das sind ziemlich gute Fragen, die auch in der Forschung noch nicht so richtig beleuchtet wurden. Ein Broker kostet definitiv Geld. Es gibt allerdings unterschiedlichste Größenordnungen. Wenn ich den SAP Event Mesh nehme, ist der enorm günstig, da er auf die Verwendung von kleinen Notification-Events ausgelegt ist und die Kunden in Abhängigkeit von ihrem Nutzungsvolumen bezahlen. Meistens läuft es hier im Monat auf mehrere Hundert Euro hinaus. Jedoch ist er auch nicht darauf ausgelegt, ein ganzes Unternehmen zu bedienen. Das heißt, man bedient hier lediglich Teilbereiche und spezielle Anwendungsfälle, dafür sind die Kosten aber auch nur sehr gering. Um die komplette Architektur eines Unternehmens zu bedienen, egal ob man andere umfassendere Event Broker anderer Hersteller oder

den SAP Integration Suite Advanced Event Mesh nimmt, ist eine ganz andere Liga und hier bewegen sich die Beträge eher im sechsstelligen Bereich und aufwärts. Zudem ist der Ansatz ein anderer: Beim Event Mesh findet pay-per-use (Bezahlen pro Benutzung) Anwendung, während man bei solchen größeren Technologien seinen eigenen Broker hat und diesen für die gesamte Zeit, die er läuft bezahlt, egal ob man ihn nutzt oder nicht. Das heißt, da reden wir über große Summen Geld. Es ist keine günstige Technologie, aber es gibt einen Break-even Punkt, ab dem man Geld spart, da die Infrastruktur sowieso da und wiederverwendbar ist. Events zu bauen hingegen ist deutlich einfacher, ich hatte hier vorhin 20 Minuten gesagt, wenn keine Standard Events da sind. In den älteren Ansätzen dauert das deutlich länger und ist wesentlich komplexer. Im Endeffekt reden wir über ca. 100 Events pro Kunde, ab denen der Break-even Punkt überschritten wird. Das kommt immer auf den Anwendungsfall an, es gibt auch Kunden, die den Break-even Punkt schon nach 2 Events erreichen.

**T:** Ok, dann zum Thema Flexibilität: Wie flexibel sind Business Events im Bezug auf spätere Anpassbarkeit, Gestaltungsmöglichkeiten bei spezielleren Anforderungen und Integration mit anderen Technologien?

**K:** Erstmal hängt das wieder an den unterschiedlichen Ansätzen, also der verwendeten Technologie. Die Standard Events waren bisher am schlechtesten anpassbar, da konnte man überhaupt nichts machen. Da aber diese Derived Events kommen werden, wird das relativ einfach werden. Diese Custom Events mit dem Add-On waren schon immer sehr flexibel, die können innerhalb kürzester Zeit angepasst werden. Kernaussage ist, dass es von der Eventquelle abhängt, im Regelfall sind die anpassbar, S/4 war bisher die Ausnahme, aber das ändert sich ab Herbst und dann sind auch diese Events sehr flexibel anpassbar. Zur Kombination mit anderen Technologien: Die Events sind zum Teil darauf ausgelegt, dass sie mit anderen Events kombiniert werden, das heißt bei den SAP Standard Events, das waren Notification Events, also sehr klein und enthalten nur die notwendigsten Daten, ist man davon ausgegangen, dass die relevanten Informationen vom Konsumenten sowieso per gesondertem API-Call angefordert werden, wenn das konsumierte Event für ihn interessant ist. Das heißt Events sind gut und einfach kombinierbar, zumindest solange man sich im SAP-Bereich befindet. Wenn man jedoch eine eventgetriebene Architektur nach der Definition umsetzt, kennt die Eventquelle die Konsumenten

der Events nicht. In Kundensystemen ist dies jedoch eigentlich immer der Fall, sodass hier eventgesteuerte Architekturen meist eher zu 80% umgesetzt werden und die Kunden biegen sich die restlichen 20% durch das Treffen von Annahmen an der Eventquelle über den Konsumenten hin.

**T:** Dann zum Kriterium der Skalierbarkeit: Wie skalierbar sind Business Events?

**K:** Business Events sind extrem skalierbar. Auch hier wieder die Unterscheidung zwischen Back-End (Quelle) und Broker. Beim Broker reden wir über Milliarden von Events pro Tag, wenn es sein muss. Als Beispiel laufen im Einzelhandel meist eventgetrieben Architekturen und hier kann man sich vorstellen, dass hier enorme Datenmengen beim Abscannen der Artikel über die Systeme laufen. Also das skaliert sehr gut, genau darauf ist es ausgelegt und das ist eine der Stärken.

