Blackboard Pattern

Elke Kniesel
Seminar Software Design Patterns im Studiengang Medieninformatik
der Hochschule der Medien SS04

Kontext

Das Blackboard- Pattern hilft bei Problemen, für die keine deterministischen Lösungsstrategien bekannt sind. Durch die Zusammenfassung von spezialisierten Subsystemen soll vorhandenes Wissen dergestalt kombiniert werden, dass eine möglichst optimale Lösung erzielt werden kann. Diese Architektur ist häufig in bisher wenig erschlossenen Domänen anzutreffen.

Typische Anwendungsbereiche

- Bildverarbeitung
- Spracherkennung
- Systemüberwachung
- Knacken kryptographischer Codes
- KI

Problem

Das Blackboard- Pattern ist geeignet für Probleme:

- für die keine deterministische Lösung existieren.
- Oft existiert keine Strategie, wie Teilproblemlöser ihr Wissen zusammenführen.

Oft reicht es aus, suboptimale Lösungen durch Vergleich alternativer Lösungen zu finden.

- Eine vollständige Durchsuchung des Lösungsraums ist in akzeptabler Zeit nicht durchführbar.
- Das Anwendungsgebiet ist noch nicht vollständig verstanden.

Lösung

Die Grundidee ist eine Sammlung unabhängiger Programme, die miteinander zusammenarbeiten. Jedes Programm ist unabhängig und auf die Lösung eines bestimmten Anteils spezialisiert. Eine direkte Zusammenarbeit findet nicht statt. Eine zentrale Kontrollkomponente evaluiert den aktuellen Zustand und koordiniert die Programme. Deren Teillösungen werden kombiniert, verändert und verworfen.

Eine Moderator-Komponente bestimmt die Zusammenarbeit.

Beispiel aus der Realität

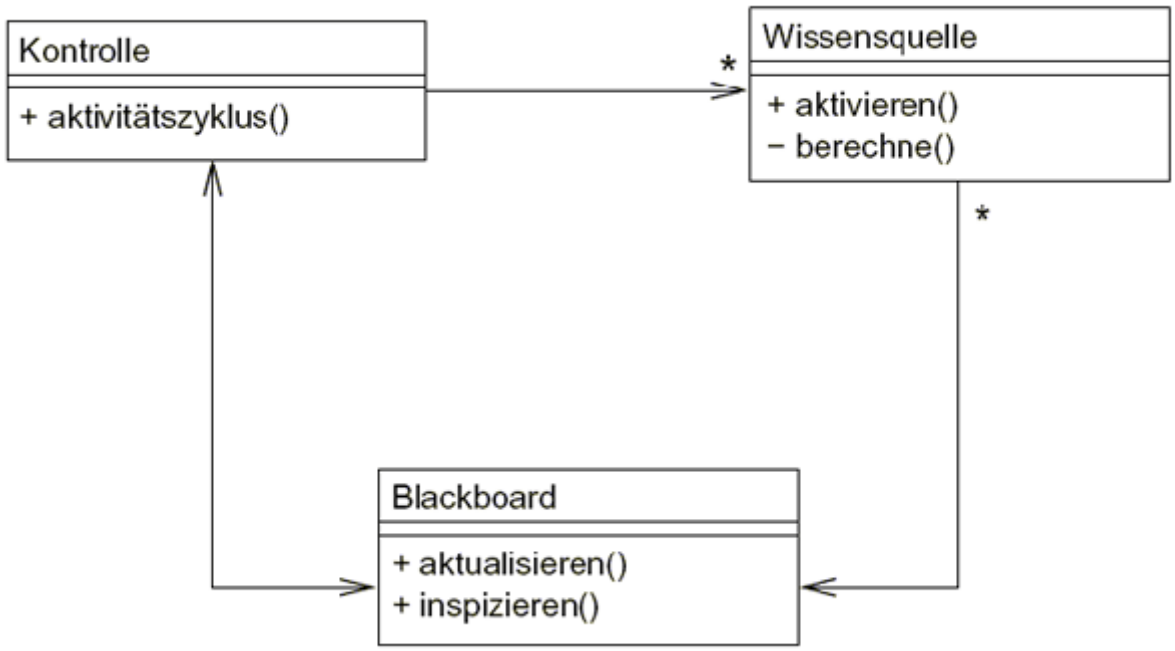
In der Realität kann man sich vorstellen, dass eine Menge von Wissenschaftlern gemeinsam an einem nicht-deterministischen Problem arbeiten. Sie sitzen an einem runden Tisch und jeder Wissenschaftler arbeitet für sich an einem Teil der Problemlösung (idealer weise auf seinem Spezialgebiet). Wenn er eine gute Lösung erzielt hat, kann er diese an das Blackboard schreiben und die anderen Wissenschaftler arbeiten auf Grundlage dieser Lösung weiter.

