Ein Werkzeug zur Nutzung von Analysemustern

Alfred Wulff

Fachhochschule Oldenburg/Ostfriesland/Wilhelmshaven
Institut für Systementwicklung in der Wirtschaftsinformatik (ISW)
26389 Wilhelmshaven
wulff@fbwi.fh-wilhelmshaven.de

Abstract: Patterns represent established concepts in object-oriented development with a main application area in design and architectural project phases. Although the potential benefits of Analysis Patterns are well-known, there is still no widespread application in the daily work. Prerequisites for an efficient utilisation of patterns in requirements and business modelling phases of a project are a fast project- or even company-wide access to analysis pattern sources, integration in the development process and appropriate tool support. E-Business projects can benefit from Analysis patterns esp. concerning the critical factor "time-to-market" because the availability of resources with initial design ideas can accelerate the early modelling phases. After a survey of pattern descriptions in the Unified Modeling Language (UML) and available CASE-Tool support, this contribution depicts the architecture and application of a Pattern-Tool that was developed in the scope of an E-Government project. The tool provides an initial set of known Analysis Patterns, supports its selection and integration in concrete system models, and enables extension and modification of the Pattern population. The Pattern-Tool is integrated as a "Plug-In"-component in the commercial CASE-Tool "Rational Rose". The open architecture enables an adaptation for other tools and modelling languages. Examples and experiences for applied Analysis patterns in the E-Government project will be illustrated.

1 Einleitung

Die Anwendung von Patterns wird seit Verbreitung objekt-orientierter Entwicklungssprachen und -methoden als wichtiges Hilfsmittel im SW-Entwicklungsprozess propagiert und kann mittlerweile als charakteristisches Paradigma objekt-orientierter Systementwicklung betrachtet werden. Aufgrund der Vielzahl der in unterschiedlichen Quellen und verschiedenen Formaten publizierten Muster ist der Zugang und damit die Nutzung in konkreten Entwicklungsprojekten jedoch stark erschwert. Zudem fehlt in den meisten gängigen Entwicklungsmethoden eine geeignete Systematik zur Pattern-Einbettung, sowie eine passende Werkzeugunterstützung. Dies trifft auch für die in der IT-Systementwicklung verbreitete Modellierungssprache Unified Modeling Language (UML) [BRJ99] und den Entwicklungsprozess Rational Unified Process (RUP) [Kr99] zu. Zwar sind hier entsprechende Notationen zur Beschreibung von Mustern vorgesehen, es fehlen jedoch Verfahrensschritte und Tools, die eine Mustereinbettung konstruktiv unterstützen.
Andererseits wird insbesondere für die Entwicklung von Internet-basierten Informationssystemen und E-Business-Anwendungen u.a. aufgrund der Komplexität der

betroffenen Anwendungsdomänen, der Vielschichtigkeit der Anwendungsarchitektur und der Anforderung nach kurzen Entwicklungszeiten, die effektive Nutzung erprobter Modelle und Lösungen immer wichtiger. Deutlich wurden diese Anforderungen bei der Entwicklung eines Internet-basierten Umwelt-Informationssystems, das als Kooperationsprojekt einer Kommunalbehörde und dem Institut für Systementwicklung in der Wirtschaftsinformatik (ISW) der Fachhochschule Oldenburg/Ostfriesland/ Wilhelmshaven durchgeführt wurde (vgl. [Wu00]). Im Rahmen dieses Projektes wurde ein Werkzeug zur Unterstützung einer Pattern-orientierten Softwaremodellierung konzipiert und prototypisch realisiert. Insbesondere wird damit die Auswahl und Einbettung von Analysemustern sowie ihre Weiterentwicklung im jeweiligen Systemmodell erleichert. Das entwickelte Werkzeug ist als Plug-In für das verbreitete Case-Tool "Rational Rose" [Ra01] realisiert worden, wobei die offene Architektur des Werkzeugs die Übertragung in andere Werkzeugumgebungen und Modellierungssprachen ermöglicht.

