(19) Deutsches Patent- und Markenamt

(10) **DE 10 2013 108 309 A1** 2015.02.05

(12) # Offenlegungsschrift

(21) Aktenzeichen: **10 2013 108 309.9**
(22) Anmeldetag: **01.08.2013**
(43) Offenlegungstag: **05.02.2015**

(51) Int Cl.: ***G06F 9/44*** (2006.01)

(71) Anmelder:
OMS Software GMBH, 81241 München, DE

(74) Vertreter:
Bettinger Schneider Schramm Patent- und Rechtsanwälte, 81679 München, DE

(72) Erfinder:
Kramer, Christian, 81243 München, DE

(56) Ermittelter Stand der Technik:

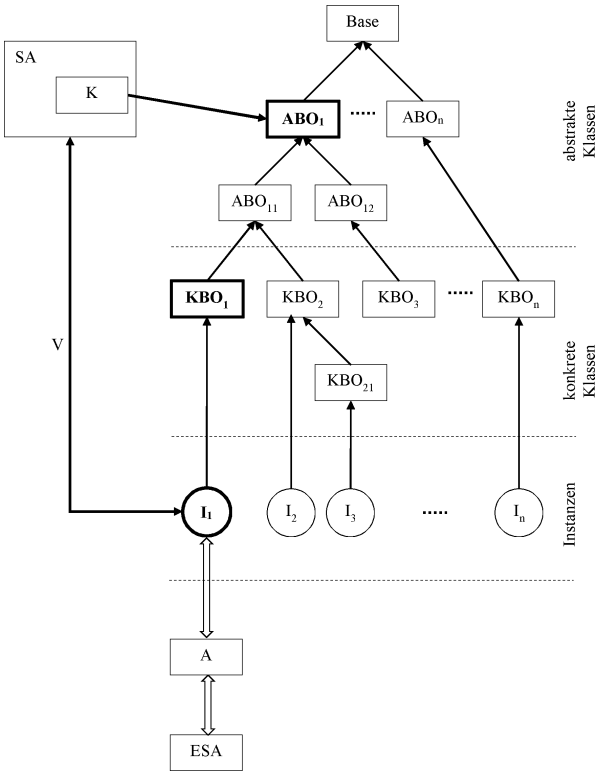
US 2003 / 0 226 136 A1
US 5 659 751 A

Prüfungsantrag gemäß § 44 PatG ist gestellt.

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

(54) Bezeichnung: **Verfahren zum Konnektieren von Objekten in einer Softwareanwendung**

(57) Zusammenfassung: Bereit gestellt wird ein Computer-implementiertes Verfahren zum Konnektieren von Businessobjekten in einer Softwareanwendung (SA), wobei
– die Softwareanwendung einen Konnektor (K) umfasst, wobei der Konnektor eine eine abstrakte Businessobjekt-Klasse (ABO_1 bis ABO_n) identifizierende Kennung umfasst,
– die Softwareanwendung zur Laufzeit die in dem Konnektor angegebene abstrakte Businessobjekt-Klasse in einer Hierarchie von abstrakten Businessobjekt-Klassen ermittelt, und
– die Softwareanwendung zur Laufzeit
– ausgehend von der ermittelten abstrakten Businessobjekt-Klasse zumindest eine davon direkt oder indirekt abgeleitete konkrete Businessobjekt-Klasse (KBO_1 bis KBO_n) ermittelt, und
– zu mindestens einer Instanz (I_1 bis I_n) der zumindest einen ermittelten konkreten Businessobjekt-Klassen eine Verbindung (V) herstellt, über die die Softwareanwendung auf Attribute und Methoden der Instanz der konkreten Businessobjekt-Klasse zugreift.
Ferner wird ein entsprechend angepasstes Computerprogrammprodukt bereit gestellt.



Beschreibung

Gebiet der Erfindung

[0001] Die Erfindung betrifft ein Verfahren zum Konnektieren von Objekten, insbesondere Businessobjekten in einer Softwareanwendung, sowie ein entsprechend angepasstes Computerprogrammprodukt.

Stand der Technik und Hintergrund der Erfindung

[0002] Im Stand der Technik sind sogenannte monolithische Softwaresysteme bekannt, bei denen die funktionalen Elemente meist untrennbar in einer einzigen Softwareanwendung zusammengefasst sind. Die Verwaltung und Bearbeitung der zugehörigen Daten wird ebenfalls durch die Softwareanwendung vorgenommen. Ein Beispiel für eine solche monolithische Softwareanwendung ist eine Leistungserfassung, bei der Leistungen erfasst und einer bestimmten Person zugeordnet werden können. Bei einer solchen Leistungserfassung werden nicht nur die einer Person zugeordneten Leistungen erfasst und gespeichert, sondern auch die Stammdaten der Personen verwaltet. Demzufolge kann die Softwareanwendung zur Leistungserfassung nur auf die in der Leistungserfassung verwalteten Personen zugreifen.

[0003] Als Alternative zu den monolithischen Softwareanwendungen sind Softwareanwendungen bekannt, die auf einer sogenannten Client-Server-Architektur beruhen oder als im Allgemeinen verteilte Softwaresysteme ausgebildet sind. Hierbei kann beispielsweise die Leistungserfassung durch zwei Softwarekomponenten realisiert werden, wobei in einer ersten Komponente Leistungen erfasst und einer Person zugeordnet werden und wobei in einer zweiten Komponente die Personen erfasst und verwaltet werden. Die Komponente zur Erfassung der Leistungen kann über eine Schnittstelle auf die Komponente zur Verwaltung der Personen zugreifen. Dadurch wird eine verbesserte Wartbarkeit der Softwarekomponenten erreicht.

[0004] Diese Vorgehensweise hat allerdings den Nachteil, dass bereits zum Entwicklungszeitpunkt der Komponente zur Leistungserfassung bekannt sein muss, auf welche konkrete Komponente zur Verwaltung der Personen zur Laufzeit zugegriffen wird bzw. zugegriffen werden soll. Ein Austausch der Komponente zur Verwaltung der Personen ist in der Regel mit einer Anpassung der Komponente zur Erfassung der Leistungen verbunden, weil beispielsweise in der Komponente zur Leistungserfassung die Schnittstellen zur Komponente der Personenverwaltung angepasst werden müssen, was zur Laufzeit nicht möglich ist und in dem Quellcode der Komponente zur Leistungserfassung vorgenommen werden muss.

