PROBEKAPITEL

Tim Mehlfeld
Dr. Christian Schneider
Kai-Uwe Stahl
Andreas Wiener

Reporting- und Business-Intelligence-Werkzeuge für den Controller

Grundlagen und praktische Tipps von Anwendern, Beratern und Herstellern

REPORTING IMPULSE

Das Team reportingimpulse wünscht Ihnen viel Freude beim Lesen von unserem Probekapitel!

Wenn Sie weitere Fragen zu unserem Buch haben, schreiben Sie uns gerne eine E-Mail.

Das gesamte Buch können Sie in unserem Shop oder auf Amazon bestellen.

REPORTING IMPULSE

Browse reportingimpulse's Courses & Learning

Own your future by learning new skills.

Become a Visual Data Analytics Expert

Structured learning path developed by experts

Conceptual knowledge presented state of the art

Practical use cases to implement the learned impulses

Personal expert feedback for implemented use cases

Best practice solutions in PowerBI, Excel & Tableau

Certification as Visual Data Analytics Expert

Evolutionsweg eines Organisations-Tools für SAP BW

Inhalt

1 Ausgangspunkt 95
2 Erster Evolutionsschritt: Die Toolsammlung 95
3 Zweiter Evolutionsschritt: Die Applikationsübersicht 96
4 Dritter Evolutionsschritt: Die Betriebsunterstützung 98
5 Vierter Evolutionsschritt: Die Entwicklungsunterstützung 99
6 Fünfter Evolutionsschritt: Die Methodenentwicklung 99
7 Sechster Evolutionsschritt: Die Integration der Anwender 100
8 Weitere Evolutionsschritte 101

Autor

Rolf Göhl ist Manager SAP BI in der Wirtgen Group und dort zuständig für die technischen Lösungen und Services im SAP BI Umfeld. Er ist seit 17-Jahren im Business Intelligence (BI) Bereich tätig. Neben dem tiefen technischen Verständnis von BI Lösungen und deren Integration in die Organisation, ist die enge Verzahnung von fachlichen Anforderungen und technischen Entwicklungsaufgaben in der täglichen Arbeit ein Schwerpunkt seiner Tätigkeit. Er ist Begründer der „wissensbasierten Entwicklung mit Smart-Solutions", einem LifeCycle-Applikations-Entwicklungs-Ansatz, in dem die Trennung zwischen Projekt/Entwicklung und Support vollständig aufgehoben wird.

Ich habe dies alles natürlich nicht alleine gemacht, deshalb möchte ich mich an dieser Stelle bei allen Beteiligten meines Teams für die hervorragende Arbeit bedanken, die sie in den letzten Jahren geleistet haben. Mein ganz besonderer Dank gilt meinem lieben Kollegen Michael Hofebauer für die vielen Ideen und Denkanstöße in und aus unseren Gesprächen, die den obigen Weg geebnet haben. Mein weiterer ganz besonderer Dank geht an Herrn Heiko Einsiedler, dem wir die schnelle und zügige Umsetzung in ABAP Code verdanken. Ohne ihn wären wir wohl mit dem Tool nicht dort, wo wir heute stehen.

1 Ausgangspunkt

Tools verändern unsere Art zu arbeiten. Solange ein Tool nicht vorhanden ist, wissen wir oft noch nicht einmal, dass es möglich ist, für unsere Aufgabe ein Tool zu nutzen.
Ich möchte diesen Wandlungsprozess an einem konkreten Tool zeigen, welches wir heute aktiv nutzen. Gleichzeitig möchte ich darstellen, wie überaus nützlich dieses Tool inzwischen für unsere Entwicklung und den Betrieb von SAP BW Applikationen geworden ist. Letztendlich ist es uns durch dieses Tool gelungen, neue Anwendungsfelder und -gruppen zu erschließen.
Ich arbeite als Leiter einer BI-Abteilung in einem bis zu 16 Mann zählenden Team von Beratern und Entwicklern im SAP-BW-Umfeld. Während meiner 18-jährigen Laufbahn in diesem Umfeld habe ich in einer Vielzahl verschiedener BI-Projekte gearbeitet. Ich habe viele verschiedene Organisationen kennengelernt und bin von der Programmierung über die Projektverantwortlichkeit bis hin zur Leitung eines BI-Bereichs vorangeschritten. In diesen Jahren habe ich viel gesehen und mir häufig die Frage gestellt, warum eigentlich alle die gleichen Probleme haben und warum es uns so schwer fällt, diese nachhaltig zu lösen?
Dabei sind mir folgende Aspekte immer wieder aufgefallen:

1. In jeder Aufgabenstellung wurde immer wieder von vorne angefangen, d. h. aus den Erfahrungen der Vergangenheit wurde in der Organisation nicht gelernt bzw. hat jeder nur für sich allein gelernt.
2. Jeder Entwickler hat so gearbeitet, wie er es gelernt hat. Gemeinsame Arbeitsweisen oder -methoden gab es oder gibt es bis heute nicht.
3. Die Qualität der Entwicklung hängt bis heute von den individuellen Fähigkeiten des jeweiligen Entwicklers ab.
4. Die Art der Entwicklungsvorgehensweise in Projekten führt in allen Organisationen zu Silos und individuellen Ergebnissen, was negative Folgen für den Gesamtbetrieb einer so entstandenen Landschaft hat.
5. Es wurden i. d. R. keine Synergien aus den vielen Projekten geschaffen.
6. Eine transparente Sicht auf alle Entwicklungen in einem System gab es in keinem der Unternehmen, in denen ich über die Jahre tätig war.