2 Patterns in der Anwendungsentwicklung

2.1 Überblick und Quellen

An dieser Stelle soll keine Einführung in das umfangreiche Gebiet der Patterns gegeben werden. Es wird vielmehr auf die im folgenden angegebenen Arbeiten verwiesen, die hier lediglich als Bezugspunkte für Patternbeschreibungen aufgeführt sind.
Schwerpunkt der Entwicklung und Nutzung von Mustern ist der Bereich der Entwurfs- und Software-Architekturmuster, wie die Veröffentlichungen z.B. in den jährlich stattfindenden PLOP-Konferenzen (Pattern Languages of Programs, auch EuroPLoP, ChiliPLoP, KoalaPLoP) zeigen. Dabei bilden die grundlegenden Arbeiten der "Gang of Four" (GOF) [Ga95] und PoSA-Autoren [Bu98] weiterhin die Basis zahlreicher Pattern-Neuerscheinungen, deren Diskussion und Präsentation über Writers-Workshops durchgeführt wird. Pattern-Systeme und Frameworks stellen Strukturierungen von Mustermengen dar und sind idealerweise über Mustersprachen (vgl. z.B. [Ev00]) definiert. Patterns für SW-Enwicklungsprozess und -organisation dokumentieren erprobte Praktiken in Vorgehensmodellen, Management und Organisation und sind etwa in den Arbeiten von Coplien [Co94] , Ambler [Am98], Larman [La98] und Brown [Br99] dargestellt. Eine neuere umfassende Zusammenstellung ist auch im Pattern Almanac 2000 [Ri00] zu finden, wobei hier nur Referenzen auf Patternquellen dokumentiert sind.
Das Wissen über Analyse-Muster, d.h. konzeptuelle Strukturen spezifischer Anwendungsdomänen wird i.d.R. als Spezialwissen von Analytikern zur (firmen)eigenen Nutzung geschützt. Quellen für Muster mit Anwendungs- oder Geschäftsmodellen sind daher nicht frei verfügbar und können allenfalls als gesonderte Bausteine kommerziell erworben werden (z.B. [TI96]). Zu den wenigen umfangreicheren, frei-verfügbaren Quellen mit Spezifikationen konzeptueller oder Analysemuster gehören die Arbeiten von Fowler [Fo97] und Eriksson [Er00].
Als elektronische Quellen sind neben den Homepages einiger Pattern-Autoren die Mustersammlungen des Portland Pattern-Repository [Cu01] und der Hillside-Group

Pattern-Homepage [Hi01] zu nennen. Teilweise liegen hier auch UML-basierte Beschreibungen einzelner Muster vor.

2.2 Muster-Einbettung in der Unified Modeling Language (UML) und im Rational Unified Process (RUP)

In der UML werden Designmuster als parametrisierte Kollaborationen dargestellt (vgl. [BRJ99]) Das zugehörige Kollaborationssymbol ist erst seit der Spezifkation der UML 1.3 als neue Strukturkomponente festgelegt, obwohl diese Darstellungsart bereits zuvor benutzt wurde (z.B. [Ös97]). Das hier dargestellte Beispiel entstammt dem in der Einleitung genannten E-Government-Projekt "Umweltinformationsssystem" und soll die Anwendung eines Analysemusters dokumentieren.
Als zu modellierender Teilbereich ist hier beispielhaft die Erfassung von Tierarten über (elektronisch verfügbare) Meldebögen gewählt. Dabei müssen u.a. Angaben zur Beobachtungszeit gemacht werden. Bei der Modellierung der Zeitangaben ist zu berücksichtigen, dass sich eine gemeldete Beobachtung entweder auf einen festen Zeitpunkt bezieht (z.B. "totes Exemplar einer Prachtlibelle am 30.07. gefunden") oder einen Zeitraum betreffen kann (z.B. "rastende Singschwäne vom 7.9. bis 12.9. beobachtet"). Die Anwendung des Analysemusters "Dual Time Record" von Fowler ([Fo97], S.47) zur Lösung dieser Problemstellung ist in Abbildung 1 als UML-Klassendiagramm mit Kollaborationssymbol dargestellt.

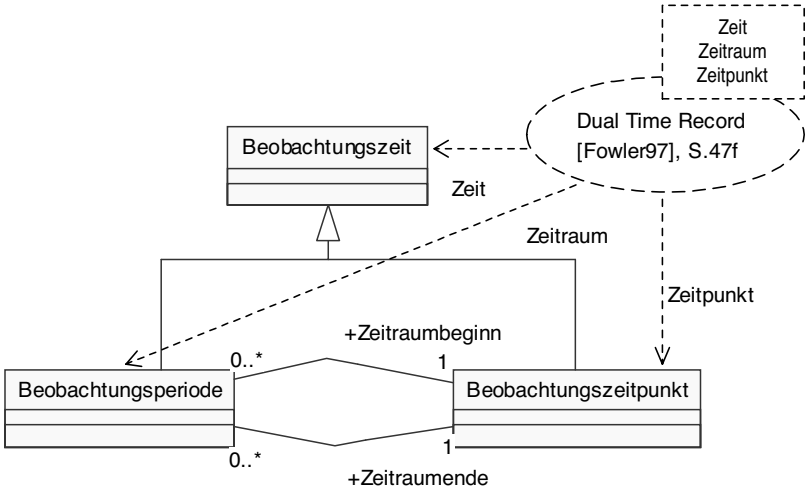

Abbildung 1: Modellierung des Zeitaspektes im E-Governmentprojekt mit Hilfe eines Analysemusters

