Herausforderungen bei der Redigierung und Versionierung fortlaufender Massendaten

Thomas Gutzke, Frank Reußner

envi-systems GmbH
An der Eschollmühle 28
64297 Darmstadt
gutzke@envi-systems.com
frank@reussner.org

Abstract: In der Wasserver- und -entsorgung werden über Prozessleitsysteme in den meisten der jeweils über 10.000 Anlagen (Wasserwerken und Kläranlagen) mehrere hundert Datenpunkte aufgezeichnet – und das z. T. im Minuten- und Sekundentakt. Zur Prozessüberwachung, -steuerung und -optimierung werden von zahlreichen dieser Datenpunkte die gesamten Datenreihen kontinuierlich und vollständig aufgezeichnet und regelmäßig ausgewertet. Für eine nachhaltige Bewirtschaftung werden diese Daten nicht nur zur zeitnahen Überwachung und Steuerung benötigt. Für die Beantwortung zahlreicher fachlicher Fragestellungen, sind dieses Daten kontinuierlich und persistent aufzuzeichnen, so dass beispielsweise Entwicklungen über mehrere Jahrzehnte innerhalb einer Auswertung erfolgen können. Da es systembedingt vorkommen kann, dass in den Datenreihen Fehler auftreten (Datenlücken, Ausreißer, Drift) werden Redigierungen erforderlich. Da die Datenreihen sich einerseits kontinuierlich fortschreiben, aber z. T. auch mehrfach redigiert werden müssen, exisiteren hohe Anforderungen an die Redigierungsmöglichkeiten, die Datenarchivierungskonzepte, die Versionierbarkeit und nicht zuletzt an die Performance..

Dieser Beitrag stellt Möglichkeiten einer performanten und gleichzeitig diagrammgestützten Datenredigierung vor. Alle Redigierungsprozesse müssen aus Gründen der Nachvollziehbarkeit versioniert in einem Datenbanksystem abgelegt werden. Insbesondere die Tatsache, dass Datenreihen nie abgeschlossen werden, sondern sich kontinuierlich fortschreiben, stellt hohe Anforderungen an die Performance der Anwendung sowie der Datenbank auf der einen und an die Versionierung aller Redigierungsprozesse auf der anderen Seite. Beide Anforderungen beeinflussen sich dabei gegenseitig negativ. Dieser Beitrag beschreibt die Herausforderungen, die sich hieraus ergeben haben, stellt aber auch einen Ansatz vor, der im ersten Schritt in einer „einfachen“ aber funktionalen Variante erfolgreich umgesetzt wurde.

1 Einführung

In Deutschland wird die Wasserversorgung von ca. 6.500 Wasserversorgungsunternehmen sichergestellt. Im Bereich der Abwasserentsorgung sind über 10.000 Anlagen in Betrieb. In beiden Bereichen fallen zahlreiche umweltrelevante Daten an, die kontinuierlich erfasst, verwaltet und ausgewertet werden müssen. Die für den Betrieb relevanten Daten werden mittlerweile in den meisten Anlagen über Prozessleitsysteme erfasst. Für die Betreiber dieser Anlagen sind diese Daten wichtig, um einerseits im Rahmen der Berichtspflichten gegenüber den Behörden den fachgerechten Betrieb zu dokumentieren und anderseits einen wirtschaftlichen und ökologischen Betrieb der Anlage durch gezielte Steuerung zu erreichen. Die Daten werden dabei über Sensoren erfasst und meist über Steuerkabel (4-20mA) oder über eine Datenfernübertragung unredigiert an das interne Speichersystem übertragen. Leistungsstarke Prozessleitsysteme ermöglichen auf dieser Basis eine Adhoc-Überwachung aller angeschlossenen Anlagenteile. Viele Anlagen erlauben darüber hinaus eine aktive Anlagensteuerung. Über die letzten Jahrzehnte wurden die Theorien, Techniken und Verfahrensweisen im Betrieb dieser Anlagen so weit verfeinert, dass diese einen sehr hohen Wirkungsgrad aufweisen. Um dies zu erreichen, wurde u. a. ein immer dichteres Datenerfassungsnetz aufgebaut, welches zusätzlich die einzelnen Daten in immer kleiner werdenden Intervallen aufnimmt. Reichten früher Tageswerte aus, so werden heutzutage mehr und mehr Daten im Minuten- und z. T. Sekundentakt erfasst. Die Daten werden in der Regel in einem Ringspeicher abgelegt, wobei die Datenbestände auf Grund ihres Datenumfangs nach festlegbaren Zeiträumen ausgedünnt bzw. überschrieben werden. Die Archivierung erfolgt daher häufig in separaten Zusatzsystemen.

