HMD (2015) 52:562–571
DOI 10.1365/s40702-015-0149-x

Ereignisverarbeitung zur Flexiblen Dynamischen Informationsverarbeitung in Smart Cities

Marc Schaaf · Gwendolin Wilke

Eingegangen: 30. Januar 2015 / Angenommen: 30. April 2015 / Online publiziert: 17. Juni 2015

Zusammenfassung Ein Pfeiler des Smart City Konzeptes ist die Verfügbarkeit von großen Datenmengen, sowie die Fähigkeit diese flexibel und intelligent in Nahe-Echtzeit analysieren und verarbeiten zu können. Als technologische Grundlage für diese oft unterschätze Datenanalyse und -verarbeitungsfähigkeit werden vielfach Technologien aus dem Bereich der Ereignisverarbeitung genannt. In diesem Kapitel werden zentrale Konzepte der Ereignisverarbeitung kurz vorgestellt und deren Nutzen, Grenzen und zukünftige Entwicklungen anhand eines Szenarios aus dem Bereich ad-hoc Car-Sharings beschrieben.

Schlüsselwörter Ereignisverarbeitung · Complex Event Processing · Big Data · Zeitnahe Verarbeitung · Flexible Verarbeitung

Abstract A pillar of the Smart City concept is the availability of big data, together with the ability to process and analyse it flexibly and intelligently in near real time. Event processing technologies are often referred to as a technical basis of the—often underestimated—big data processing and analysis capability. The chapter briefly introduces its central concepts and ideas. We outline the advantages, limits, and future developments of the technology using the example of an ad-hoc car sharing scenario in a Smart City.

M. Schaaf (✉)
Institut für Wirtschaftsinformatik, Fachhochschule Nordwestschweiz,
Riggenbachstrasse 16,
4600 Olten, Schweiz
E-Mail: marc.schaaf@fhnw.ch

G. Wilke
Institut für Wirtschaftsinformatik, Hochschule Luzern,
Zentralstrasse 9,
6002 Luzern, Schweiz
E-Mail: gwendolin.wilke@hslu.ch

1 „The world is event driven“

Im Morgenverkehr kommt es auf einer der Hauptverkehrsadern einer Großstadt zu einem Unfall und in Folge zu einem Stau. Innerhalb von Sekunden zeigen Smart Phone Apps, GPS-Geräte und digitale Anzeigetafeln am Straßenrand alternative Umgehungs-Routen an, welche per intelligenter Ampelschaltung „grüne Wellen“ zugewiesen bekommen. Die angezeigten Routen werden dynamisch an die aktuelle Verkehrssituation angepasst, wodurch Folgestaus vermieden werden.

Zeitgleich mit den ersten Staumeldungen werden automatisch die nächstgelegenen verfügbaren Einsatzfahrzeuge von Rettung, Polizei und Abschleppdienst koordiniert zugewiesen und alarmiert. Während sie ihre Stationen verlassen und sich auf den Weg zur Unfallstelle machen, hat sich dort der Stau bereits signifikant verringert und ermöglicht ihnen so, binnen Minuten am Unfallort einzutreffen.

Dort angekommen wird klar, dass eine längere Sperrung der Hauptverkehrsstraße eingerichtet werden muss. Ausgelöst durch diese Nachricht treten automatisiert berechnete Umleitungen der betroffenen Buslinien in Kraft und werden über die entsprechenden Informationskanäle angekündigt. Busse, welche in Folge der Sperrung eine Verspätung hinnehmen müssen senden Echtzeit-Statusmeldungen an stark frequentierte Anschlusslinien, und gewähren so nahtlose Transfers. Gleichzeitig werden im lokalen Metronetz Zusatzzüge eingesetzt.

Um Autofahrer zum Ausweichen auf öffentliche Verkehrsmittel zu motivieren, treten Vergünstigungen für außerstädtische P + R Parkplätze in Kraft, welche u. a. über GPS und Smart Phone angezeigt werden. Innerhalb der Stadt sorgt ein flexibles und distanzbasiertes Zahlungsverfahren für eine unkomplizierte Nutzbarkeit von Bahn-, Bus- und Metrolinien. Darüber hinaus sorgt der von der Stadt verwaltete on-demand car-sharing Dienst für ad-hoc organisierte Fahrgemeinschaften für eine weitere Entlastung des innerstädtischen Verkehrs.

