

# DYNAQUEST als Basis zur dynamischen Informationsfusion

Marco Grawunder, Frank Köster, Hans-Jürgen Appelrath  
Department für Informatik,  
Carl-von-Ossietzky Universität Oldenburg  
{grawunder|koester|appelrath}@informatik.uni-oldenburg.de

**Abstract:** Klassische Ansätze der Datenintegration lassen sich im Kontext dynamischer Informationsfusion mit einer hohen Anzahl verfügbarer Quellen nicht problemlos einsetzen. Es müssen zunächst potentiell relevante Quellen identifiziert, hieraus zur Anfrageausführungszeit eine Auswahl getroffen und diese ad-hoc integriert werden. Dabei ist der Quellenauswahlprozess von den jeweils verfolgten Anwendungszielen abhängig, da u.U. unterschiedliche Anforderungen bzgl. verschiedener Aspekte der Datenqualität erfüllt werden müssen.

Zur Wahl und Integration von Quellen sind zunächst geeignete Quellenbeschreibungen notwendig. Dabei ist es wünschenswert, Beschreibungen automatisiert ableiten zu können, wobei insbesondere die inhaltliche Distanz zwischen Beschreibung und den tatsächlichen Gegebenheiten der Quelle möglichst gering sein muss. Da dies nicht immer garantiert werden kann, ist es notwendig, die Ausführung einer Anfrage zu überwachen und im Bedarfsfall einzuschreiten. Ein solches Monitoring ist darüber hinaus auch deshalb wichtig, da die Umgebung der Quellen u.U. eine hohe Dynamik aufweist und die Quellen selbst eine hohe Autonomie besitzen, sodass Quellenverhalten i.d.R. nicht zweifelsfrei prognostizierbar ist.

In diesem Artikel wird der DYNAQUEST-Ansatz vorgestellt, worin die Quellenbeschreibung sowie Quellenauswahl, -beobachtung und Recovery-Mechanismen als Grundlage einer Anfrageverarbeitung untersucht werden. DYNAQUEST stellt damit einen wichtigen Baustein zur dynamischen Informationsfusion dar.

## 1 Einleitung

Die dynamische Fusion von Informationen aus unterschiedlichen Quellen spielt in vielen Anwendungen eine Rolle. Angefangen von solchen Systemen, die beispielsweise wirtschaftliche Daten, wie Aktienkurse, bereitstellen, um Entscheidungen über den Aktienkauf oder -verkauf treffen zu können, bis hin zu komplexen Systemen, die etwa im Rahmen des Katastrophenmanagements zeitnah die Integration unterschiedlicher Datenquellen, wie Pegelstands- oder Fließgeschwindigkeitssensoren, mit geografischen Daten und Populationsdaten, ermöglichen sollen.

Die automatische Integration verschiedener, unbekannter Quellen ist auf Grund der in praktischen Anwendungssituationen meist hohen Komplexität des Integrationsproblems nicht ohne weiteres durchführbar. Insbesondere die Schnittstellen der Quellen, die automatisiert nicht zuverlässig ermittelt werden können, müssen bekannt sein. Aus diesem Grund ist es notwendig, Quellenbeschreibungen zu etablieren, anhand derer entschieden werden

kann, ob eine Quelle für eine Anfrage von Belang ist, und wie diese zu integrieren ist. Zur Reduktion des Beschreibungs- und Integrationsaufwandes, ist es essentiell, Quellenbeschreibungen effizient generier- und wartbar zu gestalten. Dies vereinfacht zudem den Schritt, solche Beschreibungen im Rahmen einer automatisierten Adaption verarbeiten zu können.

Die Korrektheit und die Vollständigkeit von Quellenbeschreibungen sind elementare Voraussetzungen für einen effektiven Einsatz im Rahmen dynamischer Umgebungen. Des weiteren können Situationen eintreten, die nur schwer oder gar nicht antizipierbar sind. Ein im Kontext der dynamischen Informationsfusion eingesetztes Anfrageverarbeitungssystem muss aus diesem Grund in der Lage sein, auch zur Ausführungszeit einer Anfrage Probleme zu erkennen und behandeln zu können.

