Hochschule Neubrandenburg
University of Applied Sciences

D I P L O M A R B E I T
Studiengang Bauinformatik

Bauingenieuranwendungen auf einem PDA

vorgelegt von

cand. - Ing. Petra Preik

Erstprüfer:	Prof. Dr. - Ing. Andreas Wehrenpfennig
Zweitprüfer:	Prof. Dr. - Ing. Guido Bolle
Abgabedatum:	11.08.2009
URN:	urn:nbn:de:gbv:519-thesis2009-0282-9

Erklärung

Hiermit erkläre ich, dass die vorliegende Diplomarbeit selbstständig von mir angefertigt wurde. Es wurden nur die in der Arbeit ausdrücklich benannten Quellen und Hilfsmittel benutzt. Wörtlich oder sinngemäß übernommenes Gedankengut habe ich als solches kenntlich gemacht.

Bredenfelde, den 10.08.2009

Petra Preik

Kurzfassung

Diese Diplomarbeit beschäftigt sich mit dem Entwurf und der Umsetzung eines PDA[1]-Programms. Die Software PDA-SB[2] soll Berechnungen aus dem konstruktiven Ingenieurbau lösen. Das Ziel wird erreicht mit einem Rahmenprogramm, welches flexibel Aufgaben verwaltet. Die Ergebnisse werden in einer Datenbank gespeichert. Der einheitliche Aufbau der Daten ermöglicht den Austausch von Werten zwischen den Berechnungen. Damit sind verschiedene Aufgaben realisierbar, die aufeinander bauen.

Der theoretische Teil erklärt den konstruktiven Ingenieurbau, wobei der Fokus auf den Stahlbetonbau gerichtet ist. Aus diesem Bereich wird der Aufbau einer Berechnung beispielhaft geschildert. Im Anschluss wird der PDA mit seiner Leistungsfähigkeit und Funktionsweise erläutert. Darauf aufbauend werden bestehende Programme beschrieben. Die Vorzüge werden ermittelt und für das Programm PDA-SB genutzt. Im Ergebnis der Theorie entsteht ein Entwurf, der ein flexibles strukturiertes System beschreibt um Bauingenieuraufgaben zu berechnen.

Der praktische Teil beschreibt die Umsetzung des Programms. Um den aktuellen Stand festzuhalten, wird eine Leistungsbewertung vorgenommen, die auf Tests beruht. Daraus ergeben sich Möglichkeiten und Verbesserungsvorschläge, die den Prototypen weiterentwickeln. Die Diplomarbeit wird mit einer Zusammenfassung abgerundet.

1 Personal Digital Assistent, kleiner tragbarer Computer
2 PDA-SB: PDA-Programm für Stahlbetonbau Berechnungen

Abstract

This diploma thesis deals with a concept and an implementation of a PDA[3] program. The software PDA-SB[4] shall solve calculations of construction engineering. The aim is reached with a main program, which administrates flexible tasks. Results are saved in a database. The standard structure of data makes the exchange of values between the calculations possible. With this are different tasks practicable, which build up on top of each other.

The theoretical part explains the construction engineering, where the main part is justified on reinforced concrete construction. This field describes for example the design of calculations. Following the PDA is commented with performance and functionality. After that existing programs are characterized. The amenities are detected and used for the program PDA-SB. As a result of the theoretical part a concept develops, which describes a flexible structured system. This calculates duties of civil engineer.

The practical part explains an implementation of prototype. In order to keep "status quo" the appraisal of results are made, which are based on tests. Out of it possibilities and ideas of improvement are shown, which continues the development of the program. The summary completes this diploma thesis.

3 Personal Digital Assistant, little portable computer
4 PDA-SB: PDA program for calculations of reinforced concrete construction

Inhaltsverzeichnis

1 Einleitung

Im Rahmen dieser Diplomarbeit ist ein Softwareprogramm für einen PDA zu entwickeln, welches Aufgaben aus dem Bauingenieurbereich lösen soll. Typische Anwendungsgebiete eines Bauingenieurs sind der konstruktive Ingenieurbau, Bauwerkserhaltung, Planung, Wasser und Verkehr. Damit sich der Umfang der Diplomarbeit in Grenzen hält, basiert der Entwurf auf dem konstruktiven Ingenieurbau. Dabei liegt der Schwerpunkt auf dem Stahlbetonbau. Infolge der großen Anzahl von verschiedenen Aufgaben und den begrenzten Speicherplatz eines PDA soll das Programm flexibel auf die Gegebenheiten reagieren. Das bedeutet, das System muss Module in Form von Berechnungen ein- und ausladen können. Die Gemeinsamkeiten der Aufgaben bilden die Grundlage für alle Berechnungen, die das Programm umsetzen kann. Die Software erhält ihren Namen **PDA-SB** aus dem ersten umgesetzten Aufgabengebiet (**S**tahlbeton **B**erechnungen).