Zudem sind neue Fragestellungen rund um das Thema Self Service in heutigen BI-Anwendungen zu berücksichtigen, was zu einer stärkeren Integration der Fachanwender in die Entwicklung und den Betrieb von SAP-BW-Applikationen führt. Doch wie löst man solche Fragestellungen im operativen Betrieb? Gibt es überhaupt Lösungen, die mit vertretbarem Aufwand implementiert werden können?
Auf den folgenden Seiten skizziere ich unseren Weg hin zu dem heutigen Werkzeug, welches wir AC (Applikations Cockpit) nennen. Dabei beschreibe ich zum einen die entstandenen Funktionalitäten und zum anderen die organisatorischen Veränderungen und Nutzen, die diese neuen Funktionalitäten zur Folge hatten.

2 Erster Evolutionsschritt: Die Toolsammlung

Aus der täglichen Arbeit in unserem Team sind über die Zeit verschiedene kleine universelle Programme entstanden, die wir für die täglichen Aufgabenstellungen als nützlich identifiziert haben. Um nicht immer in der Entwicklungsumgebung (RSA1/SE80) suchen zu müssen, überlegten wir, ob wir nicht eine integrierte Oberfläche zur Verfügung stellen sollten, welche die Nutzung der kleinen Programme erleichtern würde.
So entstand der erste Vorläufer des AC, ein einfacher Hierarchiebaum, in den die Programme nach vermeintlich sinnvollen Kriterien gruppiert wurden. Bei diesem ersten Vorläufer unseres heutigen Tools war auch schon der Gedanke da, technikaffinen Mitarbeitern des Fachbereichs ebenfalls Zugriff auf diese Toolsammlung zu geben, so dass diese sich selbst bedienen könnten.
Schnell sammelte sich eine beachtliche Anzahl an Verknüpfungen zu Programmen, Customizing-Tabellen und anderen Systemeinstellungen, die über die Jahre in den verschiedenen Entwicklungen und Aufgabenstellungen angefallen waren.
Der Nutzen dieser ersten Oberfläche war es primär, einen zentralen Einstiegspunkt zu haben, an dem man nach einem nützlichen Programm suchen konnte.

Allein das Wissen darum, dass es jetzt solch einen Ort im System gab, führte dazu, dass immer mehr „kleine Schätze“, wie z. B. nützliche Analyseprogramme für den täglichen Betrieb, hier eingestellt wurden. Wissen, das bisher nur in den Köpfen einzelner Entwickler existierte, wurde so auch für andere Mitglieder des Teams sichtbar und somit anwendbar. Der erste Self Service für Entwickler und Anwender war geboren. Hier offenbarte sich ein wesentlicher Aspekt von Tools: Sie machen Wissen von Einzelnen für viele nutzbar. Sie schaffen Orientierungspunkte in der Fülle der Anwendungen und im Weiteren für kommunikative Prozesse.