[0005] Ein einfaches Austauschen der Komponente für die Personenverwaltung ohne Anpassung der Komponente zur Leistungserfassung ist daher nicht möglich, sodass die Komponente zur Leistungserfassung zur Laufzeit nur auf eine konkrete, in der Komponente zur Leistungserfassung definierte Komponente zur Personenverwaltung zugreifen kann.

[0006] Soll eine Komponente zur Leistungserfassung dennoch auf unterschiedliche Komponenten zur Personenverwaltung Zugriff haben, muss für jede Personenkomponente in der Komponente zur Leistungserfassung eine eigene dafür angepasste Schnittstelle implementiert werden. Neue verfügbare Komponenten zur Personenverwaltung können nur dann verwendet werden, wenn für diese neuen Komponenten ebenfalls ein entsprechendes Interface in der Komponente zur Leistungserfassung implementiert wird.

Aufgabe der Erfindung

[0007] Der Erfindung liegt daher die Aufgabe zugrunde, Lösungen bereitzustellen, welche die aus dem Stand der Technik bekannten Nachteile zumindest teilweise vermeiden und welche es ermöglichen, in einer Softwareanwendung auf eine Anzahl unterschiedlicher Softwarekomponenten, Businessobjekten bzw. weiterer Softwareanwendungen zugreifen zu können, ohne hierfür für jede einzelne Softwarekomponente, Businessobjekt bzw. weitere Softwareanwendung ein entsprechendes Interface bzw. Schnittstelle implementieren zu müssen, um die Flexibilität zu erhöhen und gleichzeitig den Wartungsaufwand zu reduzieren.

Erfindungsgemäße Lösung

[0008] Diese Aufgabe wird erfindungsgemäß durch ein Verfahren zum Konnektieren von Objekten bzw. Businessobjekten in einer Softwareanwendung, sowie durch ein entsprechend angepasstes Computerprogrammprodukt nach den unabhängigen Ansprüchen gelöst. Vorteilhafte Ausgestaltungen der Erfindung sind in den jeweiligen abhängigen Ansprüchen angegeben.

[0009] Bereitgestellt wird demnach ein Computerimplementiertes Verfahren zum Konnektieren von Businessobjekten in einer Softwareanwendung, wobei

– die Softwareanwendung einen Konnektor umfasst, wobei der Konnektor eine eine abstrakte Businessobjekt-Klasse identifizierende Kennung umfasst,
– die Softwareanwendung zur Laufzeit die in dem Konnektor angegebene abstrakte Businessobjekt-Klasse in einer Hierarchie von abstrakten Businessobjekt-Klassen ermittelt, und
– die Softwareanwendung zur Laufzeit

– ausgehend von der ermittelten abstrakten Businessobjekt-Klasse zumindest eine davon direkt oder indirekt abgeleitete konkrete Businessobjekt-Klasse ermittelt, und
– zu mindestens einer Instanz der zumindest einen ermittelten konkreten Businessobjekt-Klassen eine Verbindung herstellt, über die die Softwareanwendung auf Attribute und Methoden der Instanz der konkreten Businessobjekt-Klasse zugreift.

[0010] Eine abstrakte Businessobjekt-Klasse ist eine Klasse von der keine Instanzen erzeugt werden bzw. keine Instanzen erzeugt werden können. Eine konkrete Businessobjekt-Klasse ist eine Klasse von der Instanzen erzeugt werden bzw. Instanzen erzeugt werden können. Ein wesentlicher Vorteil des erfindungsgemäßen Verfahrens liegt darin, dass die Softwareanwendung auf die gesamte Hierarchie der abstrakten Businessobjekt-Klassen und der konkreten Businessobjekt-Klassen (weil die konkreten Businessobjekt-Klassen von den abstrakten Businessobjekt-Klassen direkt oder indirekt abgeleitet sind bilden die abstrakten Businessobjekt-Klassen und die konkreten Businessobjekt-Klassen zusammen ebenfalls eine Hierarchie) zugreifen kann und die Definitionen sowohl der abstrakten Businessobjekt-Klassen als auch der konkreten Businessobjekt-Klassen der Softwareanwendung bekannt sind. Daher kann die Softwareanwendung eine Verbindung zu Instanzen der konkreten Businessobjekt-Klassen herstellen bzw. auf Instanzen der konkreten Businessobjekt-Klassen zugreifen, weil der Zugriff über die in den abstrakten Businessobjekt-Klassen und/oder in den konkreten Businessobjekt-Klassen definierten und implementierten Methoden erfolgt. Ein Anpassen der Schnittstellen in der Softwareanwendung, um auf unterschiedliche Objekte zugreifen zu können, wird so vermieden.

[0011] In einer Ausgestaltung der Erfindung kann der Konnektor dem Quellcode der Softwareanwendung hinzugefügt werden.

[0012] Bei mehreren ermittelten konkreten Businessobjekt-Klassen kann die Softwareanwendung eine konkrete Businessobjekt-Klasse auswählen und zu mindestens einer Instanz der ausgewählten konkreten Businessobjekt-Klasse die Verbindung herstellen.

[0013] Die Auswahl der konkreten Businessobjekt-Klasse durch die Softwareanwendung kann in Reaktion auf eine Benutzereingabe erfolgen, bei der ein Benutzer der Softwareanwendung aus den mehreren ermittelten konkreten Businessobjekt-Klassen eine konkrete Businessobjekt-Klasse auswählt.

[0014] Vorteilhaft ist es, wenn die Softwareanwendung über die Verbindung eine eindeutige Kennung einer Instanz der konkreten Businessobjekt-Klasse entgegennimmt und speichert. Damit kann wird gewährleistet, dass die Softwareanwendung zu einem späteren Zeitpunkt wieder auf dieselbe Instanz zugreifen kann bzw. zur selben Instanz eine Verbindung herstellen kann.

