Hochschule Neubrandenburg
University of Applied Sciences

Fachbereich Landschaftsarchitektur, Geoinformatik, Geodäsie und Bauingenieurwesen (LGGB)

Studiengang Geoinformatik

Einsatzfähigkeit aktueller Datenbanksysteme für Überwachungsmessungen

Bachelorarbeit

Vorgelegt von *Björn Schweimler*

Eingereicht beim LGGB der Hochschule Neubrandenburg

09.September 2010

Betreuer: Prof. Dr.-Ing. Karl Foppe
Prof. Dr.-Ing. Andreas Wehrenpfennig

urn:nbn:de:gbv:519-thesis2010-0147-8

Zusammenfassung

In dieser Bachelorarbeit wird die Einsatzfähigkeit aktueller Datenbanksysteme für Überwachungsmessungen geprüft. Zum besseren Verständnis werden zu Beginn Grundlagen bezüglich Überwachungsmessungen, dem Monitoringsystem DABAMOS und ausgewählten Datenbankkonzepten erläutert. Aufbauend darauf werden Kriterien bestimmt, nach denen die Eignung dreier freier Datenbanksysteme eingestuft wird. Diese Anforderungen werden genutzt, um das geeignetste System zu wählen und schließlich einen entsprechenden Entwurf anzufertigen.

Abstract

In this bachelor thesis the operational capability of current database systems will be tested for monitoring measurements. For a better understanding there will be explained basic knowledge concerning monitoring measurements, the developed system DABAMOS and most important database concepts first. After that there will be defined criterias for the suitability of three database systems. These requirements are used to select the most suitable system and ultimately prepare a draft.

Aufgabenstellung

Einsatzfähigkeit aktueller Datenbanksysteme für Überwachungsmessungen

Motivation

Tragische Unfälle jüngster Zeit wie der Einsturz der Eissporthalle in Bad Reichenhall oder die Hangrutschungen in einem früheren Tagebaugebiet in Nachterstedt in Sachsen-Anhalt zeigen die große Bedeutung der Beurteilung der Standfestigkeit bestehender Bauwerke und Rutschhänge. Seit geraumer Zeit werden gefährdete Objekte mittels ingenieurgeodätischer Präzisionsmessungen überwacht. Modernes Instrumentarium und die rechentechnischen Möglichkeiten aktueller Datenverarbeitung und -übertragung erlauben den Aufbau vollautomatischer Monitoringsysteme zur permanenten Überwachung dieser gefährdeten Objekte.

Aufgabe

Im Rahmen eines intern geförderten Forschungsprojektes der Hochschule Neubrandenburg wird ein datenbankorientiertes Monitoring-System zur Durchführung und Analyse von permanenten Überwachungsmessungen ("DaBaMoS") entwickelt. Eine Besonderheit des Systems "DaBaMoS" stellt die konsequente Ausnutzung der Möglichkeiten moderner Datenbanksysteme dar. Der Kandidat hat die Aufgabe, im Rahmen dieser Arbeit verschiedene Datenbanksysteme auf ihre Eignung für moderne Monitoringsysteme zu untersuchen. Aus seinen Erkenntnissen ist der Aufbau einer neuartigen, für diesen Anwendungsfall optimierten, leistungsfähigen Datenbank zu planen.

Selbstständigkeitserklärung

Hiermit bestätige ich, Björn Schweimler, die vorliegende Bachelorarbeit „Einsatzfähigkeit aktueller Datenbanksysteme für Überwachungsmessungen“ selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt zu haben. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Neubrandenburg, 09.09.2010

_____
(Björn Schweimler)

Inhaltsverzeichnis

1. Einleitung

Im Rahmen der Lehrveranstaltung Sensorik an der Hochschule Neubrandenburg ist das datenbank – orientierte Monitoringsystem DABAMOS entstanden. Dieses wurde später zu einem internen Forschungprojekt erweitert. Auf Grundlage des Systems hat sich das Thema dieser Bachelorarbeit, die Überprüfung der Einsatzfähigkeit aktueller Datenbanksysteme in Bezug auf Überwachungsmessungen, ergeben.

DABAMOS ist eine Software, welche vorallem für die Ansteuerung geodätischer Sensoren entworfen wurde. Das System dient der Durchführung von Überwachungsmessungen. Mögliche Messobjekte sind hierbei Brücken, Talsperren aber auch Rutschhänge.

Ein wesentlicher Bestandteil des Programms ist die Speicherung der an die Instrumente versendeten und auch empfangenen Daten. Die wichtigsten Kriterien in diesem Zusammenhang sind Sicherheit und Effizienz. Die Daten sind Grundlage für weitere Verarbeitungsprozesse zur Auswertung der Messungen, bspw. Zeitreihenanalysen.

Ziel der Arbeit ist es zum Einen, die Vorteile von Datenbanksystemen gegenüber von häufig in der Vermessung verwendeten Dateisystemen darzustellen. Zum Anderen sollen Grundlagen, sowie Anforderungen herausgearbeitet werden, sodass eine begründete Auswahl eines Systems erfolgen kann. Auf Grundlage dieser Analyse soll außerdem ein Entwurf entstehen, der eine Implementierung der Datenbank in DABAMOS ermöglicht.

2. Geodätische Überwachungsmessungen

Geodätische Überwachungsmessungen sind ein Teilbereich der Ingenieurvermessung und befassen sich mit der Überwachung von Bauwerken und von Teilen der Erdoberfläche, bspw. Hängen und Gletschern. (1)

Zu diesem Zweck erfolgt die Bestimmung und geometrische Beschreibung von Deformationen. Das Ziel ist es dabei Abweichungen bezüglich des erwarteten Verhaltens des Messobjektes festzustellen, oder ein bestimmtes Verhalten, bspw. Tages- und Jahresgänge bei Gebäuden, nachzuweisen. Überwachungsmessungen kommen vorallem bei der Feststellung von Schadensbildern an Gebäuden, bspw. Rissbildungen, sowie zur Beweissicherung und Absicherung von Bautätigkeiten, wie dem U-Bahn-Bau, zum Einsatz. (1)

Hintergrund sind dabei zum Einen die Gewährleistung von Sicherheit, auch in Bezug auf Schadensersatz- und Haftungsfragen. Zum Anderen spielen Kostengründe eine entscheidende Rolle, da die Beseitigung von früh erkannten Mängeln meist günstiger als nachträgliche Reparatur oder Sanierung ist. Desweiteren können durch geodätische Überwachungen Beeinträchtigungen durch die Sperrung von Straßen o.Ä. vermieden werden. (2)

Der Ablauf, des auch Monitoring genannten Teilbereichs der Ingenieurvermessung, beginnt immer mit dem Messobjekt selbst. Dieses wird sowohl im Geometrie- als auch im Zeitbereich diskretitiert. Darunter wird die Zerlegung des Objektes in kleine aussagekräftige Abschnitte verstanden. Beispiele hierfür sind die Anbringung von Messmarken an vorher bestimmten Punkten und die Auswahl eines geeigneten Zeitintervalls. Auf Grundlage dessen erfolgt die eigentliche Messung. Diese kann bspw. eine tachymetrische Aufnahme eines Hanges sein. Auf Basis der erhobenen Daten folgt die Auswertung sowie die Interpretation der Daten mit Hilfe stochastischer Methoden. Bei der Erfassung eines Deformationsvorganges ist zu beachten, dass der Prozess rekursiv ist. Das bedeutet, dass der Vorgang mehrfach wiederholt wird, wobei einzelne Teilschritte, bspw. die Diskretisierung und die Wahl des Auswertemodells, zu prüfen und gegebenenfalls an die Ergebnisse vorheriger Messungen anzupassen sind. (Abb. 1) (2)

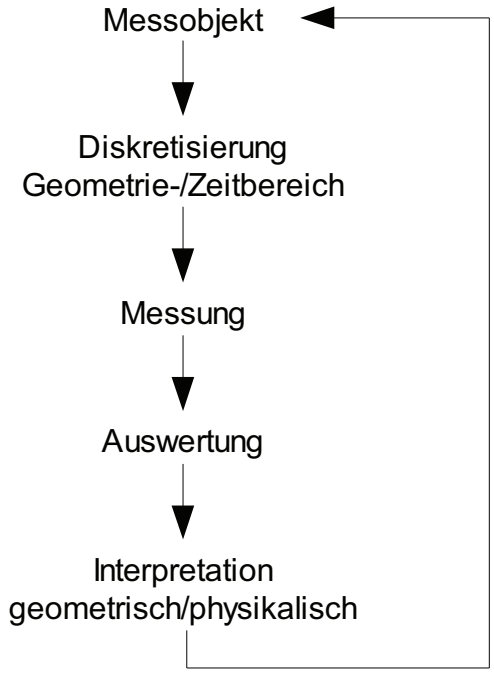

Abbildung 1: Teilschritte bei der Erfassung von Deformationsvorgängen

Während für die Ausführung der Diskretisierung vorallem die Wahl des Gerätes und die Anpassung an dieses entscheidend ist, wird unabhängig davon bei der Messung in kontinuierliche und epochale Messung unterschieden. Letzere wird vordergründig bei Bauwerken eingesetzt, deren Verhalten bekannt sind und deren Veränderungen nur langsam und gleichmäßig auftreten. Dem gegenüber steht die kontinuierliche Messung, welche bei unregelmäßigem und damit schlecht vorhersagbaren Verhalten sowie bei der Notwendigkeit zur permanenten Überwachung angewandt wird. (3)

Bei der Durchführung von Deformationsmessungen kommen häufig computer-gestützte Systeme zum Einsatz. Dabei wird das Messgerät, bspw. ein Tachymeter, über eine Schnittstellte mit einem PC verbunden. Dies führt zu einer Steigerung der Effizienz und zu einer Kostensenkung. Ein großer Vorteil ist die Anzeige der ermittelten Werte in Echtzeit. Auch die Ergebnisse stehen sofort zur Verfügung, da die Analyse der Daten parallel zur Erfassung vorgenommen werden kann. Ein wichtiger Aspekt ist die Datenspeicherung. Der integrierte Speicher in den Überwachungsgeräten ist häufig begrenzt, während computer-gestützte System eine enorme Menge an Daten speichern können. Zusätzlich dazu besteht die Möglichkeit, das System nach belieben zu optimieren. Bei einem sogenannten Stand-

alone-Überwachungsgerät werden Hard- und Softwarefunktionen häufig vom Hersteller vorgegeben und können nicht mehr verändert werden. Mit einem PC besteht die Möglichkeit, benutzerdefinierte Funktionen hinzuzufügen, um das System den individuellen Bedürfnissen anzupassen. Ein weiterer großer Vorteil ist die Netzwerkanbindung des PC. Diese ermöglicht es, Ergebnisse anzufordern, Alarmmeldungen zu versenden oder zu überprüfen, ob das System noch läuft. Auch Zugriffe auf das Messgerät sind so von außen möglich. (4)