PDA-SB soll den PDA als Adressbuch und Terminplaner um den Bauingenieurbereich erweitern. Das System soll portabel und vielleicht auf dem Handy bedienbar sein. Eine weitere Flexibilität stellt der Austausch von Berechnungsmodulen dar. Durch die Erweiterbarkeit von Aufgaben ist das Programm ein ideales Werkzeug für die Ausbildung von Bauingenieuren oder Bauinformatikern. Eine einfache Bedienung, eine angemessene Anleitung zur Benutzung der Formeln und Hintergrundinformationen zu der Thematik des Aufgabengebietes bieten eine komfortable Lernumgebung. Außerdem können angehende Ingenieure eigene Berechnungsaufgaben anfertigen, so dass eine Aktualität von Normen und DIN-Vorschriften gegeben ist. Solch eine Ausarbeitung kann das Verständnis der Aufgabe verbessern. Informatiker im Baubereich vertiefen mit der Umsetzung einer Berechnung ihre Kenntnisse in der Programmierung. Die Verwendung als erweiterter Taschenrechner ist für den Bauingenieur auf der Baustelle praktisch, denn durch den kleinen kompakten PDA, der vergleichbaren hohen Leitungsfähigkeit und den selbst erklärenden Aufgaben aus dem Bauingenieurbereich ist das Programm ideal um Bücher nicht mitnehmen zu müssen. Durch diese weitreichenden Möglichkeiten der Nutzung ist die Software ein wertvolles Hilfsmittel für nahezu alle Bereiche eines Bauingenieurs.

Für die Umsetzung der Ideen ist es notwendig ein flexibles Programm zu entwickeln. Eine Anlehnung an vorhandene Programme aus der PC-Welt und dem PDA-Bereich ist unabdingbar. Die einfache Bedienbarkeit für angehende Ingenieure soll genauso gegeben sein wie die Erweiterungsfähigkeit von Berechnungen. Damit die Eingabedaten und Ergebnisse auch für die Bearbeitung in anderen Programmen vorhanden sind, müssen diese gespeichert und exportiert werden. Werte von außen ohne manuelle Eingabe sollten über ein Import eingelesen werden. Um gegebenenfalls Berechnungsketten aufzustellen bedarf es einer normierten Datenstruktur und einem eindeutigen Übernahmekonzept.

Im Anschluss folgt das Kapitel Analyse. Zu Beginn wird der konstruktive Ingenieurbau mit dem Fach Stahlbetonbau beschrieben. Eine typische Aufgabe, die das Programm lösen soll, ist die Rechteckbemessung aus dem Massivbau. Anhand dieser Berechnung wird die Vorgehensweise und der Berechnungsumfang demonstriert. Die technischen Grundlagen und die Anwendungsbereiche eines PDA werden geschildert. Daran schließt sich die Untersuchung von Programmen an. Vorteilhafte Strukturen werden übernommen. Die Schlussfolgerungen bilden die Grundlage für den Entwurf des Programms. Der Aufbau von PDA-SB ergibt sich durch die Auslagerung der Berechnungen in Modulen. Das restliche Programm bildet den Rahmen. Es übernimmt die Steuerung und den Datenaustausch. Für die Verständigung zwischen den beiden Programmteilen dienen die Schnittstellen. Weiterhin wird im Entwurf die Datenspeicherung, der einheitliche Datencontainer und Berechnungsketten beschrieben. Das Design der Benutzeroberfläche rundet das Kapitel ab. Im Folgenden wird die Umsetzung des Programms erläutert. Als Erstes ist die Umgebung für die Programmierung erläutert. Dann folgt das Hauptprogramm mit der Beschreibung der Funktionen, die zur Verfügung stehen. Die Schnittstellen als Kommunikationsbasis zum Berechnungsmodul, die Datenstruktur und der Datenbankaufbau werden beschrieben. Als Letztes in diesem Kapitel folgt die Beschreibung des Moduls, welches die Berechnung der Bauingenieuraufgabe umfasst. Anschließend wird über Tests eine Leistungsbewertung vorgenommen, die den aktuelle Stand der Umsetzung wiedergeben. Zum Ende der Diplomarbeit schließt sich die Zusammenfassung an.

2 Analyse

In diesem Kapitel wird der konstruktive Ingenieurbau, der PDA und vorhandene Bau-Software untersucht. Die Aufgaben aus dem Bauwesen sind die Basis für das Programm. Damit die Analyse nicht zu umfangreich wird, wird der Schwerpunk auf das Fachgebiet Stahlbetonbau gelegt. Eine Rechteckbemessung aus dem Massivbau veranschaulicht eine typische Berechnung, die im Programm PDA-SB umgesetzt wird. Daran anschließend werden die Möglichkeiten und Grenzen des PDA aufgezeigt. Die Technischen Daten bis hin zum Anwendungszweck werden beschrieben. PDA und PC Software, die ähnliche Berechnungen ausführen können, werden erklärt und verglichen. Die Funktionsweise und der Aufbau dieser Programme, wenn sie vorteilhaft sind, werden für die eigene Umsetzung verwendet. Zum Ende der Analysephase ist die Zusammenfassung beschrieben, die die gewonnenen Kenntnisse als Grundlage für den Entwurf formuliert.