Hinsichtlich der Archivierung, Redigierung und Auswertung weisen diese Systeme jedoch große Schwächen auf. Werksmeister und Betriebsleiter stehen regelmäßig vor der Aufgabe, die Anlagenkomponenten in Bezug auf technische Leistungsdaten auszuwerten und effizient einzustellen. Auch die Erstellung von Berichten (z. B. im Rahmen der Eigenkontrollverordnung) erfordert Zugriff auf betriebsrelevante Datenbestände in jährlichen Zyklen. In beiden Fällen muss der Ringspeicher direkt angezapft werden, wobei die Daten i. d. R. vom Ringspeicher nach Excel kopiert und dort manuell weiterverarbeitet werden. Fehler in den Datenbeständen müssen hier z. T. sehr aufwändig und fehleranfällig korrigiert werden. Die diversen erforderlichen Auswertungen in Form von Diagrammen und Berichten erfordern zudem einen hohen manuellen Zeitaufwand.

2 Zielsetzung

Bei dem hier vorgestellten Projekt handelt es sich um eine ausgewählte Komponente des Softwaresystems „GW-Manager“ - einer Eigenentwicklung der Firma envi-systems GmbH (en14). Das im Jahr 2001 ursprünglich nur für den Bereich der Wasserversorgung entwickelte System kommt zunehmend auch in den Bereichen Abwasserentsorgung/Kläranlagenmanagement für Datenhaltungs- und Auswertungsaufgaben zum Einsatz. Das System setzt an der Schnittstelle zwischen Prozessleitsystem und den Optimierungsaufgaben der Fachanwender an und löst dabei das Problem des Handlings von Massendaten und deren Datenredigierung. Es verfolgt dabei folgende Ziele (siehe Abbildung 1):

- Automatisierte Datenübernahme in eine für Massendaten geeignete Datenbank
- Vollständige Langzeitarchivierung aller relevanten Daten
- Visuell unterstützte Datenredigierung
- Variantenbasierte Zeitreihenversionierung
- Bereitstellung komfortabler Analyse- und Auswertungsfunktionalitäten

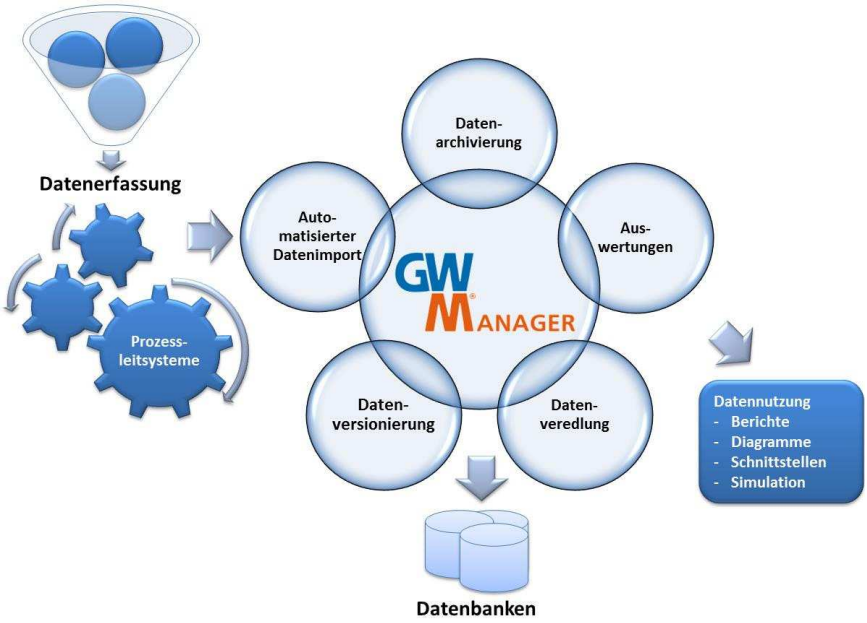

Abbildung 1: GW-Manager im Verbundsystem

3 Redigierung von Zeitreihen

Die Anforderungen, die an ein System gestellt werden, das Umweltmassendaten sowohl erfasst als auch bearbeitet, sind vielfältig. Nicht selten treten innerhalb einzelner Zeitreihen Fehler auf, die folgende Ursachen haben können:

- falsch kalibrierte Sensoren
- falsch zugeordnete Sensoren
- Stromausfall
- Stromschwankungen
- Alterungserscheinungen oder Verschmutzung von Sensoren (Drifts)

Für die Bereinigung dieser Fehler müssen Werkzeuge zur Bearbeitung von Ausreißern, Lückenschließung sowie Trend- bzw. Driftbereinigung bereitgestellt werden (siehe Abbildung 2). Die Bereinigung sollte zudem visuell unterstützt und performant umgesetzt werden.