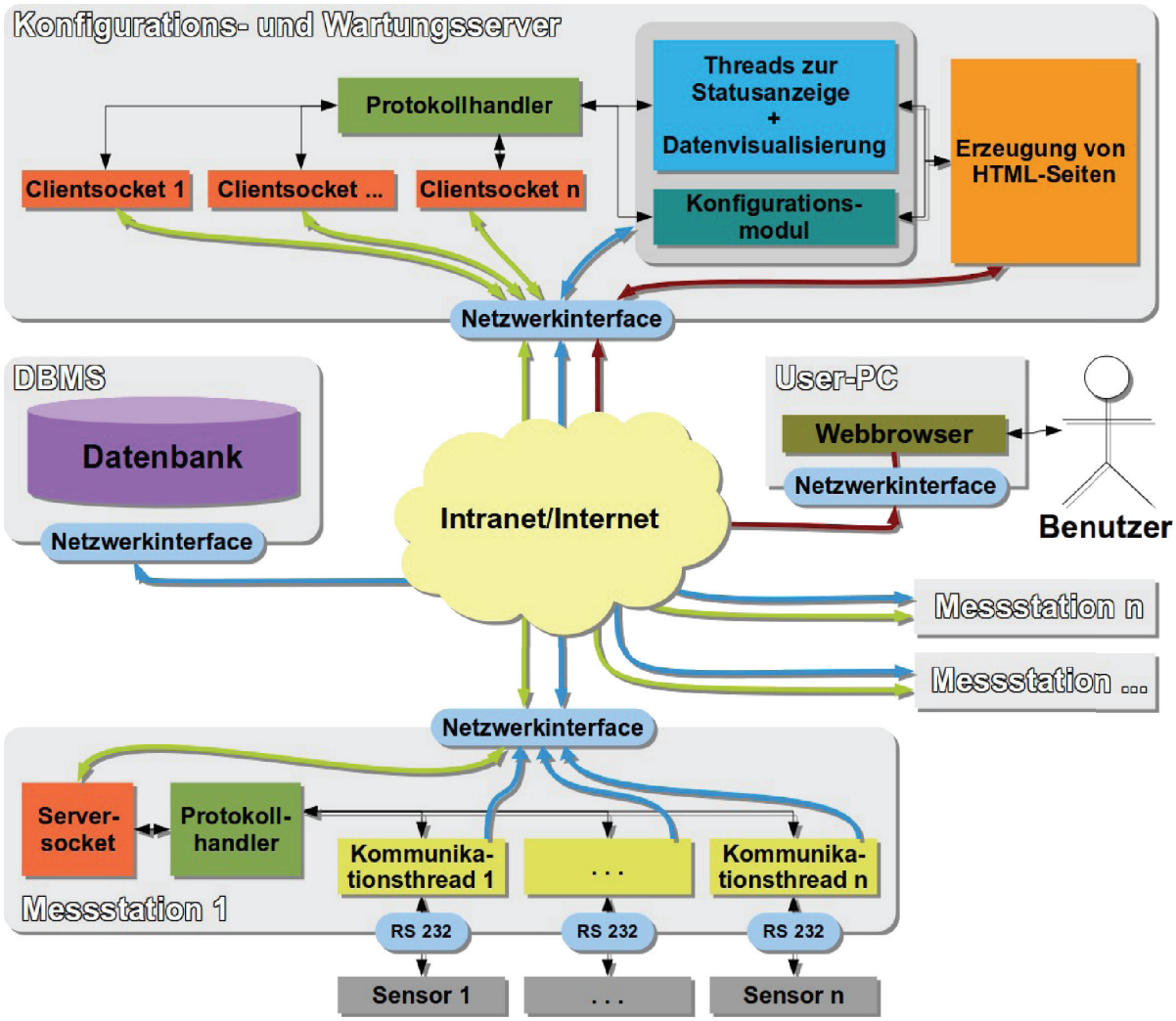

Abbildung 2: Architektur von Dabamos (5)

Ein Beispiel für ein computer-gestütztes Monitoringsystem ist das an der Hochschule Neubrandenburg entwickelte DABAMOS. Diese auf Java basierende Software ermöglicht die Ansteuerung verschiedenster, für die Überwachung geeigneter Geräte und ist dabei plattformunabhängig. Die Kommunikation mit dem Sensor erfolgt über eine serielle

Schnittstelle, über welche ASCII – Zeichen ausgetauscht werden. Das Programm kann sowohl über XML – Dateien als auch mit Hilfe eines Graphic User Interface (GUI) konfiguriert werden.

Die Architektur des Systems (Abb. 2) sieht vor, dass jeder Überwachungssensor und die dazugehörigen meteorologischen Nebensensoren, bspw. ein Thermometer, mit einem eigenen Messcomputer verbunden werden. Dieser übernimmt die persistente Speicherung der Daten in einer Datenbank sowie auch die Alarmierung im Gefahrenfall. Der Zugriff auf die Messcomputer erfolgt über eine TCP/IP – Netzwerkverbindung. Zu diesem Zweck wurde ein Übertragungsprotokoll entwickelt, welches eine Steuerung und Überprüfung der Softwarekomponente auch aus einem Webinterface heraus ermöglicht. Das Resultat der Überwachung mit DABAMOS ist die Visualisierung der erhobenen Werte einschließlich der Anzeige aussagekräftiger Graphen. Zusätzlich dazu ermöglicht eine Alarmfunktion die Benachrichtung bei Gefahr. Das Abrufen der Ergebnisse kann per eMail, über das Webinterface und mit Hilfe von PDF – Dateien erfolgen. (5)

Die beschriebenen computer-gestützten Systeme ermöglichen eine kontinuierliche Überwachung von Bauwerken oder auch Teilen der Erdoberfläche. Eingesetzt werden sie im Tunnel- oder auch im Brückenbau zur Absicherung des Bauvorhabens. Zusätzlich dazu werden diese Systeme in älteren sakralen Bauwerken genutzt, um die Standfestigkeit der Gebäude zu überwachen und so festzustellen, ob und wann teure Neugründungen notwendig werden.

3. Datenbankkonzepte

In diesem Kapitel werden Datenbanken allgemein sowie drei unterschiedliche Konzepte, welche für die Verwendung von DABAMOS in Fragen kommen, beleuchtet. Dabei werden im ersten Unterkapitel allgemeine Eigenschaften und auch Vorteile gegenüber Dateisystemen beschrieben. Die weiteren Unterkapitel beschäftigen sich mit konkreten Konzepten, wobei die Schwerpunkte auf dem Aufbau, den Eigenschaften sowie den Anwendungsbereichen und Beispielen liegen.

3.1. Eigenschaften von Datenbanken

Datenbanken sind ein Bestandteil von Datenbanksystemen (DBS), welche zusätzlich ein Datenbank-Management-System (DBMS) beinhalten (Abb. 3). Die genannten Systeme dienen der Verwaltung von Daten, wobei die Datenbank (DB) den eigentlichen Datenbestand repräsentiert, während das DBMS Aufgaben wie Verwaltung, Steuerung und Kontrolle der Daten sowie die Steuerung des Zugriffs übernimmt. (6)

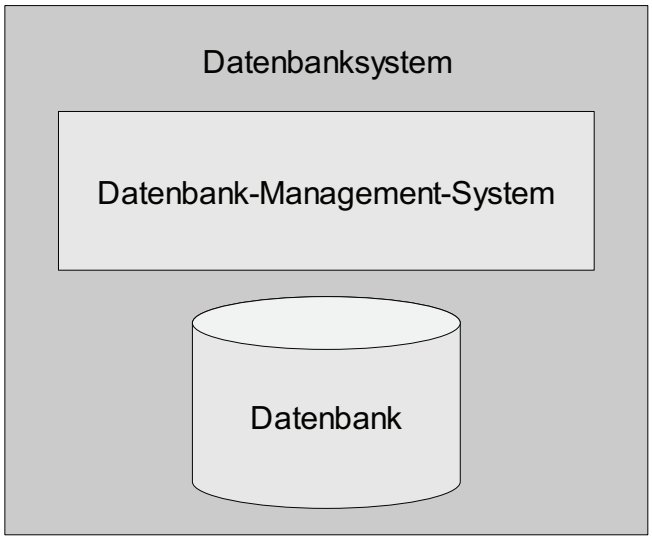

Abbildung 3: Aufbau eines DBS

Jedes Datenbanksystem, ob relational, objektorientiert oder hierarchisch, muss fünf Datenbankgrundsätzen genügen (Tabelle 1). Die Anforderungen sind vom eigentlichen Datenmodell, also der Organisation der Daten innerhalb der Datenbank, unabhängig, weswegen für dieses weitere Regeln definiert werden müssen. (7)

Datenbankgrundsätze	
D1	Dauerhaftigkeit
D2	Große Datenbestände
D3	Mehrbenutzerbetrieb
D4	Wiederherstellbarkeit
D5	Ad-hoc-Abfragemöglichkeiten

Tabelle 1: Datenbankgrundsätze (7)

Unter dem Grundsatz Dauerhaftigkeit wird die persistente Speicherung der Daten über einen beliebig langen Zeitraum zusammengefasst. Die Regel D2 besagt, dass das DBS in der Lage sein muss eine große Menge an Daten mit Hilfe entsprechender Technologien, bspw. Bufferpools, zu verwalten. Desweiteren muss ein Datenbanksystem einen reibungslosen Mehrbenutzerbetrieb ermöglichen, um der Regel D3 gerecht zu werden. Der Grundsatz D4, Wiederherstellbarkeit, gibt vor, dass im Fehlerfall Transaktionen zurückgesetzt und neu gestartet werden können. Unter der letzten Regel D5, Ad-hoc-Abfragemöglichkeiten, wird die Notwendigkeit zusammengefasst, dass ein DBS Abfragesprachen zur Verfügung stellt, um die Daten zu verwalten und zu manipulieren. (7)

Die elementarste Eigenschaft eines Datenbanksystems (DBS) im Vergleich zu einem Dateisystem stellt die Vermeidung von Datenredundanz dar. Dieses Attribut bezieht sich darauf, dass Daten gemäß Grundsatz D1 persistent gespeichert werden, aber nur einmalig im System vorhanden sind. Daraus ergibt sich, dass Speicherplatz gespart werden kann und Änderungen immer global gültig sind, d.h. es kann nicht vergessen werden, lokale Manipulationen zu aktualisieren. In einem Dateisystem gibt es häufig Redundanzen, sodass dort die genannten Vorteile nicht greifen. (8)

Zusätzlich dazu ermöglicht ein DBS eine effiziente Verwaltung und Verarbeitung der Daten. Aufbauend auf Regel D5 kommen zu diesem Zweck Abfragesprachen, zum Beispiel SQL, zum Einsatz. Mit Hilfe dieser ist es möglich, die Struktur der Daten zu definieren, die Daten zu manipulieren und sie näher zu beschreiben. Speziell bei großen Datenmengen sind die Vorteile bezüglich der Performance gegenüber anderen Softwaresystemen, wie bspw. einem

Dateisystem, sehr groß. Diese Eigenschaft wird durch den Grundsatz D2 gestützt. (6) (8)

Eine weitere Funktionalität ist der Parallelzugriff verschiedener Nutzer auf gleiche Daten. Dabei kommt das Transaktionsmanagement eines DBS zum Einsatz, welches mit Hilfe der ACID – Eigenschaften die Datenkonsistenz sichert. Somit kommt es nicht zu Datenverlusten durch unkontrolliertes Überschreiben, wie es etwa bei einem Dateisystem der Fall sein kann. Die genannte Funktionalität vereint die Regeln D3 und D4. (8)

Viele Datenbanksysteme ermöglichen es zusätzlich, die Daten vor unbefugten Zugriffen zu schützen und gewährleistet somit die Sicherheit. Es ist möglich Zugriffsprofile anzulegen, sodass jeder Nutzer nur Zugriff auf bestimmte Bereiche hat und sensible Daten abgeschirmt werden können. (8)

Die beschriebenen Eigenschaften haben zu einer weiten Verbreitung von Datenbanksystemen geführt. Dabei müssen für den Einsatz allerdings verschiedene Charakteristika der Anwendung heraus gefiltert werden, um das passende Datenbankkonzept auszuwählen.

