Architektonische Überlegungen zum Aufbau eines verteilten Küsteninformationssystems für den Jadebusen

Dr. Rainer Roosmann

C.v.O. Universität Oldenburg
Institut für Chemie und Biologie des Meeres
Arbeitsgruppe Integrative Modellierung
mailto:rainer.roosmann@uni-oldenburg.de

1. Küsteninformationssystem für den Jadebusen

Im Rahmen des Jadebusen-Projektes, einem durch die Volkswagenstiftung geförderten Forschungsprojekt der Universität Oldenburg, soll ein umfassendes natur- und kulturwissenschaftliches Gesamtbild des Jadebusens und dessen Umfeldes erstellt werden. Es handelt sich um ein interdisziplinär angelegtes Forschungsprojekt, dass die Geosphäre, Bioshphäre und Anthroposphäre betrachtet und die verschiedenen Kompartimente miteinander in Beziehung setzt. Aus diesen fachthematischen Untersuchungen ergeben sich die Anforderungen an das im Rahmen des Teilprojektes 2 zu erstellende Niedersächsische Küsteninformationssystem (NdsKIS). Räumlich wird sich NdsKIS für dieses Projekt mit dem Jadebusen befassen. Ziel ist es, die Anforderungen an ein solches Küsteninformationssystem zu erfassen, auf dieser Basis eine realisierbare Software-Architektur zu erstellen und die Einhaltung funktionaler, wie auch nicht-funktionaler Anforderungen prototypisch nachzuweisen.

2. Software-Architektur

Die Software-Architektur stellt einen relativ jungen Teilbereich des Softwareengineerings dar. Als Software-Architektur eines Softwaresystems versteht man das Konzept, das diesem System zugrunde liegt. Demnach hat jede Software eine - mehr oder weniger gute - Software-Architektur. Sollen die Wege die zu dieser Architektur geführt haben, wie auch die Ergebnisse dieses Prozesses kommuniziert werden, wird eine Architekturbeschreibung benötigt, die häufig über Sichten und entsprechende Modelle beschrieben wird.

Somit subsummiert der Begriff Software-Architektur folgende Aktivitäten und Artefakte, (Erweiterung von [Vogel, 2009]):

$$\textit{Software-Architektur}_{[\text{Disziplin}]} = \textit{Software-Architektur}_{[\text{Prozess}]} + \textit{Software-Architektur}_{[\text{Konzept}]} + \textit{[Software-Architektur}_{[\text{Beschreibung}]}\textit{]}$$

Der Software-Architektur Prozess wird in Vorgehensmodellen festgelegt und umfasst mindestens folgende Arbeitsschritte, die iterativ und evolutionär durchlaufen werden:

a. Erfassen der architektonisch relevanten Anforderungen.
b. Identifikation der wesentlichen Systembausteine, deren Kommunikation untereinander und mit der Umwelt (= Systemkontext).

c. Identifikation möglicher Umsetzungsalternativen und technischer Risiken.
d. Machbarkeitsuntersuchungen und Evaluierung der Alternativen.
e. Beschreibung und Kommunikation der Software-Architektur.
f. Umsetzung, Evolution und Validierung der Software-Architektur.

Zur Beschreibung der während des Software-Architektur Prozesses entstandenen Konzepte werden Sichtenmodelle, wie das Zachman-Framework oder der RM-ODP eingesetzt.

Die Erstellung und Beschreibung einer Software-Architektur ist gerade bei der Entwicklung komplexer Softwaresysteme, wie wir sie im Geoinformatik-Umfeld vorfinden, essentiell wichtig. Bis heute wird die Software-Architektur als Disziplin hier leider häufig vernachlässigt und findet sich auch in entsprechenden Forschungsaktivitäten nur in Ansätzen wieder.

3. Anforderungen

Die Basis zur Erstellung einer Software-Architektur stellen die architektonisch relevanten Anforderungen dar. Anforderungen lassen sich in funktionale und nicht-funktionale Anforderungen differenzieren. Funktionale Anforderungen beschreiben die Aktivitäten, die ein Anwender von dem Softwaresystem erwartet, um seine Arbeitsprozesse erfolgreich durchzuführen. Die nicht-funktionalen Anforderungen hingegen beschreiben eher technische oder organisatorische Rahmenanforderungen, aber auch Qualitätsanforderungen, wie z. B. Zeit- oder Ressourceneffizienz.

Speziell die Qualitätsanforderungen treiben die Software-Architektur. Deren Einhaltung sollte während des gesamten Entwicklungsprozesses über definierte, quantifizierbare Qualitätsszenarien geprüft werden. Grundsätzlich sollten immer die Anforderungen die architektonischen Entscheidungen beeinflussen und nicht die Technologie.

Die Realisierbarkeit der Anforderungen ist in Machbarkeitsuntersuchungen unter Berücksichtigung definierter Qualitätsszenarien zu ermitteln. Wichtig ist in diesem Zusammenhang, Anforderungen zu priorisieren. Ist es beispielsweise wichtiger INSPIRE-konform zu sein, als der Qualitätsanforderung Zeiteffizienz zu genügen?