Daraus folgen Anforderungen an ein System zur visuell unterstützten Datenredigierung, die meist gegenläufige Ziele haben. So wirkt sich z. B. eine Fokussierung auf die benutzerfreundliche Gestaltung der Datenredigierung unter Umständen negativ auf die Performanz des Gesamtsystems aus. Neben einer visuellen Datenkontrolle, die den Benutzer unterstützt, ist eine visuelle Datenbearbeitung (Drag ‘n Drop) eine weitere wichtige Anforderung. Hinsichtlich der Datenhaltung ist es zudem erforderlich, neben den erfassten Rohdaten auch eine Versionierung der redigierten Datenbestände zuzulassen.

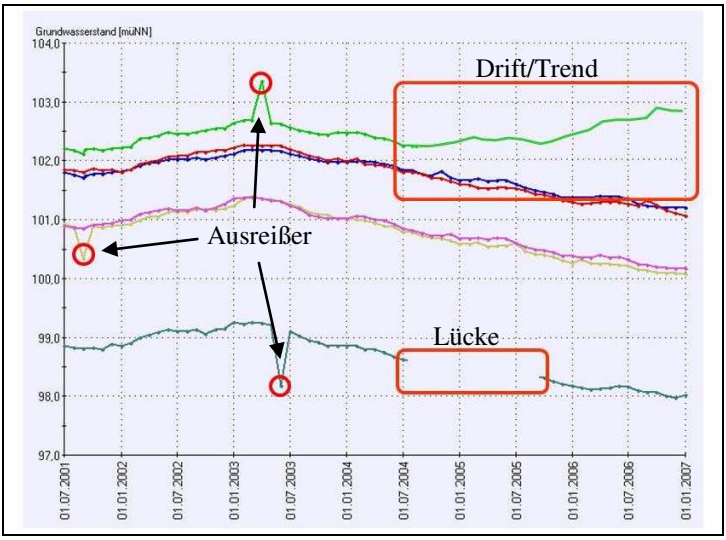

Abbildung 2: Erfordernisse der Datenredigierung

4 Versionierung fortlaufender Zeitreihen

Ziel einer Versionierung von Zeitreihen ist zum einen die Beurteilung des Qualitätszustands eines Datenpunkts. So kann beispielsweise zwischen Rohdaten oder verifizierten Daten unterschieden werden. Zum anderen ist das Ziel, mögliche Bearbeitungsschritte transparent und nachvollziehbar zu machen. Damit kann z. B. eine Bearbeitung von Datenpunkten bei Bedarf auch rückgängig gemacht werden. Denn häufig stellt sich erst im Nachhinein heraus, dass Annahmen sowie die vorgenommenen Redigierungen falsch waren.

Ein Beispiel für eine typische bearbeitete Zeitreihe ist in Abbildung 3 dargestellt. Auf der x-Achse liegen alle erfassten „Roh"-Daten. Insgesamt wurde die Zeitreihe in sieben einander folgenden Schritten bearbeitet. Alle Bearbeitungen sind in drei Zeiträumen vorgenommen worden. Je nach dem, ob man dabei alle sieben Schritte wieder nachvollziehen bzw. rückgängig machen möchte, folgen verschiedene Ansätze zur Versionierung dieser Zeitreihe.