Der Rational Unified Process (RUP) ist ein generischer Softwareentwicklungsprozess, der UML als Modellierungssprache verwendet. RUP ist ein iterativer Prozess und wird maßgeblich mit dem von Ivar Jacobsen übernommenem Use-Case-Modell gesteuert. Die Verwendung von Mustern wird im RUP-Vorgehensmodell angeregt, wobei der Fokus im Bereich der Design- und Architekturmuster liegt. Als Architekturmuster werden

beispielhaft Layers-Muster, Model-View-Controller-Muster (MVC), Pipes- und Filter-Muster, sowie Blackboard-Muster erläutert und für die Anwendung empfohlen. Eine systematische Einbettung von Mustern in die RUP-Core Process Workflows ist jedoch nicht vorhanden.

2.3 Pattern-Unterstützung in CASE-Tools

Muster stellen keine geschlossene Entwicklungsmethode dar, sondern ergänzen existierende Methoden und Prozesse. Die Verwendung von Mustern in einem Projekt wird dabei vorwiegend als rein mentale Leistung des Analytikers gesehen oder allgemein der Anwendung der methodenspezifischen "Best Practices" zugeordnet.
Methodische Unterstützung bzw. womöglich Einbettungen in SW-Entwicklungsprozesse sind nicht bekannt. John Vlissides führt in [Vl98] gar an, dass "the benefit from patterns comes mostly from applying them as they are - that is, with no support of any kind". Vlissides warnt damit jedoch vor allem vor der Erwartung, dass Pattern-Werkzeuge zur effizienten Codeerzeugung eingesetzt werden können und führt gescheiterte Projekte zur Entwicklung von Pattern-Codegeneratoren an [Bu96]. Ausführliche Pattern-Unterstützung in kommerziellen CASE-Tools ist vom Werkzeug COOL:Plex aus der COOL™-Werkzeugfamilie [CA01] von Computer Associates bekannt. Auch im Werkzeug Together[To01] ist eine Reihe von mitgelieferten GoF-Designmustern incl. Beispiel-Code enthalten. Die Unterstützung von Analyse-Mustern in Case-Tools wird soweit bekannt in keinem kommerziellen Werkzeug angeboten.

3 Anforderungen an das Musterwerkzeug

Aus den Erfahrungen im E-Government-Projekt wurden folgende globale Anforderungen an das Musterwerkzeug abgeleitet:

- Unterstützung einer systematischen Einbettung von Mustern in die verwendete Entwicklungsmethodik
- Mechanismus zur Erkennung passender Muster bereits in der Analysephase
- Erfassung, Pflege und Neuentwicklung von Mustern in einer Datenbank
- Musterbeschreibung gemäß PoSA-Format unter Verwendung der UML
- Suche über Musternamen, Synonyme, Schlüsselwörter, Kategorie incl. Reportfunktionen.

Zu diesen systemunabhängigen Zielsetzungen wurden dann implementierungsspezifische Anforderungen hinzugefügt, die die Verbindung zur verwendeten Entwicklungsumgebung herstellen:

- das Werkzeug soll in das Case-Tool Rational Rose integriert werden
- die Musterdatenbank soll als relationale Datenbank realisiert werden.

Die Entwicklung des Bausteins selbst erfolgte mit Hilfe der UML und in Anlehnung an den Rational Unified Process. Die Implementierung wurde im Rahmen einer Diplomarbeit [Le01] durchführt.

4 Anbindung an das UML-Metamodell

4.1 Konzeption

Um bereits in den frühen Phasen einer Projektentwicklung eine Unterstützung durch vorhandene Muster erhalten zu können, müssen die bisher gesammelten Konzepte einer Anwendungsdomäne mit den in Mustern enthaltenen Konzepten in Beziehung gesetzt werden können.
Der hier gewählte Ansatz betrachtet die im Laufe der Analyse ermittelten Konzepte und Strukturen der Anwendungsdomäne als Instanzen der Klassen des UML-Metamodells. Alle Modellelemente eines Entwicklungsprojektes wie etwa Anwendungsfälle (Use Cases), Klassen, Assoziationen usw. mit allen ermittelten Beschreibungen stehen in diesem Modell zur Verfügung. In Abbildung 2 ist ein Auszug des UML Metamodells als Klassendiagramm dargestellt.