Ein solches Beispiel ist der *interactive room* an der Universität in Stanford. Wie auf dem Bild unten zu sehen ist, besteht der *interactive room* aus drei großen Plasma-Bildschirmen und vielen Rechnern, an denen einzelne Studenten arbeiten können. Hat ein Student ein gutes Ergebnis gefunden, kann er dieses auf den großen Plasma-Bildschirm projizieren.
Dies ist eine ganz spezielle Art von Teamwork, bei der zwar jeder Student alleine an einer Teillösung arbeitet, es profitieren jedoch alle von den im Einzelnen erzielten Ergebnissen, wenn diese an das „Blackboard“ (in diesem Fall einer der Plasma-Bildschirme) projiziert werden.

interactive room an der Universität in Stanford

Struktur des Blackboards

Das Blackboard-Pattern besteht aus drei Kompnenten: das Blackboard, mehrere Wissensquellen („knowledge sources“) und eine Kontrollkomponente.



Die Struktur des Blackboards

Das Blackboard

Das Blackboard verwaltet die zentralen Daten (Vokabular). Sowohl Elemente des Lösungsraumes, Kontrolldaten und während der Abarbeitung erstellte Lösungshypothesen werden hier abgelegt.
Die Einträge (Entries) auf dem Blackboard bilden die Menge der Zwischenergebnisse.
Das Blackboard verfügt über eine Schnittstelle, die alle Wissensquellen in die Lage versetzt, lesend und schreibend darauf zuzugreifen.

Wissensquellen

Die Wissensquellen sind eigenständige, unabhängige Teilsysteme, welche in der Lage sind, bestimmte Teilaspekte des Gesamtproblems zu lösen. In ihrer Gesamtheit modellieren sie die zu behandelnde Problemdomäne. Keine Wissensquelle ist in der Lage, das Gesamtproblem alleine zu lösen. Die Quellen kommunizieren ausschließlich über das Blackboard, also keinesfalls direkt miteinander. Daher muss jede Wissensquelle in der Lage sein, das Vokabular des Blackboards zu verstehen.
Eine Wissensquelle verfügt über einen Bedingungs- und einen Ausführungsteil (’condition- and action-part’). Der Bedingungsteil wertet den gegenwärtigen Fortschritt in der Problemlösung aus - welcher vom Blackboard reflektiert wird - und aktiviert den zugehörigen Ausführungsteil, wenn eine Teillösung zur Problemstellung beigetragen werden kann.
Wissensquellen können zur Laufzeit hinzugefügt/ausgetauscht werden.

Kontrollkomponente

Die Kontrollkomponente (Singleton-Pattern) durchläuft eine permanente Schleife, in der sie die Veränderungen auf dem Blackboard überwacht und daraufhin entscheidet, welche Aktionen als nächstes auszuführen sind. Anhand einer Applikationsstrategie plant sie die Bewertung von Wissensquellen und deren Aktivierung. Diese Strategie kann auf spezielle Quellen gestützt sein, welche ausschließlich über Kontrollwissen verfügen (und die selbst keinen direkten Beitrag zur Problemlösung liefern).