2.1 Der konstruktive Ingenieurbau

Ein Bauingenieur aus dem konstruktiven Ingenieurbau beschäftigt sich mit der Konstruktion und der Bemessung von Tragwerken, wie z.B. Industrieanlagen, Hallen, Häuser, Tunnel und Brücken. Typische Werkstoffe sind Beton, Stahl und Holz. Die Werkstoffgesetze in Verbindung mit der Statik und der Festigkeitslehre ermöglichen eine Bemessung. Damit kann ein ebenes oder räumliches System Belastungen aufnehmen oder weiterleiten ohne Schaden zu nehmen.

Damit Aufgaben, die das Programm PDA-SB lösen soll, nachvollziehbar beschrieben werden, ist zuerst Hintergrundwissen notwendig. Dieses ergibt sich aus der Baustatik, der Festigkeitslehre und den Werkstoffgesetzen. Im Anschluss folgt die Statik im Bauwesen. [KI Wiki 09]

2.1.1 Statik

In der Statik werden Bauobjekte als Modelle angesehen, um eine rechnerische Grundlage zu schaffen. So wird eine Decke eines Hauses als Platte betrachtet.

Die einfache Variante ist der Balken. Das statische System wird auf die Schwerachse reduziert, so dass die Räumlichkeit verloren geht. Der Balken wird als Stab bzw. Träger abgebildet. Entscheidend für die Schnittkraftermittlung ist die Belastung und die Lagerung des Trägers. Die Beanspruchung ergibt sich aus der ständigen Last p (wie Eigengewicht), der Verkehrslast q (wie Wind, Schnee und Verkehr) und besondere Belastungen (wie Anprall von Fahrzeugen). Realistische Modelle werden durch verschiedene Lastfälle dargestellt. Aufgrund der Auflager des statischen Systems mit den verschiedenen Beanspruchungen ändern sich die Formeln für die Schnitt- und Verschiebungsgrößen. Es gibt Träger auf zwei Stützen mit und ohne Kragarm sowie den Träger mit einseitiger oder beidseitiger Einspannung. In Abbildung 1 ist ein Träger auf zwei Stützen ohne Kragarm zu sehen. Eingezeichnet sind die Auflagerkräfte A und B, die Momentenlinie mit dem maximalen Moment M_{max}, die Biegelinie mit der Durchbiegung in der Mitte w_{mitte} und die Beanspruchung q.

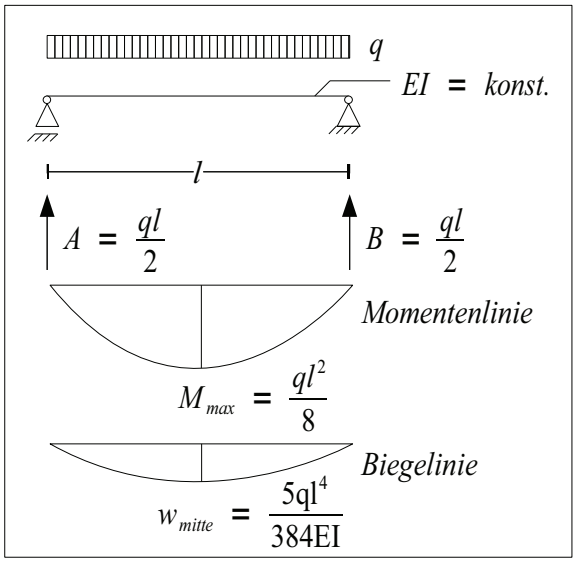


Abbildung 1: Träger auf zwei Stützen

Weitere statische Systeme sind Durchlaufträger und Rahmen. Durchlaufträger unterscheiden sich gegenüber eines Trägers auf zwei Stützen um die Anzahl der Auflager. Zum Beispiel besitzt ein Dreifeldträger drei Auflager. Die Rahmen können Eingespannte oder Zweigelenkrahmen sein. Bei allen statischen Systemen geht es um die Ermittlung der Auflagerkräfte, den Querkraftverlauf, Momentenverlauf mit dem maximalen Moment und die Biegelinie. Anhand dieser

Schnitt- und Verschiebungsgrößen kann eine Bemessung vorgenommen werden. Außerdem sind Kenntnisse aus der Festigkeitslehre notwendig. [Schneider 02]

2.1.2 Festigkeitslehre

Der Träger des statischen Systems wird als Querschnitt betrachtet. Eine Decke im Haus kann im Modell vereinfacht ein Balken oder Plattenbalken sein. Die Querschnittformen des Balkens sind beispielsweise Rechteckquerschnitte, T-, L- oder U-Profile, Kreise, Hohlprofile oder Vielecke.

In Abbildung 2 ist ein Rechteckquerschnitt mit den Formeln für die Berechnung der Querschnittswerte zu sehen. Da das gewählte Koordinatensystem y-z im Schwerpunkt S liegt und das Flächenzentrifugalmoment I_{yz} den Wert 0 annimmt, sind die y- und die z-Achse die Hauptachsen. Die Fläche A ergibt sich aus der Breite b mal der Höhe d. Die Formeln der Flächenmomente 2. Grades I_y und I_z sowie der Widerstandsmomente W_y und W_z sind im folgenden Formelfenster enthalten.