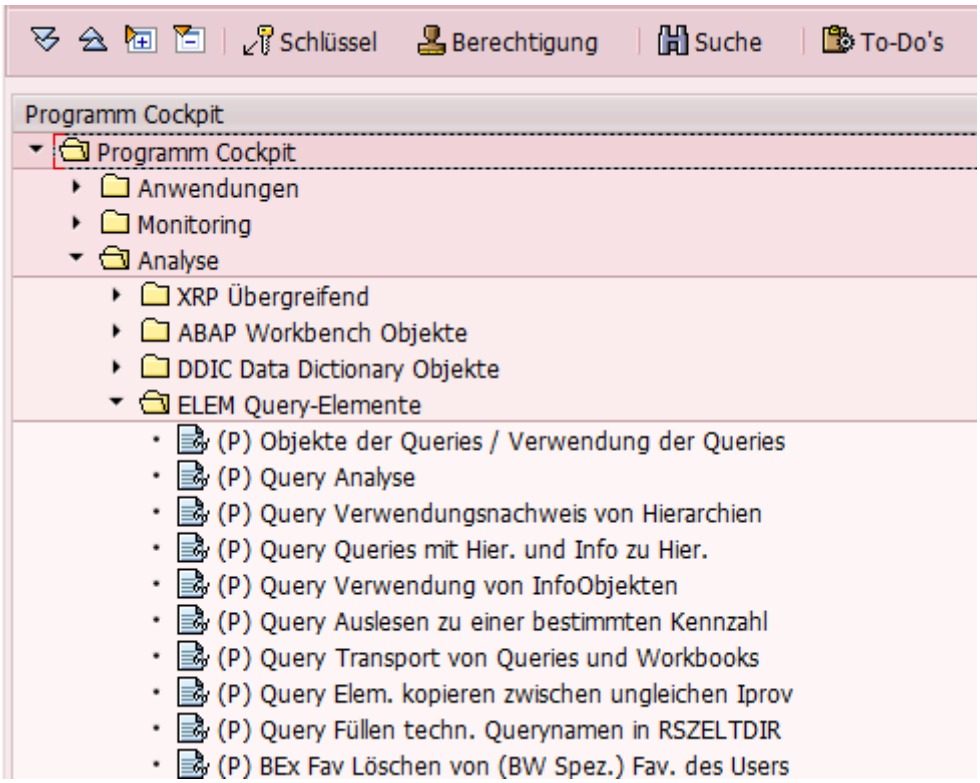


Abbildung 1: Toolsammlung

3 Zweiter Evolutionsschritt: Die Applikationsübersicht

Während sich das erste Programm Cockpit in der Praxis bewährte, stellte ich mir die Frage, wieso wir eigentlich keine Übersicht über alle unsere Applikationen im System haben. Und wie solch eine Übersicht aussehen müsste, damit sie im täglichen Betrieb einen Nutzen bringen würde.
Ausgangspunkt der Überlegungen waren die täglich anfallenden Meldungen zu unseren Applikationen. Das erste, was mir auffiel, war, dass die Mitarbeiter in vielen gemeldeten Fällen eine Einzelanalyse der Applikation starteten, was sich in manchen Fällen als sehr zeitaufwendig gestaltete. Beobachtungen über längere Zeiträume hinweg brachten immer das gleiche Ergebnis. Selbst bei ähnlichen Anfragen, die schon einmal durchgeführt wurden, kam es dazu, dass Analysen erneut durchgeführt wurden. Selbst die Verwendung gespeicherter Tickets konnte hier nur rudimentär helfen. Warum wurde nicht aus den Erfahrungen der vorangegangenen Betriebsfälle gelernt?
Ich dachte damals, das Hauptproblem sei das nicht systematische Dokumentieren. Wie ich in den folgenden Monaten lernen sollte, war ein viel einfacheres Problem von größerer Bedeutung. Nämlich die Frage, wie man zuverlässig etwas einmal Dokumentiertes wiederfindet. Hier stellt das *schnelle* Wiederfinden die eigentliche Herausforderung in einem konkreten operativen Anwendungsfall dar. Ende 2015 wurde dann eine so einfache wie zuerst für unmöglich gehaltene Idee geboren: die Idee, jedes technische Objekt im System einer Applikation zuzuordnen. Dabei verstanden wir eine Applikation als eine möglichst sinnvolle Gruppierung von technischen Objekten zu einem anwendungsbezogenen bzw. betriebswirtschaftlichen Kontext. Wichtigster Zweck

dieser Gruppierungen war es, fachliches und technisches Wissen über eine Applikation an einem Ort zusammenzuführen. Dieses Vorgehen stellte etwas Fundamentales in einem operativen Anwendungsfall sicher: Ein technisches Objekt lässt sich im Rahmen einer Fehleranalyse sehr schnell identifizieren. Mit der Zuordnung der technischen Objekte zu den Applikationen lassen sich dann auch die Dokumentationen zu dem Objektkontext schneller finden. Umgekehrt ist damit aber auch der Ort der Dokumentation definiert, d. h., wenn man ein Objekt einer Applikation verändert oder ein neues dieser hinzufüge, dann weiß man, wo man das zu dokumentieren hat.