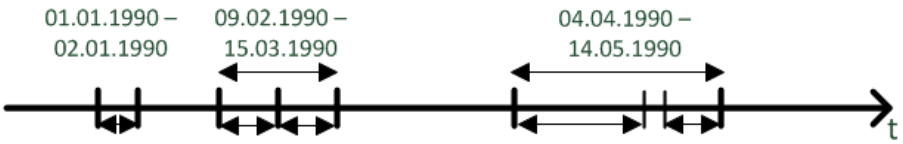

Abbildung 3: Änderungsabschnitte innerhalb einer Zeitreihe (Al12)

Die Herausforderungen bei der Versionierung von Zeitreihen liegen einerseits bei der Performanz und anderseits beim Speicherplatzbedarf, wobei sich beide Aspekte gegenseitig negativ beeinflussen. Bei der Entwicklung des GW-Managers sind für den Aufbau eines Versionierungssystems einige vielversprechende Arten der Versionierung analysiert worden, die diesen beiden Ansprüchen Rechnung tragen. Die klassischen Ansätze zur Datenversionierung beschäftigen sich u. a. mit der platzsparenden Datenhaltung sowie der performanten Datenrecherche (KSW86). Die gängigsten Muster sind:

- Vorwärtsorientiert,
    - Unterschiede von Version zu Version
    - Unterschiede von Ursprungsversion zu Version
- Rückwärtsorientiert
    - Unterschiede von Version zu Version
    - Unterschiede von Version zu Ursprungsversion
- Vorgang speichern statt Attributänderung in den Unterschieden

Eine vielversprechende Art ist z. B. die Erstellung einer Baumstruktur zu jedem Datenpunkt und damit zu einem graphentheoretischen Wald für jede Version der Zeitreihe (siehe Abbildung 4). Hier können dann Varianten aufeinander aufsetzen. So kann beispielsweise bei der Datenredigierung der Workflow abgebildet werden, bei dem als erster Schritt Ausreißer eliminiert werden, als zweiter Schritt Datenlöcher gefüllt und dann Trends/Drifts bereinigt werden. Dabei würde die Trendbereinigung als Variante auf die Daten der anderen beiden Varianten als Basis zurückgreifen.

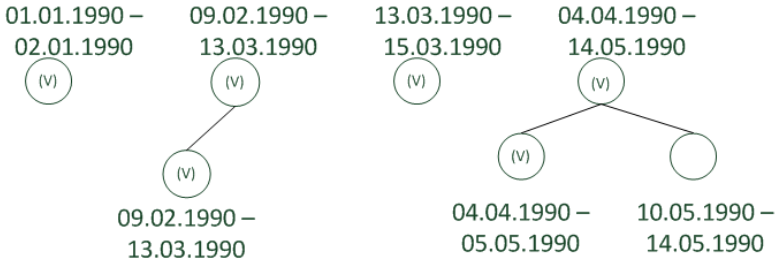

Abbildung 4: Hierarchische Datenstruktur von Zeitreihenabschnitten (Al12)

Jeder Ansatz hat seine Vor- und Nachteile in Bezug auf Redundanz und Komplexität der zu erstellenden Version. Außerdem haben die Arten auch unterschiedliche Anfälligkeiten bezüglich möglicher Konflikte durch Verklemmung (Deadlock). Allen gemeinsam ist aber, dass sie entwickelt wurden, um einen bestehenden Datenblock zu versionieren.

Die zu versionierenden Zeitreihen sind jedoch keine abgeschlossenen Datenblöcke, sondern werden fortlaufend erweitert, da die Sensoren und Logger auch fortlaufend neue Daten erheben. Durch diesen Umstand ist eine Zuordnung der neu hinzukommenden Daten zu einer bestimmten Version nicht eindeutig. Eine andere Art der Variantensysteme ist z. B. die Quellcodeverwaltung (z. B. SVN, GIT). Auch diese sind wenig geeignet fortlaufende Zeitreihen adäquat zu versionieren. Dies liegt daran, dass bei einer Zeitreihe die einzelnen Daten mehr gemeinsamen Bezug besitzen, als die Dokumente bzw. Textzeilen bei einer Quellcodeverwaltung. Beispielsweise beziehen sich Schritte zur Datenbereinigung auf dieselben Daten (Ausreißer- mit Trendbereinigung). Dies ist aber im Kontext einer Quellcodeverwaltung nicht abbildbar, da dort die zweite Änderung die Daten absolut verändert. Würde man in dem Beispiel erst die Trendbereinigung durchführen und dann die Ausreißerbereinigung, dann könnte man zwar die letzte Bereinigung zurücknehmen, nicht aber nur einzeln die Trendbereinigung. Die folgende Tabelle 1 zeigt eine Bewertungsmatrix der unterschiedlichen Variantensysteme in Bezug auf fortlaufende Zeitreihen.

