Pliable Objects zur Konzeption medizinischer Anästhesie–Informationssysteme

Markus Preißner

Arbeitsgruppe Datenbanken

Otto-von-Guericke Universität Magdeburg

Austr.12
73230 Kirchheim / Teck
Markus.Preissner@gmx.de

Abstract: Im medizinischen Fachbereich der Anästhesie ist die Informationsverarbeitung an der Erkrankung des Patienten orientiert. Die zu erfassenden Informationen orientieren sich streng an der Erkrankung und dem Grad der Erkrankung des Patienten. Aufgabe eines Anästhesie-Informationssystems ist die Verarbeitung der Gesamtheit aller Informationen, die generell in der Anästhesie zu dokumentieren sind. Moderne Informationssysteme sind für die Verarbeitung großer Datenmengen ausgelegt, scheitern jedoch in der Anästhesie, wenn neue Informationen zusätzlich zu verarbeiten oder sie an geänderte Arbeitsabläufe anzupassen sind. Im vorliegenden Beitrag wird das Modell der Pliable[1] Objects vorgestellt, die einen generischen Ansatz zur Lösung dieser Probleme anbieten.

1 Anästhesie

In den Zuständigkeitsbereich von Anästhesisten bzw. Anästhesistinnen gehören die Regional- und Allgemeinanästhesien. Die Regionalanästhesie betäubt einen größeren Körperabschnitt, und die Allgemeinanästhesie („Vollnarkose“) versetzt den Patienten in einen schlafähnlichen entspannten Zustand, in dem der gesamte Körper empfindungs- und schmerzfrei gemacht wird. Die Durchführungen von Anästhesien erfolgen in den meisten Fällen durch spezialisierte Anästhesiefachärztinnen und -fachärzte zusammen mit Anästhesiepflegefachkräften. Als Anästhesie-Team überwachen sie alle zusammen den Patienten und gewährleisten seine Sicherheit.

[1] Pliable: engl. biegsam, nachgiebig, formbar

erschienen im Tagungsband der INFORMATIK 2011
Lecture Notes in Informatics, Band P192
ISBN 978-3-88579-286-4
weitere Artikel online:
http://informatik2011.de/519.html

Anästhesien schalten nicht nur Schmerzen aus, sondern können auch andere Auswirkungen auf verschiedene wichtige Organsysteme haben, so dass anästhesierte Patienten engmaschig überwacht werden müssen. Dadurch können bei operativen Eingriffen Auswirkungen von Anästhesien auf lebenswichtige Organfunktionen vermieden oder sofort behandelt werden. Technische und organisatorische Sicherheitsstandards wurden eingeführt, um Risiken bei den Anästhesien zu minimieren.

Zur Einhaltung der Sicherheitsstandards wurde die Dokumentationspflicht für die Anästhesie eingeführt. Die Dokumentation dient der Erfüllung der Nachweispflicht, der Erfassung von Leistungen, der Sicherung der Qualität, der Informationsweitergabe und der Transparenz der Verantwortung, die gegenüber dem Patienten besteht, der seine Interessen und Grundbedürfnisse in narkotisiertem Zustand nicht selbst wahrnehmen bzw. befriedigen kann.

Die Deutsche Gesellschaft für Anästhesiologie und Intensivmedizin (DGAI) hat für die Qualitätssicherung in der Anästhesie Richtlinien verabschiedet (vgl. [OW99]). Diese geben Empfehlungen ab, welche Daten im Rahmen einer Anästhesie zu erfassen bzw. zu protokollieren sind. Diese Protokollierung wird im Rahmen des Anästhesiologischen Arbeitsablaufs vorgenommen, der sich wiederum in eine präoperative, eine intraoperative und eine postoperative Phase unterteilt.

In der präoperativen Phase werden der Zustand des Patienten und präoperative Maßnahmen dokumentiert. Die interoperative Phase beginnt mit der Einleitung der Anästhesie. In dieser Phase überwacht und dokumentiert der Anästhesist permanent den Zustand des Patienten. Die postoperative Phase beginnt mit der Verlegung des Patienten in den Aufwachraum. Hier wird der Patient aus seinem narkotischen Schlaf geholt und Komplikationen medikamentiv kompensiert. Abschließend für die Anästhesie folgt gegebenenfalls noch die postanästhesiologische Visite auf der Station, auf welcher der Patient untergebracht ist.