Das DYNAQUEST-Framework [Gr03] erlaubt die effiziente Entwicklung von Systemen zur dynamischen Informationsfusion. Das Framework umfasst ein Quellen- und Anfragebeschreibungsformat SDF [GK03], welches als Grundlage zur Integration passiver und aktiver Quellen dienen kann, einem auf SDF basierendem filtergestützten Auswahlprozess, der Quellen nach strukturellen, inhaltlichen und qualitativen Gesichtspunkten selektiert und ordnet, sowie einer Anfrageausführungsumgebung, die Techniken der dynamischen und adaptiven Anfrageverarbeitung zur fehlertoleranten Ausführung zusammenfasst. Der Schwerpunkt dieses Artikels liegt auf der Beschreibung dieser Anfrageausführungsumgebung.

## 2 Dynamische und Adaptive Anfrageverarbeitung in DYNAQUEST

Während die klassische Anfrageverarbeitung typischerweise in die drei sequentiell ausgeführten Phasen Übersetzung, Optimierung und Ausführung untergliedert wird [Gr93], kennzeichnen sich dynamische Techniken durch die Verschränkung der Ausführungs- und Optimierungsphase [BFMV00]. Dies führt beispielsweise dazu, dass eine Anfrage, die sich bereits im Status der Ausführung befindet, im Falle von Laufzeitproblemen wieder an die Optimierungskomponente zurückgegeben werden kann (vgl. Abb. 1).

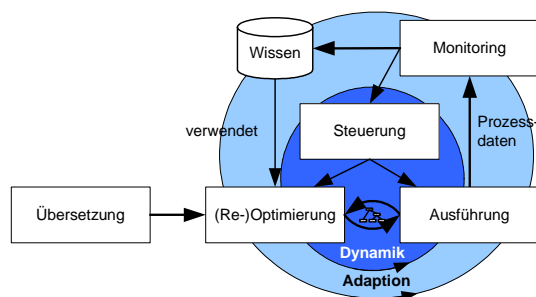


Abbildung 1: Dynamik und Adaption in der Anfrageverarbeitung

Um die Entscheidung zu treffen, ob eine Anfrage normal ausgeführt werden kann oder ob eine Intervention seitens des Anfragesystems notwendig ist, sind Daten über die Anfrageausführung von entscheidender Bedeutung. Diese Prozessdaten werden von einer Monitoring-Komponente entgegengenommen und an eine Steuerungskomponente weitergereicht. Die Steuerungskomponente entscheidet dann, ob in den aktuellen Ausführungsprozess eingegriffen wird. Dieser Kreislauf definiert den Kernprozess der dynamische Anfrageverarbeitung, da hier zur Laufzeit eine Anpassung der Verarbeitung stattfinden kann.

Aus den Prozessdaten können zudem weitere Informationen gewonnen werden, wie z.B. das Verhalten von Quellen zu bestimmten Zeiten oder der von einer Quelle gelieferte Ausschnitt ihrer Extension. Diese Information kann in zukünftigen Anfrageverarbeitungsprozessen von der Optimierungskomponente genutzt werden, um sukzessive die Leistung des Anfragesystems zu optimieren. Dieser Prozess, der anfrageübergreifend agiert, kann mit dem Stichwort adaptive Anfrageverarbeitung charakterisiert werden [HFC<sup>+</sup>00].

In Abb. 2 findet sich die Realisierung der oben beschriebenen Prozesse in DYNAQUEST. Die einzelnen Elemente der Architektur werden im Folgenden anhand eines typischen Anfrageverarbeitungsprozesses in DYNAQUEST beschrieben.