3.2. Relationale Datenbanken

Relationale Datenbanken beruhen auf dem Relationenmodell von Edgar F. Codd und müssen zusätzlich zu den fünf Datenbankgrundsätzen weiteren Regeln genügen (Tabelle 2). (7)

Relationenmodell	
R1	Tabellen
R2	Primärschlüssel
R3	Mengenorientierte Operatoren
R4	Relationenorientierte Operatoren
R5	Viewkonzept

Tabelle 2: Relationenmodell (7)

Regel R1 besagt, dass Daten und deren Beziehungen tabellarisch dargestellt werden. Dabei werden in den Spalten die Merkmale bzw. Attribute dargestellt. Die Merkmalsausprägungen,

also die Datensätze, werden als Tupel bezeichnet (Abb. 4). Die Ordnung der Spalten und Tupel innerhalb der Tabellen spielt dabei keine Rolle. (7)

Die Regel R2 fordert, dass einzelne Merkmale oder minimale Kombinationen einen Tupel innerhalb einer Tabelle eindeutig identifizieren müssen. Diese werden als Schlüsselkandidaten bezeichnet, wobei ein beliebiger Kandiat als Primärschlüssel fungieren muss. (7)

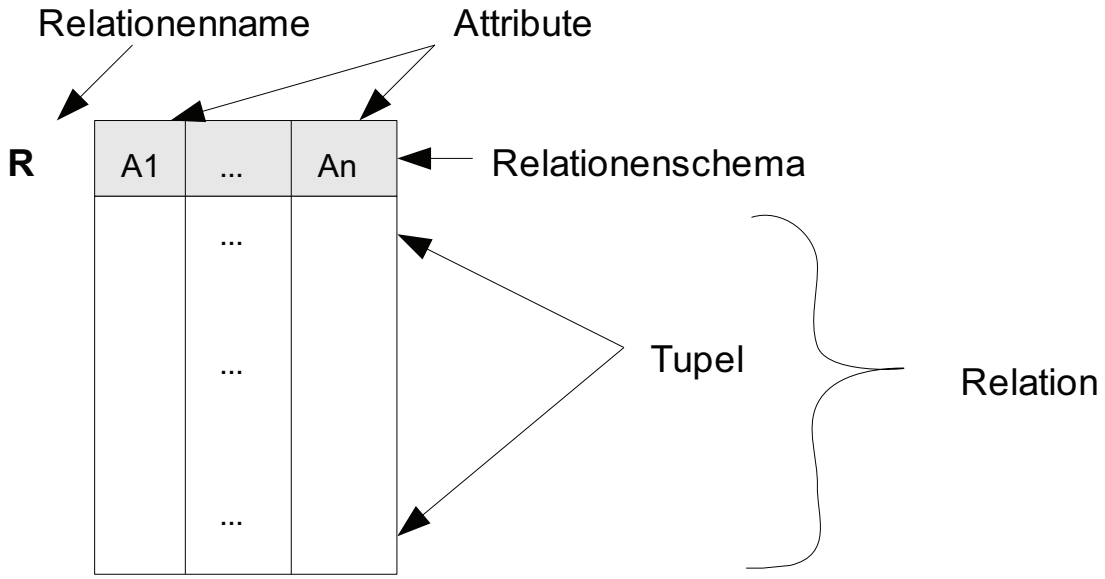

Abbildung 4: Veranschaulichung eines Relationenschemas und einer Relation (8)

In Regel R3, Mengenorientierte Operatoren, wird festgehalten, dass Tabellen mit Hilfe der Mengenoperatoren Vereinigung, Durchschnitt, Differenz und kartesisches Produkt kombiniert werden können. Dabei muss darauf geachtet werden, dass die Anzahl der Merkmale übereinstimmt und dass diese dieselben Wertebereiche haben. (7)

Die Regel relationenorientierte Operatoren gibt Werkzeuge vor, mit denen sich Tabellen durch Filteroperatoren verändern lassen. Die Selektion ermöglicht es bspw. Tupel anhand bestimmter Eigenschaften auszuwählen. Die Projektion dagegen zeigt einzelne Merkmale einer Tabelle. Zusätzlich dazu gibt es weitere Operatoren wie Verbund und Division, die aus zwei Tabellen eine einzelne generieren. (7)

Die abschließende Regel enthält Vorgaben zum Viewkonzept. Sie legt fest, dass verschiedene Sichten über einzelne Tabellen gelegt werden können. Diese dienen Auswertezwecken und

bestehen aus nicht materialisierten Tabellen. (7)

Genügt ein Softwaresystem sowohl den fünf Datenbankgrundsätzen als auch den Regeln des Relationenmodells wird es als relationales Datebanksystem (RDBS) bezeichnet.

Die Abbildung der zu speichernden Daten in einem RDBS erfolgt auf der Grundlage eines Entity Relationship Modells (ERM). In diesem werden die Daten mit Hilfe von Tabellen und Beziehungen modelliert. Die Tabellen müssen dabei so genannten Normalformen (NF) genügen, um Redundanzen, Anomalien, also Widersprüche innerhalb der Daten, sowie einen erhöhten Speicherbedarf zu vermeiden. (8)

Die Umsetzung des logischen Entwurfs erfolgt mit Hilfe einer Datendefinitionssprache (DDL). Diese ist standardmäßig in SQL integriert und ermöglicht das Erstellen von Relationenschemata mit Attributen und deren Wertebereichen sowie Schlüsselattributen, ausgehend von dem ERM. Zusätzlich dazu ermöglicht SQL das Einpflegen der eigentlichen Daten in die Tabellen und damit in die Datenbank. (8)

Sind die Daten persistent in der Datenbank gespeichert, ermöglichen Anfrage- und Manipulationssprachen verschiedenste Operationen. Dazu gehören das Ausblenden von Spalten, das Verknüpfen von Tabellen und viele weitere Möglichkeiten. (8)

Zusätzlich dazu können die Daten mit Hilfe von so genannten Object/Relationship – Mapper (O/R – Mapper) gespeichert werden. Diese Softwarekomponente bildet die Objekte einer objektorientierten Programmiersprache direkt auf der Datenbank ab. Die Grundlage hierfür ist das Datenmodell des jeweiligen Programms. Aus den Klassen, welche die persistent zu speichernden Daten enthalten, werden die Tabellen für die Datenbank generiert. Eine Instanz des Objektes wird dabei als ein Datensatz, also eine Zeile, abgebildet. Die beschriebene Software ermöglicht, equivalent zu den Anfrage- und Manipulationssprachen, verschiedene Operationen auf den Daten. Typische Implementierungen eines O/R – Mappers sind Hibernate und Castor. (9)

Das beschriebenen Datenbankkonzept wurden entwickelt um viele, einfach strukturierte und kleine Objekte abzuspeichern. Dabei gibt es optimalerweise nur wenig Objekttypen, sodass die Zuordnung weniger aufwendig ist. Die zugehörigen Transaktionen sollten dabei sehr kurz sein und nur wenige Objekte betreffen. Die vorgestellten Charakteristika treten vorallem bei

Bibliotheksausleihen, Buchhaltungssystemen, Personaldatenbanken etc. auf, weshalb das relationale Datenbankkonzept häufig in diesen Bereichen zum Einsatz kommt. (8)

Bekannte kommerzielle Datenbanksysteme werden von den Firmen Oracle sowie von Microsoft vertrieben. Diese zeichnen sich durch die professionelle Unterstützung im Einsatz, durch Optimierungs- und Erweiterungsmöglichkeiten, aber auch durch einen hohen Preis aus. Dem gegenüber steht die Open – Source – Datenbank MySQL. Diese hat einen geringen Anschaffungspreis, erfordert allerdings einen hohen Einarbeitungsaufwand. Generell sind kommerzielle Systeme besser für große Anwendungen mit vielen Nutzern geeignet, während sich OpenSource – Datenbanken bei kleinen und mittleren Projekten mit einer geringen Zahl von Anwendern lohnen können. (10)

3.3. Objektorientierte Datenbanken

Objektorientierte Datenbanken sind eine konsequente Weiterentwicklung der relationalen Datenbanken. Sie ermöglichen es Daten direkt als Objekte zu speichern und zu verwalten. Das dazugehörige DBS muss sowohl den fünf Datenbankgrundsätzen als auch den Regeln des Objektmodells (Tabelle 3) genügen.

Objektmodell	
O1	Komplexe Objekte
O2	Objektidentität
O3	Datenkapselung
O4	Typen und Klassen
O5	Vererbung
O6	Polymorphismus
O7	Vollständigkeit
O8	Erweiterbarkeit

Tabelle 3: Objektmodell (7)

Gemäß Regel O1 muss eine objektorientierte Datenbank komplexe Objekte unterstützen, d.h., dass die Objekte Attribute haben können, die wiederum andere Objekte sein können. Dabei können die Attribute sowohl einfache als auch zusammengesetzte Datentypen sein. (7)

Regel O2 gibt vor, dass Objekte unabhängig von den Werten der Attribute eindeutig identifizierbar sein müssen. Die Identität muss systemweit eindeutig und unabhängig vom Platz im Speicher sein. (7)

Der Grundsatz Datenkapselung verlangt das Verbergen von Details der Implementierung nach außen. Dies beinhaltet, dass der Zugriff auf Attribute einer Klasse nur über sogenannte getter- und setter- Methoden möglich ist (Abb. 5). (7)

```
1  private int messwert;
2
3  public void setMesswert(int i) { this.messwert = i;}
4  public int getMesswert() { return messwert;}
5
```

Abbildung 5: Zugriff auf Attribute in Java

Unter Regel O4 wird die Bestimmung zusammengefasst, nach der ein objektorientiertes Datenbanksystem (OODBS) Typen und Klassen unterstützen muss. Als Typen werden Standarddatentypen, bspw. Integer oder String, verstanden, während Klassen die Zusammenfassung gleichartiger Objekte sind. (7)

Grundsatz O5 behandelt die Vererbung in objektorientierten Datenbanken. Dabei kann eine Klasse eigene Eigenschaften haben und auch die Attribute und Methoden einer übergeordneten Klasse erben. Das DBS muss in der Lage sein, diese Unterklassen sowie auch die Oberklassen persistent zu speichern. (7)

Als Polymorphismus wird die vielgestaltige Anwendung von Methodennamen in unterschiedlichen Klassen bezeichnet. Dabei kann die Implementierung der Methoden abweichen. Ein OODBS muss diese Eigenschaft von Objekten gemäß Regel O6 berücksichtigen. (7)

Unter Vollständigkeit wird die Forderung nach einer berechnungsvollständigen Datenbanksprache zusammengefasst. Die Sprache muss es ermöglichen, dass jede Datenbankoperation durch sie ausgedrückt werden kann. (7)

Regel O8, Erweiterbarkeit, stellt die Forderung, dass eigene Typen und Klassen unterstützt werden müssen. Dabei soll kein Unterschied zwischen benutzerdefinierten und

systemdefinierten Klassen existieren. (7)

Erfüllt ein Softwaresystem sowohl die fünf Datebankgrundsätze als auch die acht Regeln des Objektmodells, wird es als objektorientiertes Datenbanksystem (OODBS) bezeichnet. (7)

Die Abbildung der Daten geht bei einem OODBS vom Objektmodell des jeweiligen Programms aus. Das Modell besteht aus einem Grundmodell, einem Strukturmodell sowie einem Verhaltensmodell. Erstgenanntes beinhaltet die Modellierung von möglichst abstrakten Klassen als Mengen von Objekten mit gleichen Attributen und Methoden. Im Strukturmodell werden die Beziehungen der Klassen zueinander beschrieben. Im Verhaltensmodell wird das dynamische Verhalten der Klassen modelliert. Zusätzlich zu den einzelnen Modellen kann durch Subsystembildung und die Verwendung von Frameworks die Wiederverwendbarkeit gesteigert werden, weshalb auch diese Bestandteil des Objektmodells sind (Abb. 6). (7)