[0015] Die Softwareanwendung kann zusätzlich zur eindeutigen Kennung der Instanz der konkreten Businessobjekt-Klasse eine eindeutige Kennung der konkreten Businessobjekt-Klasse entgegennehmen und/oder speichern.

[0016] Die eindeutige Kennung der konkreten Businessobjekt-Klasse kann beim Instanziieren der Instanz automatisch gesetzt werden, vorzugsweise durch einen Konstruktor der konkreten Businessobjekt-Klasse.

[0017] Vorteilhaft ist es, wenn die konkrete Businessobjekt-Klasse über einen Adapter mit zumindest einem externen Softwaresystem verbunden ist, in welchem Ausprägungen der konkreten Businessobjekt-Klasse gespeichert sind. Die Ausprägungen müssen nicht identisch zu den Instanzen der konkreten Businessobjekt-Klasse sein. Die Instanzen der konkreten Businessobjekt-Klasse können eine andere Datenstruktur aufweisen als die dazugehörigen Ausprägungen in dem externen Softwaresystem.

[0018] In einer Ausgestaltung der Erfindung ist jeder Instanz einer konkreten Businessobjekt-Klasse genau eine Ausprägung in dem in dem externen Softwaresystem zugeordnet.

[0019] Vorteilhaft ist es, wenn der Adapter Methodenaufrufe der konkreten Businessobjekt-Klasse entgegennimmt, diese in von dem externen Softwaresystem ausführbare Instruktionen umwandelt, und die ausführbaren Instruktionen an das externe Softwaresystem übergibt.

[0020] Weiter ist es vorteilhaft, wenn der Adapter von dem externen Softwaresystem Daten entgegennimmt und diese an eine oder mehrere Instanzen der konkreten Businessobjekt-Klasse übergibt. Damit kann eine Ausprägung in dem externen Softwaresystem in eine Instanz der jeweiligen konkreten Businessobjekt-Klasse überführt werden. Hierfür kann der Adapter entsprechende Abbildungsvorschriften und/oder Transformationsregeln vorsehen. Entsprechende Abbildungsvorschriften und/oder Transformationsregeln können auch vorgesehen bzw. verwendet werden, wenn eine Instanz einer konkreten Businessobjekt-Klasse in einer entsprechenden Ausprägung in dem externen Softwaresystem gespeichert werden soll.

[0021] Die Hierarchie der abstrakten und konkreten Businessobjekt-Klassen können in einer Konfigura-

tionsdatei gespeichert sein, auf die die Softwareanwendung zur Laufzeit Zugriff hat.

[0022] Die Softwareanwendung kann über Methoden der ermittelten abstrakten Businessobjekt-Klasse oder davon abgeleiteter abstrakter Businessobjekt-Klassen auf die Instanz bzw. Instanzen der konkreten Businessobjekt-Klasse zugreifen.

[0023] Die Softwareanwendung kann zur Laufzeit eine Instanz einer konkreten Businessobjekt-Klasse umfassen. Die Instanz der konkreten Businessobjekt-Klasse kann Bestandteil einer weiteren Softwareanwendung sein.

[0024] Die Softwareanwendung und die Instanz der konkreten Businessobjekt-Klasse und/oder die Softwareanwendung und die weitere Softwareanwendung werden zur Laufzeit vorzugsweise in einer gemeinsamen Laufzeitumgebung ausgeführt.

[0025] Bereitgestellt wird ferner ein Computerprogrammprodukt, das in den Speicher einer Datenverarbeitungseinrichtung geladen werden kann und auf dieser zur Ausführung gebracht werden kann, und das Programmabschnitte umfasst, die angepasst sind, ein erfindungsgemäßes Verfahren zur Ausführung zu bringen.

Kurzbeschreibung der Figuren

[0026] Weitere Einzelheiten und Merkmale der Erfindung sowie konkrete, insbesondere vorteilhafte Ausführungsbeispiele der Erfindung ergeben sich aus der nachfolgenden Beschreibung in Verbindung mit der Zeichnung. Es zeigt:

[0027] Fig. 1 eine erfindungsgemäße Softwarearchitektur zur Verdeutlichung des erfindungsgemäßen Verfahrens zum Konnektieren von Objekten bzw. Businessobjekten; und

[0028] Fig. 2 ein konkretes Beispiel anhand dessen das erfindungsgemäße Verfahren beschrieben wird.

Detaillierte Beschreibung der Erfindung

[0029] Fig. 1 zeigt eine Softwarearchitektur, welche für ein Verfahren zum Konnektieren von Objekten bzw. Businessobjekten in einer Softwareanwendung angepasst ist.

[0030] Eine Softwareanwendung SA soll zur Laufzeit auf ein bestimmtes Businessobjekt, welches außerhalb der Softwareanwendung SA implementiert ist, Zugriff haben. Die Softwareanwendung SA wird auch als sogenannte Mikro-APP bezeichnet. Das Businessobjekt, auf welches die Softwareanwendung SA zur Laufzeit Zugriff haben soll, kann ebenfalls von einer Softwareanwendung bzw. Mikro-APP zur Verfügung gestellt werden bzw. in einer Softwareanwendung implementiert sein. Das Businessobjekt, auf welches die Softwareanwendung SA zur Laufzeit zugreifen soll, gehört zu einer bestimmten Klasse von Businessobjekten, beispielsweise Personen oder Dokumente.

[0031] Erfindungsgemäß muss nicht bereits zum Entwicklungszeitpunkt der Softwareanwendung SA feststehen, auf welches konkrete Businessobjekt die Softwareanwendung zur Laufzeit Zugriff haben muss. Vielmehr ist es erfindungsgemäß ausreichend, wenn zum Entwicklungszeitpunkt feststeht, auf welche Klasse von Businessobjekten die Softwareanwendung zur Laufzeit Zugriff haben soll. Dadurch wird vermieden, dass für verschiedene Businessobjekte in der Softwareanwendung SA jeweils entsprechende Schnittstellen implementiert werden müssen. Das konkrete Businessobjekt wird dann zur Laufzeit der Softwareanwendung, vorzugsweise von der Softwareanwendung, ermittelt bzw. ausgewählt.