In allen Phasen der Anästhesie ist die Protokollierung verpflichtend vorgeschrieben, um dadurch die Transparenz und Nachvollziehbarkeit ärztlicher Entscheidungen zu gewährleisten. Derzeitige Realisierung sind klinikspezifische Papierprotokolle, die vom Anästhesisten auszufüllen sind.

2 Probleme anästhesiespezifischer Informationssysteme

Die manuelle Papierdokumentation soll durch Anästhesie-Informationssysteme bzw. Anästhesie-Informations-Management-Systeme (AIMS) abgelöst werden. Ziel solcher Systeme ist die Überführung des papierbasierten Anästhesieprotokolls in ein einheitliches, lesbares und vollständiges elektronisches Dokument. Auf Basis dieses elektronischen Dokumentes müssen Datenaggregationen für betriebswirtschaftliche, wissenschaftliche und qualitätssichernde Zwecke möglich sein. Die Erfassung des elektronischen Dokumentes orientiert sich dabei vollständig an dem anästhesiologischen Arbeitsablauf (vgl. [B07]).

Anästhesie–Informationssysteme bieten umfangreiche Funktionen, um die kontinuierliche Dokumentation von Anästhesien mit hoher Qualität zu gewährleisten. Aufgrund der Verfügbarkeit in einer Datenbank sind durch die Vergleichbarkeit der Daten statistische Auswertungen möglich. Eine zusätzliche Validierung von Daten ergibt sich durch die Verfügbarkeit zusätzlicher Stammdaten, wie Informationen zu Personal, Operationssälen, Dienstplänen, etc.

Neben den Vorteilen offenbaren sich im Einsatz von Anästhesie-Informationssystemen auch Nachteile. Die Einsatzumgebung der Anästhesie-Informationssysteme erfährt Änderungen, wie durch neu bzw. zusätzlich zu erfassende Informationen oder Änderungen im Arbeitsablauf; beispielsweise könnten sich Angaben zu Eigenschaften ändern, wie die Erweiterung der Auswahl *normal, somnolent, komatös, sediert* bei der Zustandsbeurteilung des Bewusstseins eines Patienten um die Option *verwirrt* - oder eine Ergänzung der Patienten-Zustandsbeurteilung um eine neue Eigenschaft, wie die Erfassung des *psychischen Zustands* mit den Optionen *unauffällig, depressiv, euphorisch* und ähnlichem. Die Erfassung neuer Eigenschaften führt zur Beeinflussung des anästhesiologischen Arbeitsablaufes, da entschieden werden muss, wann und wo die neue Eigenschaft erfasst wird.

Aufgrund derartiger Ergänzungen ist ein stetiger, fachspezifischer Anpassungsbedarf der Software notwendig, die das jeweilige Informationssystem realisiert. Dies betrifft das Datenbankschema der dem Informationssystem zugrundeliegenden Datenbank, die interne Programmlogik der Software sowie die Anpassung der Bedienoberfläche.

Neben fachspezifischen können auch produktspezifische Änderungen auftreten, wenn sich z.B. Hardware- und/oder Softwaresysteme ändern; so könnten beispielsweise neue Anästhesiemonitore zusätzliche Informationen erfassen, die im Protokoll zu hinterlegen wären.

Selten berücksichtigte Einwirkungen haben auch Änderungen an Methoden und Werkzeugen bei der Softwareerstellung zur Folge. So könnten durch neue Funktionen in Betriebssystemen, Compilern, Interpretern, Datenbank-Management-Systemen etc. Performancegewinne und Optimierungspotentiale in Aussicht gestellt werden, wenn die Software, die das jeweilige Informationssystem realisiert, dahingehend angepasst wird, diese neuen Funktionen auch zu nutzen.

Die permanenten Anpassungen verursachen Zusatzkosten, die das Budget der Anästhesieabteilungen belasten. Daher wird die Entwicklung eines Informationssystems gefordert, welches mit überschaubarem Aufwand an die jeweiligen anästhesiespezifischen Bedürfnisse angepasst werden kann.

3 Pliable Objects

Um die Anforderung zu erfüllen, muss ein Informationssystem mit einer Generik versehen werden, die es ermöglicht, die gewünschten, bislang nicht absehbaren Anpassungen des Informationssystems vornehmen zu können. Einen Ansatz bildet das Modell der Pliable Objects. Sie stellen ein neues Konzept zur Abbildung realer Entitäten dar und beinhalten ferner einen generischen Ansatz, um bestehenden Objekten eine neue Form zu geben bzw. derart zu „verbiegen“, dass sie neuen Ansprüchen gerecht werden.