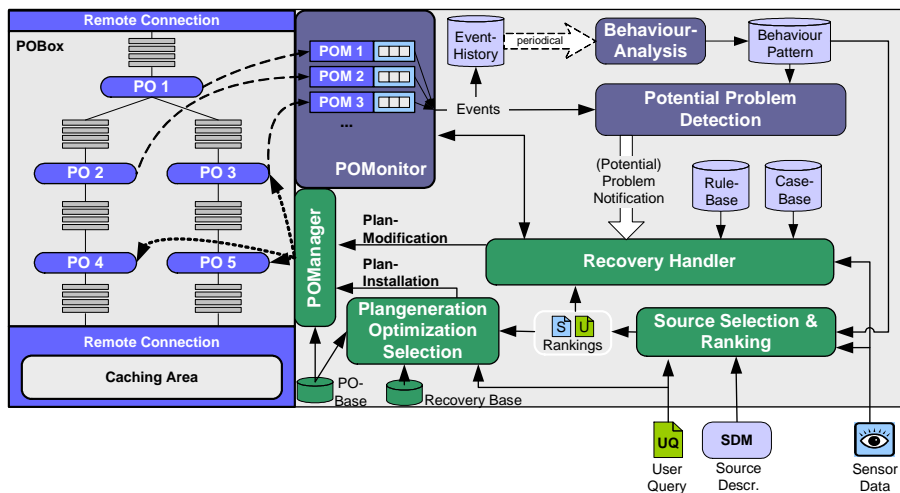


Abbildung 2: Eine Architektur für dynamische und adaptive Anfrageverarbeitung

Im ersten Schritt wird basierend auf der Anfrage des Nutzers (diese definiert intensionale, extensionale und qualitative Anforderungen an das Ergebnis), der Menge der verfügbaren Quellenbeschreibungen, Informationen über vergangenes Verhalten von Quellen, sowie aktuellen Sensordaten, z.B. über die aktuelle Auslastung von Quellen, eine Wahl potentiell relevanter Quellen durchgeführt. Diese Quellenwahl erfolgt in einem mehrstufigen Filter- und Rankingprozess, der nacheinander die Quellen bezüglich ihrer intensionalen, extensionalen und qualitativen Relevanz anhand eines Abstandsmaßes beurteilt. Das Ergebnis besteht aus einer Liste von Quellen bzw. Quellengruppen, die nach ihrer Eignung bzgl. der obigen Aspekte geordnet ist.

Aus den Quellenordnungen und der Nutzeranfrage werden mögliche Anfragepläne generiert, optimiert und auf Basis eines Kostenmodells selektiert. Um Problemen bei der Ausführung zu begegnen, wie bspw. dem Ausfall einer Quelle, können zusätzliche unterstützende Maßnahmen in den Anfrageplan integriert werden. Diese Maßnahmen umfassen beispielsweise die Protokollierung der von einer Quelle gelieferten Datensätze (z.B. anhand eindeutiger Kennungen), um bei einem Quellenausfall, die bereits verarbeiteten Daten identifizieren zu können. Der schließlich generierte Plan wird mit Hilfe des POManagers, einer Komponente zur Verwaltung von Anfrageplänen, in einer Ausführungsumgebung ausgeführt.

Diese POBox genannte Ausführungsumgebung erfüllt im Wesentlichen den Zweck der Steuerung und Überwachung eines Anfrageplans. Die Ausführung der Anfrage folgt dem bekannten Open-Next-Close-Protokoll [Gr93]. Die Realisierung weicht jedoch von der typischerweise eingesetzten auftragsgetriebenen Verarbeitung ab. Die in DYNAQUEST eingesetzten Planoperatoren produzieren Ausgaben, solange in ihrem Eingabebereich Daten vorliegen und in ihrem Ausgabebereich Platz zur Ablage vorhanden ist. Sie setzen damit eine datengetriebene Verarbeitung um. Der Vorteil dieses Ansatzes liegt darin, dass schwankende Datenraten von Quellen („Bursty Arrival“) [UFA98] auf diese Weise sehr gut ausgeglichen werden können. Die Größe der Ein- bzw. Ausgabebereiche kann zur Laufzeit den aktuellen Anforderungen angepasst werden.

Ein Planoperator generiert neben den angefragten Daten auch Mitteilungen über Zustandsänderungen und Fehlersituationen. Von jedem Operator ist jederzeit bekannt, ob er gerade Daten produziert, auf Eingabedaten wartet oder auf Platz in seinem Ausgabebereich. Auf diese Weise lässt sich ein sehr genaues Bild über den aktuellen Zustand der Planausführung gewinnen. Die Mitteilungen der Planoperatoren werden sowohl in einer Datenbank gesichert, als auch an eine Komponente weitergereicht, die hieraus Fehler- oder Problemsituationen zu erkennen versucht. Fehlersituationen können dabei explizit (ein Operator meldet einen Fehler) oder implizit (der Ausfall einer Quelle wird durch ein Timeout registriert) sein. Falls die Problemerkennungskomponente eine mögliche Problemsituation identifiziert wird der Recovery-Handler informiert, der weitere Aktionen veranlasst.