**T:** Zwei Fragen habe ich noch: Einmal zum Thema Wartbarkeit: Wie gut wartbar sind Business Events?

**K:** Business Events sind sehr einfach wartbar, solange man sich an die Grundsätze hält. Standardmäßig wartet man Events nur an der Eventquelle. Wenn man eventgesteuerte Architektur, rein wie in der Lehre umsetzt, da die Events hier sowieso so konstruiert sind, dass sie unabhängig vom Konsumenten sind. In der Realität ist es nicht so ganz einfach, hier muss man etwas an der Eventquelle, am Broker und am Konsumenten machen. Das heißt, je nachdem welcher Ansatz gewählt wird, klassisch eventgetrieben ist sehr einfach wartbar. Angepasst eventgetrieben ist der Aufwand höher, weil potenziell mehrere Stellen angepasst werden, weil z. B. der Konsument Annahmen über die Events schon im Vorhinein trifft. Hier merkt man auch, dass das noch ein neues Thema ist, weil es auf solche Fragen noch keine endgültigen Antworten gibt. Hier reicht das Spektrum von nur kleinen Notification Events fast ohne Daten, wo man immer einen API-Call machen müsste bis hin zu großen Daten Events die immer gleich alle Daten enthalten und wo dann der Konsument erst entscheiden muss, was er mit dem Event bzw. den Daten macht. Das S/4 ist genau in der Mitte, hier gibt es Decision Events, die genügend Informationen bieten, aber nicht alle. Bei Bedarf können dann noch mehr Informationen per API-Call angefordert werden.

**T:** Meine letzte Frage betrifft die Abwärtskompatibilität: Gerade im Hinblick auf heterogene Kunden-Systemlandschaften mit teilweise auch älteren Systemen: Wie

215       abwärtskompatibel sind hier Business Events im Bezug auf ältere Releases oder  
216       ältere Systeme?

217 **K:**   Es kommt wie immer auf die gewählte Variante an. Beim S/4 muss man bei den  
218       Standard Events immer die neueste Version haben, da mit den Versionen immer  
219       mehr neue Features hinzugefügt werden und es keine Downports gibt. Das ist ein  
220       Problem, weil die Kunden immer die neueste Version haben müssen. Bei dem Event  
221       Enablement Add-On ist es genau anders herum, da das darauf ausgelegt ist auch  
222       mit sehr alten ECC-Systemen zu arbeiten. Man kann Events zwar versionieren,  
223       aber dieses Feature kommt auch erst in Zukunft.

224 **T:**   Gut, das wars dann auch schon mit meine Fragen. Ich bedanke mich für das  
225       Interview und würde dir das Transkript dann noch im Nachgang zum drüberlesen  
226       zuschicken.

227 **K:**   Gerne.

### **A.2.3. bgPF**

**Befragender:** Tom Wolfrum (Abkürzung: **T**)

**Befragter:** Robert Wang (Abkürzung: **R**)

**Datum:** 27.07.2023

1   **T:**   Hello Robert. Todays interview is about the bgRFC. Thank you for taking the time.  
2       I'm just going to start with a few general questions. Who are you and what do you  
3       do at SAP? Maybe you can also say something about where you encounter bgRFC  
4       in your daily work.

5   **R:**   My name is Robert, as you know. Im working on the pure ABAP-Stack in the  
6       component SAP-Basis, the Netweaver, as it was called and the connectivity team,  
7       which belongs to CST Group, which means Client-Server-Technology. I am the code  
8       owner of the bgRFC Framework. Basically we maintain and develop new features  
9       for the bgRFC Framework and also help other application colleagues to use bgRFC  
10       efficiently. And for my daily work, I would say in 10% of the time I develop new  
11       features and the other 90% are allocated to maintenance of the bgRFC. I work

12 with ABAP-Systems, that are shipped to the customer or internally with other  
13 applications using bgRFC.

14 **T:** Ok, then for the record: May I record, transcribe and use the interview for my  
15 project work?

16 **R:** Yes, definitely.

17 **T:** Ok, thanks. So now the first question about the bgRFC is about the implementation:  
18 Can you say something about how complex the implementation of the bgRFC is,  
19 maybe describe it with a bit more details or say how many arefacts need to be  
20 created. And explain just how it works in general.