3.1 Definition Pliable Object

Ein *Pliable Object* π ist eine Beschreibungsdomäne Γ, die um eine Menge von *Attributes* A, einer Menge von *Actions* X und einer Menge von *Rules* P zu einem Tupel $\pi = (\Gamma, A, X, P)$ erweitert wird.

- Die Beschreibungsdomäne setzt sich zusammen aus einem Fachausdruck zur Benennung und einer (umgangssprachlichen) Erklärung der Bedeutung und des Zweckes, die dem Pliable Object durch die Experten zugewiesen werden.
- Ein **Attribute** charakterisiert eine Eigenschaft, die für das Pliable Object relevant ist und als Angabe einer Zustandsinformation dient.
- Eine **Action** definiert einen bestimmten Zustandswechsel, der ein spezielles Verhalten eines Pliable Objects beschreibt.
- Eine **Rule** beschreibt eine Verhältnismäßigkeit, die das Verhalten, die Beziehungen und die Attributes eines Pliable Objects verwaltet. Dies schließt die Erkennung von Ereignissen in seiner Umgebung, die Änderung seiner Attributes und die Interaktion mit anderen Pliable Objects mit ein.

3.2 Verständnis von Pliable Objects

Die Definition von Pliable Objects ist der Definition von Business Objects der OMG sehr ähnlich [C96]. Die von der OMG vertretene Auffassung und Interpretation der Definition als Grundlage für Komponenten wird nicht vertreten, da kein Bezug auf ein näheres Architekturkonzept gegeben ist. Stattdessen wird die Theorie entwickelt, dass Pliable Objects nicht auf einer Ebene oberhalb des objektorientierten Modells aufsetzen, sondern dass es sich bei den Pliable Objects um ein Modell handelt, welches mittels des objektorientierten Modells ausgedrückt wird.

Das semantische Datenmodell von Pliable Objects ist in Abbildung 1 wiedergegeben. Ein Pliable Object entspricht quasi einem Objekt des objektorientierten Modells.

erschienen im Tagungsband der INFORMATIK 2011
Lecture Notes in Informatics, Band P192
ISBN 978-3-88579-286-4
weitere Artikel online:
http://informatik2011.de/519.html

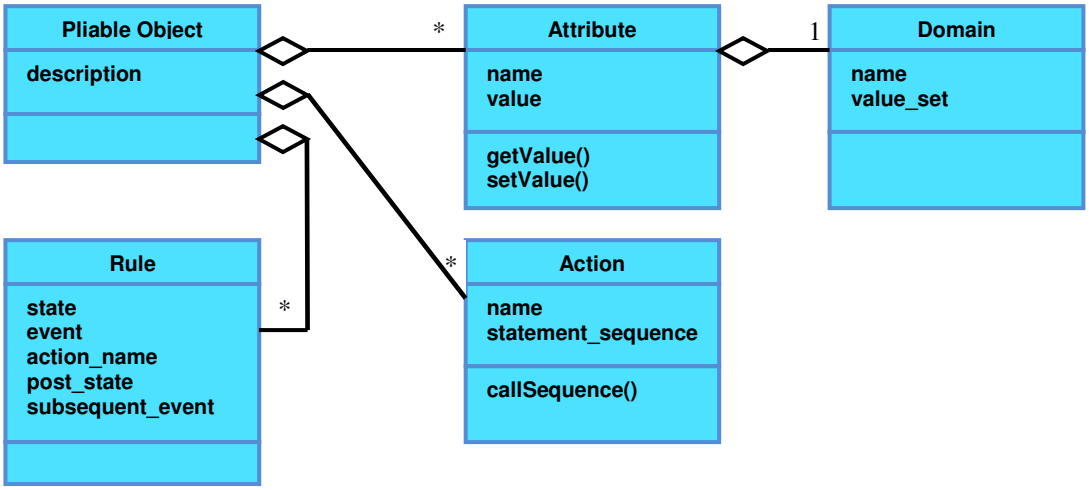

Abbildung 1: Semantisches Pliable-Object-Modell

Im Gegensatz zum Objekt des objektorientierten Modells besteht das Pliable Object nicht aus Attributen und Methoden, sondern besitzt jeweils eine Menge von Attributes und Actions. Die Mengenbetrachtung von Attributes und Actions erlaubt einen flexibleren Umgang mit Pliable Objects. Über einfache Mengenoperationen wird es möglich, die Gestalt eines Pliable Objects zu ändern, nicht nur hinsichtlich seiner Eigenschaften, sondern auch in Bezug auf sein Verhalten. Besondere Dynamik erhält ein Pliable Object durch die Zuweisung von Rules. Eine Regel erweitert ein Pliable Object um einen Zustand inklusive Transitionen, die Bearbeitungs- oder Verfahrensanweisungen darstellen, wenn der Zustand aktiv ist, und das Ereignis, auf das die Regel reagieren soll, eintrifft.