Theoretisch kann der Zustand des Blackboards zwischen zwei Extremen pendeln: es wird ein Zustand erreicht, in dem keine weiteren Wissensquellen mehr anwendbar sind (wo das System also zu keiner Lösung des Gesamtproblems gelangt), bzw. jeder Verarbeitungsschritt erzeugt viele neue Hypothesen, so dass der Lösungsraum explodiert.
Durch eine ausgezeichnete Wissensquelle, oder durch eine Funktion in der Kontrollkomponente muss bestimmt werden, wann das System zu einem Abschluss kommen sollte und was das erzielte Resultat ist. Das System kommt idealer weise dann zum Abschluss, wenn entweder eine akzeptable Hypothese gefunden wurde, oder aber die Systemresourcen erschöpft wurden.

Problemlösungszyklus (vereinfacht):

1. Eine Wissensquelle verändert das Blackboard.
2. Speicherung in der globalen Datenstruktur
3. Jede Wissensquelle gibt an, welchen Beitrag sie zu dem neuen Stand der Problemlösung beitragen kann.
4. Diese Information werden von einem Kontrollmodul genutzt den Fokus einzustellen.
    i. Falls der Fokus auf eine Wissensquelle gerichtet ist, dann wird eine Menge von Blackboard-Objekten ausgewählt, die als Kontext für die Ausführung der Wissensquelle dienen (Wissensgetriebener Ansatz: knowledge-scheduling approach)
    ii. Wenn der Fokus auf ein Blackboard-Objekt gerichtet ist, wird diejenige Wissensquelle ausgesucht, die das Objekt verarbeiten kann.
    iii. Falls Wissensquelle und BB-Objekt im Fokus stehen, wird die Wissensquelle im Kontext des BB-Objektes ausgeführt.

Implementierung

1. Problem definieren

- Anwendungsgebiet und Wissensgebiete festlegen
- System-Eingaben analysieren, Besonderheiten festlegen, Muster erkennen
- System-Ausgaben festlegen: was ist eine korrekte Lösung?
- Bewertungskriterien
- Sind Benutzerinteraktionen notwendig, wenn ja wie?

2. Lösungsraum definieren

- Wie sehen Zwischenlösungen aus?
- Was sind Teillösungen, Gesamtlösungen, ...
- Wie kann er strukturiert werden?
- Was ist eine Lösung auf oberster Ebene?
- Welche Ebenen gibt es?
- Lösungsraum in Hierarchien und Partitionen einteilen. Dies ermöglicht unabhängiges Arbeiten.

3. Den Problemlösungsprozess in Schritte aufspalten

- Wie werden aus einzelnen Lösungen Gesamtlösungen
- Wie entstehen Hypothesen ??
- Wie können Hypothesen validiert werden
- Nach welchen Kriterien kann der Lösungsraum eingeschränkt werden

4. Spezialisierte Wissensquellen definieren

- Anhand von Teilaufgaben
- Spezialisten für besondere Strategien oder Taktiken
- Einbindung menschlicher Experten

5.Vokabular des Blackboards bestimmen (Version I des Lösungsraums)

- Datenstrukturen des Blackboards und der Blackboard-Einträge festlegen
- Eventuell sind Konverter notwendig, die Datenstrukturunterschiede konvertieren

6.Kontrollkomponente spezifizieren

- Welche Wissensquelle darf wann und wo Änderungen vornehmen.
- Ziel ist es, eine Hypothese zu konstruieren, die als Ergebnis akzeptabel ist.
- Hypothesen können bewertet werden. Nur glaubwürdige bleiben bestehen
- Aus den anwendbaren WQ muss eine (mehrere) ausgewählt werden
- zufällig (meist sehr ineffizient)
- Problemlösungsstrategien (Induktion, Wiederspruch, Abstandsmasse,...)
- Heuristiken, Expertenwissen.
- Priorisierung anhand der Bedingungen und erwarteten Ergebnisse oder anhand der Ebene, auf der die Wissensquelle arbeitet

Beurteilungskriterien für die Anwendung und Auswahl von Wissensquellen

Welche Änderungen entstehen am Blackboard

- Werden neue Wissensquellen anwendbar
- Werden keine neuen Wissensquellen anwendbar

Eventuell lassen sich über die Art der Änderung Wissensquellen gruppieren

- Sollten immer angewendet werden (im allgemeinen nützlich)
- Unkritisch (Die Existenz der Lösung bleibt unberührt)
- Nur mit Vorsicht (eventuell lässt sich die Lösung nun nicht mehr finden)