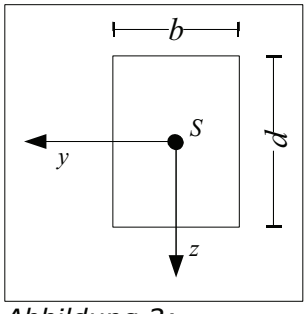


Abbildung 2: Rechteckquerschnitt

Breite b, *Höhe* d

Schwerpunkt S, *Fläche* $A = bd$

Flächenzentrifugalmoment $I_{yz} = 0$

Flächenmomente 2. Grades: $I_y = \frac{bd^3}{12}$; $I_z = \frac{db^3}{12}$

Widerstandsmomente: $W_y = \frac{bd^2}{6}$; $W_z = \frac{db^2}{6}$

Ein weiterer wichtiger Punkt in der Festigkeitslehre ist die Ermittlung der Spannungen infolge der Momente, Normal- und Querkräfte. Diese teilen sich in Normal-, Schub-, Rand- und Hauptspannungen auf.

In Abbildung 3 sind die Kräfte im Querschnitt des Trägers eingetragen. Die Bezugsachsen x, y und z sind Hauptachsen. Entlang der y-Achse wirkt die

Querkraft Q_y. Auf der z-Achse ist die Querkraft Q_z angesetzt. Die Normalkraft N wirkt in Richtung der x-Achse. Die Momente drehen um jede Achse.

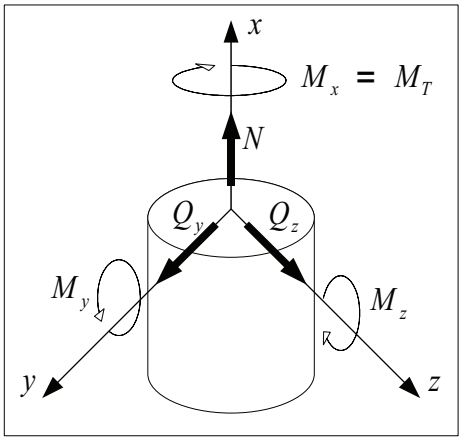


Abbildung 3: Momente, Querkräfte und Normalkraft eines Querschnitts

Für einfach- und doppeltsymmetrische Querschnitte, wobei die Schwerpunktachsen y und z identisch mit den Hauptachsen sind, gelten folgende Formeln:

Normalspannung σ

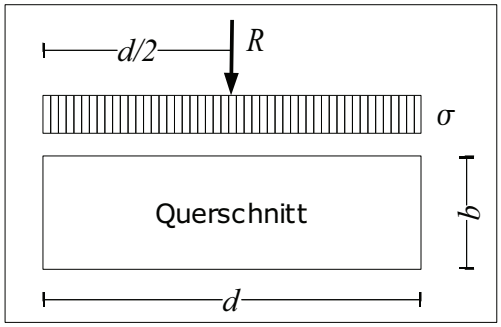
$$\sigma_x = \frac{N}{A} + \frac{M_y}{I_y} z - \frac{M_z}{I_z} y$$

Schubspannung τ

$$\tau_{xz} = \tau_{zx} = \frac{Q_z \cdot S_y}{I_y \cdot b}$$

Die Normalspannung σ und die Schubspannung τ sind für eine Bemessung erforderlich. Die Randspannungen sind ebenfalls wichtig für die weiteren Berechnungen.

In Abbildung 4 ist ein rechteckiger Querschnitt mit mittiger resultierender Kraft R und einer gleichmäßigen rechteckigen Spannungsverteilung σ abgebildet. Die Formel für die Randspannung σ ist im unteren Formelfenster zu sehen.



$$Randspannung\ \sigma \qquad \sigma = \frac{R}{bd}$$

Abbildung 4: Belastungs- und Spannungsschema

Nachdem die Querschnittswerte und die Spannungen erläutert wurden, ist im folgenden das Material des Trägers entscheidend. Die Decke des Hauses kann aus Stahlbeton oder Holz gefertigt werden. Jeder Werkstoff hat bestimmte Eigenschaften und verhält sich unter Belastung anders. Daher sind besondere Kenntnisse auf den entsprechenden Gebieten erforderlich. Im Weiteren werden die Werkstoffgesetze für Beton und Betonstahl beschrieben. [Schneider 02]

2.1.3 Werkstoffgesetze

Um die Bemessung auszuführen, sind die Ausgangswerte für die Berechnung notwendig. Diese ergeben sich durch die Materialeigenschaften, die in den Werkstoffgesetzen beschrieben werden. Da die Berechnung im Fachgebiet des Stahlbetonbaus liegt, wird Beton und Betonstahl näher analysiert.