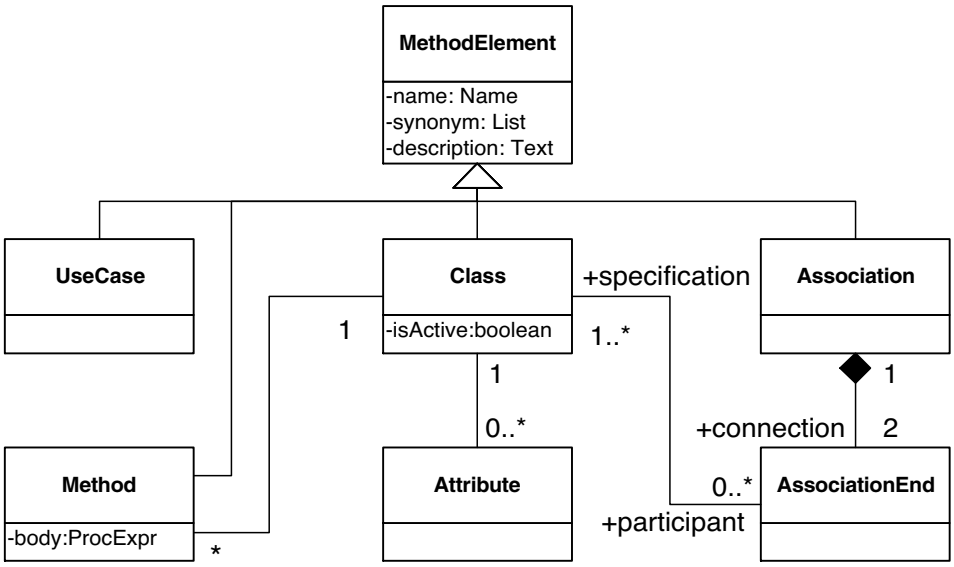

Abbildung 2: Auszug aus UML-Metamodell V.1.3 (Foundation Package) mit Erweiterung

Die aufgeführten Klassen sind dabei dem Foundation-Package und dem Use Cases-Package der OMG-UML-Dokumentation entnommen [OMG01]. Die Attribute "synonym" und "description" wurden hinzugefügt, da diese wichtige Modellinformationen für die geplante Toolkomponente darstellen und zudem in Case-Tool-Implementierungen des Metamodells üblicherweise verfügbar sind.
Die in Patterns enthaltenen Konzepte sollen über die gemäß PoSA-Format vorgesehenen Spezifikationsbestandteile beschrieben und in einer Datenbank abgelegt werden.
Nach diesen Vorbereitungen ist es möglich, Beziehungen zwischen Mustern und Modellelementen der Anwendungsdomäne herzustellen. Die semantische Affinität von Mustern und Modellelementen wird hier über Identität oder terminologische Ähnlichkeit

(Match) von Namen, Synonymen oder sonstigen textuellen Bestandteilen der betroffenen Quellen definiert. Die Zusammenführung passender Objekte wird dabei über ein Vermittlerobjekt vorgenommen, das die passenden Muster- und Anwendungsmetainstanzen in Beziehung setzt. Der Entwurf verwendet implizit das Entwurfsmuster Mediator (Vermittler), das gerade die lose Kopplung verschiedener Objekt ermöglicht, ohne explizite Objektreferenzen einrichten zu müssen.