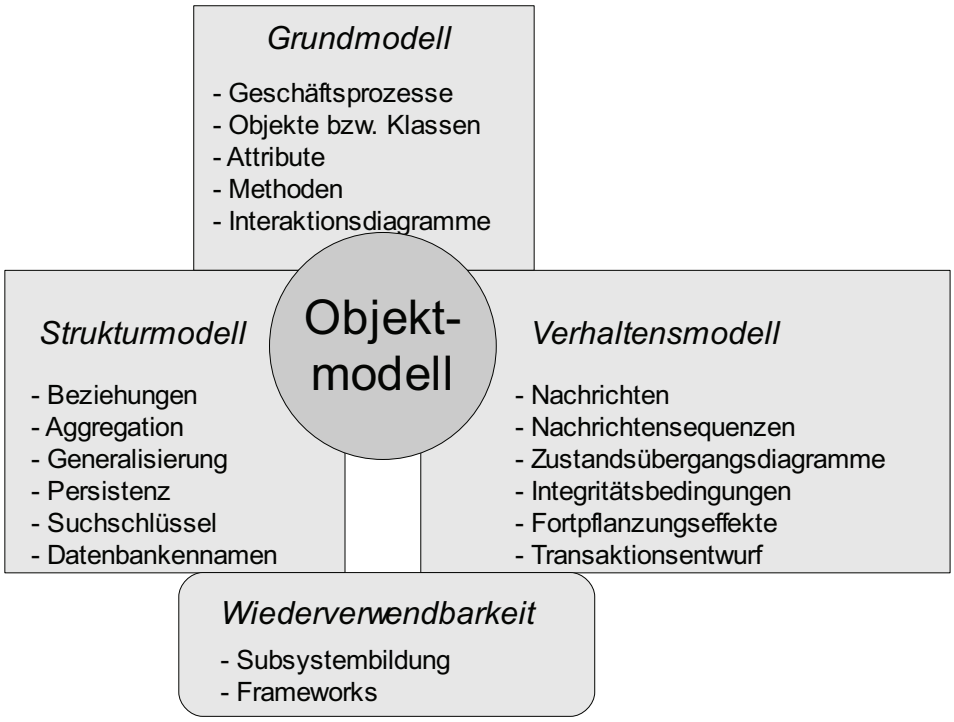

Abbildung 6: Wesentliche Komponenten eines Objektmodells (7)

Der Aufbau des Objektmodells gibt die Vorgehensweise beim Entwurf des Modells und damit auch der Datenbank vor. Auf Basis der Analyse werden geeignete Klassenkanditaten bestimmt. Nachfolgend wird diesen relevante Attribute und Methoden zugeordnet. Der

nächste Schritt ist das Aufdecken von Beziehungen sowie das Festlegen von Persistenzeigenschaften. Abschließend werden Nachrichtensequenzen erfasst, um die Kommunikation der Klassen untereinander zu modellieren, und geeignete Frameworks zur Steigerung der Wiederverwendbarkeit vorgeschlagen. (7)

Ausgehend vom Entwurf kann mit Hilfe von objektorientierten Programmiersprachen wie Java oder C++ die Realisierung und der Zugriff auf die Datenbanken erfolgen. Die Sprachen sind dabei um Datendefinitions- und Datenmanipulationskonstrukte erweitert worden. Zusätzlich dazu gibt es Abfragesprachen. Wichtig ist, dass die Sprachschnittstelle dabei eine strukturbasierte Sprachunabhängigkeit aufweist. Diese Eigenschaft gewährleistet, dass gleichzeitig und mit mehreren Datenbankprogrammiersprachen auf eine Datenbasis zugegriffen werden kann (Abb. 7). Eine weitere Forderung an die Schnittstelle ist die verhaltensbasierte Sprachunabhängigkeit, welche die Verwendung von Methoden der einen Sprache in einer anderen objektorientierten Sprache ermöglicht. Aufbauend darauf ist es möglich, Daten in der Datenbank persistent zu speichern sowie sie zu manipulieren oder zu löschen. (7)

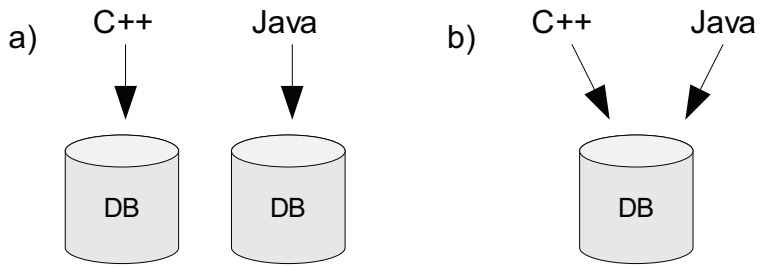

Abbildung 7: a) sprachabhängige und b) sprachunabhängige Schnittstelle (7)

Objektorientierte Datenbanksysteme wurden entwickelt, um den hohen Anforderungen von bspw. CAD – Anwendungen oder auch Expertensystemen gerecht zu werden. Diese enthalten viele stark strukturierte Objekte, welche vielen Objekttypen zugeordnet werden können. Des Weiteren sind die Transaktionen häufig sehr aufwendig und haben eine lange Dauer. Die Verwendung von relationalen Datenbanksystemen ist für diese Aufgaben nicht optimal. (8)

Bekannte kommerzielle Systeme sind FastObjects, GemStone oder auch Itasca. Im Bereich der objektorientierten Datenbanken gibt es, anders als bei RDBS, sehr viele Anbieter, welche sich um diesen relativ kleinen Markt bemühen. Ein sehr bekanntes Open - Source Projekt ist db4o. (7)

3.4. Objektrelationale Datenbanken

Ein objektrelationales Datebanksystem (ORDBS) ist ein Softwaresystem, welches den fünf Datenbankgrundsätzen, den fünf Regeln des Relationenmodells sowie den acht Regeln des Objektmodells genügt. Zusammenfassend ist es ein relationales DBS, welches um Eigenschaften der Objektorientierung erweitert wurde.

Die Speicherung der Objekte erfolgt bei einem ORDBS in Tabellen, die Speicherungskomponente ist also relational. Für die Verwaltung komplexer Strukturen des Objektmodells müssen dabei allerdings erweiterte Datentypen und Speicherungskonzepte zur Verfügung gestellt werden. Ein Beispiel hierfür sind die Binary Large Objects (BLOBs). Diese ermögliches es, lange Textteile oder auch Multimediadateien in der Datenbank abzulegen. Die beschriebenen Strukturen können bei Bedarf verkettet werden und auch in Beziehungen zueinander gestellt werden.

Aus der Speicherung in Relationen ergibt sich, dass bekannte Abfragesprachen, bspw. SQL, mit all ihren Möglichkeiten genutzt werden können. Durch die Erweiterung des Standards erlauben diese dem Nutzer zum Beispiel Hierarchien von Relationen zu bearbeiten. Dadurch können typspezifische Methoden von komplexen Objekten aufgerufen werden.

Das Konzept kommt, ähnlich dem OODBS, vor allem bei komplexen Anwendungen zum Einsatz. Es kann seine Stärken ausspielen, wenn es mit vielen Objekttypen und komplizierten Operationen umgehen muss. Ein wichtiges Einsatzgebiet sind deshalb die Geoinformationssysteme. (11)

Bekannte kommerzielle System sind DB2 UDB und Oracle9i. Im Open – Source – Bereich ist PostgreSQL ein bekannter Vertreter von objektrelationalen Datenbanken. (7)

4. Analyse

Aufbauend auf den Eingangsdaten von DABAMOS werden in diesem Kapitel alle zu speichernden Informationen sowie die Anforderungen an die Datenbank analysiert. Zu diesem Zweck werden im ersten Unterkapitel anfallende Metadaten beleuchtet. In 3.2. geht es um die eigentlichen Sensoren mit ihren Eigenschaften und den Messwerten. Im letzten Abschnitt werden die Anforderungen an die Datenbank zusammengefasst.

4.1. Metadaten

Als Metadaten werden in der Informatik Informationen über andere Daten bezeichnet. Im Fall von DABAMOS handelt es sich um nähere Beschreibungen zur eigentlichen Messung. Der Nutzer muss einmalig vor Beginn einer Überwachung Angaben zum Projekt sowie zu den einzelnen Messstationen und Sensoren machen. Diese Daten müssen persistent in der Datenbank gespeichert werden, um eine Zuordnung der Messwerte zu ermöglichen.

Die Projektdaten werden mit Hilfe einer Eingabemaske (Abb. 8) an das Programm übergeben und dann gespeichert. Standardmäßige Informationen zur Einordnung sind der Projektname, die Beschreibung, der Ort sowie auch der Leiter des Projektes. Wichtige Daten sind vorallem die eMail – Adresse sowie die Telefonnummer. An diese werden im Alarmfall Benachrichtigungen gesendet. Zur Übersicht kann ein Status angegeben werden und wann die letzte Änderung vorgenommen wurde.

Abbildung 8: Eingabemaske Projektdaten

Auf Basis der Architektur von DABAMOS können zu einem Projekt mehrere Messstationen gehören. Der Nutzer muss alle Informationen zur Kommunikation mit dem Rechner, bspw. IP – Adresse und Port, über eine Eingabemaske an das Programm übergeben. Weitere Informationen sind optional.

Zu einer Messstation können beliebig viele Sensoren gehören, wobei es aus Gründen der Übersicht und der Performance ratsam ist, nur jeweils einen Hauptsensor, zum Beispiel ein Tachymeter, sowie die zugehörigen meteorologischen Nebensensoren anzuschließen. Der Nutzer muss alle Informationen bezüglich der Ansteuerung und Kommunikation mit dem Messgerät angeben. Des Weiteren müssen Details zum Messablauf, bspw. das Intervall, sowie Listen mit Befehlen hinterlegt werden. Die Eingabe der Informationen kann über eine GUI oder auch über Konfigurationsdateien erfolgen.

Die beliebig vielen Messwerte eines Sensors sind abhängig vom Gerät und werden nachfolgend detaillierter beschrieben.

4.2. Sensor-Daten

Die in diesem Unterkapitel beschriebenen geodätischen Sensoren sind exemplarisch, denn DABAMOS ermöglicht die Ansteuerung aller per ASCII – Zeichen kommunizierender Geräte. Aus Gründen der Verfügbarkeit wurden allerdings ein Tachymeter, ein Neigungsmesser sowie ein Nivellier ausgewählt, um die Sensordaten beispielhaft darzustellen und so die Anforderungen an die Datenbank zu konkretisieren.

4.2.1. Tachymeter

Das Tachymeter ist ein Allround-Sensor und aus diesem Grund sehr gut für Überwachungsmessungen geeignet. Dieses Messinstrument ist in der Lage Richtungen und Strecken von einem Standpunkt zu mehreren Messpunkten zu messen. Daraus ergeben sich die Lage und Höhe der Punkte, sodass eine Veränderung in allen Richtungen erfassbar und somit eine umfassende Überwachung der Punkte möglich ist. Die Geräte werden nach Genauigkeiten eingeteilt, für das Bauwerksmonitoring werden bevorzugt Präzisionstachymeter mit einer Richtungsgenauigkeit von 0,1 bis 0,5 mgon und einer Streckengenauigkeit im Bereich von 1mm + 1ppm aber auch Ingenieurtachymeter eingesetzt. Zusätzlich zu der hohen Genauigkeit bieten viele heutige Tachymeter weitere Funktionen, wie eine automatischen Zielerfassung (ATR). Des Weiteren sind viele Sensoren mit Schnittstellen ausgestattet, über die mittels Computer mit dem Gerät kommuniziert werden kann. Die genannten Eigenschaften erleichtern den Einsatz bei Überwachungsmessungen und ermöglichen den automatischen Betrieb des Sensors, was häufig zu einer Kostenersparnis führt.