Das Verhalten des Recovery-Handlers wird mit Hilfe von ECA-Regeln und Fällen eines Case-Based-Reasoning-Systems<sup>1</sup> spezifiziert. Falls Probleme erkannt werden, entscheidet der Recovery-Handler ob und welche Aktionen angestoßen werden müssen. Diese Aktionen werden über den POManager umgesetzt und können beispielsweise den Austausch von Quellen, die zeitweise Stilllegung eines Planoperators oder die Anpassung von Ein- bzw. Ausgabebereichen beinhalten. Damit unterstützt der beschriebene Prozess die Integration von Techniken der dynamischen Anfrageverarbeitung in die Ausführungsumgebung.

Die während der Überwachung abgelegten Informationen über das Planoperatorverhalten werden periodisch analysiert und in Form von Verhaltensmustern verdichtet. Diese Verhaltensmuster können beispielsweise festlegen, dass bestimmte Quellen zu bestimmten Zeiten nur schwer oder gar nicht erreichbar sind. Diese Muster, die entweder durch einfache statistische Verfahren oder durch Zeitreihenanalysen gewonnen werden, dienen in nachfolgenden Läufen zum einen dazu, potentielle Probleme zu erkennen und zum ande-

---

<sup>1</sup>Creek, Universität Trondheim, <http://dionysus.idi.ntnu.no/creekdistro/>

ren die Wahl von Quellen über die Zeit effektiver zu gestalten. Auf diese Weise werden Techniken der adaptiven Anfrageverarbeitung in DYNAQUEST zur Verfügung gestellt. Als Grundlage werden hierzu deskriptive statistische Methoden oder z.Zt. auch neuronale Netze untersucht.

### 3 Fazit

In diesem Artikel wurde DYNAQUEST als Grundlage zur Realisierung der Informationsfusion im Kontext dynamischer Umgebungen dargestellt. DYNAQUEST bietet durch seinen Framework-Charakter wesentliche Eigenschaften, wie sie in diesem Anwendungskontext relevant sind, und besitzt darüber hinaus einen anderen Fokus als andere Ansätze in diesem Bereich, wie beispielsweise [Iv02] oder [BFMV00]. Spezielle dort vorgeschlagene Techniken zur dynamischen und adaptiven Anfrageverarbeitung lassen sich in DYNAQUEST integrieren.

Die Erweiterbarkeit an neue Anforderungen und die einfache Anpassbarkeit an spezielle Einsatzszenarien durch die flexible Verarbeitungsarchitektur stellen wesentliche Eigenschaften von DYNAQUEST dar. DYNAQUEST bildet damit einen wichtigen Baustein, um gerade auch im Bereich der dynamischen Informationsfusion die Entwicklung und Umsetzung komplexer Systeme zu unterstützen.

### Literatur

- [BFMV00] Bouganim, L., Fabret, F., Mohan, C., und Valduriez, P.: Dynamic query scheduling in data integration systems. In: *ICDE 2000*. S. 425–434. San Diego. 2000. IEEE.
- [GK03] Grawunder, M. und Köster, F.: The dynaquest-framework for dynamic and adaptive source selection. In: *Proceedings of the 2003 International Symposium on Collaborative Technologies and Systems CTS 2003 Special Session on Advanced Information Systems*. 2003.
- [Gr93] Graefe, G.: Query evaluation techniques for large databases. *ACM Computing Surveys*. 25(2):73–170. 1993.
- [Gr03] Grawunder, M.: Technical report of the 20th british national conference on databases, bncod 20, coventry, uk, july 15-17, 2003, poster papers. In: James, A. E. und Younas, M. (Hrsg.), *BNCOD Posters*. S. 27–29. Coventry University. 2003.
- [HFC<sup>+</sup>00] Hellerstein, J. M., Franklin, M. J., Chandrasekaran, S., Deshpande, A., Hildrum, K., Madden, S., Raman, V., und Shah, M. A.: Adaptive query processing: Technology in evolution. *Bulletin of the Technical Committee on Data Engineering*. 23(2):7–18. 2000.
- [Iv02] Ives, Z. G.: *Efficient Query Processing for Data Integration*. PhD thesis. University of Washington. 2002.
- [UFA98] Urhan, T., Franklin, M. J., und Amsaleg, L.: Cost-based query scrambling for initial delays. In: Haas, L. M. und Tiwary, A. (Hrsg.), *SIGMOD*. S. 130–141. Seattle, Washington. 1998. ACM.