Beton besteht aus Zement, Gesteinskörnungen, Wasser und ggf. aus Betonzusatzmitteln. Durch die Zusammensetzung erhält der Beton seine Eigenschaften. Es gibt Unterscheidungen zwischen Frisch- und Festbeton. Für die weitere Betrachtung ist der Festbeton von Bedeutung. Die Expositionsklassen beschreiben Umwelteinflüsse. Um die Widerstandskraft des Betons gegen aggressive Stoffe zu erhöhen, werden Zusatzmittel hinzugefügt. Zum Schutz der Bewehrungslagen gegen Korrosion wird die Betondeckung erhöht. Im Beispiel ist die Expositionsklasse XC1 gewählt, die für ein Bauteil innerhalb eines umbauten Raumes angenommen wird. Damit kann die Betondeckung c_{nom} ermittelt werden, wie im Folgenden zu sehen ist.

$$\text{Betondeckung } c_{nom}$$
$$c_{nom} = c_{min} + \Delta c = 10\,mm + 10\text{mm} = 20\,mm$$

Die Druckfestigkeiten sind in Klassen nach Leicht-, Normal- und Schwerbeton sortiert. Die charakteristische Festigkeit f wird durch einen Versuch ermittelt. Dazu wird der Beton in Probekörpern nach 28 Tagen verwendet. Die ermittelte Festigkeit eines zylindrischen Probekörpers wird $f_{ck,cyl}$ und die Probe eines Würfels wird $f_{ck,cube}$ genannt. Es folgt ein Beispiel für den Normalbeton C20/25.

$$\text{Druckfestigkeitsklasse } C20/25 \text{ für Normalbeton}$$
$$f_{ck,cyl} = 20\,N/mm^2 \qquad f_{ck,cube} = 25\,N/mm^2$$

Nachdem der Beton mit seinen Eigenschaften beschrieben wurde, folgen im Anschluss die Eigenschaften des Betonstahls (BSt). Unterschieden werden die Betonstahlsorten durch die Verarbeitungsform (Betonstabstahl, Betonstahlmatte, Bewehrungsdraht), die Festigkeitseigenschaften (Streckgrenze, Zugfestigkeit), die Oberflächengestaltung und dem Herstellverfahren. Im Folgenden sind die Eigenschaftswerte des Betonstabstahls BSt 500 S abgebildet.

$$\text{Betonstabstahl } BSt\,500\,S$$
$$\text{Nenndurchmesser } d_s = von\ 6\ bis\ 28\ mm$$
$$\text{Streckgrenze } R_e \text{ bzw. Dehngrenze } R_{p0,2} = 500\,N/mm^2$$
$$\text{Zugfestigkeit } R_m = 550\,N/mm^2$$
$$\text{Bruchdehnung } A_{10} = 10\,Prozent$$

Der Betonstahl wird in den Duktilitätsklassen normal und hoch eingestuft. Das bedeutet, dieser muss eine angemessene Dehnfähigkeit aufweisen. Für die normale Duktilität ist der charakteristische Wert der Dehnung $\varepsilon_{uk} = 25\ ‰$.

Für die Querschnittsbemessung ist die Spannungs-Dehnungs-Linie wichtig, die in Abbildung 5 zu sehen ist. Lt. DIN 1045-1 sind zwei Annahmen zugelassen, welche in der unteren Abbildung die gezeichneten Linien I und II darstellen.

Dabei ist die Linie I bei der Stahlspannung auf f_{yk} bzw. f_{yk}/γ_s und der Dehnung ε_s auf $\varepsilon_{uk} \leq 25\ ‰$ begrenzt. Die Linie II zeigt den Anstieg der Stahlspannung zur Zugfestigkeit f_{tk} bzw. f_{tk}/γ_s. Die Dehnung darf maximal $\varepsilon_{su} \leq 25\ ‰$ betragen. Der Rechenwert der Zugfestigkeit $f_{tk,cal}$ beträgt $525\ N/mm^2$.

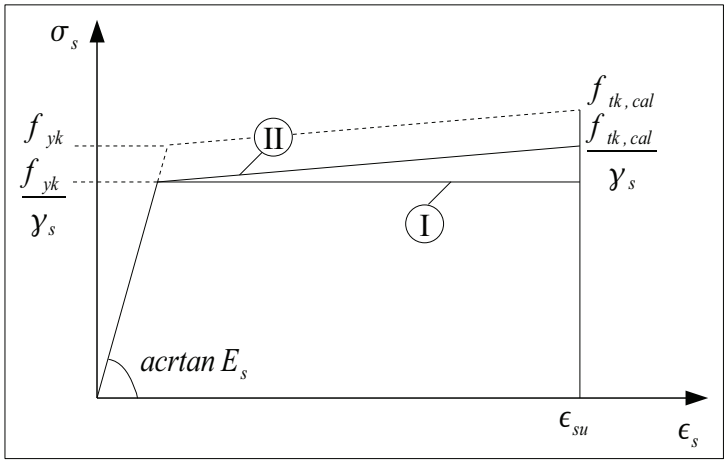


Abbildung 5: Spannungs-Dehnungs-Linie

Die Bemessung ist in der Theorie und mit Formeln als Beispiel einer Rechteckbemessung veranschaulicht. Diese ist im nächsten Kapitel beschrieben. [Schneider 02]

2.1.4 Rechteckbemessung

Um beispielhaft die Anwendungen eines Bauingenieurs zu demonstrieren, wird aus dem Bereich Stahlbetonbau die Bemessung eines Rechteckquerschnitts erläutert. Die Formeln beschreiben eine Näherungsberechnung. Es gelten folgende Anwendungsgrenzen:

- einfache Bewehrung (eine Bewehrungslage)
- Normalbeton C ≤ 50/60
- reine Biegung ohne Normalkraft

Innerhalb dieser Grenzen gilt die Theorie der Bemessung, die in Abbildung 6 veranschaulicht ist.