[0032] Um dies zu bewerkstelligen, ist erfindungsgemäß eine Hierarchie von Businessobjekten bzw. eine Anordnung von Businessobjekten in einer hierarchischen Baumstruktur vorgesehen.

[0033] Die Baumstruktur umfasst eine abstrakte Basis-Businessobjekt-Klasse Base, von der eine oder mehrere abstrakte Businessobjekt-Klassen abgeleitet sind. Von diesen abgeleiteten abstrakten Businessobjekt-Klassen ABO_1 bis ABO_n können jeweils wiederum eine Anzahl von abstrakten Businessobjekt-Klassen ABO_{11} bis ABO_{n1} abgeleitet sein. Abgeleitet bedeutet, dass eine abgeleitete abstrakte Businessobjekt-Klasse Attribute und Methoden der Vaterklasse erbt. Die Tiefe der Baumstruktur, in der die abstrakten Businessobjekt-Klassen organisiert sind, kann beliebig sein.

[0034] Von den abstrakten Businessobjekt-Klassen ABO können jeweils eine Anzahl von konkreten Businessobjekt-Klassen KBO abgeleitet werden. Von jeder konkreten Businessobjekt-Klasse KBO können wiederum eine Anzahl von konkreten Businessobjekt-Klassen KBO abgeleitet werden, wobei eine konkrete Businessklasse die Methoden und Attribute der übergeordneten abstrakten Businessobjekt-Klasse bzw. der übergeordneten konkreten Businessobjekt-Klasse erbt.

[0035] Jeder konkreten Businessobjekt-Klasse KBO können eine oder mehrere Instanzen zugeordnet sein. Beispielsweise kann der konkreten Businessobjekt-Klasse KBO_1 eine Instanz I_1 zugeordnet sein, wobei die konkrete Businessobjekt-Klasse KBO_1 von der abstrakten Businessobjekt-Klasse ABO_{11} abgeleitet ist, die wiederum von der abstrakten Businessobjekt-Klasse ABO_1 abgeleitet ist. Mit Hilfe der hierarchischen Anordnung der abstrakten Businessobjekt-

Klassen in einer Baumstruktur werden die darunterliegenden konkreten Businessobjekt-Klassen strukturiert, wobei die abstrakten Businessobjekt-Klassen ABO Methoden implementieren können, die von der direkt bzw. indirekt abgeleiteten konkreten Businessobjekt-Klasse KBO verwendet werden oder überschrieben werden können.

[0036] Bei dem in **Fig.** 1 gezeigten Beispiel soll die Softwareanwendung SA zur Laufzeit Zugriff auf eine bestimmte Klasse von Businessobjekten haben, wobei im Entwicklungszeitpunkt die konkrete Businessobjektklasse nicht bekannt sein muss.

[0037] Hierzu ist es vorgesehen, in der Softwareanwendung SA einen Konnektor K vorzusehen, der eine abstrakte Businessobjekt-Klasse identifizierende Kennung umfasst. In dem in **Fig.** 1 gezeigten Beispiel identifiziert die Kennung des Konnektors K die abstrakte Businessobjekt-Klasse ABO_1.

[0038] Zur Laufzeit ermittelt die Softwareanwendung SA die in dem Konnektor K angegebene abstrakte Businessobjekt-Klasse ABO_1 in der die abstrakten Businessobjekt-Klassen umfassenden Baumstruktur. Ausgehend von der ermittelten abstrakten Businessobjekt-Klasse ABO_1 ermittelt die Softwareanwendung SA zur Laufzeit zumindest eine davon direkt oder indirekt abgeleitete konkrete Businessobjekt-Klasse. In dem Beispiel gemäß **Fig.** 1 ermittelt die Softwareanwendung SA zur Laufzeit die konkreten Businessobjekt-Klassen KBO_1, KBO_2, KBO_3 und KBO_{21}, weil diese vier konkreten Businessobjekt-Klassen direkt oder indirekt von der abstrakten Businessobjekt-Klasse KBO_1 abgeleitet sind.

[0039] Nachdem die konkreten Businessobjekt-Klassen ermittelt worden sind, kann die Softwareanwendung SA eine Verbindung zu zumindest einer konkreten Businessobjekt-Klasse bzw. zu zumindest einer Instanz der ermittelten konkreten Businessobjekt-Klassen herstellen.

[0040] Wurden mehrere konkrete Businessobjekt-Klassen ermittelt, die direkt oder indirekt von der abstrakten Businessobjekt-Klasse abgeleitet sind, kann die Softwareanwendung SA aus den ermittelten konkreten Businessobjekt-Klassen eine konkrete Businessobjekt-Klasse auswählen. In dem in **Fig.** 1 gezeigten Beispiel hat die Softwareanwendung SA die konkrete Businessobjekt-Klasse KBO_1 ausgewählt, sodass die Softwareanwendung SA zur Laufzeit eine Verbindung V zu einer Instanz I_1 der konkrete Businessobjekt-Klasse KBO_1 herstellen kann. Die Auswahl einer konkreten Businessobjekt-Klasse aus mehreren konkreten Businessobjekt-Klassen kann die Softwareanwendung SA gemäß vorbestimmter Auswahlkriterien selbst vornehmen.

[0041] In einer Ausgestaltung der Erfindung kann die Auswahl einer konkreten Businessobjekt-Klasse in Reaktion auf eine Benutzereingabe erfolgen, bei der ein Benutzer der Softwareanwendung SA zur Laufzeit aus den mehreren ermittelten konkreten Businessobjekt-Klassen eine konkrete Businessobjekt-Klasse auswählt. Hierfür kann die Softwareanwendung SA dem Benutzer eine Auswahlliste mit den ermittelten konkreten Businessobjekt-Klassen zur Auswahl einer bestimmten konkreten Businessobjekt-Klasse zur Verfügung stellen.