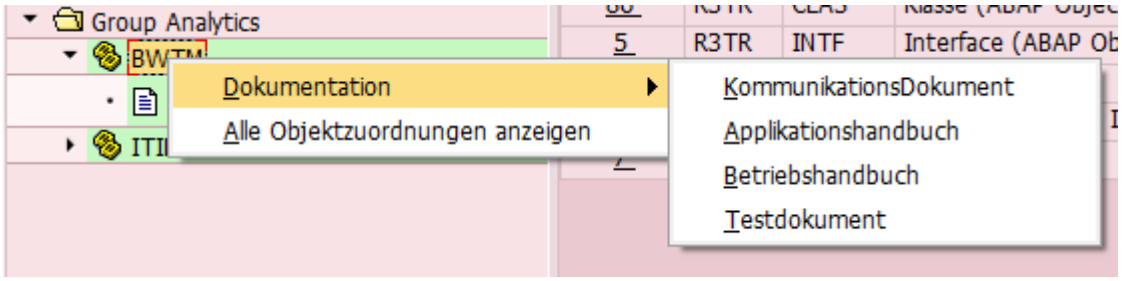
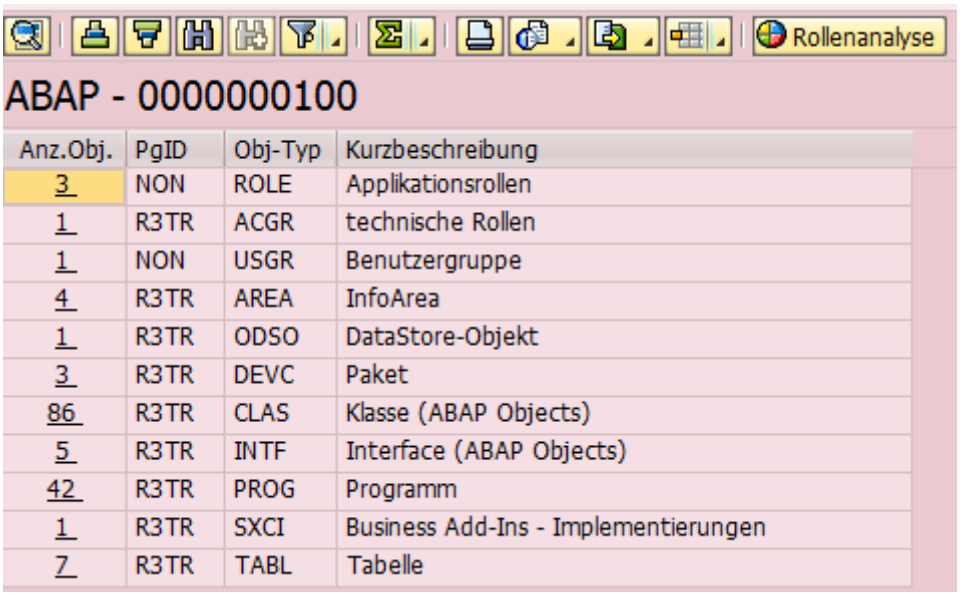


Abbildung 2: Applikation und Zugriff auf die Dokumente

Nach zweiwöchiger Implementierungsphase konnten wir mit dem ersten Prototyp starten. Dieser bestand im Prinzip nur aus einer Hierarchiedarstellung der Applikationen und einer Anzeige der zugeordneten technischen Objekte. Neben der Funktionalität, Objekte einer Applikation zuzuordnen, bestand auch die Möglichkeit, auf die Dokumente einer Applikation zugreifen zu können. Die Grundidee war, dass zu jeder Applikation genau ein Handbuch existiert, d. h. ein eindeutiger Ort der Wahrheit.



Anz.Obj.	PgID	Obj-Typ	Kurzbeschreibung
3	NON	ROLE	Applikationsrollen
1	R3TR	ACGR	technische Rollen
1	NON	USGR	Benutzergruppe
4	R3TR	AREA	InfoArea
1	R3TR	ODSO	DataStore-Objekt
3	R3TR	DEVC	Paket
86	R3TR	CLAS	Klasse (ABAP Objects)
5	R3TR	INTF	Interface (ABAP Objects)
42	R3TR	PROG	Programm
1	R3TR	SXCI	Business Add-Ins - Implementierungen
7	R3TR	TABL	Tabelle