3.3 Beispiel

Bei Patienten, die einmal einen Herzinfarkt erlitten haben, ist die Reinfarktrate innerhalb des ersten halben Jahres nach dem Infarkt besonders groß. Daher muss sich ein Anästhesist vergewissern, ob ein Patient bereits einmal einen Herzinfarkt erlitten hat. Liegt ein Infarkt maximal ein halbes Jahr zurück, wird empfohlen, einen neuen Operationstermin festzulegen, sofern die Operation aufschiebbar ist - andernfalls sollten zusätzliche subtile Überwachungsmethoden zum Einsatz kommen (vgl. [KS01]).

Exemplarisch wird im Folgenden die Realisierung obiger Überprüfung mittels Pliable Objects skizziert. Dabei steht das Prinzip der Einzelinformationsabbildung vorrangig im Fokus, weshalb die Ermittlung des Patientenzustandes außen vor gelassen wird.

1. Eine Domain mit den *Herzinfarktzeiten { < 3 Monate, < 6 Monate, > 6 Monate}* ist anzulegen.

2. Anschließend ist ein Attribute *Herzinfarkt* mit der Domain *Herzinfarktzeiten* und dem Defaultwert *nicht bekannt* anzulegen.

3. Dieses Attribute ist dem Pliable Object *Patientenzustand* zuzuweisen, welches die Daten enthält, die der Anästhesist bei der Zustandsbeurteilung erfasst.

4. Folgende Actions sind anzulegen und ebenfalls dem Pliable Object *Patientenzustand* zuzuweisen:

    a. *getHeartData*: Action, die herzinfarktrelevante Daten erfasst.

    b. *checkHeartData*: Action, die die erfasste Herzinfarktangabe mit der Domain abgleicht. Bei möglichem Konflikt, der besagt, dass der Patient innerhalb des letzten halben Jahres einen Infarkt erlitten hat, überschreibt die Action das Folgeereignis mit dem Ereignis *E_Attention*, um die Prüfung einer möglichen Verschiebung anzustoßen.

    c. *checkMovement*: Action, die eine Verschiebung der Operation prüft. Wenn die Verschiebung nicht möglich ist, überschreibt die Action das Folgeereignis mit dem Ereignis *E_NoMove*, um zu signalisieren, dass keine Verschiebung möglich ist.

    d. *cancelOperation*: Action, die eine geplante Operation verschiebt.

    e. *getBloodValues*: Action, die die Blutwerte des Patienten hinsichtlich möglicher operationsrelevanter Probleme prüft.

5. Abschließend werden in tabellarischer Form folgende Rules vereinbart, die indirekt den Ablauf der Beurteilung des Patientenzustands während der präoperativen Phase widerspiegeln.

State	Event	Action	Post state	Subsequent Event
S_IDLE	E_Start	getHeartData	S_Heart	E_Check
S_Heart	E_Check	checkHeartData	S_Heart	E_OK
S_Heart	E_Attention	checkMovement	S_Move	E_Move
S_Heart	E_OK	getBloodValues	S_Blood	E_Check
S_Move	E_Move	cancelOperation	S_IDLE	-
S_Move	E_NoMove	getBloodValues	S_Blood	E_Check

Tabelle 1: Skizzierte Regeln

ISBN 978-3-88579-286-4

In Tabelle 1 ist ein Ablauf skizziert, in dem angenommen wird, dass im Rahmen der Patientenzustandsermittlung irgendwann die Frage hinsichtlich eines etwaigen Herzinfarktes geklärt wird. Im Normalfall werden keine Probleme erwartet, so dass generell nach dieser Klärung beispielsweise mit der Bewertung von Blutwerten fortgefahren werden kann. Die Blutwerte selbst spielen in diesem Beispiel keine Rolle.