[0042] Erfindungsgemäß kann die Softwareanwendung SA nicht nur eine Verbindung V zu Instanzen einer einzigen ausgewählten konkreten Businessobjekt-Klasse herstellen, sondern auch Verbindungen zu Instanzen unterschiedlicher konkreter Businessobjekt-Klassen. In diesem Fall muss keine Auswahl einer speziellen konkreten Businessobjekt-Klasse durch die Softwareanwendung SA bzw. durch einen Benutzer erfolgen. Damit wird es beispielsweise möglich, dass die Softwareanwendung SA eine Liste von Instanzen unterschiedlicher konkreter Businessobjekt-Klassen dem Benutzer zur Auswahl einer Instanz bereitstellt, ohne dass hierfür für jede unterschiedliche konkrete Businessobjekt-Klasse eine eigene Schnittstelle in der Softwareanwendung implementiert werden muss.

[0043] In dem in **Fig.** 1 gezeigten Beispiel kann beispielsweise die abstrakte Businessobjekt-Klasse ABO_{11} eine Methode implementieren, mit der die Instanzen der von der abstrakten Businessobjekt-Klasse ABO_{11} direkt oder indirekt abgeleiteten konkreten Businessobjekt-Klassen KBO_1, KBO_2 und KBO_{21} ermittelt und in der Softwareanwendung SA zur Anzeige gebracht werden. Eine solche Methode kann aber auch von den konkreten Businessobjekt-Klassen implementiert bzw. überschrieben werden, sodass beispielsweise die konkrete Businessobjekt-Klasse KBO_1 eine Liste aller Instanzen der konkreten Businessobjekt-Klasse KBO_1 ermittelt und beispielsweise die konkrete Businessobjekt-Klasse KBO_2 eine Liste aller Instanzen der konkreten Businessobjekt-Klasse KBO_2 und der konkreten Businessobjekt-Klasse KBO_{21} ermittelt.

[0044] Auf diese Weise muss in der Softwareanwendung SA nur mehr ein Konnektor K definiert werden, der auf eine abstrakte Businessobjekt-Klasse referenziert, um eine Verbindung zu Instanzen der von dieser abstrakten Businessobjekt-Klasse abgeleiteten konkreten Businessobjekt-Klassen herstellen zu können bzw. auf diese Instanzen zugreifen zu können.

[0045] Weil in der Ableitungshierarchie der abstrakten Businessobjekt-Klassen bzw. konkreten Businessobjekt-Klassen die entsprechenden Methoden zum Zugriff auf die jeweiligen Instanzen implemen-

tiert sind, müssen für den Zugriff auf Instanzen unterschiedlicher Klassen keine speziellen Schnittstellen bzw. Interfaces in der Softwareanwendung SA implementiert werden. Zudem kann die Softwareanwendung bzw. ein Benutzer der Softwareanwendung zur Laufzeit entscheiden, auf welche Instanzen welcher konkrete Businessobjekt-Klasse bzw. konkreten Businessobjekt-Klassen die Softwareanwendung zugreifen soll bzw. Zugriff haben soll.

[0046] Je weiter unten sich die abstrakte Businessobjekt-Klasse in der Hierarchie der Baumstruktur befindet, umso spezieller wird der Zugriff auf die entsprechenden Instanzen. D.h., wenn der Konnektor K eine Referenz zur abstrakten Businessobjekt-Klasse ABO_{11} aufweist, kann die Softwareanwendung auf Instanzen der konkreten Businessobjekt-Klassen KBO_1, KBO_2 und KBO_{21} zugreifen, während bei einer Referenz auf die abstrakte Businessobjekt-Klasse ABO_1 zusätzlich auf die Instanzen der konkreten Businessobjekt-Klasse KBO_3 zugegriffen werden kann.

[0047] Ferner können nachträglich zusätzliche konkrete Businessobjekt-Klassen der Hierarchie hinzugefügt werden, wobei die Softwareanwendung SA ohne weitere Anpassung auch auf die Instanzen dieser zusätzlichen konkreten Businessobjekt-Klassen zugreifen kann. Hierzu ist es lediglich notwendig, dass die zusätzlichen konkreten Businessobjekt-Klassen jeweils von einer abstrakten Businessobjekt-Klasse abgeleitet werden. Beispielsweise könnte von der abstrakten Businessobjekt-Klasse ABO_{12} eine zusätzliche konkrete Businessobjekt-Klasse KBO_4 abgeleitet werden, sodass bei dem in **Fig.** 1 gezeigten Beispiel die Softwareanwendung SA über die abstrakte Businessobjekt-Klasse ABO_1 auch Zugriff auf die Instanzen der neuen konkreten Businessobjekt-Klasse KBO_4 hat.

[0048] Über die Verbindung V erhält die Softwareanwendung SA beispielsweise Zugriff auf eine eindeutige Kennung einer ausgewählten Instanz, die in der Softwareanwendung vorzugsweise in Kombination mit einer Klassenkennung abgespeichert werden kann.

[0049] In einer Ausgestaltung der Erfindung ist eine Instanz einer konkreten Businessobjekt-Klasse über einen Adapter A mit einem externen Softwaresystem ES verbunden. In dem externen Softwaresystem ES werden die realen Ausprägungen einer konkreten Businessobjekt-Klasse gespeichert und verwaltet. Das externe Softwaresystem ES kann physikalisch getrennt von dem Softwaresystem sein, auf dem die Softwareanwendung SA zur Ausführung gebracht wird. Das Softwaresystem, auf dem die Softwareanwendung zur Ausführung gebracht wird, kann über eine Kommunikationsschnittstelle mit dem externen Softwaresystem gekoppelt sein.

[0050] Der Adapter A ist angepasst, Methodenaufrufe der jeweiligen konkreten Businessobjekt-Klasse bzw. Instanz entgegenzunehmen und die entgegengenommenen Methodenaufrufe an das externe Softwaresystem ES zu übergeben bzw. die entgegengenommenen Methodenaufrufe in von dem externen Softwaresystem ausführbare Instruktionen umzuwandeln. Beispielsweise kann die abstrakte Businessobjekt-Klasse ABO_1 eine Methode „getList()“ definieren bzw. implementieren, mit der eine bestimmte Anzahl von Instanzen der konkrete Businessobjekt-Klasse KBO_1 ermittelt werden sollen. Die Softwareanwendung SA kann diese Methode der Instanz I_1 aufrufen, wobei der Adapter A diese Methode in eine entsprechende Methode des externen Softwaresystems ES umwandelt und zur Ausführung bringt. Als Ergebnis der Ausführung übermittelt das externe Softwaresystem ES dem Adapter A eine Anzahl von Datenobjekten, die von dem Adapter A in eine Liste von Instanzen I_1 transformiert wird bzw. in eine Liste transformiert wird, die von der Instanz I_1 referenziert wird. Es müssen also lediglich Adapter zur Verfügung gestellt werden, über die die Instanzen der konkreten Businessobjekte auf externe Softwaresysteme zugreifen können. Die Softwareanwendung SA muss hierfür nicht angepasst werden.