Fokussierungsmechanismen

- Teilergebnisse, an denen weitergearbeitet werden soll
- Wissensquellen, die anderen vorgezogen werden.
- Warteschlangen anwendbarer Wissensquellen

Implementierung der Wissensquellen

- Aufspalten in Bedingungs- und Aktionsteil
- Keine Annahmen über andere Wissensquellen verwenden Unabhängigkeit und Austauschbarkeit bleibt erhalten
- Eventuell kommen verschiedenen Technologien zum Einsatz
    - ➔ Regelbasierte Systeme
    - ➔ Neuronale Netze
    - ➔ Entscheidungstabellen
    - ➔ Gewöhnliche Funktion
- oder auch verschiedene Programmiersprachen (Prolog, Lisp, C#)
- Andere Paradigmen erfordern gegebenenfalls eine Kapselung (Facade-Pattern)
- Wissensquellen sind oft große Subsysteme, die ihrerseits nach diversen Architektur- und Entwurfsmustern organisiert sind, Hypothesen können bewertet werden.

Bekannte Anwendungen

HEARSAY-II

Das Spracherkennungssystem Hearsay II (hear say) ist das erste bekannte Blackboard-System. Es wurde Mitte der 1970er Jahre entwickelt.
Das System hatte die Aufgabe, natürlich-sprachliche Anfragen an eine Literaturdatenbank zu beantworten. Die Eingaben an das System bestanden aus akustischen Signalen, welche einer semantischen Interpretation unterzogen wurden und anschließend in eine Datenbankanfrage übersetzt wurden.

Bei HEARSAY-II manifestierte sich eine Abstraktionshierarchie, bei der die einzelnen Zwischenlösungen in Form von akustischen Signalen, akustisch-phonetischen Segmenten, Phonemen, Silben, Wörtern und Ausdrücken festgehalten wurden.

Die während des Problemlösungsprozesses erstellten Hypothesen wurden ebenfalls in der Blackboard-Komponente gespeichert.

Hearsay II hat Komponenten zur

- Segmentierung von phonetischen Einheiten
- Silbenerzeugung
- Worterzeugung
- Phrasenerzeugung

Bei jeder Spracherkennungsaufgabe wurde das Problem in die oben beschriebenen Teilprobleme mit den verschiedenen Abstraktionsebenen zerlegt, wobei das Schema der Abstraktion aber stets das Gleiche geblieben ist.

Die Knowledge Sources bestanden bei HEARSAY-II aus unabhängigen Subsystemen, welche in der Lage waren, spezialisierte Teilaufgaben des Gesamtproblems selbständig zu lösen. Beispielsweise konnten die Knowledge Sources akkustisch-phonetische Segmente definieren, sie konnten Phoneme, Silben, Wörter und Ausdrücke erzeugen.

In HEARSAY-II verfügte jede Knowledge Source über einen Stimulus-Frame, welcher eine aktuelle Zustandsbeschreibung des Blackboard enthielt, sodass bei dessen Vorliegen die jeweilige Knowledge Source mit einem zugehörigen Response-Frame reagierte.

Bei HEARSAY-II (Spracherkennungsproblem) gehörte zum Anwendungskontext, dass es sich um eine bisher wenig durchdrungene Anwendungsdomäne (’immature domain’) handelte, für die obendrein keine bekannten deterministischen Problemlösungsstrategien vorhanden waren. Der mögliche Lösungsraum war bei dieser Anwendung dadurch charakterisiert, dass er riesengroß war und mithin ungeeignet, um in überschaubarer Zeit durchsucht zu werden. Betrachtet man Sätze, welche bis zu 10 Wörter Länge umfassen können, so ergibt sich bei einem Vokabular bestehend aus 1000 Wörtern bereits ein Suchraum aus 1000^{10} möglichen Wort-Permutationen.