Termin-sensible Dienste wie Zustelldienste, Lieferdienste, Krankentransporte, mobile Hausbetreuung oder Geldtransporte erhalten über Abonnement-Dienste detaillierte real-time Information über die aktuelle und prognostizierte Verkehrssituation, wodurch eine zeitnahe Information der betroffenen Kunden, sowie eine schnelle Anpassung der eigenen Logistik-Planung des Unternehmens möglich wird.

Bereits wenige Minuten nach dem Unfall verlaufen die Geschäfte und Services der Stadt größtenteils in ihren gewohnten Bahnen. Größere Unterbrüche konnten aufgefangen werden.

So oder so ähnlich könnten die Abläufe in einer Smart City zukünftig aussehen, und schon heute weisen Leuchtturmprojekte wie Singapurs „Intelligent Transport System“[1] den Weg in die Zukunft. Was das obige Beispiel jedoch auch verdeutlicht ist, dass die Welt praktisch in allen Bereichen ereignisgetrieben ist. Ereignisse erfordern Entscheidungen. Wann immer etwas „Relevantes“ auftritt muss hierauf reagiert werden. Reaktionen auf Ereignisse werden in der heutigen digital vernetzten Welt in immer kleineren Zeitfenstern erwartet.

Schon heute sind viele IT-Systeme auf hohe Geschwindigkeiten (bspw. SCADA) oder auf hohe Datenmengen (bspw. Kartendienste) optimiert. Neu ist hier jedoch

[1] http://www.lta.gov.sg, aufgerufen am 22.04.2015

die Problematik, beide Herausforderungen gleichzeitig bewältigen zu müssen. Ausserdem sind die Prozesse vieler Systeme häufig zu unflexibel, um sich schnell und dynamisch an wechselnde Situationen anpassen zu können.

Ereignisverarbeitende Systeme können an dieser Stelle Abhilfe schaffen: Sensordaten werden dabei als „Datenströme“ behandelt, welche parallel und in Echtzeit verarbeitet und direkt danach verworfen werden. Relevante Änderungen in einzelnen Messdaten gelten hierbei als „Ereignisse“. Nur diese werden zur weiteren Verarbeitung zwischengespeichert, wodurch eine zeitnahe Bewältigung hoher Datenvolumina ermöglicht wird. Komplexe Situationen können flexibel als Kombination einfacher Ereignis-Ströme dargestellt werden.

Im nächsten Kapitel beschreiben wir kurz die Ursprünge ereignisverarbeitender Systeme, ihre grundlegende Architektur, sowie die wichtigsten heute verbreiteten Typen von Ereignisverarbeitungssystemen.

2 Grundlagen der Ereignisverarbeitung

Ereignisverarbeitung hat seinen Ursprung in den 1950ern in der Simulation von diskreten Ereignissen beispielsweise in der Luftfahrttechnik oder in der Simulation natürlicher Systeme wie der Wettervorhersage. Später, in den frühen 1990ern wurden klassische Datenbanksysteme erweitert um die Unterstützung so genannter Event Condition Action (ECA) Regeln und boten damit eine grundlegende Unterstützung für Ereignisverarbeitung. In Datenbanksystemen erlauben noch heute ECA Regeln eine *direkte Reaktion* auf eintretende Ereignisse wie beispielsweise das Einfügen oder Ändern eines Datensatzes. Im Laufe der Zeit wurde diese Funktionalität auch außerhalb der begrenzten Umgebung eines Datenbanksystems verfügbar und findet sich heute in diversen Ausprägungen in einer Vielzahl von Systemen.

Bezüglich des Kommunikationsmodus verfolgt die ereignisgetriebene Verarbeitung einen anderen Ansatz als die klassische Anfrage-Antwort Kommunikation: Während bei dieser aktiv nach Informationen angefragt werden muss, steht bei der Ereignisverarbeitung die Verfügbarkeit einer neuen Information bzw. eines Ereignisses im Zentrum. Diese dient als Treiber für die Kommunikation zwischen Anwendungen und Anwendungsteilen (siehe Tab. 1). Analog zur Kommunikation ist zudem auch die Verarbeitung getrieben durch das Vorhandensein neuer Ereignisse. Entsprechend erfolgt die Verarbeitung häufig über die anfangs erwähnten ECA-Regeln oder ähnlichen Regeltypen, wobei die jeweils passenden Regeln für den neu eingetroffenen Ereignistyp gewählt und gegen die eingetroffenen Ereignisdaten evaluiert werden. Um eine einheitliche Definition der verfügbaren Ereignistypen zu gewährleisten, verfügen Ereignisverarbeitungssysteme in der Regel über ein Ereignismodell (Abb. 1) welches allen beteiligten Komponenten gleichermaßen bekannt ist.