[0051] **Fig.** 2 zeigt ein konkretes Beispiel einer Softwareanwendung, die über einen Konnektor Zugriff auf Instanzen konkreter Businessobjekt-Klassen gemäß dem vorstehend beschriebenen Verfahren erhält.

[0052] Bei dem in **Fig.** 2 beschriebenen Beispiel handelt es sich bei der Softwareanwendung SA um eine sogenannte Leistungserfassung, bei der Leistungen erfasst und einer bestimmten Person zugeordnet werden. Das Erfassen der Leistungen selbst ist hier in der Softwareanwendung SA implementiert.

[0053] Um die erfassten Leistungen einer konkreten Person zuordnen zu können, muss die Softwareanwendung beispielsweise eine Auswahlliste anbieten können, aus der ein Benutzer eine bestimmte Person, der die Leistung zugeordnet werden soll, auswählen kann. Um Personen auswählen zu können, die aus verschiedenen Anwendungen stammen, wird zunächst eine abstrakte Businessobjekt-Klasse „Personen“ definiert, von der zwei weitere abstrakte Businessobjekt-Klassen „Personen (SAP)“ und „Personen (Facebook)“ abgeleitet sind. In der Softwareanwendung SA wird dann ein Konnektor K definiert, der die Kennung der abstrakten Businessobjekt-Klasse „Personen“ umfasst bzw. der die abstrakte Businessobjekt-Klasse „Personen“ referenziert.

[0054] Dadurch, dass der Konnektor die abstrakte Businessobjekt-Klasse „Personen“ referenziert, kann die Softwareanwendung SA eine Liste aller Personen zur Auswahl anbieten, die Instanzen der konkreten

Businessobjekt-Klassen repräsentieren, die von den abstrakten Businessobjekt-Klassen „Personen (SAP) “ und „Personen (Facebook)“ direkt oder indirekt abgeleitet sind. In dem in **Fig.** 2 gezeigten Beispiel sind zur besseren Übersicht lediglich die Instanzen dieser konkreten Businessobjekt-Klassen gezeigt. Die jeweiligen konkreten Businessobjekt-Klassen sind hingegen nicht gezeigt. Bei den Instanzen P_{S1} bis P_{Sn} handelt es sich um Personenobjekte aus einem SAP-System. Bei den Instanzen P_{F1} bis P_{Fn} handelt es sich um Personen aus der Social-Network-Plattform Facebook. Über in **Fig.** 2 nicht gezeigte Adapter können die jeweiligen Instanzen auf das SAP-System bzw. auf Facebook zugreifen.

[0055] Zur Laufzeit der Softwareanwendung SA ermittelt diese die von der abstrakten Businessobjekt-Klasse „Personen“ abgeleiteten abstrakten Businessobjekt-Klassen, sowie die von diesen abgeleiteten konkreten Businessobjekt-Klassen. Die Softwareanwendung SA kann dann eine Verbindung zu den konkreten Businessobjekt-Klassen bzw. zu den Instanzen der konkreten Businessobjekt-Klassen herstellen und auf die Instanzen der konkreten Businessobjekt-Klassen zugreifen, ohne hierfür eine spezielle Schnittstelle bzw. Interface implementieren zu müssen. Die für den Zugriff notwendigen Methoden sind in den abstrakten Businessobjekt-Klassen definiert bzw. implementiert

[0056] Die Definition der Baumstruktur der abstrakten Businessobjekt-Klassen und der konkreten Businessobjekt-Klassen kann die Softwareanwendung SA aus einer Konfigurationsdatei KD beziehen, auf die die Leistungserfassung zur Laufzeit Zugriff hat.

[0057] In einer Ausgestaltung der Erfindung werden die Leistungserfassung und die konkreten Businessobjekt-Klassen bzw. die Instanzen der konkreten Businessobjekt-Klassen in einer gemeinsamen Laufzeitumgebung aufgeführt, um zu gewährleisten, dass eine Verbindung zu den Instanzen hergestellt werden kann. Die gemeinsame Laufzeitumgebung ist in **Fig.** 2 durch den Bereich „Solution“ gekennzeichnet.

[0058] In einer Ausgestaltung der Erfindung kann auf einen Adapter A auch verzichtet werden, beispielsweise wenn die Instanzen der konkreten Businessobjekt-Klasse „echte“ Instanzen sind, auf die die Leistungserfassung zugreifen kann.

Bezugszeichen:

A	Adapter
ABO_1 bis ABO_n	abstrakte Businessobjekt-Klassen
Base	abstrakte Basis-Businessobjekt-Klassen
ES	externe Softwareanwendung
I_1 bis I_n	Instanzen einer oder mehrerer konkreten Businessobjekt-Klassen
K	Konnektor
KD	Konfigurationsdatei
KBO_1 bis KBO_n	konkrete Businessobjekt-Klassen
SA	Softwareanwendung, z.B. eine sogenannte Mikro-APP
V	Verbindung

Patentansprüche

1. Computer-implementiertes Verfahren zum Konnektieren von Businessobjekten in einer Softwareanwendung (SA), wobei
– die Softwareanwendung einen Konnektor (K) umfasst, wobei der Konnektor eine eine abstrakte Businessobjekt-Klasse (ABO_1 bis ABO_n) identifizierende Kennung umfasst,
– die Softwareanwendung zur Laufzeit die in dem Konnektor angegebene abstrakte Businessobjekt-Klasse in einer Hierarchie von abstrakten Businessobjekt-Klassen ermittelt, und
– die Softwareanwendung zur Laufzeit
– ausgehend von der ermittelten abstrakten Businessobjekt-Klasse zumindest eine davon direkt oder indirekt abgeleitete konkrete Businessobjekt-Klasse (KBO_1 bis KBO_n) ermittelt, und
– zu mindestens einer Instanz (I_1 bis I_n) der zumindest einen ermittelten konkreten Businessobjekt-Klassen eine Verbindung (V) herstellt, über die die Softwareanwendung auf Attribute und Methoden der Instanz der konkreten Businessobjekt-Klasse zugreift.