Wie schon [Goodchild, 2000] feststellt, geht es bei vielen Fragestellungen im marinen und Küstenzonen-Bereich um Anforderungen, die mit heutigen Geoinformationssystem (GIS) nicht oder nur sehr schwer umsetzbar sind. Seit vielen Jahren wird eine Integration der dritten räumlichen Dimension in die Geoinformationssysteme gefordert, die allerdings bis heute nicht umfassend gelungen ist. Eine weitere Herausforderung stellt die Berücksichtigung von Veränderungen im Laufe der Zeit dar. Diese Veränderungen verhalten sich im marinen Umfeld größtenteils nicht linear. [Langran, 1992] veröffentlichte eine erste Zusammenfassung möglicher Lösungsansätze, die von verschiedenen anderen Forschern aufgegriffen und erweitert wurden [Worboys, 1998], [Couclelis, 1999], [Roosmann, 2003] . Einen Einzug in GIS haben diese Konzepte nur in Ansätzen gefunden. Dies liefert die Begründung für den aktuellen Zustand: Bei den Projektpartnern des Jadebusen-Projektes finden sich viele spezialisierte, proprietäre Speziallösungen, die in das NdsKIS zu integrieren sind.

4. Architekturstile und Standardarchitekturen

Aufgrund der Anforderungen wird das NdsKIS als verteiltes Softwaresystem aufgebaut. In der Informatik existieren Architekturstile, die Vorgaben für bestimmte Softwarefamilien festlegen. Diese Vorgaben ergeben sich aus Vorgehensweisen, die in der Praxis nachweislich zum Erfolg geführt haben. Eine Renaissance erlebten die Architekturstile mit der Dissertation von Roy Fielding zum Thema REST [Fielding, 2000]. Architekturstile gehen auf die Arbeit von [Shaw, 1996] zurück, die über Komponenten, Konnektoren und deren Komposition, Leitlinien für die Gestaltung verschiedener Softwaretypen beschreiben.

Der REST-Architekturstil ist die Grundlage für eine Ressourcen-Orientierte Architekturen (ROA), wie von [Richardson, 2007] vorgestellt. Während eine ROA Ressourcen und die Arbeitsweise des WWW in den Mittelpunkt stellt, sowie auf dem REST-Architekturstil aufsetzt, versuchen Service-Orientierte Architekturen die Art und Weise wie Desktop-Applikationen entwickelt werden, auf verteilte Systeme zu übertragen. Einer SOA können verschiedene Architekturstile zugrunde liegen, wobei häufig der Stil "Brokered Distributed Objects" Anwendung findet, bei dem Objekte bestimmte Dienste realisieren. Diese Dienste werden über einen zentralen Verzeichnisdienst angesteuert, so dass die Identität der implementierenden Objekte in den Hintergrund tritt. Ein Enterprise Service Bus (ESB) fungiert häufig als Message-Broker, um eine lose Kopplung zwischen den Objekten zu realisieren und andere häufig verwendete Dienste zentral anzubieten.

Die Diskussion, welcher dieser Architekturstile im Geoinformatik-Umfeld optimal eingesetzt werden kann, wird seit Jahren sehr kontrovers geführt. INSPIRE und GDI-DE favorisieren einen Service-Orientierten Ansatz unter Verwendung von OGC Web-Services (OWS) mit SOAP als Anwendungsprotokoll. Die Grundlage dieser Entscheidung stellen entsprechende Untersuchungen dar, wie beispielsweise [JRCSOAP, 2008] und [JRCREST, 2008].

Die Beschreibung der Arbeitsprozesse durch die Projektbeteiligten fordert eine ressourcen-orientierte Vorgehensweise. Komplexe Prozessierungen, wie wir sie in der marinen Geoinformatik anfinden, zu nennen sei beispielweise die räumliche Interpolation von Sedimentparametern über ein "Krigging with External Drifts" [Verfaillie, 2009], würde niemals als Service angeboten werden. Stattdessen werden die Ressourcen angeboten, nämlich entweder die original Punktdaten oder die prozessierten Rasterdaten.

Der Blog-Eintrag "SOA ist Dead; Long Live Services" von Anne-Thomas Mannes Anfang 2009 führte zu heftigen Diskussionen, bringt aber den wesentlichen Aspekt für die Probleme auf den Punkt: SOA ist keine Technologie! Entspricht die Arbeitsweise nicht den SOA-Erfordernissen, kann eine SOA nicht zum Erfolg führen. Grundsätzlich gilt, dass Services und Daten nah beinander liegen sollten, da es ansonsten zu großen Performance-Problemen kommt. Derjenige der die Daten verwaltet, hat häufig kein Interesse, einen Service zu implementieren und in sein System zu integrieren, den ein Dritter benötigt. Dies spiegelt das eigentliche Problem einer SOA wieder, das auch in diesem Projekt dazu führen würde, dass wieder nur reine Basisdienste angeboten würden.