Tab. 1 Taxonomie der Kooperationsmodelle nach Mühl et al. (2006)

	Empfänger initiiert Kommunikation	Sender initiiert Kommunikation
Direkte Adressierung	Anfrage-Antwort Zyklus	Callback
Indirekte Adressierung	Anonymer Anfrage-Antwort Zyklus	Ereignisbasiert

Ein Konzept, welches in den letzten Jahren besonders an Bedeutung gewonnen hat ist das von David Luckham (2001) beschriebene Complex Event Processing (CEP). CEP ist spezialisiert auf das Erkennen von Mustern in eingehenden Ereignissen und ermöglicht es hierbei höherwertige Ereignisse zu erzeugen um hierdurch (a) die Menge der Ereignisse, welche weiterverarbeitet werden müssen zu reduzieren und (b) die nachfolgende Verarbeitung auf einer höheren Abstraktionsstufe, in der Regel einfacher durchführen zu können. Hierbei ist die Abstraktion explizit nicht auf einen Schritt begrenzt, sondern findet häufig in zahlreichen Stufen statt, wobei sich hierbei in der Regel ein Übergang von quantitativen Eingabedaten hin zu Qualitativen vollzieht um so die nachfolgende Verarbeitung zu vereinfachen.

In einem Ereignisverarbeitungssystem registrieren Komponenten ihr Interesse an bestimmten Ereignistypen bei einer Kommunikationsmiddleware (Abb. 2). Die Middleware kümmert sich um die Verteilung der Ereignisse an Komponenten, welche sich für die jeweiligen Ereignistypen registriert haben. Die Komponenten müssen hierbei nicht wissen, welche Anwendung das geforderte Ereignis generieren wird. Ebenso muss die ereignisgenerierende Komponente kein Wissen über den oder die Empfänger eines Ereignisses haben. Hierdurch wird die häufig enge Bindung von Anwendungskomponenten aneinander aufgehoben. Das genutzte Kommunikationsmodell resultiert zudem in eine zeitliche Entkoppelung der Verarbeitungsabläufe zwischen den Komponenten, da eine ereignisgenerierende Komponente ein neues Ereignis direkt in die Obhut der Middleware übergibt, welche die Ereignisse asynchron an registrierte Empfänger weiterleitet. Die Entkoppelung der Komponenten ermöglicht des Weiteren das einfache Einbinden neuer Komponenten oder das Anpassen von

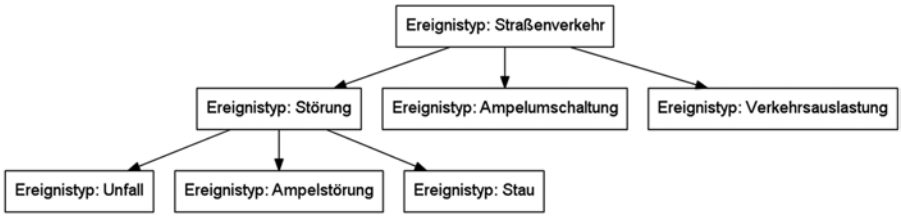

Abb. 1 Ein beispielhaftes Ereignismodell im Kontext Smart Traffic Management

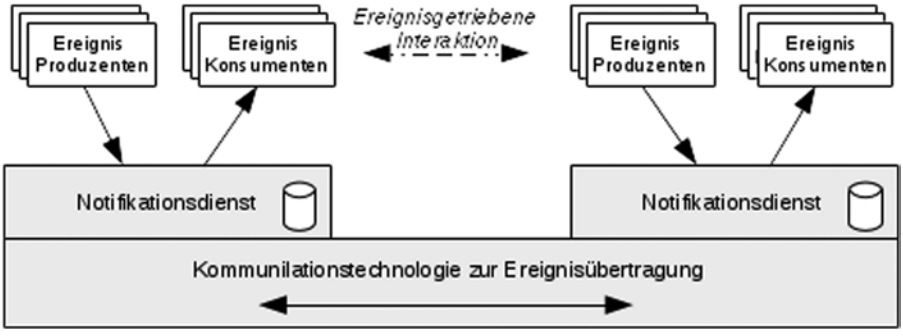

Abb. 2 Notifikationsdienst nach Mühl et al. (2006)