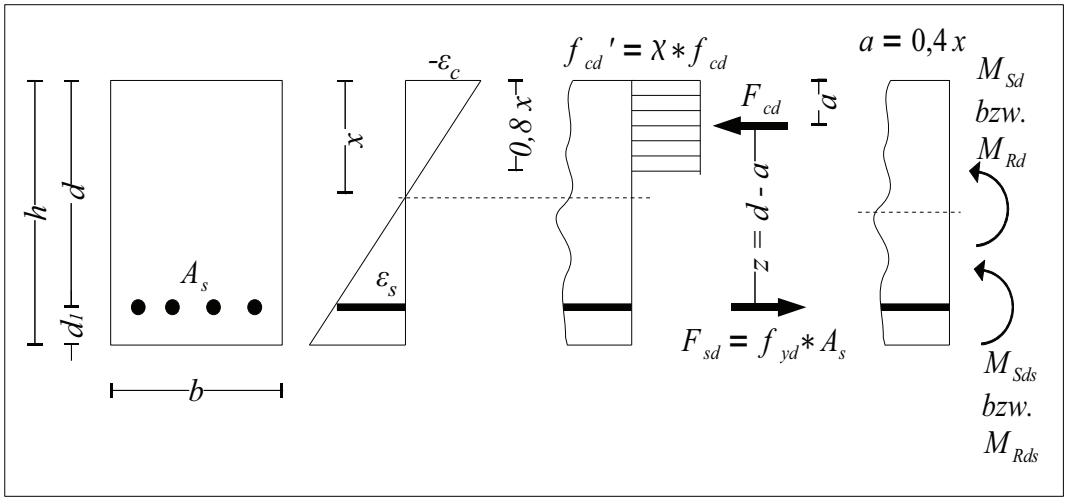


Abbildung 6: Rechteckbemessung

In Abbildung 6 ist links das Rechteck mit einlagiger Bewehrung zu sehen. Dabei ist b die Breite und h die Höhe des Objektes. Die Nutzhöhe d errechnet sich aus der Höhe h minus d_1, wobei d_1 die Betondeckung c_{nom} plus die Hälfte des Stabdurchmessers d_s ist. A_s ist die Fläche des Betonstahls. Rechts neben dem Rechteck ist erst die Dehnungsverteilung mit der Dehnung ε_c für den Beton und ε_s für den Stahl angegeben. Dann ist die Druck-Zug-Verteilung aufgezeichnet. In der Betondruckzone, die im oberen Bereich des Rechtecksquerschnitts liegt, wird näherungsweise eine rechteckige Spannungsverteilung angenommen. Die Betondruckkraft F_{cd} wirkt am Schwerpunktabstand a. Die Stahlzugkraft F_{sd} ergibt sich aus dem Bemessungswert der Stahlstreckgrenze f_{yd} und der Fläche des Betonstahls A_s. Im Querschnitt bei reiner Biegung sind die beiden Kräfte im Gleichgewicht. Dabei ist z der Hebelarm. Der reduzierte Bemessungswert der Betondruckfestigkeit f_{cd}' ergibt sich aus der Abminderung des Bemessungswert der Betondruckfestigkeit f_{cd} um den Faktor χ. Das einwirkende Bemessungsmoment M_{Sds} ist das auf die Zugbewehrung „versetzte" Bemessungsmoment M_{Sd}.

Es folgen die Formeln für die Bemessung des Rechteckquerschnitts mit einfacher Bewehrung. Die Anwendungsgrenzen für diese Aufgabe wurde zu Beginn erläutert.

Bemessung nach DIN 1045-1

Gegeben:

Beanspruchung:

$$Bemessungsmoment \quad M_{Sd} = M_{Sds}$$

Materialkennwerte:

$$Bemessungswert\ der\ Stahlstreckgrenze \quad f_{yd}$$

$$Bemessungswert\ der\ Betondruckfestigkeit \quad f_{cd}'$$

Querschnittsgeometrie:

$$Breite \quad b, \quad Höhe \quad h$$

$$Nutzhöhe \quad d = h - d_1 = h - (c_{nom} + \frac{d_s}{2})$$

Gesucht:

$$Fläche\ des\ Betonstahls \quad A_s$$

Rechenweg:

$$\mu_{cd} = \mu_{cds} = \mu_{Sds} = \frac{M_{Sds}}{f_{cd}' \cdot b \cdot d^2}$$

$$\xi = 1{,}25 \cdot (1 - \sqrt{1 - 2 \cdot \mu_{cd}})$$

$\xi \leq \lim \xi$

$$\nu_{cd} = 0{,}8 \cdot \xi$$

$$A_s = \frac{\nu \cdot f_{cd}' \cdot b \cdot d}{f_{yd}}$$

$\xi > \lim \xi$

→ *Querschnitt vergrößern oder doppelte Bewehrung*

Diese Bemessung eines Rechteckquerschnitts mit reiner Biegung wird für die Umsetzung eines Berechnungsmoduls verwendet. Damit wird eine typische Bauingenieuranwendung auf dem PDA realisiert. Vorteilhaft ist diese Aufgabe wegen der einfachen Formeln und den wenigen Bedingungen.