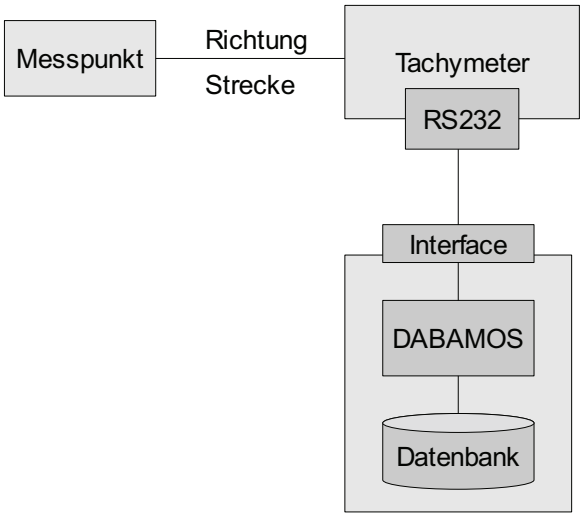

Abbildung 9: Tachymetermessung mit DABAMOS

Das für den Test von DABAMOS eingesetzte Tachymeter ist das Leica TCRP 1203. Diese so genannte Totalstation gehört zur Genauigkeitsklasse der Ingenieurtachymeter und hat eine Richtungsgenauigkeit von 1 mgon und eine Streckengenauigkeit von 1mm + 1,5ppm. Das Gerät bietet zusätzlich zu den normalen Funktionen eines Tachymeters eine ATR. Diese Zielerfassung arbeitet mit Hilfe einer integrierten CMOS – Kamera, um das Prisma zu finden und die Motoren des Gerätes entsprechend zu steuern. Das Resultat ist die autormatische Ausrichtung des Fadenkreuzes auf die Prismenmitte. Eine weitere wichtige Eigenschaft für den Einsatz mit DABAMOS ist die serielle RS232 - Schnittstelle des Sensors (Abb. 9). Auf Grundlage der Leica GeoCOM – Spezifikation können über dieses Interface ASCII - Befehle an das Gerät übertragen und auch die Antworten empfangen werden. Die beschriebene Spezifikation beschreibt sowohl die Übertragungsparameter, bspw. die Baudrate, als auch das eigentliche Protokoll. Es wird demnach genau vorgegeben, wie die Syntax der Anfrage aufgebaut ist und wie die entsprechende Antwort aussieht (Tabelle 4). Der Befehl zum Messen setzt sich dabei aus zwei Anfragen an das Gerät zusammen. Zum Einen muss sich das Tachymeter auf den Punkt ausrichten. Zum Anderen muss das Prisma gefunden und gemessen werden. Die Frequenz der Aufnahme hängt von der Anzahl der Punkte ab. Häufig liegt diese allerdings im Bereich von einem Befehlssatz pro halbe bis eine Stunde je Punkt. Wichtig ist, dass die Software DABAMOS sowohl die gesendeten als auch die empfangen Strings speichert, um eine genaue Dokumentation aller Vorgänge innerhalb der Software zu

gewährleisten. (12)

Sende – und Antwortstring	
Ausrichten	%R1Q,9027:*Hz,V*
	%R1P,0,0:0
Messen	%R1Q,2107:
	%R1P,0,0:0,*Hz,V,SlopeDistance*

Tabelle 4: Leica TCRP 1203 Kommunikationsprotokoll

Zusätzlich zu den Strings des Tachymeters müssen die Werte der so genannten meteorologischen Nebensensoren gespeichert werden. Dabei handelt es sich häufig um multifunktionale Messinstrumente, welche Klimadaten wie Temperatur, Luftdruck und Luftfeuchte erfassen können. Das im Test eingesetzte Gerät ist die Thommen MeteoStation HM30. Auch dieser Sensor kommuniziert über eine RS232 – Schnittstelle und ermöglicht so eine direkete Abfrage (Tabelle 5) der ermittelten Werte. Die Frequenz liegt auf Grund der Trägheit der klimatischen Eigenschaften bei einer Abfrage pro Stunde. (13)

Sende – und Antwortstring	
Abfrage	readall
	(tab)BARO_"value"_"unit"_QNH_"value"_ ...

Tabelle 5: Thommen MeteoStation HM30 Kommunikationsprotokoll

4.2.2. Neigungssensor

Der Neigungssensor ist ein weiteres Gerät, welches bei dem Test von DABAMOS eingesetzt wurde. Die Sensoren sind in der Lage Neigungen und Verformungen von Messobjekten, bspw. Bauwerken, im mm – Bereich zu bestimmen. Vorteile beim Einsatz sind die schnelle Erfassung von Neigungsänderungen sowie die vielseitige Einsetzbarkeit und einfache Handhabung. Nachteilig ist häufig, dass diese Instrumente nicht auf geneigten Flächen eingesetzt werden können, da sie auf dem Prinzip der Flüssigkeitssensoren beruhen. (14)

Vor Beginn einer Messung wird der Sensor mit den Aufsatzflächen auf das Messobjekt aufgesetzt und festgeschraubt. Im Idealfall sollte der Sensor bei einer horizontalen Fläche eine Neigung von Null gon anzeigen. Aufgrund von Dejustierung des Systems ist dies im

allgemeinen nicht der Fall. Bei einer auftretenden Neigungsänderung wird diese durch das Instrument registriert und kann über eine Schnittstelle mit dem PC oder über eine analoge Ausgabe ausgelesen werden. (15)

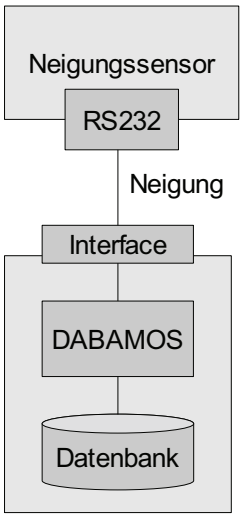

Abbildung 10: Leica Nivel 210

Der im Test verwendete Neigungssensor war das Nivel 210 von Leica. Dieses Gerät gestattet die Messung der Neigung auf zwei Achsen mit einer Auflösung von 0,001 mrad. Die Messfrequenz beträgt bis zu 50 Hz. Zusätzlich zur Neigung wird die Temperatur im Geräteinneren ermittelt. Der Sensor ist ebenfalls mit einer RS232 – Schnittstelle ausgestattet, welche das Anfordern und Auslesen von Messergebnissen mit Hilfe eines Übertragungsprotokolls ermöglicht (Abb. 10). Äquivalent zur Messung mit dem Tachymeter übernimmt DABAMOS die Ansteuerung des Gerätes sowie die Speicherung der Messwerte. Es werden ebenfalls wieder Anfrage- und Antwortstring gespeichert. Eine Auflistung der zu sendenen und empfangenden Messstrings können aus Tabelle 6 entnommen werden.

Sende – und Antwortstring	
Messen	<16><2>NxCX G A <3><13><10>
	<22><2>CxNX X: [...] Y: [...] T:[...]<3><13><10>

Tabelle 6: Leica Nivel 210 Kommunikationsprotokoll

4.2.3. Nivellier

Das Nivellier ist der dritte Sensor, welcher examplarisch für die Speicherung von Messwerten in der Datenbank ausgewählt wurde. Bei Deformationsmessungen werden diese Geräte häufig eingesetzt, um Fassaden zu überwachen und um ein Absacken von Gebäuden festzustellen und so rechtzeitig zu warnen.

Die Höhenbestimmung mittels eines Nivelliers gehört zur geometrischen Höhenmessung. Dabei wird mit Hilfe der Zielachse des Gerätes und den vertikal aufgestellten Maßstäben ein Höhenunterschied ermittelt. Bei der Überwachungsmessung ist dabei zu beachten, dass der Standpunkt des Nivelliers so gewählt wird, dass dieser durch eine mögliche Bewegung des Messobjektes unbeeinflusst ist. (16)

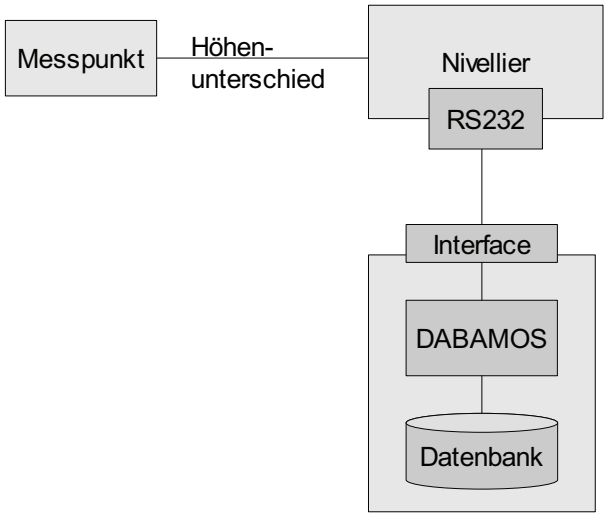

Abbildung 11: Leica DNA03

Der im Test genutzte Sensor war das Leica DNA 03. Dieses Gerät ist ein Digital-Nivellier, welcher für hochpräzise Messungen verwendet wird. Mit einer Genauigkeit von 0,3 mm auf einen Kilometer eignet sich das Gerät hervorragend für Messungen in Nivellements 1. und 2. Ordnung sowie für Überwachungsmessungen. Neben der Genauigkeit ist das Vorhandensein einer seriellen Schnittstelle ein wichtiger Vorteil dieses Gerätes. Die Kommunikation erfolgt, äquivalent zu den anderen Sensoren, über ein ASCII – Übertragungsprotokoll, welches das Setzen von verschiedenen Eigenschaften sowie das Auslösen der Messung ermöglicht (Abb. 11). Tabelle 7 zeigt exemplarisch einen Anfrage – und einen Antwortstring. Das Intervall, mit

der die Messungen ausgelöst werden, kann unterschiedlich gewählt werden. Im Fall einer Beispielmessung mit DABAMOS wurde ein Intervall von 10 Minuten gewählt, sodass in diesem Zeitraum auch die Strings gespeichert werden müssen.

Sende – und Antwortstring	
Messen	GET/M/WI33<CR/LF>
	33..00+00001119

Tabelle 7: Leica DNA 03 Kommunikationsprotokoll

4.3. Zusammenfassung der Anforderungen

Ausgehend von den zuvor beschriebenen Metadaten von DABAMOS und den Messwerten der Beispielsensoren werden in diesem Unterkapitel die Anforderungen an das Datenbanksystem erläutert.

Grundsätzlich kann eine Speicherung der Daten mit jedem der drei Konzepte aus Kapitel 3 erfolgen. Von daher ist die Art der Abbildung der Daten auf der Datenbank kein Ausschlusskriterium, weswegen tiefergehende Bewertungsmerkmale heraus gearbeitet werden müssen (Tabelle 8).

Anforderung	**Priorität**
Open – Source	höchst
Speicherung der Daten	hoch
Geschwindigkeit	hoch
Erreichbarkeit im Netzwerk	hoch
Trigger und stored procedures	hoch
Updates und Stabilität	mittel
Backup und Restore	gering
Clustering und Replication	gering

Tabelle 8: Übersicht Anforderungen und Priorität

Ein erster wichtiger Punkt stellt hierbei die Verfügbarkeit der Datenbank für die eigene Anwendung dar. Aus Kostengründen muss das System frei verwendet werden können, ohne das Gebühren anfallen. Die Schlussfolgerung daraus ist die Verwendung einer Open – Source – Datenbank mit einer entsprechenden Lizenz ohne Kosten.

Das Ziel bei der Verwendung einer Datenbank ist die persistente Speicherung aller übergebenen Daten. Im Vordergrund stehen dabei die Meta- und Sensordaten. Das System muss in der Lage sein, große Mengen an Daten gleichzeitig zu speichern, wobei mehr als 100 Datensätze in der Sekunde von einem Gerät anfallen können. Dies ist dem geschuldet, dass das Nivel 210 in der Lage ist mit einer Frequenz von 50 Hz zu messen. Da immer Anfrage- und Antwortstring gespeichert werden, muss die Datenbank die entsprechenden Anzahl von Daten speichern und auch das Umwandeln des Strings durchführen können.

Zusätzlich zur Speicherung ist die Zugriffszeit beim Abfragen der Daten ein wichtiges Kriterium. Die Messwerte der einzelnen Sensoren sind Grundlage für bspw. Zeitreihenanalysen und müssen dementsprechend verarbeitet werden. Die beschriebene Zeitreihenanalyse ist die Basis für die Auswertung der Überwachungsmessungen, da dadurch Veränderungen an überwachten Objekten detektiert werden können. Das schnelle Auslesen der Informationen aus der Datenbank ist eine Möglichkeit, diesen langwierigen und rechenaufwendigen Prozess zu verkürzen und so zu optimieren.

Neben der eigentlichen Speicherung sind einige weitere Eigenschaften der Datenbank wichtig für die Nutzung mit DABAMOS. Eine große Rolle spielt bspw. die Erreichbarkeit der Datenbank im Netzwerk. Dabei soll nicht nur über das eigentliche Programm der Zugriff auf die Daten ermöglicht werden, sondern auch die Möglichkeit bestehen, mit Hilfe eines Webfrontends Einsicht zu nehmen und auch einen Export von Daten durchzuführen. Hintergrund hierfür ist das Einlesen der Daten in andere Programme, bspw. ein Import in ein CAD – Programm. Zusätzlich dazu sollte die Wartung des Systems über das Frontend durchführbar sein.