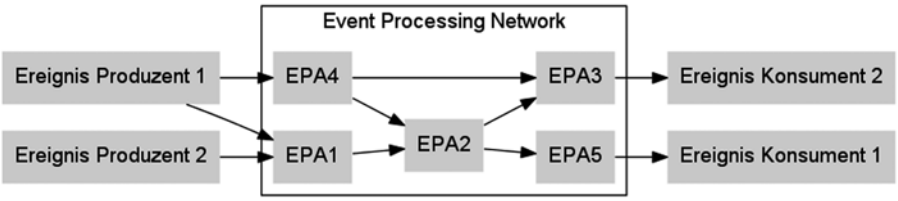

Abb. 3 Exemplarische Darstellung eines Event Processing Networks

bereits existierenden. Dies stellt einen klaren Vorteil dieses Architekturstils in Bezug auf den eingangs erwähnten Bedarf einer flexibilisierten Verarbeitung da, welche sich kurzfristig an neue Gegebenheiten anpassen muss.

In einem Ereignisverarbeitungssystem werden die Verarbeitungskomponenten in der Regel als Event Processing Agents (EPA) bezeichnet. Ein solcher Agent realisiert eine Ereignisverarbeitung basierend auf einem komponentenübergreifenden Ereignismodell und den für den aktuellen Agenten nötigen Ereignisverarbeitungsregeln. Die Resultate der Verarbeitung werden von dem Agenten dann wieder als Ereignisse publiziert, um so von anderen Agenten genutzt werden zu können. Neben den verarbeiteten EPAs können reine Ereignisquellen und reine Ereigniskonsumenten als Spezialformen gesehen werden.

Aus der Menge der EPAs und deren Kommunikationsbeziehungen ergibt sich ein gerichteter Graph welcher auch als Event Processing Network (EPN) bezeichnet wird (Abb. 3). Basierend auf dem Graphen können Entscheidungen zur Verteilung, Skalierung sowie zur Optimierung des Verarbeitungssystems durchgeführt werden.

Im folgenden Kapitel erläutern wir die Funktionsweise eines Ereignisverarbeitungssystems exemplarisch anhand eines flexiblen on-demand Car-Sharing Dienstes, welcher als Teil des Mobilitäts-Konzeptes einer zukünftigen Smart City umgesetzt werden könnte.

3 Anwendungsbeispiel: Ad-hoc Car-Sharing als Location Based Service

Ein im Kontext von Smart Mobility zunehmend populäres Konzept ist vehicle sharing, also die gemeinschaftliche Nutzung von privaten oder geleasten Fahrzeugen wie Fahrrädern (bike sharing) oder Autos (car sharing). Beispiele sind CITYBIKE WIEN[2] oder das Schweizerische Mobilty[3]. Insbesondere car sharing wird in Ballungszentren als zukunftsorientiertes Lösungskonzept zur Reduzierung der Abgasbelastung, der Verkehrsdichte, sowie des Energieverbrauchs gesehen.

Heute eingesetzte Car-Sharing Systeme haben jedoch den wesentlichen Nachteil mangelnder Flexibilität: Autos müssen an fest vorgegeben Standorten abgeholt werden und können nicht ohne erhebliche Kosten über Nacht zu Hause behalten werden; Sie müssen meist am Abholort zurückgegeben werden, wodurch one-way-trips unmöglich gemacht werden; darüber hinaus sinkt die räumliche Abdeckung mit

[2] http://www.citybikewien.at, aufgerufen am 28.01.2015.

[3] https://www.mobility.ch, aufgerufen am 28.01.2015.

zunehmender Distanz vom Zentrum eines Ballungsraumes, wodurch das sogenannte „last mile problem“ geschaffen wird: Pendler von außerhalb können das System nur bedingt nutzen, da in den letzten Kilometern kein Rückgabestandort vorhanden ist (Vishwanath 2014). Diese mangelnde Flexibilität führt dazu dass viele Menschen ein eigenes Auto dem Car-Sharing Konzept vorziehen.