HASP

HASP wurde für militärische Zwecke als System zu Erkennung von feindlichen U-Booten aufgrund von Sonarsignalen entwickelt. Bei dieser Implementierung des Blackboard-Patterns werden unterschiedliche Objektklassen auf verschiedenen Abstraktionsebenen gespeichert, außerdem ist kein separates Kontrollelement vorhanden. Die Lösungen werden auf dem Blackboard erzeugt und Lösungsgruppen werden zu Wissensinseln zusammengefasst („island driving“). Die Wissensquellen sind hierarchisch in verschiedene Arten gegliedert, die zum Teil schon Aufgaben der Kontrollkomponente implementiert haben.

CRYSALIS

Verarbeitung der 3D-Struktur von Proteinmolekülen.

SUS

Software Understanding System. Suche nach wiederverwendbaren Teilen basierend auf einer Muster-DB.

Rover

Ein autonomer, mobiler Roboter, der zur Aufspürung, Erkennung und Klassifikation von hochgiftigen Flüssigkeiten verwendet wird. Dabei handelt es sich um eine Umsetzung des Prinzips der „virtuellen Blackboard-Architektur“.
An „Rover“ sind viele Instrumente und Messgeräte montiert, die zur Erfassung der Umgebung benötigt werden. Die gesamte Struktur wird vom autonomen Roboterkontrollsystem überwacht und hat eine Funkverbindung zur Leitstelle. Während seiner Mission wird „Rover“ eine Vielzahl von situationsbedingten Operationen durchführen müssen. Zum Beispiel muss er sich in unbekanntem Gelände bewegen können und er muss mit Situationen zurechtkommen, die er zum Beginn seiner Mission nicht vorhersehen konnte.

Blackboard – Varianten

Virtuelles Blackboard-Modell

Das virtuelle Blackboard-Modell stellt die Blackboard-Architektur in ihrer ursprünglichen Konzeption dar, nämlich als Gruppe von Experten, die gemeinsam versuchen ein Problem zu lösen. Jede Kontrollstation entspricht einem Experten. Alle Wissensquellen im gesamten System sehen nur das gemeinsame Blackboard, obwohl das Blackboard an sich auf die kooperierenden Instanzen verteilt ist – daher auch der Name virtuelles Blackboard. Die einzelnen Kontrollstation sind für Teile des gemeinsamen Blackboards zuständig.

Die Aufgabe der Synchronisation wird dadurch gelöst, dass sie auf höherer Ebene stattfindet als auf Höhe der Wissensquellen. Damit lässt sich eine gleichzeitige Bearbeitung mehrerer Aufgaben durch Kontrolle der Interaktionen zwischen Wissensquellen und Blackboard implementieren.

Ein weiteres Problem ist der gegebenenfalls unvertretbar hohe Aufwand für das Message Passing. Da im virtuellen Blackboard beliebige Wissensquellen alle Datensätze einsehen und bearbeiten können, ist ein schnelles Reagieren auf Änderungen möglich und das Aufwandsproblem für Message Passing entfällt weitgehend.

Ein anderes Problem eines globalen Blackboards ist die Möglichkeit, dass das Blackboard zum Flaschenhals werden kann. Wird das Blackboard aber derart in Sub-blackboards aufgeteilt, dass die einzelnen Wissensquellen vorwiegend nur auf lokale Zugriffe angewiesen sind, kann dieser Engpass behoben werden. Eine solche Aufteilung ist allerdings nicht einfach durchzuführen, außerdem kann sich die Güte dieser Aufteilung zeitabhängig ändern (z.B. durch geändertes Zugriffsverhalten).

Die Arbeitsweise wird im Folgenden an einem Beispiel erläutert:
Eine aktiv gewordene Wissensquelle ändere ein bestimmtes Sub-blackboard ab.
Sofern dies den Zuständigkeitsbereich der eigenen Kontrollstation betrifft, passiert weiter nichts. Falls aber das zu ändernde Sub-blackboard im Zuständigkeitsbereich einer anderen Kontrollstation liegt, muss eine Meldung an die dafür zuständige Kontrollstation abgeschickt werden, so dass diese die Modifikation vornehmen kann.
Änderungen der Hierarchie werden in der zugehörigen Instanz festgehalten, die sich auf dem Niveau befinden, auf dem die Änderung vorgenommen wurde.
Falls die Änderung für eine Instanz von Interesse ist, die nicht diejenige ist, welche die Änderung verursacht hat, wird die betreffende fremde Kontrollstation durch eine Nachricht informiert.