In Tabelle 1 bedeutet dieses Verhalten, dass grundsätzlich die Abarbeitung durch die Sequenz *Zeile1, Zeile 2, Zeile 4* gegeben ist, d.h. zunächst wird die Angabe hinsichtlich eines Herzinfarktes entgegengenommen. Es folgt die Prüfung, ob die Angabe operationsrelevante Folgen haben kann. Die allgemeine Erwartungshaltung ist, dass keine operationsrelevanten Folgen festgestellt werden, so dass sich die Ermittlung der Blutwerte anschließt.

Die Ausnahme tritt ein, wenn bei der Bewertung eines etwaigen Herzinfarktes eine Sonderprüfung vorgenommen werden muss. Dieser Sachverhalt wird von der Prüf-Action *checkHeartData* über das Ereignis *E_Attention* signalisiert. Die Verarbeitung dieses Ereignisses, gegeben durch *Zeile 3*, unterbricht den Standardablauf und sorgt für die Prüfung einer Operationsverschiebung, in dem in den Zustand *S_Move* gewechselt und die Prüf-Action *checkMovement* aufgerufen wird. Als Vorgabe für diese Ausnahme wird angenommen, dass die Operation verschoben werden kann. Die Ausnahmeverarbeitung wird über *Zeile 5* geführt, d.h. die aktuelle Patientenzustands-ermittlung ist hinfällig und wird abgebrochen. Falls die Ausnahme in der Ausnahme eintritt und die Operation nicht verschoben werden kann, werden innerhalb der Prüf-Action *checkMovement* Vorgaben für die subtilen Überwachungsmethoden getroffen. Der weitere Ablauf wird über *Zeile 6* geführt, und es wird mit der gewohnten Überprüfung der Blutwerte fortgefahren.

Auf eine Darstellung der Regeln in einem Zustandsdiagramm ist bewusst verzichtet worden, da die Auflistung der Rules in tabellarischer Form in Tabelle 1 eine andere Sicht auf Abläufe erlaubt. Durch die tabellarische Form wird eine Ablaufsteuerung sichtbar, mit deren Hilfe Einfluss auf die Reihenfolge der Action-Aufrufe genommen werden kann. Prinzipiell kann ohne Änderung an den Actions die Reihenfolge der Action-Aufrufe durch simple Änderungen in der Tabelle vorgenommen werden. Existiert beispielsweise eine Action *getDiabetes*, die eine eventuelle Zuckererkrankung des Patienten erfasst und wünscht ein Anästhesist, dass die eventuelle Zuckererkrankung nach der Erfassung des Herzinfarktes erfasst werden soll, wäre in der Tabelle nur die Action *getBloodValues* gegen die Action *getDiabetes* auszutauschen.

Die Speicherung von Tabellen und ihren Inhalten ist eine Kernaufgabe von relationalen Datenbanken. Mit der Interpretation der Tabelle 1 als spezielle Relation einer relationalen Datenbank wird ein Vorteil von Pliable Objects deutlich. Die Ablauflogik, die in Programmen fest codiert wird, wird in die Datenbank verschoben und kann dort mittels SQL-Anweisungen verändert werden. Relationale Datenbanken wiederum bilden häufig die Basis für Informationssysteme. Wären deren Ablauflogiken in den Datenbanken hinterlegt, könnte geschultes Fachpersonal mittels generalisierter Zugriffe die Ablauflogiken in die gewünschte Form bringen.

erschienen im Tagungsband der INFORMATIK 2011
Lecture Notes in Informatics, Band P192
ISBN 978-3-88579-286-4
weitere Artikel online:
http://informatik2011.de/519.html

3.4 State Machine

Wenn die Rules in der vorgeschlagenen Form in der relationalen Datenbank eines Informationssystems gespeichert werden, sollte das Informationssystem in die Lage versetzt werden, die Verarbeitung entsprechend der Rules zu steuern.

Die Rules sind einem Pliable Object zugewiesen. Durch die Zuweisung von Rules wird das Pliable Object selbst indirekt zu einer Zustandsmaschine. Es gilt nun, die Zustandsverarbeitung, gegeben durch das Pliable Object, in einem Informationssystem auf simple Art und Weise vorzunehmen.

Generell ist eine Zustandsmaschine durch eine Menge von Zuständen gegeben (vgl. Abbildung 2). Ein Zustand wird innerhalb der Zustandsmaschine als aktiv markiert, und in einer Eigenschaft wird das Ereignis hinterlegt, welches als nächstes zu verarbeiten ist.