Als Reaktion auf die mangelnde Flexibilität existierender Car-Sharing Konzepte wurden kooperative, personalisierte on-demand Car-Sharing Systeme vorgeschlagen (siehe z. B. (Winter und Nittel 2006; Miller 2013; Vishwanath 2014), welche dynamisch und ereignisgetrieben Fahrgelegenheiten und Transportanfragen in Echtzeit miteinander abgleichen. Diese erlauben es insbesondere, Mitfahrgelegenheiten auf gemeinsamen (Teil-) Strecken on-demand zu koordinieren bzw. diese ohne feste Übergabestandorte von einem zum anderen Nutzer zu übergeben.

Ein Szenario einer ad-hoc Mobilitätsanfrage an ein on-demand Car-Sharing System könnte z. B. folgendermaßen aussehen: Eine Nutzerin telefoniert mit einer Freundin. Diese lädt sie spontan zu einem Treffen in Wien Mitte anlässlich einer Lesung in ihrem Lieblings-Café ein. Die Vorstellung beginnt in einer Stunde. Die Nutzerin stellt daraufhin eine Mobilitätsanfrage per Smart Phone, um vor Beginn der Vorstellung von ihrem Wohnort in Wien Liesing nach Wien Mitte zu gelangen. Sie gibt dazu ihren aktuellen Standort in Liesing, ihren gewünschten Zielort in Wien Mitte, das frühestmögliche Abfahrtsdatum („sofort“), sowie das spätest mögliche Ankunftsdatum („in einer Stunde“) bekannt. Sie wählt die Option „one-way“, da sie plant, im Anschluss an die Lesung bei ihrer Freundin zu übernachten, sowie die Option „Mitfahrgelegenheit“, da Parkplätze in Wien Mitte schwer zu finden sind.

Basierend auf den aktuellen und vorhergesagten raumzeitlichen Standorten teilnehmender Fahrzeuge sowie der aktuellen und vorhergesagten Verkehrslage berechnet der Algorithmus des ad-hoc Car-Sharing Systems mögliche Mitfahrgelegenheiten und die zugehörigen Routen. Basierend auf den geschätzten Fahrzeiten pro Route, der Anzahl der notwendigen Umstiege von einer zur anderen Mitfahrgelegenheit, sowie der geschätzten Wartezeiten und Distanzen zwischen Transferpunkten wird der Nutzerin die beste Routenoption über ein intuitives Benutzer-Interface vorgeschlagen (Rigby und Winter 2014). Ein solcher Routenvorschlag könnte bspw. wie in Abb. 4 dargestellt aussehen: Die Route besteht aus drei aufeinanderfolgenden Teilstrecken, zwischen welchen die Nutzerin jeweils von einer zur nächsten Mitfahrgelegenheit umsteigen muss. Die grauen Kreise zeigen dabei die räumlichen Distanzen zwischen Umsteigepunkten an, die gesamte Fahrzeit, sowie Transferwartezeiten werden in

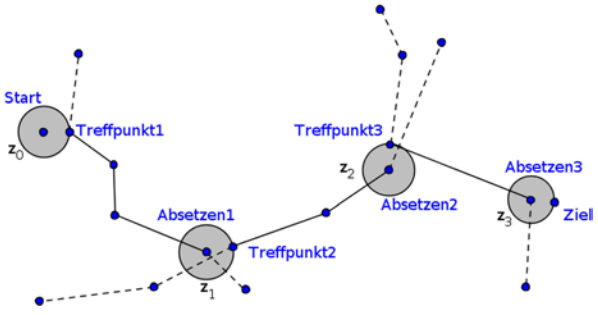

Abb. 4 Beispielhafte Routenfindung für eine Routenoption über drei Mitfahrten

einem separaten Fenster gelistet. Die Nutzerin kann daraufhin dem Routenvorschlag zustimmen, ihn ablehnen, oder weitere Routenvorschläge anzeigen lassen.