Von einer Instanz, welche über eine Änderung benachrichtigt wurde sind zwei weitere Aktionen durchzuführen:
Die Benachrichtigung wird benutzt, um eine Kopie der Ereignisstruktur anzufertigen.
Dies reaktiviert gegebenenfalls Wissensquellen, die in einer Warteschlange blockiert waren, weil sie auf das Eintreffen eines Ereignisses dieser Art warteten.

Die Fähigkeit, dass Wissensquellen in jedem Sub-blackboard Änderungen vornehmen und in gleicher Weise direkt auf Änderungen in einem anderen Sub-blackboard reagieren können, ohne dass die Wissensquellen oder sonstige Hierarchien geändert werden müssen, ist das besondere Merkmal des virtuellen Blackboard-Patterns.

Parallele Blackboard-Systeme

Parallele Blackboard-Systeme wie zum Beispiel CAGE und POLIGON sind Erweiterungen von AGE (An Attempt to Generalise), die eine Parallelverarbeitung erlauben.

CAGE

Die Zielmaschine ist eine Multiprozessormaschine mit gemeinsamen Hauptspeicher (shared Memory)

POLIGON

Die Zielmaschine ist ein verteiltes System ohne gemeinsamen Hauptspeicher (message passing).

Parallelisierung:

- Einzelne Wissensquellen können parallel ausgeführt werden und mehrere Wissensquellen können zueinander parallel ausgeführt werden.
- Pipeline - Folgen von Wissensquellen als Pipelining
- Datenparallelität - An einzelnen Lösungskomponenten kann parallel gearbeitet werden (z.B. kann das BB in mehrere Ebenen unterteilt werden, die parallel bearbeitet werden können)
- Kontrollparallelität

Probleme der Parallelisierung

- Konkurrenz um Ressourcen, bei shared memory, um Speicherzugriff, bei verteilten Systemen z.B. Kommunikationsmittel
- Prozess-Erzeugung - um Teilprobleme parallel ablaufen zu lassen müssen sie durch einen eigenen Prozess ausgeführt werden.

Wichtige Konzepte für die Parallelisierung

- Definition von Atomaren Operationen
- kritische Bereiche ununterbrechbar ablaufende, komplexere Programmteile
- Definition von Synchronisationsprimitiven
- Sperrmechanismen
- Pipeline - eine Menge von Operationen, die parallel ausgeführt werden können aber sequentiell voneinander abhängen (vgl. Montagestrasse)

Repository-Systeme

Repository-Systeme sind Blackboard-Systeme ohne interne Kontrolle. Die zentrale Datenstruktur heißt jetzt **Repository**.
Die Kontrolle liegt bei externen Anwendern und Anwendungen, wie zum Beispiel bei Datenbanksystemen oder Programmierumgebungen.

Klassifikation nach Buschmann et al.

Bei Buschmann basieren die Architekturstile auf der Überlegung, in welchem Stadium des Entwurfs oder auf welchem Wissenstand man sich befindet. Es gibt 4 Stile:

(a) From Mud to Structure
Zu Beginn eines Entwurfs nehmen wir im optimistischen Fall an, das die Anforderungen an unser System wohl überlegt und stabil sind. Es geht darum, den unüberschaubaren „ball of mud“ in eine arbeitende Struktur zu bringen und dabei alle Aspekte der Anforderungen zu berücksichtigen. Das Blackboard-Pattern kann hier zugeordnet werden.

(b) Distributed Systems
Zwei maßgebliche Trends führten zur Entstehung dieses Architekturstils:

1. Computersysteme mit mehreren Prozessoren und unterschiedlichsten Betriebssystemen finden zunehmend Einsatz, auch in kleineren Firmen.
2. Das Verbinden von heterogenen Rechnersystemen über ein gemeinsames LAN wird alltäglich.