21 **R:** Yes, sure. If you want to do something based on the bgRFC, you have to do certain  
22 things beforehand. For example you have to configure inbound destinations and  
23 also tune do some configuration for the framework. This is more or less from the on  
24 premise system, where the customer has a maintenance team. So for each product,  
25 they also have online documentation to tell the customer support team what what  
26 steps should be taken to configure bgRFC correctly, and then you can use the  
27 bfRFC as technologies to develop your own code. While this part is quite simple,  
28 because it is just a syntax where you you call something in the background unit  
29 but the function module itself, you have to develop beforehand. So in my opinion  
30 is it's quite simple, it's just a couple lines of code.

31 **T:** Ok. Then my next question would be about the system landscape of the customer  
32 that's implementing it. Does implementing bgRFC impose any kind of implications  
33 or impact on the system landscape so e.g. are additional system components required?  
34 Do I need to restructure my processes? Do I need cloud components? Or is everything  
35 already part of the shipped system and happens there locally?

36 **R:** For this question a I would say yes of course. Any new code will impact the bgRFC-  
37 calls and but the RFC's will consume the dialogue work processes, so from this  
38 point of view, yes. But in bgRFC framework we also have resource control to not  
39 screw up the entire system. You can tune the configuration and parameters to  
40 either have more resources or less allocated to bgRFC, depending on how quick  
41 you you want the bgRFC unit to be processed now, yes I would say yes, but not  
42 much of an impact.



43 **T:** Ok, but regarding additional system components, because for example another  
44 technology that I'm comparing with it needs a a cloud component, so this is not  
45 required?

46 **R:** Oh ok, no, not really, because the bgRFC is on the basis layer of the ABAP System,  
47 so its available on every shipped instance and directly ready to use.

48 **T:** That's what I meant, so it all happens locally in the system?

49 **R:** Exactly, yes.

50 **T:** Ok, then my next question is about performance. How can you classify bgRFC in  
51 terms of performance? I think, you already mentioned, that you can change, how  
52 many ressources it can take up.

53 **R:** So first of all, we classify this as asynchronised technology in the ABAP world  
54 and it's not a synchronised call, meaning if one application tries to use bgRFC to  
55 perform something, they have to expect some delay but it depends on the specific  
56 task to be performed. If you e.g. create one million calls in a very short time, so the  
57 unit might be running for a long time. So we cannot execute one million calls in for  
58 example ten minutes. So this is given and because this is a asynchronous technology,  
59 you have to expect some sort of delay but from the pure performance point of  
60 view, it depends on the configuration. So you can tune the system to allocate more  
61 resources or less resources to be used for bgRFC, so this is very flexible.

62 **T:** Ok, so I think the next question might be a pretty short one, but I have to ask the  
63 question to compare the technology to others: Does the implementation of bgRFC  
64 result in any additional costs for the customer implementing it?

65 **R:** No, it's ready in any ABAP System.

66 **T:** You already mentioned flexibility. How flexible would you say are bgRFCs in terms  
67 of their adaptibility, or if I have special requirements, how flexible is it to meet  
68 those requirements and what about integration options with other technologies?

69 **R:** bgRFC, as I said is a asynchronised technology, So what we did is basically when  
70 you create a bgRFC unit in your own application, the unit will not be called  
71 immediately, the call is just saved into the database table. So this is the first  
72 step and the second step is, we have a scheduler running all the time and it will

73 then pick up the unit to trigger it. So regarding the integration possibilities: Any  
74 application can use the bgRFC, so there are no scalability problems. We have a  
75 scheduler running on each application server. You can basically configure one or  
76 more schedulers and configure when the unit will be triggered. So for example if  
77 you have 10 application servers and you really just want have three application  
78 servers running the unit, it's also flexible, so you can configure it to not be triggered  
79 on the other seven servers.

80 **T:** Ok, so my next question would be about scalability. How scalable are bgRFC's?  
81 Maybe there is some kind of load balancing across multiple application servers  
82 possible? Or the framework is already handling that? I think you already talked  
83 about having multiple schedulers.

84 **R:** From the bgRFC point, we are only talking about the units. If we have multiple  
85 units and multiple queues, the load balancing is done from two perspectives: The  
86 first is the scheduler itself and the other one is the execution of the unit. In your  
87 ABAP System you can configure for each application server, whether it should run  
88 a scheduler or not. So I can configure how many application servers are running a  
89 scheduler and how many schedulers are running in one application server. So here it  
90 is very flexible. This was the scheduler point of view. When the scheduler triggers a  
91 unit, it is also possible to configure on which application server the unit is executed.  
92 It can be executed by all application servers or only by certain or one server.

93 **T:** Ok, the next question would be about maintainability. Can you describe the  
94 maintainability of bgRFCs? E.g. is the maintenance possible centrally from one  
95 point, or do you have to change something in multiple places?

96 **R:** So basically the bgRFC is shipped in default configuration with any ABAP System.