Ein weiteres wichtiges Feature sind Trigger und sogenannte stored procedures. Diese ermöglichen die Durchführung von bestimmten Operationen auf dem Datenbankserver. Denkbar wäre eine Zerlegung der Sensorstrings in die einzelnen Messwerte auf der Datenbankseite, bspw. die Verarbeitung eines Strings von einem Tachymeter und die Speicherung von Strecke und Richtung oder auch von Koordinaten. Das hat den Vorteil, dass der Traffic minimiert wird und die Anforderungen an die Messstationen herunter gesetzt werden können, da diese dann keine rechenlastigen Aufgaben übernehmen müssen. Die entsprechenden Operationen werden vom Datenbankserver übernommen.

Ein in Vergleichen von Datenbanken oft vernachlässigter Punkt ist die Verfügbarkeit von Updates und die Stabilität der einzelnen Versionen. Von Vorteil ist hierbei eine aktive Community, welche bei der Pflege der Datenbank, dem Aufdecken von Fehlern und beim Ausbau der Funktionalität wertvolles Feedback gibt. Auch die Erweiterbarkeit durch Zusatzmodule spielt bei der Auswahl eines geeigneten Systems eine Rolle.

Ebenfalls nicht zu unterschätzende Features sind Backup und Restore. Diese ermöglichen sowohl eine regelmäßige Sicherung als auch eine Wiederherstellung der Daten im Fall eines Ausfalls. Die zu wählende Datenbank sollte über entsprechende Funktionen verfügen, um die Sicherheit der Daten zu gewährleisten. Dazu gehört auch eine Funktion zur Protokollierung von relevanten Ereignissen, um ständig über den Zustand des Datenbestandes informiert zu sein.

Zusätzlich zu den genannten Eigenschaften ist die Möglichkeit des Clusterings sehr interessant. Als Cluster wird eine bestimmte Anzahl von vernetzten Computer bezeichnet, auf die die Last verteilt wird. Daraus ergibt sich eine erhöhte Verfügbarkeit und auch eine erhöhte Ausfallsicherheit der Datenbank. Ein wichtiger Vorteil ist die Steigerung der Effizienz und Performance, sodass aufwendige Operationen, wie die Zeitreihenanalyse, schneller durchgeführt werden können. Zusätzlich zum Clustering sorgt das Feature Replication dafür, dass die Datenbank auf andere Systeme gespiegelt und so die Datensicherheit erhöht wird, da ein Ausfall eines Servers keinen Verlust des Datenbestandes zur Folge hat.

5. Entwurf

In diesem Kapitel geht es um die Auswahl einer geeigneten Datenbank auf Grundlage der Anforderungen und um den Entwurf des Datenmodells.

5.1. Auswahl der Datenbank

Die erste Vorgabe ist die Verwendung einer Open – Source – Datenbank. Zu diesem Zweck wird eine entsprechende Vorauswahl getroffen, deren Resultat die Einschränkung auf MySQL, db4o und PostgreSQL ist. Diese drei sind die bekanntesten Vertreter von freien relationalen, objektorientierten und objektrelationen Datenbanken.

Bei db4o, ausgesprochen Database for Objects, handelt es sich um eine objektorientierte Datenbank für Java und .NET. Das System ermöglicht ein direktes Abspeichern der Objekte in der Datenbank und macht so die Modellierung eines ERM unnötig. Der Zugriff auf die gespeicherten Daten erfolgt ebenfalls über die Programmiersprache, eine Verwendung von SQL ist nicht vorgesehen. Die Software wird von der Firma Versant vertrieben und kann sowohl unter einer freien als auch unter einer kommerziellen Lizenz erworben werden. Nachteilig bei db4o in Bezug auf die Verwendung mit DABAMOS ist das Fehlen eines Webfrontends sowie die Problematik des Datenaustausches. Für den Export aus der Datenbank und den Import in eine andere Software müsste ein eigenes Modul entwickelt werden, welches die entsprechende Funktionalität bietet. Dieses Modul kann allerdings in das angedachte Webinterface integriert werden und so den Comfort der Anwendung erhöhen. Ein großer Vorteil ist die geringe Größe der Datenbank kombiniert mit den geringen Anforderungen an die Hardware. Ein entsprechendes System ist also mit wenig Aufwand und Kosten zur Verfügung zu stellen. Zusammenfassend ist db4o ein sehr interessantes und schnelles Datenbanksystem, welches seine Stärken vor allem im Bereich der mobilen Anwendungen ausspielen kann, aber auch für Echtzeitsysteme interessant ist. (17)

Das von Oracle betriebene MySQL ist ein sehr populäres relationales Datenbanksystem. Es ist gekennzeichnet durch gute Leistungsfähigkeit, sowie hohe Zuverlässigkeit und Benutzerfreundlichkeit. Eine große und sehr aktive Community gibt sehr viel Feedback und

Hilfestellungen bei Problemen. Praktische Tools wie bspw. phpMyAdmin ermöglichen die Einsichtnahme und Verwaltung der Daten über ein Webfrontend. Ein besonderes Feature von MySQL ist das Angebot verschiedener Speichersubsysteme. Diese sind auf unterschiedlichste Anwendungsfälle zugeschnitten und ermöglichen so eine gewisse Anpassung der Datenbank. Zusätzlich dazu ist in einigen Subsystemen die Verwendung von Triggern, Views und Stored Procedures möglich. Das System wird mit sehr hoher Geschwindigkeit weiter entwickelt und verbessert. Allerdings sind relativ häufig Fehler in den Updates aufgetreten, weswegen die Qualitätssicherung von MySQL keinen guten Ruf genießt (18). Ein weiterer Kritikpunkt ist das zum Teil sehr starke Abweichen vom ANSI – Standard für SQL. Zusätzlich dazu führt die Verwendung einiger Speichersubsystemen zur Einschränkung der Funktionalität, bspw. unterstützt das System MyISAM als einziges Volltextsuche, allerdings muss dafür auf referentielle Integrität verzichtet werden. Zusammenfassend ist MySQL ein sehr schnelles und effizientes relationales Datenbanksystem. Leider ist die Verwendung von Triggern und prozedures nur mit einigen Einschränkungen möglich. (19)

PostgreSQL ist ein objektrelationales Datenbanksystem, dessen Entwicklung in den 1980er Jahren begann. Das System ist dadurch gekennzeichnet, dass es, im Gegensatz zu MySQL, den gesamten SQL92 – sowie große Teile des SQL99 – Standards unterstützt. Zusätzlich dazu gibt es eine ganze Reihe eigener Erweiterungen, ein Beispiel hierfür ist die Speicherung und Verarbeitung geometrischer Datentypen. Die Datenbank bietet zudem Features wie referentielle Integrität, Transaktionen, stored procedures und etliche mehr. PostgreSQL besitzt darüber hinaus objektrelationale Eigenschaften, die es dem Nutzer ermöglichen, nichtatomare Datentypen zu erstellen und zu speichern. Zusätzlich dazu können eigene Datenbankfunktionen in verschiedenen Sprachen, bspw. Java, geschrieben werden. Eine kleine aber aktive Community beteiligt sich auch bei diesem Projekt an der Weiterentwicklung und Verbesserung des Systems und gibt in diversen Foren Hilfestellungen bei Problemen. Ähnlich wie bei MySQL gibt es auch bei PostgreSQL Werkzeuge zur Administration der Datenbank. In Anlehnung an phpMyAdmin trägt ein sehr umfassendes Programm den Namen pgAdmin III. Das Kriterium Geschwindigkeit war bis vor einigen Jahren ein Nachteil von PostgreSQL gegenüber MySQL. Seit den letzten Versionen übertrifft die Geschwindigkeit, gerade bei großer Last, allerdings die von MySQL (20). Insgesamt ist die

PostgreSQL eine sehr umfangreiche Datenbank. Sie verbindet die Vorteile von relationalen Datenbanken mit objektorientierten Ansätzen. Dabei sind viele Funktionen vorhanden, sodass sich unerfahrene Nutzer schnell der Möglichkeiten des DBS bedienen können. Für spezielle Anforderungen sind allerdings häufig Anpassungen nötig. (18) (21)

Anforderung	**db4o**	**MySQL**	**PostgreSQL**
Open – Source	erfüllt	erfüllt	erfüllt
Speicherung der Daten	erfüllt	erfüllt	erfüllt
Geschwindigkeit	erfüllt	erfüllt	erfüllt
Erreichbarkeit im Netzwerk	erfüllt	erfüllt	erfüllt
Trigger und stored procedures	Eigenentwicklung	teilweise	erfüllt
Updates und Stabilität	erfüllt	teilweise	erfüllt
Backup und Restore	Eigenentwicklung	Erweiterung	erfüllt
Clustering und Replication	erfüllt	Erweiterung	Erweiterung

Tabelle 9: Übersicht Datenbankanforderungen

Tabelle 9 fasst die Anforderungen an das DBS sowie die Eigenschaften der vorgestellten Datenbanken zusammen.

Bei den ersten vier Kriterien genügen alle Systeme den gestellten Anforderungen. Bei db4o besteht die Notwendigkeit, einige der benötigten Funktionen selbst zu entwickeln. Dazu gehören die Trigger und Stored Procedures sowie die Backup und Restore – Funktionalität. Von Vorteil ist allerdings, dass die Datenbank sehr gut an das bestehende System angepasst werden kann und alle anderen Anforderungen erfüllt. MySQL dagegen hat vorallem im Bereich von Update und Stabilität Nachteile gegenüber den anderen beiden Vertretern. Auch die Trigger und Stored Procedures werden nicht in allen Speichersubsystemen unterstützt. Die Backup und Restore – Funktion kann, ebenso wie Clustering und Replikation, über Erweiterungen hinzugefügt werden. PostgreSQL bietet von vornherein sehr viele Funktionen, einzig das Clustering und die Replication muss in Form einer Erweiterung nachinstalliert werden.

Insgesamt ist die Entscheidung immer beeinflusst von persönlichen Vorlieben und Erfahrungen, aber auch von den gesammelten Fakten. Auf Basis dieser Faktoren wird

deutlich, dass sich MySQL und PostgreSQL relativ stark ähneln, wobei MySQL klare Nachteile aufweist. Aus diesem Grund kommen für eine Verwendung nur db4o und PostgreSQL in Frage. Die Nutzung von letztgenannter DB ist vorallem dann interessant, wenn nicht viel Zeit in die Entwicklung und Pflege der Datenbank investiert werden kann. Im Falle der Entwicklung von DABAMOS sind allerdings einige Anpassung notwendig, sodass sich der Vorteil des guten Gesamtpakets egalisiert.

Begründet am Interesse an neuen Technologien und auf Basis der Eigenschaften des Systems hat sich das DABAMOS – Team für die Verwendung von db4o entschieden. Diese Datenbank bietet sehr hohe Geschwindigkeit und Stabilität. Dabei kann das DBS den eigenen Bedürfnissen am Besten angepasst werden und so optimal im Zusammenspiel mit DABAMOS genutzt werden. Zusätzlich dazu ist der Aufwand für die Anpassung der Datenbank gering, da der Programmentwurf nicht angepasst werden muss.

5.2. Datenmodell

Aufbauend auf den Daten der Analyse wird in diesem Unterkapitel das Datenmodell beschrieben. Zum Zweck der besseren Darstellung werden UML – Klassendiagramme verwendet. Diese stellen die Klassen mit ihren Attributen und Methoden dar. Aus Gründen der Übersicht wird auf die Methoden verzichtet, da diese nur aus gettern und settern bestehen.