Tragfähigkeit nach DIN 1045-1

Gegeben:

Materialkennwerte:

Bemessungswert der Stahlstreckgrenze f_{yd}

Bemessungswert der Betondruckfestigkeit f_{cd}'

Querschnittsgeometrie:

Breite b, *Höhe* h, *Fläche des Betonstahls* A_s

Nutzhöhe $d = h - d_1 = h - (c_{nom} + \frac{d_s}{2})$

Gesucht:

auf die Zugbewehrung „versetztes" aufnehmbares Moment

$M_{Rd} = M_{Rds}$

Rechenweg:

$$\nu_{cd} = \frac{f_{yd} \cdot A_s}{f_{cd}' \cdot b \cdot d}$$

$$\xi = 1{,}25 \cdot \nu_{cd}$$

$\xi \leq \lim \xi$ $\qquad$ $\xi > \lim \xi$

$\qquad\qquad\qquad\qquad\quad \nu = 0{,}8 \cdot \lim \xi$

$$M_{Rd} = M_{Rds} = \nu_{cd} \cdot (1 - 0{,}5 \cdot \nu_{cd}) \cdot f_{cd}' \cdot b \cdot d^2$$

Nachweis der Tragfähigkeit: $M_{Rds} \leq M_{Sds}$

Der Tragfähigkeitsnachweis wird in diesem Fall zur Kontrolle ausgeführt. Dabei wird die ermittelte Fläche des Betonstahls A_s aus der Bemessung als Eingabewert benötigt. Das Ergebnis der Berechnung ist das auf die Zugbewehrung „versetzte" aufnehmbare Moment M_{Rds}. Als Nachweis wird das aufnehmbare Moment M_{Rds} mit dem einwirkenden Moment M_{Sds} verglichen. [Massivbau 06]

2.2 Aufbau eines PDA

In diesem Kapitel wird der PDA als Gerät beschrieben. Ausgehend von den Technischen Grundlagen wird die Funktionsweise erklärt. Dann werden die Standardanwendungen genannt.

2.2.1 Technische Grundlagen

Ein PDA ist ein kompakter tragbarer Computer, der 1993 zum ersten Mal erschienen ist. Nach dem heutigen Standard sind die Geräte mit einem Touchscreen ausgestattet. Dieser ermöglicht eine interaktive Bedieneingabe über das LC-Display[5] mit Hilfe eines Stiftes.

Wichtige Einstellungen und persönliche Daten wie Adress-, Termin- oder Kontaktdaten werden von dem schnell startenden Betriebssystem im RAM[6] gespeichert. Einen ausgeschalteten Zustand gibt es nicht, sondern nur einen Standby-Modus. Daher ist die Benutzeroberfläche sofort nach dem Einschalten sichtbar, allerdings entleert sich der Akku schnell. Damit die Daten in Folge der Akkuentleerung nicht verloren gehen, gibt es eine zusätzliche Sicherungsbatterie. Die meisten Geräte verfügen weiterhin über externe Speicherkarten, z.B. im SD-Format[7], auf denen Daten und Anwendungen dauerhaft gespeichert werden können.

Um eine Vorstellung von einem PDA zu erhalten, werden zuerst die technischen Daten vorgestellt. Anhand dieser Auflistung wird deutlich, wo die Grenzen liegen. Begrenzte Kapazitäten und Ressourcen schränken die Möglichkeiten der Umsetzung von Berechnungen ein, wenn das Programm PDA-SB auf vielen Geräten laufen soll.

5 LCD (englisch liquid crystal display) = Flüssigkristallbildschirm
6 RAM (englisch random access memory) = Arbeitsspeicher
7 SD Memory Card (englisch secure digital memory card) = sichere digitale Speicherkarte

Technische Daten

Es gibt verschiedene PDA mit unterschiedlichen Eigenschaften. Das Spektrum variiert von der Größe des LCD-Monitors über die Auflösung bis hin zum Betriebssystem. Der Arbeits- und Festplattenspeicher ist ein wesentlicher Punkt, der komplexe Berechnungen ermöglicht.