Legende: **++: geeignet** **+ : bedingt geeignet** **- : eher ungeeignet** **--: ungeeignet**	**Speicherplatz-bedarf**	**Deadlocks**	**Abhängige Versionen**	**Zuordnung neu erhobener Daten**
Versionsabschnittsbildung an einer Zeitreihe	+	++	+	++
Autarke Variantenbildung	--	++	-	--
Hierarchische Variantenbildung	+	-	++	+
Variantenbildung mit Bezug auf n-Varianten	++	-	+	+
Quellcodeverwaltung	+	+	--	-

Tabelle 1: Vor- und Nachteile der Zeitreihen-Versionierungsstrategien

5 Projektstand

Bei der Umsetzung des Projekts wurde ein Schwerpunkt auf die Performanz gelegt, so dass der Benutzer effizient mit seinen Massendaten arbeiten kann. Hierfür wurden aus dem Bereich des Big Data die klassischen Werkzeuge analysiert. Durch die Anforderungen der Interaktion ist aber keines der gängigen Big Data Werkzeuge für dieses Projekt einsetzbar. Die Werkzeuge „Hadoop“, „Big Table“, „Map Reduce“ etc. sehen alle den Informationsfluss in eine Richtung vor, so dass zwar schnell Daten-Aggregationen aus-

geführt werden können (DG08), der Informationsfluss zurück in die Datenhaltung jedoch nicht vorgesehen ist bzw. keine Vorteile gegenüber herkömmlichen Werkzeugen (wie relationalen Datenbanksystemen) aufweist. Aus diesem Grund wurde das bestehende Datenhaltungssystem des GW-Managers (ein relationales Datenbanksystem auf Basis des Microsoft SQL-Servers oder alternativ Oracle) genutzt, das bereits alle wesentlichen Aufgaben erfüllt. Hinsichtlich der Verarbeitung von Massendaten, wurde die Datenbank sukzessive optimiert. Hierbei wurden insbesondere die Tabellen-Indexierung sowie die Auslagerung von SQL-Abfragen in Stored Procedures stark ausgebaut: Die Optimierungen, die vorgenommen wurden, sind zum einen die Überprüfung der Indizes einzelner Tabellen, vor allem jedoch die Optimierung aller SQL-Abfragen. Neben der Syntax der SQL-Abfragen konnte durch die Verlagerung der Abfrage aus dem Programmcode in die Datenbank selbst – als Stored Procedure – die Geschwindigkeit der Abfrage um den Faktor 15 erhöht werden.

Auch hat sich herausgestellt, dass im Programmcode selbst (.Net 4.5) noch eine Reihe von Optimierungsmöglichkeiten liegt. Trotz der in .Net 4.5 enthaltenen Garbage-Collection (GC) ist die Datenhaltung der Zeitreihen halbautomatisch implementiert worden. So wird zwar die GC normal von der Anwendung genutzt, wenn aber Zeitreihen entfernt werden, wird explizit die GC angesteuert, so dass der Speicher schnellstmöglich freigegeben wird. Dies ist erforderlich, da .Net-Anwendungen immer nur einen limitierten Hauptspeicher belegen dürfen. Bei 32-bit Betriebssystemen (die das Projekt auch bedienen soll) sind dies ca. 1,5 GByte maximale Speicherbelegung.

Ein weiterer wichtiger Schritt der Umsetzung lag in der Suche nach einem geeigneten Chart-Werkzeug. Hier hat sich gezeigt, dass die meisten Hersteller von sich aus nicht angeben, mit wie vielen Datenpunkten das Werkzeug noch effizient bedienbar ist. Lediglich einige Werkzeuge sind auf den Einsatz von Massendaten (> 500.000 Datenpunkten) ausgelegt. Die Wahl ist auf das Werkzeug TeeChart von Steema gefallen (St13). Hier sind mit den aktuellen Tests keine Performanzprobleme mit großen Zeitreihen aufgetreten. Auch eine Bearbeitung mittels Drag ‘n Drop kann durchgeführt werden.

6 Fazit und Ausblick

Das hier vorgestellte Projekt ist abgeschlossen und wird bereits produktiv eingesetzt. Die Massendaten können dabei direkt aus den Sensoren und Loggern in das relationale Datenbanksystem des GW-Managers importiert werden. Auch können Prozessleitsysteme angebunden werden, so dass alle wasser- sowie abwasser-relevanten Daten vollständig verfügbar sind. Mit dem Schritt der Datenerfassung in das bestehende Datenhaltungssystem, wurden alle Anforderungen bezüglich Datenarchivierung erfüllt. Dabei mussten jedoch zahlreiche Datenbankoptimierungen vorgenommen werden, um die Performance zu verbessern. Die Verbesserung von Indexierungen sowie der Einsatz von Stored Procedures führten hierbei zu Performance-Steigerungen um den Faktor 15.