4.2 Realisierung der Werkzeugkomponente und Population der Muster-DB

Zur Realisierung der hier dargestellten Konzeption ist es notwendig, dass Zugriff auf die Projekt-Population des UML-Metamodells besteht. Da die Implementierungen des Metamodells der verschiedenen CASE-Tool-Entwickler stark voneinander abweichen, muss die Realisierung der Pattern-Komponente für jede spezifische Werkzeugumgebung angepasst werden.
Die Implementierung des UML-Metamodells in dem hier verwendetete CASE-Tool "Rational Rose" kann über das Rose Extensibility Interface (REI) genutzt werden und bietet Zugriffsmöglichkeiten auf alle Modellkomponenten eines Enwicklungsmodells. Die Umsetzung der im vorigen Kapitel erläuterten Konzeption kann damit erfolgen. Der Prototyp wurde mittels Rose Automation Objects als ActiveX-DLL mit Visual Basic (VB) realisiert und dann als Plug-In in Rose eingebunden. Der Prototyp wird in Kapitel 5 an Hand eines Modellierungsbeispiels des zugrundeliegenden E-Government-Projektes demonstriert. Voraussetzung zur Nutzung des Pattern-Tools ist die Verfügbarkeit einer ausreichenden Anzahl spezifizierter Muster. Deshalb wurden Spezifikationen verschiedener publizierter Muster im PoSA-Format mit dem entwickelten Werkzeug erfasst und in der Pattern-DB gespeichert. Entsprechende Beschreibungen lagen über Vorarbeiten im ISW bereits vor. Der Schwerpunkt der Erfassung lag dabei im Bereich der Analysemuster, da hier aufgrund der wenig strukturierten Muster-Beschreibungen (z.B. aus [Fo97]) ein hoher Arbeitsaufwand zur Transformation in das PoSA-Format anfiel. Z. Zt. liegen etwa 40 dem PoSA-Schema entsprechende Beschreibungen vorwiegend von Analysemustern, aber auch einigen ausgewählten Design- und Architekturmustern vor. Diese Grundpopulation wird stetig erweitert. In Abbildung 3 ist die Beschreibung des Musters "GeoObjektrolle" (vgl. [Ba97]) im Pattern-Tool für das PoSA-Kapitel "Problem" dargestellt. Auf dieses Muster wird im Kapitel 5 noch wieder verwiesen.

5 Anwendungsbeispiel: E-Government Projekt einer kommunalen Verwaltung

Zur Demonstration des Tools wird hier ein Auszug eines am Institut durchgeführten Projektes zur Entwicklung eines Internet-basierten Umweltinformationssystems für Kommunalverwaltungen herangezogen (vgl. [Wu00]). Dabei soll die Nutzung in einer frühen Projektphase verdeutlicht werden.
Ziel des Projektes ist der Aufbau eines multimedialen Informationssystems auf der Basis zu entwickelnder integrierter Fachanwendungen. Dabei soll verschiedenen

Interessensgruppen der zielgruppengerechte Zugriff auf aktuelle, dynamisch erzeugte Umweltdaten der Kommune per Internet ermöglicht werden. Bestimmte Informationen sollen nur gegen Gebühr abrufbar sein.