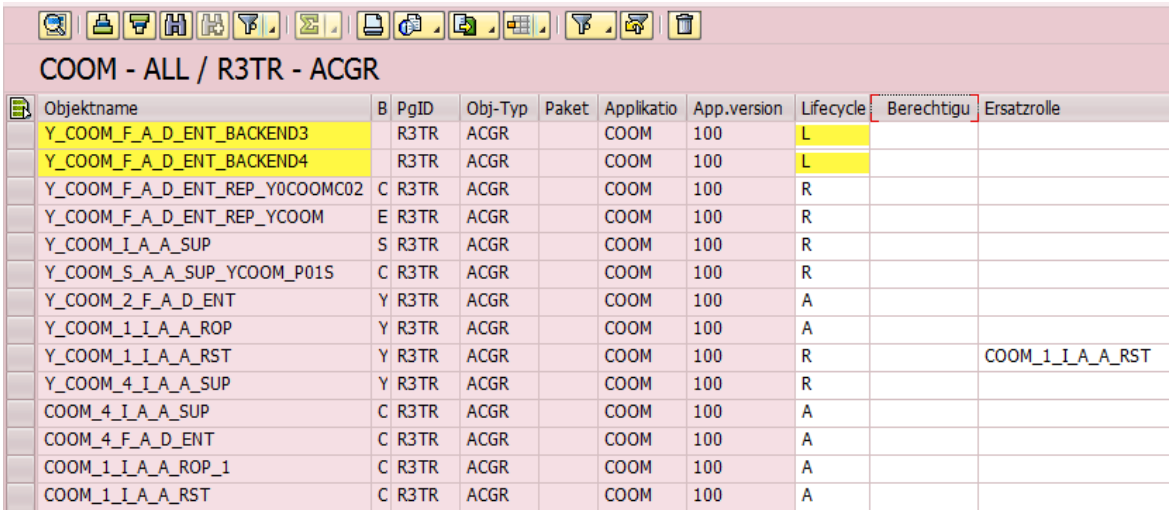
Abbildung 3: Objekte einer Applikation

Bei der Implementierung dieser Oberfläche gestaltete sich zu Beginn die Frage nach der Pflege der Objekte als eine unüberwindbare Hürde, da wir mit einer Objektzahl im fünfstelligen Bereich arbeiten mussten. Diese vermeintliche Hürde führte fast dazu, dass wir die Idee verworfen hätten. Mit Hartnäckigkeit fanden wir für dieses Problem schließlich eine einfache Lösung in Form einer Transportprüfung mit integrierter Objektpflege. Diese führte dazu, dass innerhalb weniger Monate fast alle Entwicklungsobjekte im Rahmen der täglichen Arbeit zu Applikationen zugeordnet wurden. Der Pflegeaufwand für die erstmalige Objektzuordnung liegt heute im Minutenbereich, für Objekte mit bestehender Zuordnung fällt kein Aufwand mehr an.
Der Nutzen dieser Version des AC ist der Zugriff auf die Informationen zu Applikationen über einen Einstiegsort. Einfaches Finden von dokumentiertem Wissen zu den jeweiligen Applikationen. Die Einfachheit und Klarheit dieser neuen Struktur führte im Team dazu, dass einzelne Entwickler einen Sinn darin erkannten, Wichtiges

zu einer Applikation aufzuschreiben. Sie konnten es ja jetzt sicher wiederfinden und damit im Fall einer Fehlermeldung oder Änderungsanforderung wieder nutzen.
Damit funktionierte plötzlich das Dokumentieren von alleine, was vorher all die Jahre mit vielen Vorschriften und Anweisungen nie vollständig funktioniert hatte.

4 Dritter Evolutionsschritt: Die Betriebsunterstützung

Mit der Verfügbarkeit des neuen AC veränderte sich der Betrieb bei uns nachhaltig.
Rund um die Objektlisten entstanden neue Funktionalitäten, die im täglichen Betrieb zu einer verbesserten Betriebsunterstützung führten.
So wurden z. B. alle Prozessketten attributiert, in der Form „ausführbar", „nicht ausführbar", „Regelbetrieb" oder „für Wartungsarbeiten". Damit war es für alle Teammitglieder einfacher zu erkennen, welche Prozessketten in welchem Anwendungsfall zu nutzen waren, wodurch sich die restliche textliche Dokumentation auf Spezialfälle reduzierte. Das Tool half, die Aufwände für Dokumentation und deren Aktualisierung zu reduzieren. Gleichzeitig führte es dazu, dass die Anwendungsfälle für Prozessketten standardisiert wurden.
Des Weiteren wurde für alle Objekte ein Lebenszyklusattribut eingeführt, d. h., Objekte können einzeln mit Eigenschaften wie z. B. „in Entwicklung" oder „aktiv" oder „zum Löschen vorgesehen" markiert werden. Dies gibt jedem, der das AC benutzt, Auskunft darüber, ob ein Objekt noch genutzt werden darf oder nicht. Beispielhaft seien hier als Anwendungsfall die technischen Berechtigungsrollen genannt. Über den Lebenszyklus eines Systems oder einer Organisation verändern sich gerade die Berechtigungen sehr stark. Welche Rollen noch verwendet werden sollen und welche nicht, sieht man dem technischen Objekt in der Regel nicht an.



COOM - ALL / R3TR - ACGR

Objektname	B	PgID	Obj-Typ	Paket	Applikatio	App.version	Lifecycle	Berechtigu	Ersatzrolle
Y_COOM_F_A_D_ENT_BACKEND3		R3TR	ACGR		COOM	100	L		
Y_COOM_F_A_D_ENT_BACKEND4		R3TR	ACGR		COOM	100	L		
Y_COOM_F_A_D_ENT_REP_Y0COOMC02	C	R3TR	ACGR		COOM	100	R		
Y_COOM_F_A_D_ENT_REP_YCOOM	E	R3TR	ACGR		COOM	100	R		
Y_COOM_I_A_A_SUP	S	R3TR	ACGR		COOM	100	R		
Y_COOM_S_A_A_SUP_YCOOM_P01S	C	R3TR	ACGR		COOM	100	R		
Y_COOM_2_F_A_D_ENT	Y	R3TR	ACGR		COOM	100	A		
Y_COOM_1_I_A_A_ROP	Y	R3TR	ACGR		COOM	100	A		
Y_COOM_1_I_A_A_RST	Y	R3TR	ACGR		COOM	100	R		COOM_1_I_A_A_RST
Y_COOM_4_I_A_A_SUP	Y	R3TR	ACGR		COOM	100	R		
COOM_4_I_A_A_SUP	C	R3TR	ACGR		COOM	100	A		
COOM_4_F_A_D_ENT	C	R3TR	ACGR		COOM	100	A		
COOM_1_I_A_A_ROP_1	C	R3TR	ACGR		COOM	100	A		
COOM_1_I_A_A_RST	C	R3TR	ACGR		COOM	100	A		

Abbildung 4: Objekte und deren Lebenszyklus (A= aktiv, R = In Ablösung, L = zum Löschen vorgemerkt)