Ein Zustand selbst besitzt nur zwei Actions, mit denen die Verhaltensänderung vorgenommen werden kann, die gewünscht wird, bevor der Zustand aktiviert bzw. durch Aktivierung eines anderen Zustands deaktiviert wird.

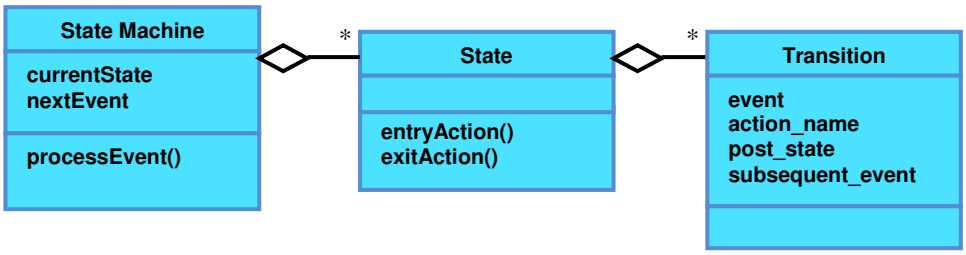

Abbildung 2: Beschreibung einer einfachen State Machine

Für jeden Zustand kann eine Menge von Transitions festgelegt werden. Jede Transition beschreibt, auf welches Ereignis hin sie aktiv wird, d.h. nach Eintreffen des Ereignisses wird die mit der Transition verknüpfte Action verarbeitet, und die Zustandsmaschine wird angewiesen, den in *post_state* hinterlegten Zustand zu aktivieren. Abschließend ist das eventuell in der Transition angegebene Folgeereignis zu verarbeiten. Falls kein Folgeereignis angegeben ist, wird dies als Abschluss der gewünschten Verarbeitung interpretiert. Ein Beispiel einer Relation, die Transitions verwaltet, ist in Tabelle 1 gegeben.

Der Algorithmus, der das Verhalten der Zustandsmaschine realisiert, ist in Abbildung 3 wiedergegeben. Er selbst realisiert prinzipiell wieder eine abstrakte Zustandsmaschine, die nur zwei Zustände kennt, entweder der Algorithmus wartet auf ein Ereignis oder er verarbeitet ein Ereignis.

erschienen im Tagungsband der INFORMATIK 2011
Lecture Notes in Informatics, Band P192
ISBN 978-3-88579-286-4
weitere Artikel online:
http://informatik2011.de/519.html

Der Ausgangs- oder Startzustand ist der Wartezustand. Wenn ein Ereignis eintrifft und zu verarbeiten ist, wechselt der Algorithmus in den Verarbeitungszustand. Darin wird zunächst die Transition-Action bestimmt. Die Transition-Action ist die Action, die im aktiven Zustand mit dem Ereignis verknüpft ist. Anschließend wird das Folgeereignis aus der Transition bestimmt, welches das Ereignis ist, das nach Beendigung des Aufrufs der Action zu senden bzw. zu verarbeiten wäre.

Die eigentliche Verarbeitung startet mit der Überprüfung, welcher Zustand während des Action-Aufrufes als aktiv zu markieren ist.

Falls es sich um den Zustand handelt, der aktuell aktiv markiert ist, wird nur die Action aufgerufen und anschließend das Folgeereignis verarbeitet. Es sei angemerkt, dass die Action die Option besitzt, das Folgeereignis zu überschreiben, um damit eine Abweichung vom Standardablauf zu initiieren.