Voraussetzung für das reibungslose Funktionieren eines kooperativen, personalisierten on-demand Car-Sharing Systems ist daher ein IT-System, welches die parallele Berechnung komplexer und interaktiver raumzeitlicher Problemstellungen über große Mengen an Datenströmen (wie die ständig wechselnden Standorte beteiligter Fahrzeuge und Nutzer, die sich ändernde Verkehrslage, etc.) in Echtzeit bewältigen kann. Dabei müssen die vorhergesagten Standorte teilnehmender Fahrzeuge mit Mobilitäts-Anfragen von Benutzern so abgeglichen werden, dass für die Benutzer angenehme Transportbedingungen resultieren: Die gewünschte Route muss end-to-end bedient werden können, Wartezeiten bei Fahrtantritt und Transfer dürfen nicht zu lange sein, Fußwege zwischen Transferstandorten dürfen nicht zu weit sein. Abhängig von der Routenlänge oder dem Benutzerwunsch darf nur eine begrenzte Anzahl von Transfers stattfinden, die Zuverlässigkeit des Transports muss abgesehen von angemessenen zeitlichen Schwankungen gewährleistet sein.

Entsprechend des beschriebenen Szenarios gibt Abb. 5 eine grobe schematische Übersicht über einen beispielhaften Einsatz von Ereignisverarbeitung für die Umsetzung eines Car Sharing Management Systems (CSMS).

Fahrzeuge, welche bereit sind Fahrgäste aufzunehmen übermitteln zusammen mit dem Ereignis ‚Fahrtantritt' ihr Fahrtziel und die geplante Route an das CSMS. Das beim CSMS eingehende Ereignis wiederum initiiert einen Abgleich der Fahrtroute mit bisherigen Mitnahmeanfragen. Analog initiieren neue Mitnahmeanfrageereignisse einen Abgleich mit bereits registrierten Fahrzeugen und ihren Fahrtrouten. Sobald eine akzeptable Mitnahmemöglichkeit gefunden wurde, wird der Fahrer und der Fahrgast durch ein Treffpunktangabe Ereignis vom CSMS informiert. Im weiteren Verlauf kann das CSMS dann periodische Aktualisierungen der Fahrzeugposition und Fahrgastposition nutzen, um die jeweilige Ankunftszeit am Treffpunkt zu prognostizieren und bei erwarteten Abweichungen den Fahrer und Fahrgast zu informieren.

Hierbei kann der nötige Verarbeitungsablauf auf mehrere Komponenten aufgeteilt werden (z. B. Positionsvorhersage, regionsbezogene Abstimmung mit Mitnahmeanfragen, Bestimmen von Weiterreisemöglichkeiten). Aus der Menge der Komponenten und ihren Kommunikationsbeziehungen ergibt sich somit ein Event Processing Network, welches im Laufe der Verarbeitung komplexe Ereignisse generiert, welche

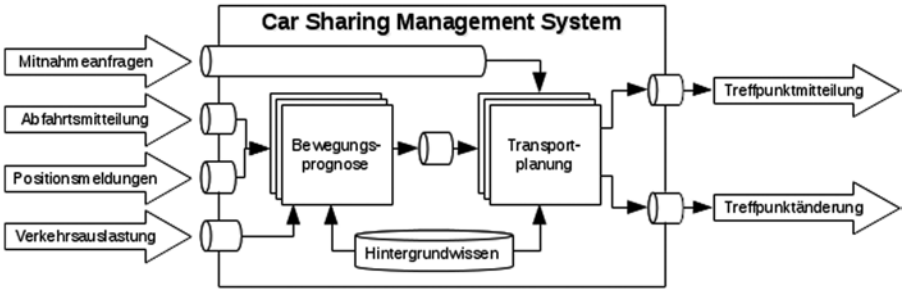

Abb. 5 Beispielhafte Struktur eines Ad-hoc Car Sharing Management Systems

z. B. die Routen für die nahe Zukunft beschreiben. Bedingt durch die Aufteilung auf mehrere Komponenten ergibt sich zudem ein guter Ansatzpunkt zur Verteilung und Skalierung der Anwendung.

Neben den reinen Ereignisdaten benötigt das CSMS Zugriff auf Hintergrundinformationen wie das Layout des Straßennetzes, geplante Baustellen sowie größere Veranstaltungen, bei welchen Verkehrsbeeinträchtigungen wahrscheinlich sind. Diese werden den Komponenten in Form von externen Datenquellen, nicht ereignisgetrieben, zum Abruf bereitgestellt und können als Teil der ereignisgetriebenen Regelverarbeitung genutzt werden.