Mit der Einführung des Lebenszyklusattributes lässt sich jedoch für eine Applikation eindeutig sagen, welchen Zustand die technischen Rollen haben. Damit können auch Übergangsszenarien abgebildet werden, z. B. bei der Umstellung von technischen Rollen.
Der Nutzen für die Organisation besteht darin, dass selbst neue Teammitglieder innerhalb einer angemessenen Einarbeitungszeit schon viele Betriebsaufgaben übernehmen können. Das bedeutet, dass uns das Tool bei der Wissensverteilung und -entwicklung im Team unterstützt, was zu einer Verbesserung der Ausfallsicherheit in Fällen von Krankheit und Fluktuation führt. Hierzu muss allerdings bemerkt werden, dass die Einarbeitungszeiten in einer solchen technischen Umgebung, wie sie bei uns im Unternehmen vorliegt, bis zu zwölf Monate betragen kann, bevor ein neuer Mitarbeiter in diesem Umfeld selbstständig handeln kann.

5 Vierter Evolutionsschritt: Die Entwicklungsunterstützung

Standardisierung ist eines der wichtigsten Schlüsselwörter im Bereich der Applikationsentwicklung. Die Einführung des AC markiert einen Meilenstein. Durch die gewonnene Übersicht über die Applikationen stellte sich recht schnell heraus, welche Applikationen wartungsintensiv waren und welche weniger Wartungsaufwand benötigten. Indikatoren, die im AC ersichtlich sind:

- abgebrochene Prozessketten,
- Größe des Applikationshandbuches,
- Anzahl der technischen Objekte,
- etc.

Der Nutzen des AC liegt hier in der Einfachheit, mit der solche Sachverhalte zentral und transparent dargestellt werden.

Des Weiteren kann jetzt durch die Verfügbarkeit von Applikationshandbüchern zu jeder Applikation über alle Applikationen hinweg systematisch an der Vereinheitlichung der Inhalte gearbeitet werden. Damit können wir uns jetzt mehr damit beschäftigen, *wie* zu dokumentieren ist. Insbesondere, was noch schriftlich dokumentiert und was über Programmfunktionalitäten in dem AC abgebildet wird.

Eine weitere Vereinheitlichung findet aktuell bei den implementierten Lösungen statt. Durch die Übersicht aller Applikationen und die Einführung einheitlicher Methoden, diese zu warten, zeigten und zeigen sich zunehmend Gemeinsamkeiten und Unterschiede von den verschiedenen Entwicklungen auf Detailebene. Damit erkennen wir jetzt schneller Gemeinsamkeiten und können den Fachbereichen konzeptionelle Lösungen in Form von sog. Solutions bei neuen Anforderungen zur Verfügung stellen – also vorgedachte Lösungen, in denen nur noch die betriebswirtschaftlichen Inhalte angepasst werden müssen. Hier stehen wir noch am Anfang einer sehr vielversprechenden neuen Art und Weise, Programmentwicklung zu gestalten.

Neben diesen sehr wertvollen Verbesserungen wurden allerdings auch Grenzen der ersten Version des AC deutlich. Die einfache Hierarchiedarstellung ermöglichte keine weiteren parallelen Sichten auf Applikationen. Sichten sind hier so zu verstehen, dass sie eine andere Art der Gruppierung des gleichen Sachverhaltes ermöglichen, d. h., je Begriff bzw. Denkwelt sind Sichten nur andere Einstiege zu den immer gleichen grundlegenden Elementen und Objekten. Sie sind nützlich für die leichtere Anwendbarkeit des Tools durch unterschiedliche Anwendergruppen. Solche Sichten wie z. B. „Entwicklungs-", „Betriebs-" oder „Fachbereichssichten" konnten mit der ersten Version des AC nicht dargestellt werden.

6 Fünfter Evolutionsschritt: Die Methodenentwicklung

Die größte Änderung in der Version 2 des AC ist die Implementierung von Sichten.

Zwei weitere wesentliche Veränderungen dieser Version des AC sind hier von Bedeutung für die Arbeitsweisen im Team. Die erste Veränderung zielt auf die systematische Standardisierung von Anwendungen ab, und zwar in Form von Meta-Dokumenten, in denen Best Practices dokumentiert werden. Mir die Frage zu stellen, *wie* wir entwickeln und welche Methoden wir nutzen, beschäftigt mich persönlich schon seit langem.

Besonderes Augenmerk gilt den Fragen, wie wir schneller gemeinsam im Team lernen können, und was bessere oder schlechtere Methoden der Entwicklung sind. Meta-Dokumente – also Dokumente darüber, *wie* wir Dinge tun, sind hier der erste Versuch, ein Werkzeug zum selbstreflektierten Lernen in die operative Arbeitsumgebung zu bringen.

Hier sollen sowohl Methoden, Konzepte oder auch Templates (standardisierte Vorlagen) abgelegt werden. So stehen z. B. die Meta-Dokumente, was bei der Initialisierung einer neuen Applikation zu tun ist, in der Entwicklungssicht neben den Applikationen im Meta-Fenster direkt zur Verfügung. Analoges gibt es für den Going-Live-Prozess.