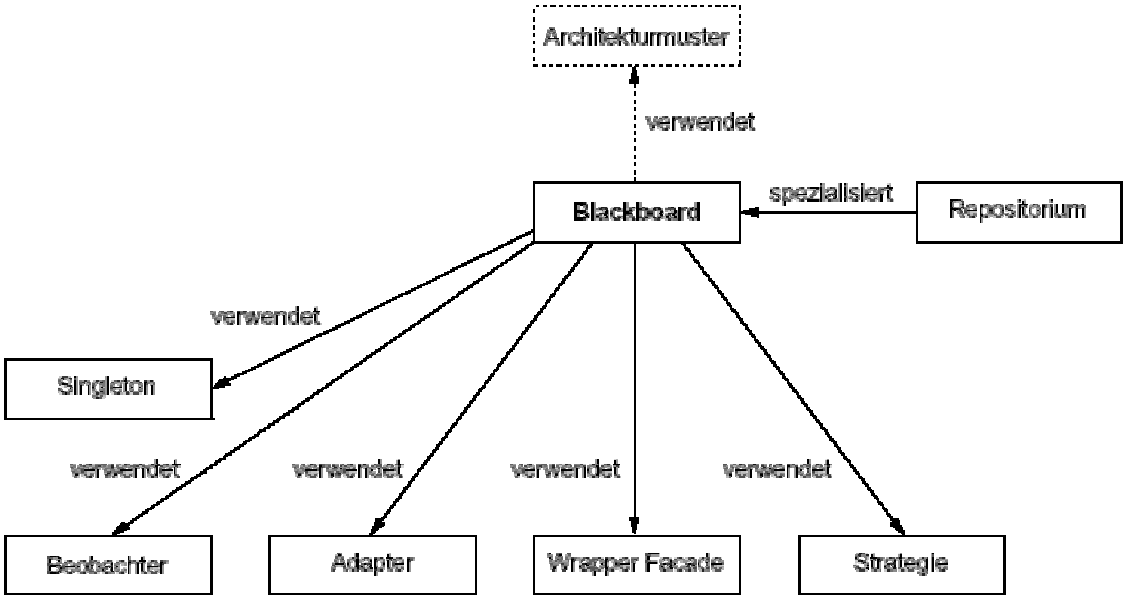
(c) Adaptable Systems
Anpassbare Systeme müssen die Eigenschaft besitzen, neue Versionen zu unterstützen, auf neuen Betriebssystemen zu laufen oder neue Standards von Hardware oder Software übernehmen zu können. Während des Systementwurfs und der Implementierung fordert der Auftraggeber immer wieder neue Funktionalität oder ändert seine bestehenden Anforderungen, und dies meist in einer späten Phase des Projektes und überraschend. Aus diesem Grund muss der Entwurf des Systems auf Wandel und Zuwachs ausgelegt werden und die Architekturen dieses Stils sollen das in den Vordergrund stellen.

(d) Interactive Systems
Dieser Stil betrifft alle Systeme die einen hohen Grad an Benutzerinteraktion mit aufwendigen grafischen Ausgaben und intelligenten Hilfesystemen haben. Ziel dieser Systeme ist es ein einheitlichen und leicht zu erlernenden Zugang zu der Funktionalität des Systems zu geben, so dass schnell Ergebnisse erzielt werden können.

	Architektural Patterns	Design Patterns	Idioms
From Mud to Structure	Layers Pipes and Filters **Blackboard**		
Distributed Systems	Broker Pipes and Filters Microkernel		
Interactive Systems	MVC PAC		
Adaptable Systems	Microkernel Reflection		
Structural Decomposition		Whole-Part	

Beziehungen zu anderen Mustern

Wie aus der unten stehenden Grafik ersichtlich wird, verwendet und erweitert das Blackboard-Pattern verschiedene andere Basis-Design-Patterns.
Das Blackboard-Pattern ist eine Erweiterung des Repository-Patterns, da es im Gegensatz zum Repository-Pattern eine zentrale Kontrollstruktur besitzt.
In den einzelnen Komponenten des Blackboard-Patterns können verschiedene Basis-Patterns entdeckt werden: das Blackboard an sich wird als Singleton implementiert. Die Wissensquellen, die einfach austauschbar sein müssen, werden meist in das Facade-Pattern gekapselt. Es gibt genau eine zentrale Kontrollkomponente (Singelton-Pattern), die das Blackboard zyklisch beobachtet (Observer-Pattern) und das Moderator-Pattern zur Bestimmung der Zusammenarbeit verwendet. Die Daten, die auf dem Blackboard gespeichert werden, sind zwar Anwendungsspezifisch, werden jedoch oft mit Hilfe von Composite Messages und Mementos gekapselt.