Nachfolgend sei kurz auf die Vorteile und Grenzen des umrissenen Verarbeitungsansatzes eingegangen. Des Weiteren wird ein kurzer Ausblick auf aktuelle Entwicklungen in diesem Bereich gegeben.

4 Vorteile Ereignisverarbeitender Systeme

Vorteile bietet der ereignisgetriebene Ansatz vor allem in der zeitnahen Verarbeitung, der einfachen Skalierbarkeit sowie der einfachen Integration bzw. Erweiterbarkeit der Anwendung. Bedingt durch die ereignisgetriebene, *zeitnahe Verarbeitung* kann das CSMS der heutigen Erwartungshaltung der Anwender besser gerecht werden, indem es möglichst kurze Verarbeitungszeiten und zudem kurzfristiges, aktives Informieren der Nutzer über Abweichungen auf einfache Weise ermöglicht. Zudem werden kommende Smart City Anwendungen aufgrund großer Datenmengen und gleichzeitig großen Anwenderzahlen hohe Anforderungen an die *dynamische Skalierbarkeit* eingesetzter Systeme haben, welchen mithilfe der inhärenten Komponentenentkoppelung und der Verteilbarkeit der Komponenten eines ereignisgetriebenen Anwendungskonzeptes begegnet werden kann.

Bedingt durch das Erzeugen von Zwischenzuständen seitens des CSMS in Form von Ereignissen können auf einfache Weise weitere Systeme integriert werden. Beispielsweise können Treffpunktereignisse, welche einen gewissen Fußweg seitens des Fahrgastes erfordern würden, durch die Integration mit einem Taxi Anbieter genutzt werden, um dem Fahrgast, bei Verfügbarkeit eines Taxis in der Gegend, eine Fahrt zum Treffpunkt anzubieten. Eine derartige Erweiterbarkeit kann gerade in der aufstrebenden Anwendungsdomäne Smart Cities, wo viele Szenarien bisher nur unscharf definiert sind, einen großen Vorteil darstellen, um das Potenzial neuer Szenarien oder Anpassungen der existierenden Systeme kurzfristig evaluieren zu können.

5 Anwendungsbereiche Ereignisverarbeitender Systeme

Heute werden CEP Systeme häufig im Bankensektor im Bereich der Betrugserkennung oder in der Logistikbanche zur Lieferkettenüberwachung eingesetzt. Ein Beispiel für den Einsatz von CEP im Smart City Bereich ist die Urbane Informations- und Service-Plattform (UISP)[4], welche in der Stadt Darmstadt u. a. zum Monitoring

[4] http://martinfowler.com/articles/microservices.html, aufgerufen am 22. April 2015

von Verkehrsfluss und Emissionsbelastung eingesetzt wird. Viele neu aufkommende Anwendungen im Smart City Bereich, wie bspw. die oben beschriebene Car-Sharing Anwendung, sind jedoch oft komplexer und dynamischer als herkömmliche Anwendungen, da sie sich nicht nur auf statische Sensordaten beziehen, sondern situationsbezogene Verarbeitungsabläufe mit dynamisch-adaptiver Verarbeitung unter Einbeziehung von Hintergrundwissen benötigen. Um beispielsweise längere Routen mit mehreren Mitfahrgelegenheiten zu koordinieren ergibt sich eine Verarbeitung, welche sich auf die jeweilige Situation der Route fokussieren muss, indem sich das zuständige EPA gezielt für Ereignisse in einem (in diesem Fall geografischen) Fokusbereich registriert. Erst durch diese Fokussierung kann der nächste Schritt im Routenverlauf gezielt bestimmt werden. Für diese Art der dynamischen situationsadaptiven Ereignisverarbeitung lässt dich Dynamic Complex Event Processing (DCEP), eine Erweiterung der herkömmlichen CEP, einsetzen (Schaaf 2013). DCEP wird derzeit für den Einsatz im Telekommunikationsbereich (Schaaf et al. 2015), sowie für Anwendungen im Smart Grid Bereich (Wilke et al. 2014) getestet.