2. Verfahren nach Anspruch 1, wobei der Konnektor (K) dem Quellcode der Softwareanwendung hinzugefügt wird.

3. Verfahren nach einem der vorhergehenden Ansprüche, wobei bei mehreren ermittelten konkreten Businessobjekt-Klassen (KBO_1 bis KBO_n) die Softwareanwendung (SA) eine konkrete Businessobjekt-Klasse auswählt und zu mindestens einer Instanz (I_1 bis I_n) der ausgewählten konkreten Businessobjekt-Klasse die Verbindung (V) herstellt.

4. Verfahren nach Anspruch 3, wobei die Auswahl der konkreten Businessobjekt-Klasse durch die Softwareanwendung (SA) in Reaktion auf eine Benutzereingabe erfolgt, bei der ein Benutzer der Softwareanwendung aus den mehreren ermittelten konkreten Businessobjekt-Klassen (KBO_1 bis KBO_n) eine konkrete Businessobjekt-Klasse auswählt.

5. Verfahren nach einem der vorhergehenden Ansprüche, wobei die Softwareanwendung (SA) über die Verbindung (V) eine eindeutige Kennung einer

Instanz der konkreten Businessobjekt-Klasse (KBO_1 bis KBO_n) entgegennimmt und speichert.

6. Verfahren nach Anspruch 5, wobei die Softwareanwendung (SA) zusätzlich zur eindeutigen Kennung der Instanz (I_1 bis I_n) der konkreten Businessobjekt-Klasse (KBO_1 bis KBO_n) eine eindeutige Kennung der konkreten Businessobjekt-Klasse speichert.

7. Verfahren nach einem der vorhergehenden Ansprüche, wobei die konkrete Businessobjekt-Klasse (KBO_1 bis KBO_n) über einen Adapter (A) mit zumindest einem externen Softwaresystem (ES) verbunden ist, in welchem Ausprägungen der konkreten Businessobjekt-Klasse gespeichert sind.

8. Verfahren nach Anspruch 7, wobei der Adapter (A) Methodenaufrufe der konkreten Businessobjekt-Klasse (KBO_1 bis KBO_n) entgegennimmt, diese in von dem externen Softwaresystem (ES) ausführbare Instruktionen umwandelt, und die ausführbaren Instruktionen an das externe Softwaresystem übergibt.

9. Verfahren nach Anspruch 7 oder 8, wobei der Adapter von dem externen Softwaresystem (ES) Daten entgegennimmt und diese an eine oder mehrere Instanzen der konkreten Businessobjekt-Klasse übergibt.

10. Verfahren nach einem der vorhergehenden Ansprüche, wobei die Hierarchie der abstrakten und konkreten Businessobjekt-Klassen in einer Konfigurationsdatei (KD) gespeichert ist, auf die die Softwareanwendung (SA) zur Laufzeit Zugriff hat.

11. Verfahren nach einem der vorhergehenden Ansprüche, wobei die Softwareanwendung (SA) über Methoden der ermittelten abstrakten Businessobjekt-Klasse oder davon abgeleiteter abstrakter Businessobjekt-Klassen (ABO_1 bis ABO_n) auf die Instanz (I_1 bis I_n) der konkreten Businessobjekt-Klasse (KBO_1 bis KBO_n) zugreift.

12. Verfahren nach einem der vorhergehenden Ansprüche, wobei die Softwareanwendung (SA) zur Laufzeit eine Instanz einer konkreten Businessobjekt-Klasse umfasst und/oder wobei die Instanz der konkreten Businessobjekt-Klasse Bestandteil einer weiteren Softwareanwendung ist.

13. Verfahren nach einem der vorhergehenden Ansprüche, wobei die Softwareanwendung (SA) und die Instanz der konkreten Businessobjekt-Klasse und/oder die Softwareanwendung (SA) und die weitere Softwareanwendung zur Laufzeit in einer gemeinsamen Laufzeitumgebung ausgeführt werden.

14. Computerprogrammprodukt, das in den Speicher einer Datenverarbeitungseinrichtung geladen werden kann und auf dieser zur Ausführung gebracht werden kann, und das Programmabschnitte umfasst, die angepasst sind, ein Verfahren nach einem der vorhergehenden Ansprüche zur Ausführung zu bringen.