Die zweite wesentliche Veränderung bezieht sich auf die Verwaltung von Funktionalitäten. So wurde die oben bereits erwähnte Toolsammlung in dieser zweiten Version vollständig in das AC integriert, was zur Folge hatte, dass sehr spezifische Ideen zur Qualitätsverbesserung im Betrieb und der Entwicklung von Applikationen jetzt sehr viel schneller umsetzbar sind.

So hatten wir bereits vorher viele gute applikationsspezifische Ideen zur Verbesserung, die alle wegen der hohen Arbeitslast aus den operativen Aufgabenstellungen nicht zur Umsetzung kamen, d. h. ständig verschoben wurden. Mit dem AC steht jetzt ein Rahmen zur Verfügung, in dem die initialen Rüstzeiten für die Programm-entwicklung von Hilfsprogrammen für einzelne Anwendungen massiv reduziert wurden. So hatte beispielsweise einer der Entwickler eine Programmidee für eine immer wieder angefragte Analyse in Crystal-Reports. Da wir die Applikationen, in denen Crystal-Report Verwendung findet, in dem AC dokumentiert haben, konnten wir unverzüglich ein Programm erstellen und dieses Programm im Kontext der Crystal-Applikationen zur Verfügung stellen.
Diese neue Funktionalität stand umgehend jedem Nutzer des AC im Kontext der Crystal Anwendung zur Verfügung, d. h. sofortiger Anwendungsnutzen für alle Teammitglieder.
Der Aufwand für Verbesserungsmaßnahmen amortisiert sich durch die damit erreichten Zeiteinsparungen innerhalb weniger Wochen. So war es auch in diesem Beispiel. Innerhalb weniger Wochen bekamen wir keine Anfragen mehr zu der Analyse in Crystal-Reports. Diese können die Anwender jetzt selbst in dem AC durchführen. Der Nutzen für das BI-Team entsteht durch die Entlastung im operativen Geschäft durch die Anwender und damit verbunden eine insgesamt bessere Qualität, weil wir uns nämlich jetzt, anstatt die oben beschriebenen Analysen durchzuführen, um dringendere Aufgaben kümmern können. Während wir also in der „Toolsammlung" allg. Hilfsprogramme erstellt hatten, sind wir jetzt in der Lage, für alle regelmäßig auftretenden Aufgabenstellungen im Kontext der einzelnen Applikationen passgenaue Lösungen zeitnah und leicht wiederauffindbar anzubieten.
Als eine weitere Neuerung sei hier noch die Erweiterung des Objekt-Konzeptes (bis dahin nur SAP-Systemobjekte), um organisatorische Objekte, z. B. Benutzergruppen oder SLAs (Service Level Agreements) zu nennen. Was ist nun der entscheidende Unterschied zur ersten Version des AC?
Mit den Sichten können jetzt alle Fragestellungen rund um die BI-Organisation abgedeckt werden. D. h., alle Themen können vollständig und transparent unter einer Oberfläche verfügbar gemacht werden, z. B. aus Sicht von ITIL = IT Infrastructure Library. Dabei ist in den meisten Fällen der Bezug zu den Systemobjekten direkt herstellbar. Hier ist wieder der Aspekt von der Verbindung des dokumentierten Wissens mit bestehenden Objekten und dessen aktiver Nutzung von zentraler Bedeutung, erweitert um den Aspekt, dieses Wissen intuitiv finden zu können.
Es können jetzt Informationen zu Entwicklungsprozessen und Vorgaben direkt in die operative Umgebung integriert werden, d. h., der Zugriff ist schnell und einfach verfügbar. Und mit der einfachen Implementierung von applikationsspezifischen Hilfsprogrammen ist eine wichtige Voraussetzung für einen kontinuierlichen Verbesserungsprozess in Gang gesetzt worden und damit ein erster Ansatz, Lernprozesse in den operativen Alltag unter einer Oberfläche zu integrieren.

7 Sechster Evolutionsschritt: Die Integration der Anwender

Wie sich im Rahmen der Entwicklung des AC zeigte, ist eine bessere Integration von technischen Lösungen und fachlichen Anforderungen der Schlüssel zur Komplexitätsreduktion in der Kommunikation mit den Fachbereichen. Dadurch wird der Übersetzungsaufwand (fachlich nach technisch und umgekehrt), welcher heute noch in BI-Projekten notwendig ist und viel Zeit in Form von Abstimmungsprozessen und Konzepten in Anspruch nimmt, deutlich reduziert.
Mit dem Sichten-Konzept im AC besteht die Möglichkeit, Fachanwendern neue Sichten, die ihrem Begriffsraum entsprechen, zur Verfügung zu stellen. Damit ist es möglich, fachliche Begriffswelten direkt mit technischen oder auch organisatorischen Objekten zu verbinden und abzugleichen. Dies ist hilfreich, um gemeinsame Begrifflichkeiten zu schaffen und die Kommunikation zu vereinheitlichen. Als Beispiel seien hier die Abbildung von Berichten im Kontext der fachlichen Aufgaben als eine Form von fachlichen Sichten genannt.
Durch die Einbeziehung aller Beteiligten der BI-Anwendungsentwicklung, also des Entwicklers sowie des fachlich Verantwortlichen, in einem gemeinsam genutzten AC, verbesserte sich die Kommunikation nachhaltig.