Zukünftige Smart City Anwendungen unterscheiden sich von herkömmlichen Anwendungsbereichen ereignisverarbeitender Systeme außerdem in ihrem Fokus auf partizipative Strukturen. Dadurch resultiert eine besondere Wichtigkeit der Benutzerfreundlichkeit: meist sollen möglichst alle Bevölkerungsschichten und Bevölkerungsgruppen in den Stand gesetzt sein eine solche Anwendung nutzen zu können. Wegweisend hierfür sind natürlichsprachliche Eingabesysteme wie Siri[5]. Um die Benutzerfreundlichkeit und damit den Erfolg von kollaborativen und mobilen ortsbezogenen Diensten wie dem beschriebenen ad-hoc on-demand Car-Sharing System sicherzustellen gibt es zunehmend Bestrebungen, ungenaue umgangssprachliche Ortsangaben wie „Wien Stephansplatz“ als Eingabemöglichkeiten zuzulassen, siehe z. B. (Wilke 2009). Die besondere Herausforderung hierbei ist die verschiedenen Arten von Ungenauigkeit (wie bspw. statistische und possibilistische) Unsicherheiten zu kombinieren, und gleichzeitig verlässliche Ergebnisse zu liefern (Rigby und Winter 2014; Zadeh 2006; Wilke 2012). Gerade in diesem Gebiet bedarf es noch weiterer Entwicklungen, welche die Umsetzung derartiger Funktionalitäten ermöglichen und hinreichend vereinfachen.

Insgesamt jedoch ergeben sich Methoden und Systeme aus dem Bereich der Ereignisverarbeitung auch schon heute als besonders geeignet für Smart City Anwendungen, da ihre Fähigkeit, dynamisch sich ändernde, komplexe Ereignisse auf qualitativer (d. h. von den einzelnen Messwerten abstrahierter) Ebene zu erkennen es ermöglicht auf diese direkt, d. h ohne explizite Anfrage, flexibel reagieren zu können. Zudem bieten sie eine gute Grundlage für die dynamische Skalierung von Anwendungen um mit den teils erheblich fluktuierenden Datenmengen in Smart Cities umgehen zu können.

[5] http://www.apple.com/chde/ios/siri, aufgerufen am 22.04.2015

Literatur

Schaaf (2013) Event processing with dynamically changing focus. Research Challenges in Information Science (RCIS) IEEE Seventh International Conference on, pp.1, 6, 29–31 Mai 2013 doi: 10.1109/RCIS.2013.6577727

Luckham D (2001) The power of events: an introduction to complex event processing in distributed enterprise systems. Wesley Longman Publishing Co. Inc., Boston

Miller H (2013) Beyond sharing: cultivating cooperative transportation systems through geographic information systems. J Transp Geogr 31:296–308

Mühl G, Ludger F, Pietzuch P (2006) Distributed event-based systems. Springer, Berlin

Rigby M, Winter S (2014) Flexible decision making under uncertainty for intelligent mobility on-demand. Workshop on visually-supported reasoning with uncertainty, GIScience, Vienna, Austria

Schaaf M, Wilke G, Mikkola T, Bunn E, Wache H, Gatziu Grivas S (2015) Engineering systems towards a timely root cause analysis for complex situations in large scale telecommunications networks. Knowledge based and intelligent information and engineering systems, 19th International Conference on (to be published)

Vishwanath A, Gan H-S, Kalyanaraman S, Winter S, Mareels I (2014) Personalised public transportation: a new mobility model for urban and suburban transportation, IBM Research Report RC25472 (MRL1406-001)

Wilke G (2009) Approximate geometric reasoning with extended geographic objects. ISPRS-COST Workshop on quality, scale and analysis aspects of city models, Lund, Sweden

Wilke G (2012) Approximate tolerance geometry—an error calculus for geometric reasoning under positional tolerance. Dissertation, Technische Universität Wien

Wilke G, Schaaf M, Bunn E, Mikkola T, Remo R, Wache H, Gatziu Grivas S (2014) Intelligent dynamic load management based on solar panel monitoring. Smart Grids and Green IT Systems, Proceedings of the 3rd International Conference on, 76–81

Winter S, Nittel S. (2006) Ad-hoc shared-ride trip planning by mobile geosensor networks. Int J Geographical Information Science 20(8):899–916

Zadeh L (2006) Generalized theory of uncertainty (GTU)—principal concepts and ideas. Comput Stat Data Anal 51:15–46