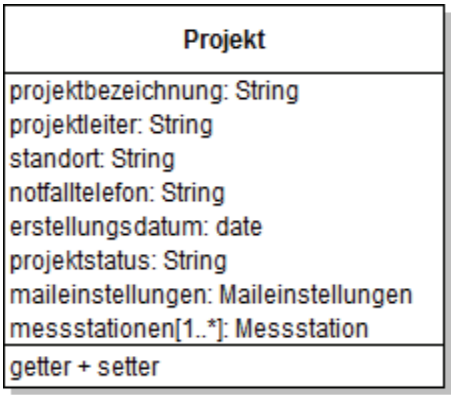

Abbildung 12: UML - Klassendiagramm Projekt

Die Klasse Projekt (Abb. 12) ist die Hauptkomponente des Datenmodells. Über dieses Objekt lassen sich alle weiteren Komponenten referenzieren. Mit Hilfe der entsprechenden Attribute

nimmt die Klasse die Projektbezeichnung, den Projektleiter, den Standort, Telefon und Maileinstellungen sowie das Erstellungsdatum auf. Um die Referenzierung zu gewährleisten, wird zusätzlich eine Liste mit beliebig vielen aber mindestens einer Messstationen verwaltet. Über diese ist es möglich, zuzuordnen welche Messstation zu welchem Projekt gehört. Der Zugriff auf die Attribute erfolgt, wie bei allen nachfolgend beschriebenen Klassen, über getter und setter.

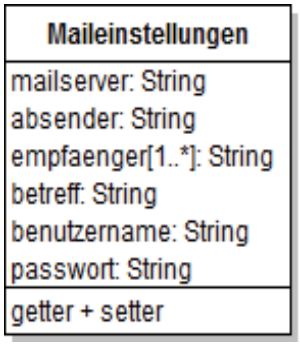

Abbildung 13: UML Klassendiagramm Maileinstellungen

Die Klasse Maileinstellungen (Abb. 13) nimmt Informationen bezüglich des Mailservers auf. Dazu gehören die Daten des Servers, Nutzername und Passwort. Zusätzlich dazu werden eine Empfängerliste, die Adresse des Absenders sowie der Betreff hinterlegt. Die Beziehung ist 1:1, da zu einem Projekt immer genau ein Objekt mit Maileinstellungen gehört.

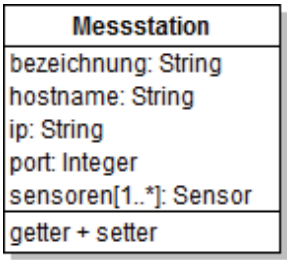

Abbildung 14: UML - Klassendiagramm Messstation

Das Objekt Messstation (Abb. 14) speichert Informationen bezüglich der Verbindung zum entsprechenden Rechner. Dazu gehören der Hostname, sowie IP – Adresse und Port. Zusätzlich dazu kann im Objekt eine Bezeichnung hinterlegt werden. Ähnlich der Liste im Projekt verwaltet die Klasse Messstation eine beliebige Menge an Sensoren. Diese ermöglicht eine Zuordnung, welche Geräte an die Station angeschlossen sind, wobei auch

hier mindestens ein Gerät verbunden sein muss.

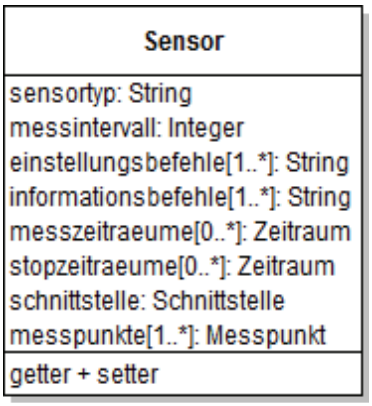

Abbildung 15: UML - Klassendiagramm Sensor

Informationen bezüglich der Sensoren werden im Objekt Sensor (Abb. 15) gespeichert. Das Attribut Sensortyp nimmt hierbei auf, mit welcher Art Gerät das Programm kommunizieren soll. Die Ansteuerung wird mit Hilfe des Objektes Schnittstelle realisiert. In diesem sind Informationen bezüglich des Ports, der Baudrate sowie weiterer Parameter hinterlegt. Im Attribut Messintervall wird festgelegt, in welchem Zeitraum Messungen ausgelöst werden. Zusätzlich dazu ist es möglich, Mess – und Stopzeiträume aufzunehmen, in denen gemessen oder eben nicht gemessen werden soll. Bei Einstellungs – und Informationsbefehle handelt es sich um Listen mit Befehlen, welche für Statusabfragen und zum Setzen von Einstellungen an das Gerät gesendet werden. Ein Beispiel hierfür ist das Übertragungsprotokoll zum Abfragen der Gerätenummer. Äquivalent zum Projekt wird auch hier wieder eine Liste mit weiteren Komponenten verwaltet, da zu jedem Sensor mindestens ein, aber auch beliebig viele Messpunkte gehören können.

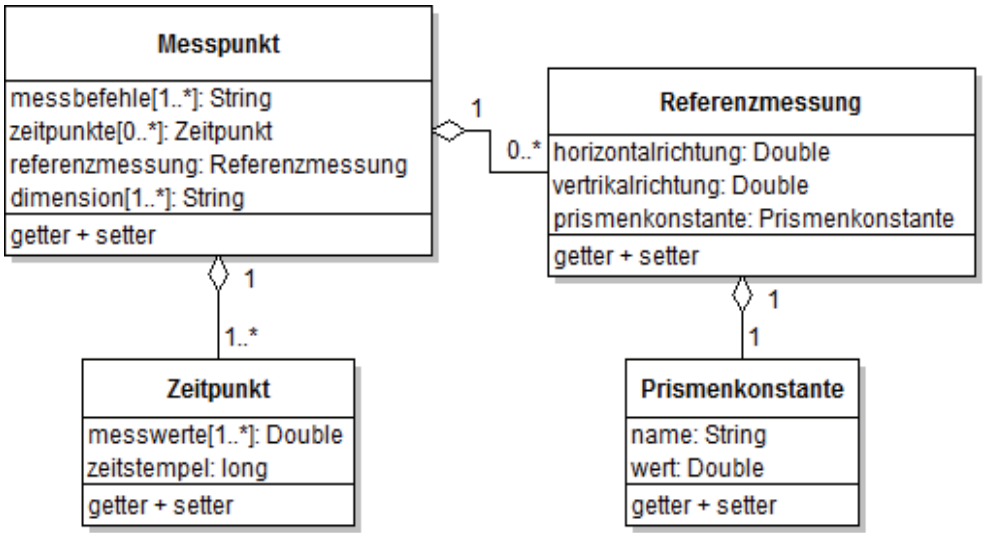

Abbildung 16: UML - Diagramm Messpunkt, Zeitpunkt und Referenzmessung

Eine genaue Zusammensetzung des Objektes Messpunkt zeigt Abbildung 16. Die Klasse nimmt Informationen bezüglich der Messbefehle und der Dimension auf. Zusätzlich dazu wird der Zeitpunkt, also die Messwerte mit einem Zeitstempel, sowie eine optionale Referenzmessung verwaltet. Letztgenanntes Attribut ist für die Tachymetermessung bestimmt, da das Ausrichten des Gerätes auf die Punkte Näherungswerte voraussetzt. Als weitere tachymeterspezifische Eigenschaft wird die Prismenkonstante gespeichert, da diese abhängig vom verwendeten Prisma auf dem Messpunkt ist.

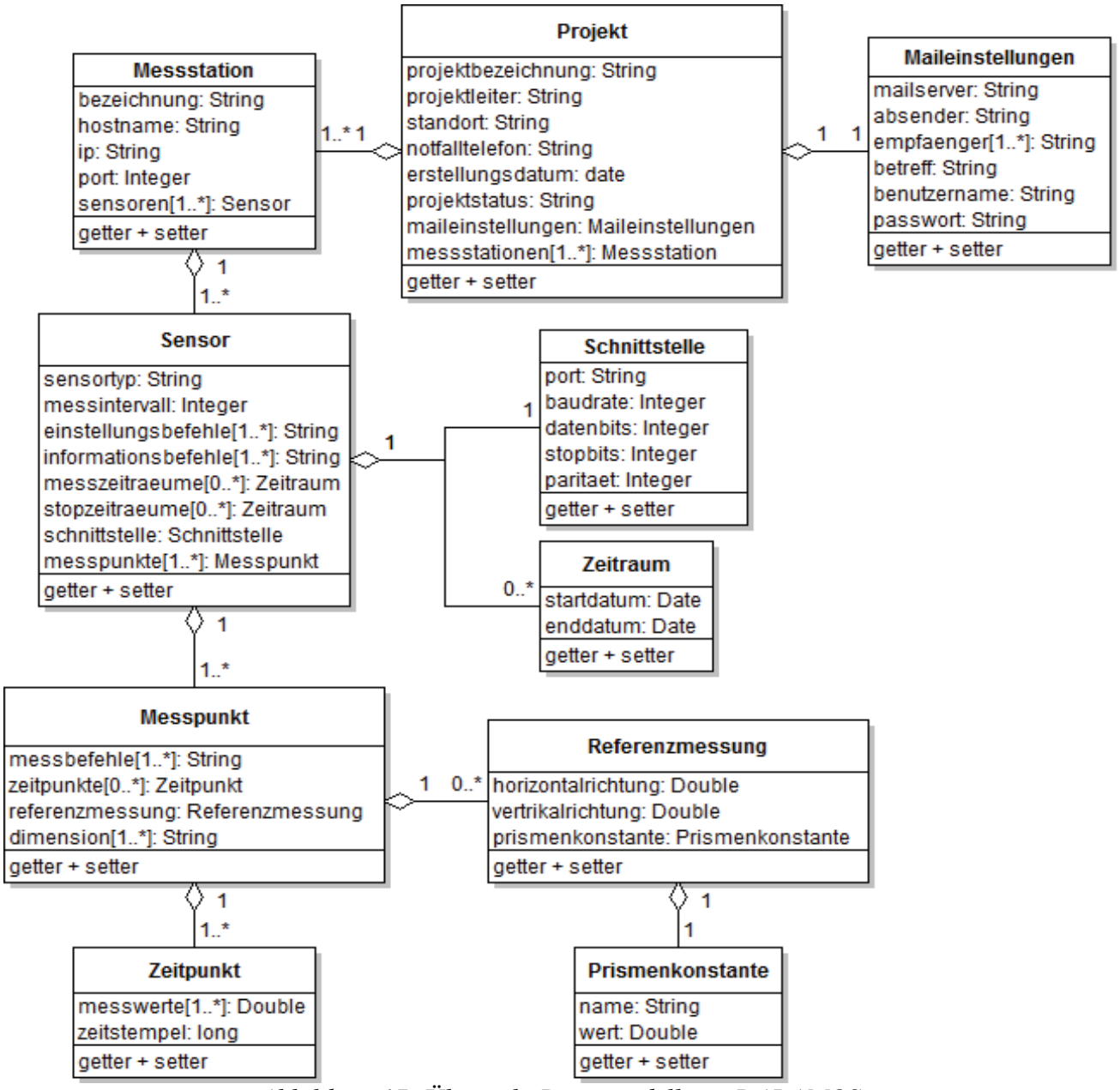

Abbildung 17: Übersicht Datenmodell von DABAMOS

Zur besseren Übersicht zeigt Abbildung 17 den Zusammenhang aller Komponenten inklusive der Beziehungen und Kardinalitäten. Aufbauend auf diesem Modell werden die Daten auf der Datenbank abgebildet. Aufgrund der Verwendung einer objektorientierten DB müssen die Daten nicht erst umgewandelt werden, sondern können direkt persistent gespeichert werden.

6. Realisierung

In diesem Kapitel werden Details zur Realisierung der Speicherlösung beschrieben. Es wird darauf eingegangen, wie die Datenbank installiert wird und wie die Kommunikation abläuft. Nachfolgend beschäftigt sich Kapitel 6.2. mit der Implementierung der Schnittstelle, den sogennanten Data Access Objects (DAO's).