Gleichzeitig zeigte sich, dass das ursprünglich als Entwicklertool gebaute Instrument heute auch von den Fachbereichen sinnvoll genutzt werden kann.
Das AC zukünftig so zu nutzen, ermöglicht aktives abgestimmtes Lernen. Alle Erkenntnisse in der Arbeit können jetzt direkt abgelegt und gleichzeitig jederzeit von allen Mitgliedern der Teams genutzt werden.
Dadurch wird ein teamübergreifendes, also auch ein abteilungsübergreifendes Arbeiten ermöglicht, in dem alle Teammitglieder neues Wissen integrieren und gleichzeitig von neuem Wissen profitieren können.
Mit dem so abgelegten Wissen und den Objektverknüpfungen sind fortan neue Modelle des verteilten Arbeitens, auch über die Abteilungsgrenzen hinweg, möglich.
Voraussetzung ist die Anwendung von wenigen einfachen Vorgaben durch die Anwender. Diese Vorgaben sind integrativ in der Systemumgebung verfügbar und zugreifbar. Wie gelingt es jetzt, dass jeder in der Entwicklung dieses Wissen nutzt?

- Selbstverantwortung: Jeder Anwender übernimmt die Verantwortung für das, was er tut. Es ist eine Holpflicht.
- Definierte Abläufe: Jeder Entwickler ist verpflichtet, Objekte seiner Anwendung seiner Applikation zuzuordnen.

Der Nutzen für die Organisation besteht darin, verfügbare Ressourcen in der Unternehmensgruppe, die über unterschiedliche Standorte verteilt sind, in zentralen Entwicklungen einsetzen zu können und damit gleichzeitig Wissen und neue Ideen im Unternehmen zu streuen.
Ein noch viel größerer Nutzen ist allerdings, dass hiermit der Kampf zwischen IT-Abteilungen und Fachbereichen endlich beendet und der prioritäre Nutzen, gemeinsam an der Weiterentwicklung des Unternehmens zu arbeiten, endlich ausgeschöpft werden kann.

8 Weitere Evolutionsschritte

Die oben beschriebenen Entwicklungsschritte mit Hilfe des AC sind in einem Zeitraum von 14 Monaten während der operativen Arbeit entstanden. Die Erfahrungen und die neue Flexibilität, die wir dank des Tools erlangt haben, werden wir in nächsten Schritten gezielt zu einem neuartigen modularen Entwicklungskonzept im BI-Umfeld ausbauen, welches wir dann in zukünftigen Applikationsentwicklungen im HANA-Umfeld verwenden möchten. Ideen und erste Lösungen sind bereits verfügbar.
Eine weitere wichtige Aufgabe ist es, das AC in das Geflecht der Organisation zu integrieren. Aktuell liegen unsere Dokumentationen noch in MS Office Dokumenten vor. Wir sind aktuell dabei, diese Inhalte in eine zentrale organisationsübergreifende Wissensdatenbank einzupflegen und mit dem AC zu verlinken.
Auch die Integration in eine Anforderungs- und Aufgabenverwaltung ist bereits in vollem Gange. Schließlich möchten wir das AC auch in zentrale Prozessübersichten und -abläufe integrieren.
Wir verstehen heute Tools als Werkzeuge, die uns helfen, neue Ideen für die nächste Generation von analytischen Anwendungen umzusetzen. Sie schaffen Transparenz und Bedienbarkeit, reduzieren den Dokumentationsaufwand massiv und ermöglichen somit eine permanent aktuell dokumentierte Anwendungsentwicklung. Gleichzeitig ermöglichen diese Informationen Entwicklern und Fachbereichen, qualitativ hochwertige Entwicklungen abgestimmt durchführen zu können. Damit sind ganz neue Wege der Zusammenarbeit zwischen IT und Fachbereichen möglich, was die Einführung von Self-Service-Anwendungen in einer bisher nicht dagewesenen Art und Weise erlaubt. Sie ermöglichen zusätzlich einen integrierten Lern- und Verbesserungsprozess in der täglichen Arbeitsumgebung: Lernen im Team und im Kontext ist damit umsetzbar. Nachhaltiger geht Lernen aus meiner Sicht nicht. Und damit helfen solche Tools, Lernprozesse in Unternehmen zu initiieren und nachhaltig zu beschleunigen.
Als Technologie wurde in den ersten Versionen des AC die ABAP Umgebung im SAP BI System genutzt mit ALV – Grid Darstellungen. In den zukünftigen Entwicklungsphasen werden wir die Entwicklung auf SAP UI5 umstellen mit dem Ziel, zukünftig von Endgeräten unabhängige Anwendungsfelder für unsere Anwender zu erschließen.