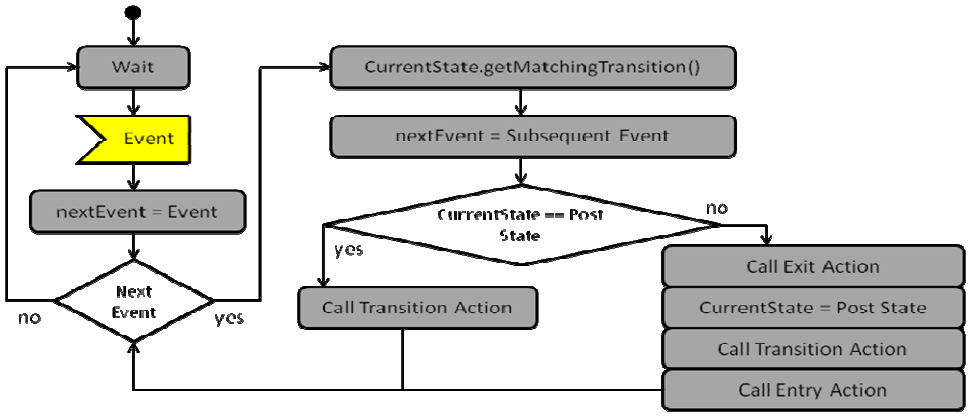

Abbildung 3: Verhalten der einfachen State Machine

Falls ein anderer Zustand als aktiv markiert werden soll, wird zunächst die eventuell angegebene Exit-Action aufgerufen. Es folgen die Aktiv-Markierung des neuen Zustands und der Aufruf der Transition-Action. Abschließend wird die Entry-Action des neuen Zustands aufgerufen und ein eventuell vorliegendes Folgeereignis verarbeitet.

Die Verarbeitung stoppt, wenn kein Folgeereignis mehr vorliegt. In diesem Fall kehrt der Algorithmus in den Ausgangszustand zurück.

3.5 Skizzierung des Datenbankschemas

Ein Informationssystem, das das obige Verhalten einer State Machine bietet, kann einer Ablauflogik folgen, die in der Datenbank des Informationssystems abgelegt ist. Ein vereinfachtes, am Entity-Relationship-Modell angelehntes Datenbankschema wird in Abbildung 4 skizziert, um den Fokus auf die Basisidee zu lenken.

ISBN 978-3-88579-286-4

Die Ablauflogik wird in einer Form, wie sie Tabelle 1 zeigt, über die Entity Rule mit einem Pliable Object verknüpft. Analoge Verknüpfungen werden für Attributes und Actions angelegt. Um das im obigen Beispiel gezeigte Verhalten hinsichtlich Domains zu unterstützen, verfügen Attributes jeweils über eine Domain. Actions werden hier nicht tiefergehend betrachtet. Es sollen darunter Funktionseinheiten verstanden werden, die eine kleine Verarbeitungsroutine realisieren. Im Rahmen dieser Verarbeitungsroutine wird über Standardmechanismen, wie z.B. SQL-Anweisungen, auf Werte von Attributen in der Datenbank zugegriffen, diese Werte verarbeitet und gegebenenfalls die Ergebnisse in der Datenbank gespeichert. Eine derartige Verarbeitung kann entweder eine Datenerfassung oder eine Datenprüfung bzw. eine Plausibilitätskontrolle sein.

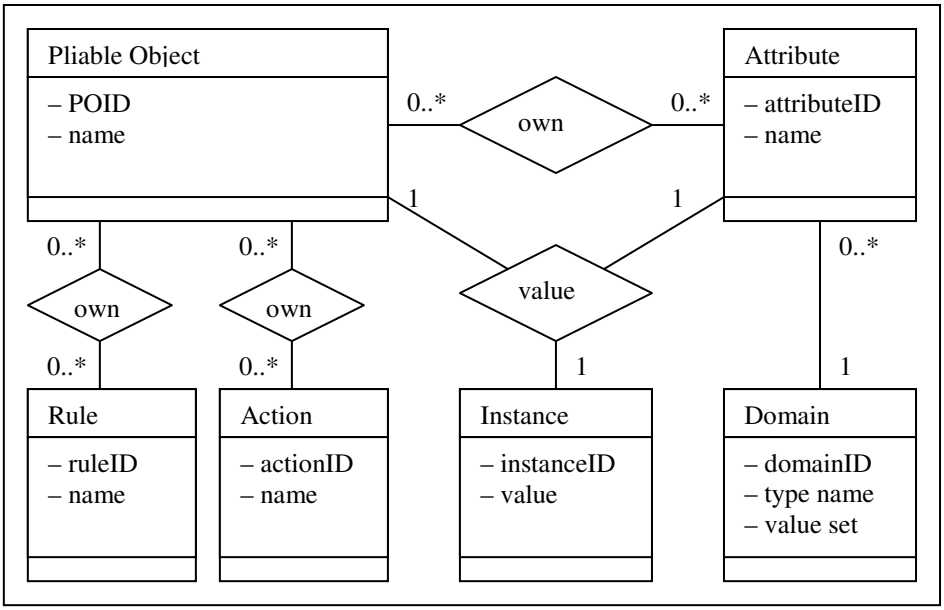

Abbildung 4: Vereinfachte Darstellung des Datenbankschemas

Eine Besonderheit stellt die Instance dar. Sie speichert eine Menge von Values, wobei jeder einzelne Wert einer eingegebenen oder berechneten Information entspricht. Mittels der ternären Assoziation *value* wird einerseits die Information an ein Attribut gebunden und andererseits wird die is-a-Beziehung zum Pliable Object angegeben, um über dessen Rules und Actions darin gespeicherte Plausibilitätskontrollen zu triggern.