Es folgen 2 Seiten Zeichnungen

Anhängende Zeichnungen

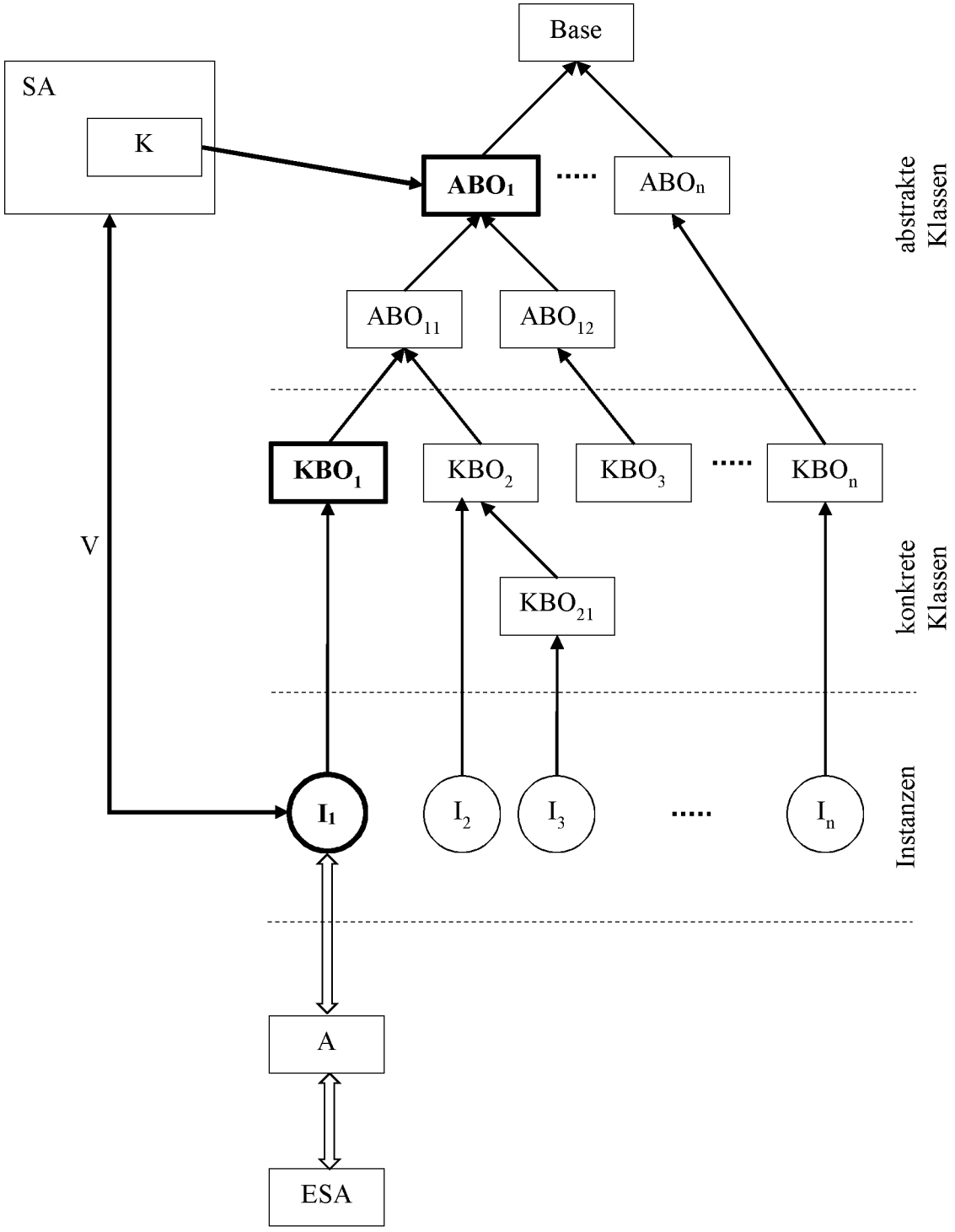


Fig. 1

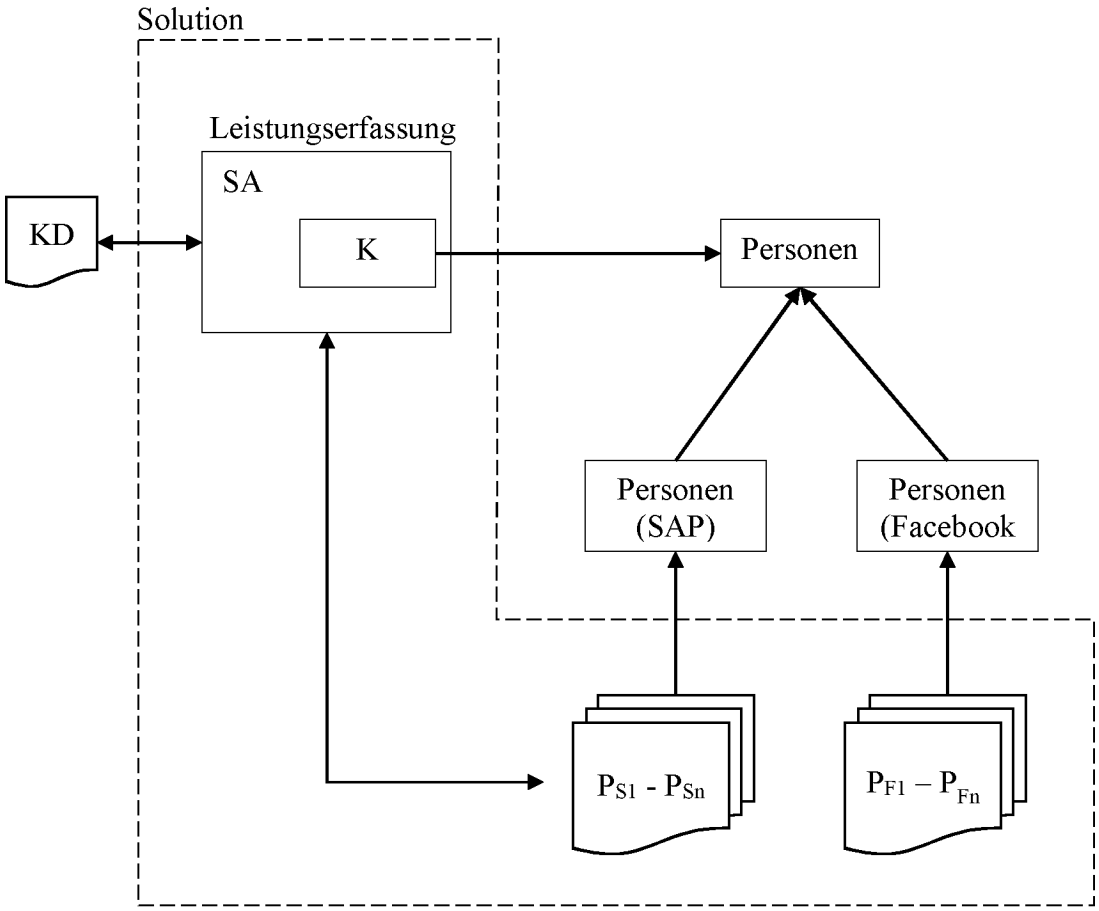

Solution
Leistungserfassung
SA
K
KD
Personen
Personen (SAP)
Personen (Facebook
P_{S1} - P_{Sn}
P_{F1} – P_{Fn}

Fig. 2