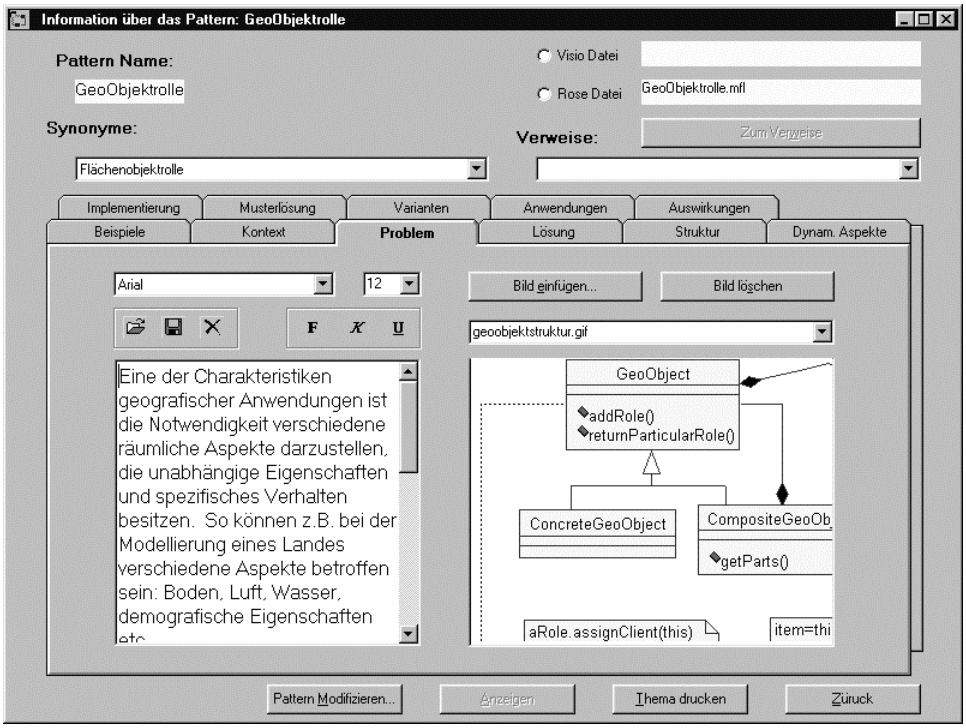

Abbildung 3: Darstellung des Pattern GeoObjektrolle im Pattern-Tool

Zudem soll das System die Möglichkeit bieten, dass umweltrelevante Beobachtungen direkt über passende Formulare gemeldet werden können. Der Systembetrieb soll über Advertisement-Angebote lokaler Unternehmen finanziert werden. Ein zentrales Konzept des Anwendungskomplexes ist die Fläche. Flächen können zusammengesetzt sein und verschiedene unabhängige Rollen einnehmen, wobei Flurstücke als elementare Flächeneinheiten zu berücksichtigen sind. Folgt man dem RUP, so werden aus dem 'Problem Statement' die 'preliminary analysis classes' gebildet, womit ein erstes konzeptuelles Modell des Anwendungsbereiches vorliegt. Ein Auszug aus dieser Sammlung ist im Diagramm der Abbildung 4 dargestellt.
Nach Installation des Pattern-Tools kann über das Tools-Menu das "Patterns"-Plug-In gestartet werden. Mittels des Eintrags "Pattern Suchen" lässt sich eine Suche über den vorhandenen Patternbestand nach verschiedenen Kriterien durchführen. Die gefundenen Muster können dann wie in Abbildung 3 dargestellt, inspiziert werden. Neue Muster lassen sich mit dem Menüpunkt "Pattern erstellen" einrichten. Da hier die Klasse "Fläche" markiert wurde, kann jetzt mit der Funktion "Pattern Match" die Suche nach semantisch ähnlichen Mustern aus der Pattern-DB durchgeführt werden. Dazu wird nach Patterns gesucht, deren Namen bzw. Synonyme mit dem Namen oder einem Synonym

der betrachteten Klasse übereinstimmt oder als Teil enthalten ist. Zusätzlich werden auch die Kapitel "Problem" und "Kontext" auf das Vorkommen gleichlautender Namen untersucht.

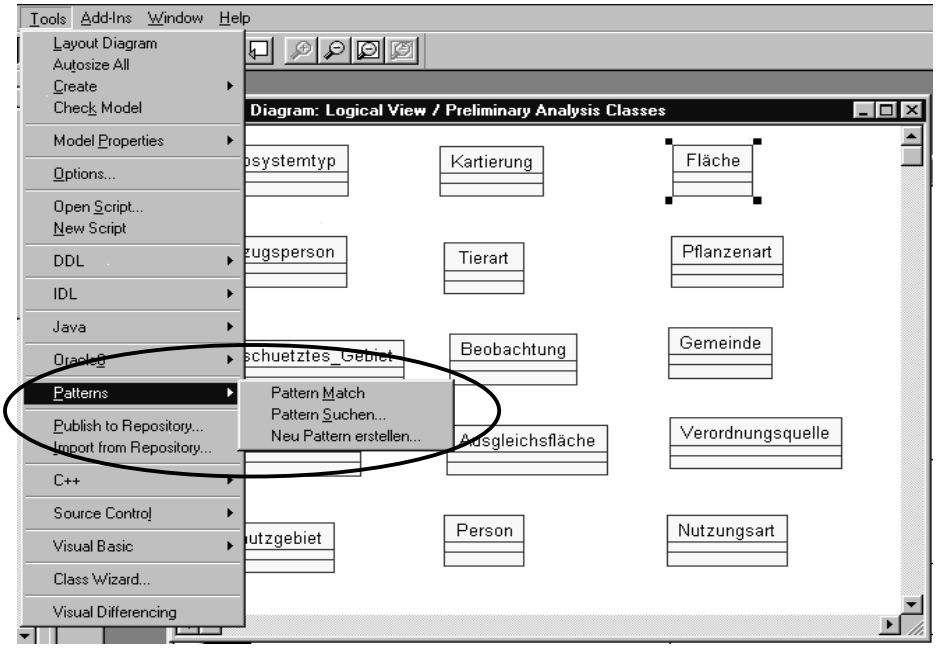

Abbildung 4: 'Preliminary analysis classes' (Auszug) mit Pattern-'Plug-In'

Als Ergebnis wird hier das Pattern "GeoObjektrolle" [Ba97] gefunden (siehe auch Abbildung 3), da als Synonym der Begriff "Flächenobjektrolle" existiert. Dieses Muster kann nun vom Entwickler untersucht werden und falls gewünscht über das in der Datenbank gespeicherte Rose-Modell in das vorliegende Analysemodell übernommen werden. Die Einbettung in das Klassendiagramm des Modells wird vom Pattern-Werkzeug dann automatisch durchgeführt. Nach entsprechender Anpassung und Dokumentation sieht das Analysemodell dann z.B. wie in Abbildung 5 gezeigt aus. Die Kennzeichnung mit dem Kollaborationssymbol ist hier als externe Grafik hinzugefügt worden, da die verwendete Rose-Version 98EE diese Darstellungsart noch nicht unterstützt.