ISBN 978-3-88579-286-4

4 Zusammenfassung

Pliable Objects können einerseits Informationen generisch aufbereiten, andererseits bieten sie die Möglichkeit zur Definition von Aufrufsequenzen durch eine geeignete Aufbereitung von Actions. Dies geschieht dadurch, dass die häufig in Methoden hinterlegte bedingte Verarbeitungslogik in ein tabellarisches Regelsystem verschoben wird. Dieses Regelsystem kann in einer Datenbank hinterlegt werden, so dass mit den Zugriffsmitteln eines Datenbankmanagementsystems das Regelsystem geändert werden kann. Damit wird dem Datenbankmanagementsystem erlaubt, durch Änderung von Aufrufsequenzen eine Verhaltensänderung in der Software oder gar dem Informationssystem herbeizuführen.

Anästhesie-Informationssysteme, die das Konzept der Pliable Objects integrieren und die Informationen mittels Pliable Objects erfassen und verarbeiten, würden die Optionen anbieten können, die Erfassung und die Verarbeitung an geänderte Umstände anpassen zu können, ohne dass eine Softwareaktualisierung vorzunehmen wäre.

Eine Realisierung von Pliable Objects in Informationssystemen steht noch aus.

Literaturverzeichnis

[BS02] Broy, M., Siedersleben, J.: Objektorientierte Programmierung und Softwareentwicklung – Eine kritische Einschätzung, Informatik Spektrum, 25.2.2002

[B97] Bruder, M., Rehfeld, W., Seeger, T., Strauch, D. (Hrsg.): Grundlagen der praktischen Information und Dokumentation, 4. völlig neu gefasste Ausgabe, Band 1 und 2, K.G.Saur Verlag, 1997

[B07] Branitzki et al.;Spezielle Empfehlungen und Anforderungen zur Implementierung eines Anästhesie-Informations-Management-Systems; Anästh. Intensivmed., 2007;48:282-290

[C96] Casanave, Cory (ed.): OMG Common Facilities RFP-4 Common Business Objects and Business Object Facility, OMG TC Document CF/96-01-04, www.omg.org

[CYA94] Coad, P., Yourdon, E.: OOA – Objektorientierte Analyse, Prentice Hall Verlag, 1994

[CYD94] Coad, P., Yourdon, E.: OOD – Objektorientiertes Design, Prentice Hall Verlag, 1994

[CC01] Cuhls, M., Cuhls, H.: Patienteninformation – Ablauf einer Narkose in Bildern, 15.06.2001, http://www.meb.uni-bonn.de/institute/kliansint/patient/inhalt.htm

[GHJV95] Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Entwurfsmuster, Addison-Wesley, 1995

[H92] Heuer, A.: Objektorientierte Datenbanken, Konzepte, Modelle, Systeme, Addison-Wesley, 1992

[KS01] Kretz, F.J., Schäfer, J.: Anästhesie, Intensivmedizin, Notfallmedizin, Schmerztherapie, Springer-Verlag, 2001

[OW99] Opderbecke, H.W., Weissauer, W.(Hrsg.): Entschließungen, Empfehlungen, Vereinbarungen, Ein Beitrag zur Qualitätssicherung in der Anästhesie, Kerndatensatz Anästhesie – Version 2.0/1999, http://www.dgai.de/06_1_00tabelle.htm#zu_viii

[Pr05] Preißner, M., Modellierung und Entwicklung von Pliable Objects zur Unterstützung des Aufbaus von Informationssystemen im medizinischen Anwendungsgebiert der Anästhesie, 17. Workshop Grundlagen von Datenbanken, Tagungsband, 2005

[RT04] Roewer, T., Thiel, H.: Taschenatlas der Anästhesie, 2. aktualisierte Auflage, Thieme Verlag, 2004

[S94] Sims, O.: Business Objects, Delivering cooperative objects for client-server, McGraw – Hill, 1994

[SPCH97] Sutherland, J., Patel, D., Casanave, C., Hallowell G., Miller, J.(eds): Business Object Design and Implementation, OOPSLA'95 Workshop Proceedings, Springer Verlag, 1997

[V00] Vossen, G.: Datenmodelle, Datenbanksprachen und Datenbankmanagementsysteme, Oldenbourg, 2000