Fazit

Nachteile

Ein wesentlicher Nachteil im Einsatz von Blackboard-Systemen liegt darin, dass Berechnungen oft nicht nachvollziehbar sind, daher ist es schwierig diese Systeme zu testen. Darüber hinaus bieten Blackboard-Systeme keine Lösungsgarantie. In der Regel wird nur ein gewisser Prozentsatz der Aufgaben gelöst. Da ein Großteil der Schritte falsch oder unnötig sind und wieder verworfen werden müssen, hält sich die Effizienz einer Blackboard-Anwendung in Grenzen.
Auch der Entwicklungsaufwand ist sehr hoch, da auf einem schlecht strukturierten Anwendungsgebiet operiert wird. Außerdem wird oft „Trial-and-error“ bei der Definition des Vokabulars, der Kontrollstrategie und der Wissensquellen verwendet.

Kontrollstrategien sind oft schwierig zu finden, d.h. in der Regel bedarf es eines oder mehrerer Experten, um auch nur annähernd geeignete Heuristiken zu finden. Bei ungeeigneter Auswahl der Kontrollstrategie kann das System in Sackgassen laufen (wenn keine Wissensquelle mehr anwendbar ist) oder das System kann explodieren, wenn die Wissensquellen immer neue Hypothesen erzeugen und immer mehr Wissensquellen anwendbar werden.

Daraus folgen komplexe, heuristische Kontrollalgorithmen, die oft interaktiv mit einem Benutzer interagieren.

Vorteile

Die Vorteile des Blackboard-Patterns liegen darin, dass man komplexe Lösungsstrategien entwickeln kann, für die kein Lösungsmechanismus existiert.
Falls keine geschlossene Lösung existiert und eine vollständig Suche unmöglich ist (z.B. in der Kryptographie) hat man die Möglichkeit, zu experimentieren.

Die Änderbarkeit und Wartbarkeit einer Blackboard-Anwendung wird durch die strikte Trennung zwischen Wissensquellen und Kontrolle (evtl. auch zu den Datenstrukturen optimal unterstützt. Wissensquellen sind individuelle Spezialisten und können beliebig ausgetauscht und widerverwendet werden.

Alles in Allem sind Blackboard-Systeme sehr Fehlertolerant und stabil, z.B. könnten bei Hearsay II die Eingabedaten rauschen und die Anwendung gibt trotzdem mit hoher Wahrscheinlichkeit ein gutes Ergebnis aus. Daraus kann man schließen, dass Blackboard-Systeme vor allem für das Experimentieren in schlecht erschlossenen Domänen geeignet ist.

Quellen

Autonome mobile Roboter und deren Steuerung, LuFG Informatik V, Prof. G. Lakemeyer Ph. D., WS 1999/2000

http://atbruegge27.informatik.tu-muenchen.de/teaching/ss02/muster/Vorlesung4_Architektur.pdf

Objektorientierte Architekturen, Frameworks und Architekturmuster, Fabian Wallwitz, Universität Stuttgart
(http://cybop.berlios.de/papers/2003_design_patterns_for_backup/studienarbeit.pdf)

A Distributed Computing Pattern Language, Frank Buschmann, Siemens AG, Corporate Technology, Software and Engineering

http://www.vico.org/pages/PatronsDisseny/Pattern%20Blackboard

http://chat.carleton.ca/~narthorn/project/patterns/BlackboardPattern-display.html

http://www.openloop.com/softwareEngineering/patterns/architecturePattern/arch_Blackboard.htm
http://www.agcs.com/supportv2/techpapers/patterns/papers/tutnotes/sld037.htm

ftp://ftp.teccomm.les.inf.puc-rio.br/pub/docs/05_otavio.pdf

http://www.cs.bilkent.edu.tr/~bedir/CS411/Slides/ArchitecturalPatternsOverview.pdf