6.1. Vorbereitung

Der Unterschied von db4o im Gegensatz zu einer relationalen oder objektrelationalen Datenbank macht sich auch im Installationsvorgang bermerkbar. Dieser beschränkt sich lediglich auf das Einbinden einer Bibliothek in das entsprechende Programm. Die dafür benötigte Jar – Datei kann von der Webseite heruntergeladen werden. Nach dem Einbinden kann der Zugriff auf die Methoden, äquivalent zu anderen Bibliotheken, erfolgen.

"Eine bewährte Grundregel im Umgang mit db4o lautet, nach Möglichkeit so zu programmieren*, als sei überhaupt keine Persistenzschicht vorhanden* – als sei db4o eine einfache Collection, die rein zufällig auch persistiert werden kann." ((22), Seite 81, Z. 8) Dieser Satz spiegelt wider, dass die Nutzung von db4o keine Auswirkungen auf das Programmdesign hat. Der Programmierer muss die Software demzufolge nicht an die Datenbank anpassen und kann dem entwickelten Entwurf folgen.

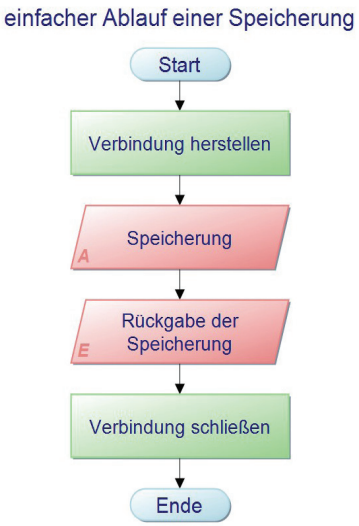

Abbildung 18: Programmablaufplan einer Speicherung

Das Speichern und Abfragen von Daten (Abb. 18) erfolgt ähnlich dem anderer Datenbanken. Der erste Schritt im Programm ist das Öffnen der Verbindung. Danach erfolgt das Senden der gewünschten Befehle und schließlich das Schließen der Verbindung. Beim Austausch der Befehle handelt es sich bei db4o um das programmseitige Aufrufen der entsprechenden Methoden der eingebundenen Bibliothek. So ist es möglich, die Objekte mit Hilfe von wenigen Code – Zeilen zu speichern. Das Abrufen der gespeicherten Daten kann mit Hilfe von drei verschiedenen Abfragekonzepten erfolgen. Das sogenannte QueryByExample (QbE) ist dabei die einfachste und ausdrucksschwächste Variante. Die Abfrage bildet ein Beispielobjekt, dem die Rückgabewerte ähneln. Das S.O.D.A. - Konzpet erlaubt das Integrieren von QbE, ist allerdings deutlich umfangreicher und mächtiger. Es werden komplexere Anfragen, bspw. mit Hilfe von logischen Operatoren, ermöglicht. Eine S.O.D.A. - Abfrage besteht aus einer abstrakten Spezifikation, welche ebenfalls als Musterobjekt genutzt wird und so die entsprechenden Ergebnisse liefert. Als Native Queries (NQ) wird das dritte Konzept bezeichnet. Dieses ist die Hauptabfragesprache für db4o, da sie das Konzept von S.O.D.A. um Typsicherheit erweitern und auch einen Performancegewinn bringen. Eine NQ – Abfrage muss immer die Klasse Predicate der Bibliothek implementieren und gibt den zu erwartenden Typ, welcher generisch festgelegt wird, zurück. (22)

Die Datenbank kann sowohl im Solo – Modus als auch im Client – Server – Modus betrieben werden. Ersterer bedeutet, dass der Zugriff auf die DB in ein und derselben Virtual Machine (VM) erfolgt. Der Client – Server – Modus dagegen ermöglicht das Speichern und Abfragen von Daten über ein beliebiges Netzwerk. Die Kommunikation verläuft dabei über TCP/IP – Sockets. Beide Modi unterscheiden sich bei der Nutzung nur in Details. Im Solo – Modus werden beim Erzeugen eines Server lediglich Platzhalter verwendet, während diese im zweiten Modus durch echte Adressen, Nutzernamen und Passwörter ersetzt werden. Ein Client kann sich mit diesen Parametern einfach mit der Datenbank verbinden und und nach erfolgter Authentifizierung beliebige Operationen durchführen. (22)

Die Konfiguration der Datenbank erfolgt ebenfalls über Methoden aus der Bibliothek. Die Einstellungen können für einen Container, ein Objekt oder aber für den Client oder Server erfolgen und müssen häufig vor der Erstellung der Bereiche, in denen sie gültig sind, getätigt werden. Neben Möglichkeiten zur Konfigurationen stehen zusätzliche Operationen, bspw

Replikation, Defragmentierung sowie Logger – und Statistik – Tools zur Verfügung. Die drei letztgenannten sind allerdings nicht Bestandteil der Bibliothek sondern werden als eigene Unterprogramme mitgeliefert und müssen separat importiert und ausgeführt werden. (22)

6.2. Implementierung

In diesem Unterkapitel geht es um die Implementierung der Datenbank und ihrer Schnittstellen in DABAMOS.

Vor der Durchführung beliebiger Operationen muss die Verfügbarkeit der DB geprüft werden. Dies geschieht beim Start des Programmes durch eine entsprechende Methode. Diese prüft, ob die Datenbank erreichbar und vorhanden ist. Ist das der Fall, wird der Programmstart wie geplant fortgesetzt. Ist die Datenbank nicht verfügbar, wird sie mit entsprechenden Vorgaben angelegt. Die Konfiguration kann dabei über XML – Dateien oder über eine GUI erfolgen. Es ist zusätzlich möglich, den Zustand der Verbindung periodisch abzufragen, um eine Statusanzeige im Webinterface zu realisieren.

Die Speicherung der Objekte aus dem Datenmodell erfolgt mit Hilfe von sogenannten Data Access Objects (DAO's). Ein DAO ist ein Interface, welches Methoden zur Verfügung stellt, mit denen Operationen bezüglich eines Objektes auf der Datenbank durchgeführt werden können. Typische Methoden spiegeln die sogenannten CRUD – Operationen wider. Die einzelnen Buchstaben stehen dabei für Create, Read, Update und Delete. Häufig werden zusätzliche Methoden implementiert, welche bspw. alle Objekte eines Typs in einer Liste ausgeben.

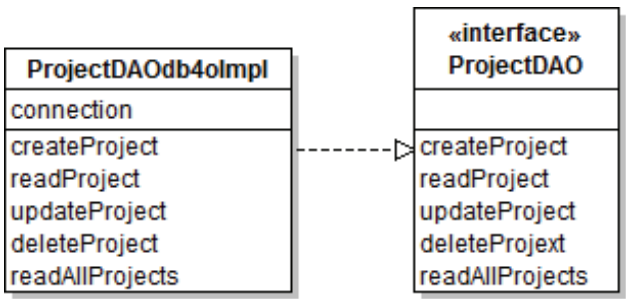

Abbildung 19: UML Klassendiagramm ProjekDAO

Exemplarisch wird nachfolgend das DAO für das Objekt Projekt beschrieben. Das Interface ProjectDAO gibt die Methoden vor (Abb. 19). Die Implementierung findet sich in der Klasse ProjectDAOdb4oImpl wieder. Auf Grundlage dieser Architektur ist es möglich, die Persistenzschicht einfach auszutauschen, ohne große Änderungen am Programm durchführen zu müssen. Es ist bspw. denkbar eine Klasse ProjectDAOHibernateImpl zu implementieren, um die Daten mit Hibernate zu speichern.

Wie beschrieben, werden die Methoden create, read, update und delete in der Klasse ProjectDAOdb4oImpl implementiert. Zusätzlich dazu können alle Objekte des Typs Projekt aus der Datenbank gelesen werden. Das Attribut connection wird dem Konstruktor der Klasse übergeben und stellt die Verbindung zur Datenbank dar. Diese globale Variable kann damit in allen Methoden des Objektes genutzt werden.

Die beschriebene DAO – Klasse wird zuätzlich für die Objekte Messstation, Sensor, sowie Mess- und Zeitpunkt implementiert. Die Klassen ähneln sich sehr stark, sodass auf eine einzelne Beschreibung verzichtet wird. Alle verfügen über die dargestellten Methoden, welche allerdings angepasste Methodennamen und Rückgabewerte haben. Mit Hilfe dieser Klassen ist es möglich, die Daten effizient persistent zu speichern und abzurufen.

7. Fazit

Zusammenfassend ist die Einsatzfähigkeit von Datenbanksystemen für Überwachungsmessungen ein sehr umfangreiches Themengebiet. Verschiedene Monitoringsysteme haben, aufgrund der großen Anzahl verfügbarer Sensoren und Einsatzmöglichkeiten, unterschiedlichste Anforderungen.

Im Rahmen dieser Bachelorarbeit wurden drei Konzepte zur Speicherung von Daten in Datenbanken beleuchtet. Aufbauend auf diesen wurden drei unterschiedliche Systeme vorgestellt. Anhand der speziellen Bedürfnisse von DABAMOS ist die Datenbank db4o als Persistenzschicht ausgewählt worden. Entscheidende Eigenschaften sind hierbei hohe Performance, Einfachheit der Datenbank sowie Flexibilität. Die objektorientierte Datenbank ermöglicht eine einfache und schnelle Speicherung der anfallenden Daten und eine gute Integration in DABAMOS.

Auf Basis des ausgewählten Systems ist eine Erweiterung des Webinterfaces denkbar. Es ist geplant, eine Datenbankadministration zu integrieren, mit der die Objekte visualisiert und verwaltet werden können. Zusätzlich dazu sind Features, wie die Ausgabe von Statistiken und Logs oder auch das Defragmentieren der Datenbank, in Planung.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

1: DVW, Zeitabhängige Messgrößen - Verborgene Schätze in unseren Daten, 2009

2: Michael Möser, Gerhard Müller, Harald Schlemmer, Hans Werner, Handbuch Ingenieurgeodäsie, 2000

3: Michael Neid, Michael Platte, Deformationsüberwachung an der Kanalbrücke des Wasserstraßenkreuzes Magdeburg, 2007

4: National Instruments, Vorteile PC-gestützter Überwachungsanwendungen, Juli 2010, http://zone.ni.com/devzone/cda/tut/p/id/11274

5: Christian Wolff, Entwicklung eines Webinterfaces zur Fernwartung von Überwachungsmesssystemen, 2010

6: Roland Gabriel, Datenbankmanagementsystem, Juli 2010, http://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/daten-wissen/Datenmanagement/Datenbanksystem/Datenbankmanagementsystem

7: Andreas Meier; Thomas Wüst, Objektorientierte und objektrelationale Datebanken, 2003

8: Andreas Heuer; Gunter Saake, Datenbanken: Konzepte und Sprachen, 2000

9: Torsten Horn, Objektrelationales Mapping (O/R-M) mit Hibernate 3.2.5, Juli 2010, http://www.torsten-horn.de/techdocs/java-hibernate.htm

10: Andreas Wehrenpfennig, Skript Datenbanken, 2009

11: unbekannt, Definition bzw. Erklärung: Objektrelationale Datenbank, Juli 2010, http://www.bullhost.de/o/objektrelationale-datenbank.html

12: Leica, Handbuch TCRP1203, 2010

13: Thommen, Handbuch HM30, 2010

14: Hans-Jürgen Larisch, Skript Sensorik, 2010

15: Harald Schlemmer, Grundlagen der Sensorik, 1996

16: Günter Petrahn, Taschenbuch Vermessung, 2007

17: Versant, db4o HP, August 2010, http://www.db4o.com/

18: Torsten Zühlsdorff, MySQL oder PostgreSQL?, August 2010, http://www.php-faq.de/q-db-vergleich.html

19: Oracle, Warum MySQL?, August 2010, http://www.mysql.de/why-mysql/

20: unbekannt, MySQL vs PostgreSQL Benchmarks, August 2010, http://www.randombugs.com/linux/mysql-postgresql-benchmarks.html

21: PostgreSQL, Überblick, August 2010, http://postgresql.de/postgresql_outline.html

22: Larysa Visengeriyeva, Patrick Römer, db4oschnell + kompakt