Ansätze aus dem Bereich der komponentenbasierten Softwareentwicklung, wie sie z. B. mit der Service-Component Architecture (SCA) in [Marino, 2009] aufgezeigt werden, könnten hier eine mögliche Lösung sein, die aktuell untersucht wird.

5. Zusammenfassung und Fazit

Die Software-Architektur stellt eine wichtige Teildisziplin des Softwareengineerings dar, die im Geoinformatik-Umfeld häufig vernachlässigt wird, aber gerade bei großen, verteilten Softwaresystemen essentiell wichtig ist. Die Software-Architektur beschäftigt sich mit den wesentlichen Systembausteinen, also der Struktur der Software, aber auch mit der Kommunikation der einzelnen Systembausteine untereinander und der Systemumgebung zur Laufzeit, was dem dynamischen Verhalten entspricht.

Bei dem neu aufzubauenden Niedersächsischen Küsteninformationssystem (NdsKIS) handelt es sich um ein komplexes, verteiltes Softwaresystem. Getrieben wird die Software-Architektur dieses Systems von den Anforderungen der Projektteilnehmer des Jadebusen-Projektes. Herausforderungen stellt u. a. die Integration verschiedener verteilter, häufig proprietärer Legacy Systeme dar, die räumliche, temporale, wie auch spatio-temporale Objekte nutzen und verarbeiten. Dynamik ist weder in Geoinformationssysteme, noch in den OGC Web Services (OWS) umfassend integriert.

Die Entwicklung einer Software-Architektur für komplexe verteilte Systeme entsteht immer auf den Erkenntnissen vorheriger, erfolgreich durchgeführter Projekte. Dies minimiert das Risiko des Scheiterns enorm. Architekturstile und Standardarchitekturen definieren Vorgaben, die in der Praxis erfolgreich angewendet werden konnten. Zu nennen sind die Ressourcen-Orientierte Architektur (REST), die Service-Orientierte Architektur (SOA), wie auch die Service-Component Architecture (SCA).

Die Arbeitsweise der Projektteilnehmer zeigt eindeutig, dass eine serviceorientierte Philosophie diesem Projekt nicht zugrunde gelegt werden kann. Der ressourcen-orientierte Ansatz entspricht diesem weit mehr. Da die Ressourcen verteilt im Netz liegen, müssen wir uns über die Reduzierung der Datenmenge Gedanken machen. Hier können Ideen der komponentenbasierten Softwareentwicklung, wie sie in der Service-Component Architecture (SCA) zu finden sind, zielführend sein.

Literatur:

[Couclelis, 1999] Couclelis, H., Space, Time And Geography, in: Longley, P., et.al., Geographic Information Systems, John Wiley & Sons, 1999

[Fielding, 2000] Fielding, R., Architectural Styles and the Design of Network-based Software Architecture, Dissertation, University of California, Irvine, 2000

[Goodchild, 2000] Goodchild, P., Foreword, In: Wright, W., Wright, D., Bartlett, D., Marine and Coastal Information Systems, Taylor & Francis, 2000

[JRCSOAP, 2008] JRC Scientific and Technical Reports, Inspire Network Services - SOAP Framework, 2008, http://inspire.jrc.ec.europa.eu/reports/ ImplementingRules/network/INSPIRE_NETWORK_SERVICES_SOAP_ Framework.pdf

[JRCREST, 2008]. JRC Scientific and Technical ReportsRessource Oriented Architecture And REST, 2008 http://inspire.jrc.ec.europa.eu/reports/

ImplementingRules/network/Resource_orientated_architecture_and_REST.pdf

[Langran, 1992] Langran, G., Time in Geographic Information Systems, Taylor & Francis, 1992

[Marino, 2009] Marino, j., Rowley, M., Understanding SCA, Addision-Wesley Longman, 2009

[Roosmann, 2003] Roosmann, R. et.al., Modeling Spatiotemporal Objects And Processes As A Basis For Monitoring The Environmental Influences Caused By Deep Hard Coal Mining, In: Troch, I., Breitenegger, F., Proceedings 4th MATHMOD Vienna, Feb. 2003, ARBESIM-Report 24, Vol. 1, ARGESIM-Verlag, 2003

[Shaw, 1996] Shaw, M., Garlan, D., Software Architecture: Perspectives on an Emerging Discipline, Upper Saddle River, NJ: Prentice-Hall, 1996

[Verfaillie, 2009] Verfaillie, E., et.al., Geostatistical Modeling Of Sedimentological Parameters Using Multi-Scale Variables: Application Along the Belgian Part Of The North Sea, International Journal of Geographic Information Science, Vol. 23, Nos. 1-2, Jan. - Feb., Taylor & Francis, 2009

[Vogel, 2009] Vogel, O., et.al., Software-Architektur: Grundlagen - Konzepte -Praxis, Spektrum Akademischer Verlag, 2. Auflage, 2009

[Worboys, 1998] A Generic Model For Spatio-temporal Geographic Information, In: Egenhofer, M., Golledge, R., Spatial And Temporal Reasoning In Geographic Information Systems, Oxford University Press, 1998