6 Erfahrungen und Ausblick

Der Prototyp des Pattern-Tools konnte bereits für den Projektabschnitt "Schutzgebiete" im E-Government-Projekt verwendet werden. Die Nutzung für die weiteren Projektbausteine wie "Kompensationsflächen" und "Flora und Fauna" ist vorgesehen. Schon der bisherige Einsatz konnte die Nützlichkeit deutlich machen und ist vor allem in folgenden Punkten zu sehen. Den projektbeteiligten Experten der Behörde (u.a. Dipl.-

Landschaftsplaner, Dipl.-Biologen) ließ sich die Verwendung der Muster gut erläutern und der toolgestützte Verwendungsnachweis erhöhte die Akzeptanz des Modells bei allen Mitwirkenden. Das in Abbildung 1 aufgeführte Beispiel bietet einen mit dem Tool sofort verfügbaren Ansatz für eine Modellierung des Zeitaspektes, die für weniger erfahrene Analytiker erst nach längeren Modellierungsaufwand erreicht worden wäre. Die bekanntlich diffizile und zeitraubende Modellierung räumlicher Strukturen (hier Schutzgebiete-Flächen) konnte wie in Abschnitt 5 gezeigt wurde, mit der Anwendung des Musters "GeoObjektrolle" schnell und zufriedenstellend gelöst werden. Die Modelldokumentation lässt sich mit dem Tooleinsatz verbessern und beschleunigen. Die Begründung für bestimmte Entwurfsentscheidungen in einem Modell sind leicht festzuhalten und für später im Projekt eingesetzte Akteure gut nachzuvollziehen.

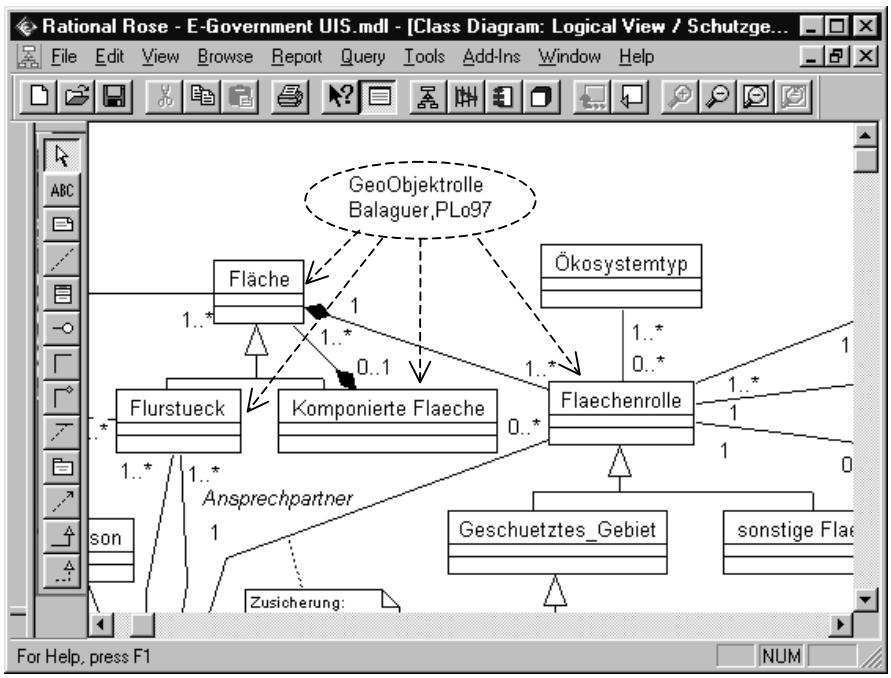

Abbildung 5: E-Government-Klassenmodell mit eingebautem Analysemuster (Auszug)

Natürlich kann das beschriebene Werkzeug nur eine Hilfestellung im Modellierungsprozess bieten. Voraussetzung für den nutzbringenden Einsatz des Hilfsmittels ist die sorgfältige Erarbeitung und Spezifikation der betroffenen Anwendungsdomäne in der Definitionsphase. Insbesondere ist hier der große Wert einer gewissenhaften Erstellung der Fachterminologie, Synonymdefinition und textuellen Beschreibungen der einzelne Fachkonzepte zu erkennen, d.h. langbekannter "best practices" der Systemanalyse. Deutlich wird auch, dass zumindest Namen und Synonyme von Analysemustern in Landessprachenversionen vorliegen müssen, wenn diese automatisiert eingesetzt werden sollen. Für den Einsatz von Designmustern ist dies weniger notwendig, da diese vorwiegend für die Software-technische Realisierung verwendet werden.