Die Komponente zur Datenredigierung wurde fertiggestellt, so dass hier nun leistungsstarke Funktionalitäten zur Verfügung stehen (siehe Abbildung 5). Die Komponente verfügt über Werkzeuge zur Datenredigierung (z. B. Datenübernahme aus anderen

Messstellen, lineare Trendbereinigung), direkte grafische Datenbearbeitung (Drag ‘n Drop) und komfortable Datenvisualisierung (Zoom, Diagrammtyp Linie, Balken und Punkt etc.). Zur Datenanalyse sind verschiedene Statistiken (Mittelwert, Varianz, gleitendes Mittel usw.) oder automatisierte Berichte ad-hoc erstellbar.

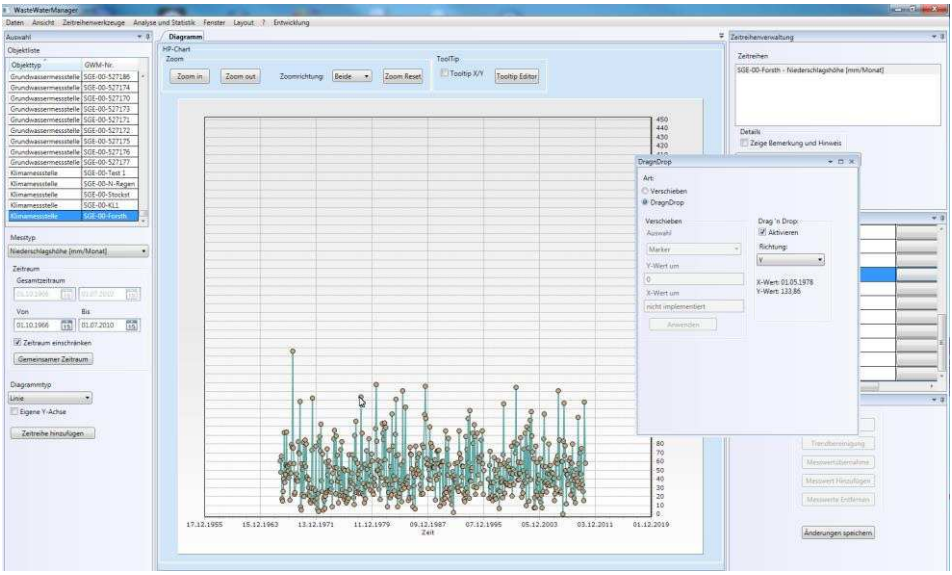

Abbildung 5: Datenredigierung mit dem GW-Manager

Die Versionierung redigierter Zeitreihen wurde in einer ersten „einfachen“ Variante realisiert. Die gewählte Variante unterstützt dabei lediglich eine Versionsebene. Die in diesem Beitrag vorgestellte „Baumstruktur“ (siehe Abbildung 4) soll Gegenstand zukünftiger Entwicklungen sein. Bei allen Weiterentwicklungen muss dabei die Wechselbeziehung zwischen einer performanten Datenbank-Anwendung und den hohen Anforderungen an eine fortlaufende Versionierung berücksichtigt werden.

Literaturverzeichnis

[Al12] Albrecht, Andreas: Entwicklung eines Prototyps zur hochperformanten grafischen Darstellung, Editierung und versionierbaren Speicherung von Massendaten. Abschlussarbeit Hochschule Darmstadt. Darmstadt, 2012.

[DG08] Dean, J.; Ghemawat, S.: MapReduce: simplified data processing on large clusters. Communications of the ACM, 51(1), S.107-113. New York, 2008.

[en14] envi-systems GmbH: GW-Manager, Available at http://www.gw-manager.com/, 2014.

[KSW86] Klahold, P.; Schlageter, G.; Wilkes, W.: A general model for version management in databases. Gesamthochschule, FernUniversität Hagen, Hagen 1986. Available at: http://www.vldb.org/conf/1986/P319.PDF?origin=publication_detail [Zugegriffen am 21.April 2014].

[St13] Steema: Steema Support Central - Charting Components by Steema, Chart for .NET, VCL, Delphi, ActiveX, Java, PHP, Mobile Chart for Android and Phone 7. Available at: http://www.teechart.net/, 2013.