Das Werkzeug lässt einen kontinuierlichen Aufbau eines Musterbestandes zu und kann die Grundlage für domänenspezifische Mustersysteme innerhalb eines Unternehmens werden und dabei teamübergreifend eingesetzt werden. Die Integration in andere CASE-Werkzeuge ist möglich, sofern diese "offen" sind bzgl. des Zugriffs auf die Implementierung des verwendeten Metamodells. Die Vermittlungsregeln zum Pattern-Matching können ebenfalls leicht erweitert und verfeinert werden.

Literaturverzeichnis

[Am98] Ambler, S.: Process Patterns, Cambridge University Press, Cambridge (UK),1998.

[Ba97] Balaguer, F. et al.: Patterns for GIS Application Design. In (Hanmer, R.S.; Roberts, D. Hrsg.):Proceedings on the 4th Annual Conference on the Pattern Languages of Programming , Washington, 1997.

[BRJ99] Booch, G.; Rumbaugh, J.; Jacobsen, I.: The Unified Modelling Language User Guide, Addison Wesley Longman, Inc., Reading MA, 1999.

[Br99] Brown, W.J. et. al.: Anti-Patterns and Patterns, Wiley & Sons, New York, 1999.

[Bu96] Budinsky F..J. et. al.: Automatic code generation from design patterns. In: IBM System Journal, Vol. 35, No. 2, 1996 - Object technology , URL: http://www.research.ibm.com/journal/sj/budin/budinsky.htm, 1996.

[Bu98] Buschmann, F. et. al.: Pattern-orientierte Software-Architektur, Addison-Wesley, Bonn;Paris [u.a.], 1998.

[CA01] Computer Associates Entwicklungswerkzeuge (2001), Internetquelle URL:http://ca.com/products/, 2001.

[Cu01] Cunningham, W. : Portland Pattern Repository, Internetquelle URL: http://c2.com/ppr/index.html, 2001.

[Co94] Coplien, J.O. (1994): A generative Development-Process Pattern Language. In (Coplien, J.O.; Schmidt, D. Hrsg.): Proceedings on the 1th Annual Conference on the Pattern Languages of Programming, Reading MA, Addison-Wesley, 1997; S. 183-237.

[Er00] Eriksson, H.-E. et. al.: Business Modeling with UML - Business Patterns at work, OMG-Press, Wiley & Sons, New York, 2000.

[Ev00] Evitts, P.: A UML Pattern Language, Macmillan Technical Publishing, Indianapolis, 2000.

[Fo97] Fowler, M.: Analysis Patterns, Addison-Wesley, Menlo Park CA, 1997.

[Ga95] Erich Gamma et. al.: Design Patterns, Addison-Wesley, Reading MA, 1995.

[Hi01] Hillside Group: Pattern Hompage, URL http://hillside.net/patterns/, 2001.

[Kr99] Kruchten, P.: The Rational Unified Proecess - An Introduction,, Addison Wesley, Reading MA, 1999.

[La98] Larman, C.: Applying UML and Patterns, Prentice Hall, Upper Saddle River,1998.

[Le01] Lerena, S.: Implementierung eines Pattern-Tools, Fachhochschule Oldenburg/Ostfriesland/Wilhelmshaven und Universidad Rey Juan Carlos Mostoles/Madrid, 2001.

[Ös97] Oesterreich, B.: Objektorientierte Softwareentwicklung mit der UML, Oldenbourg-Verlag, München, 1997.

[OMG01] OMG: Unified Modeling Language Specification, Internetquelle URL: http://www.omg.org/technology/documents/unified_modeling_language.htm, 2001.

[Ra01] Rational Software Corporation, Internetquelle URL http://www.rational.com/products/rose/index.jsp, 2001.

[Ri00] Rising, L.: The Pattern Almanac 2000, Software Pattern Series, Addison Wesley, Boston, 2000.

[TI96] Texas Instruments Composer, SW Referral Catalogue, 1996.

[To01] TogetherSoft Corporation (2001), Internetquelle URL: http://www.togethersoft.com , 2001.

[Vl98] Vlissides, J.: Pattern Hatching - Design Patterns Applied, Software Pattern Series, Addison Wesley, Reading MA, 1998.

[Wu00] Wulff, A. et. al.: Ein Umweltinformationssystem für Landkreisverwaltungen. In (Greve, K. Hrsg.): 14. Int. Symposium "Informatik für den Umweltschutz", Metropolis-Verlag, Marburg, 2000; S.295-305.