Der Produktvergleich im Internet ergab eine Übersicht von verschiedenen Produkten. Ein kleiner Auszug ist in Tabelle 1 zu sehen. [Günstiger 09]

Produkt	Palm Z22	Fujitsu-Siemens Pocket Loox N560	HP-Hewlett-Packard iPAQ 214	Nokia N810
Gruppe	PDA	Pocket PC	PDA	Internet Tablet
Anschlüsse, Schnittstellen PC		Wireless LAN	Bluetooth, Wireless LAN	
Display LCD-Monitor Auflösung Anz. d. Farben	 160 x 160	 3,5'' 480 x 640 65.563	 4'' 460 x 480 262.144	 4,13'' 800 x 480
Eingabe	Touchscreen	Touchscreen	Touchscreen	Touchscreen, Tastatur
Prozessor Takt		Intel 624 MHz	Sonstige 624 MHz	
Betriebssystem	Palm OS	Microsoft Windows Mobile 5.0	Microsoft Windows Mobile 6.0	
Speicher ROM[8] Speicher RAM Speichermedien	32 MB	128 MB 64 MB MMC, SD	 128 MB CF, SD, SDHC	256 MB 128 MB microSD, MiniSD, SD

Tabelle 1: Auszug aus dem Produktvergleich

Aus der Tabelle 1 ist ersichtlich, dass die Produkte stakt schwanken. Die Bildschirmauflösung der Geräte liegt zwischen 160x160 und 800x480. Der Speicher des Palm OS Gerätes ist wesentlich niedriger als der anderen Modelle. Durch diese unterschiedlichen Eigenschaften ist es nicht möglich, einen typischen PDA zu klassifizieren.

8 ROM (read-only memory) = Festwertspeicher oder Nur-Lese-Speicher

Betriebssysteme

Im Laufe der Zeit wurden viele Betriebssysteme für verschiedene PDA erstellt. Die folgende Tabelle 2 zeigt die Systeme in Abhängigkeit zum Gerät bzw. zur Firma.

Betriebssystem	Gerät / Firma
Newton OS	Newton
PEN/GEOS	z.B. Casio Z-PDA „Zoomer", HP Omnigo 100/120
Palm OS	Palm
Pocket PC / Windows Mobile	z.B. Dell, HP, Acer
Linux	z.B. HP/Compaq iPAQ, SHARP Zaurus, Zimputer
NetBSD	SHARP Zaurus
OpenBSD	SHARP Zaurus
EPOC, Basis von Symbian OS	Psion Tastatur-PDA, Smartphones
Windows CE (HandheldPC)	Psion Teklogix netBook pro, HP600 und ältere

Tabelle 2: Übersicht über die Betriebssysteme von PDA

Es gibt eine große Auswahl von Betriebssystemen. Das sollte im Entwurf berücksichtigt werden. Wenn es möglich ist, sollte das Programm auf vielen Geräten funktionieren.

Anschlüsse

Der PDA kann verschiedene Anschlüsse besitzen. Darüber können Daten ausgetauscht werden. Welche es gibt, zeigt ein Auszug in Tabelle 3.

Anschluss	Kurze Beschreibung
Infrarot	kabelloses, optisches Übertragen von Daten
Bluetooth	drahtlose Funkübertragung zwischen Bluetooth-Geräte
Wireless LAN	drahtlose Funkübertragung zwischen W-LAN fähigen Geräten
RS232	zur Synchronisation oder zum Anschluss von GPS-Empfängern
USB Slave	kabelgebundene Synchronisation mit einem PC
USB Host	direkte Kommunikation mit anderen USB-Geräten

Tabelle 3: Anschlüsse eines PDA

Eine wichtige Schnittstelle ist der USB-Anschluss, den fast jedes Gerät vom Handy über Kamera bis hin zum PDA besitzt. Ein weiterer wichtiger Anschluss ist Wireless LAN. Dieser ermöglicht den Datenaustausch übers Internet. Für die Umsetzung des Programms ist die Information nützlich, denn die Berechnungen können ggf. übers Internet nachgeladen werden. [PDA Wiki 08]

2.2.2 Anwendungszweck

Es sind viele Anwendungen auf einem PDA möglich. Standardmäßig wird die PIM[9]-Software mitgeliefert. Diese umfasst das Adressbuch, den Terminplaner, den Kalender, das Notizbuch, den Aufgabenplaner, das E-Mail Postfach und den Projektmanager. Oft sind Office-Programme (z.B. Textverarbeitung, Tabellenkalkulation und Taschenrechner) und Spiele auf den Geräten installiert. Die Aufnahme und Wiedergabe von Musik und Videos sind bei aktuellen Modellen möglich. Software kann auf den PDA geladen werden. Es funktioniert mit oder ohne Kabel, auch übers Internet. Der PDA kann infolge der gestiegen Leistungsfähigkeit für mobile Datenerfassung genutzt werden. Durch die stetige Entwicklung ist die Verwendung als IP-Telefon oder Navigationssystem mit integriertem GPS-Empfänger umgesetzt. Die Synchronisation eines PDA mit dem PC erfolgt problemlos über Programme wie HotSync oder ActiveSync. Damit können E-Mails, Adressen und Termine ausgetauscht oder andere Datenbanken aktualisiert werden. Anwendungen anderer Einsatzbereiche nutzen die Vorteile eines PDA. [PDA Wiki 08]

Ein Einsatzgebiet ist die Baustelle, wo der PDA durch hohe Leistungsfähigkeit und kompakter Größe vorteilhaft nutzbar ist. Ein programmierbarer Taschenrechner wäre eine Alternative. Dieser kann die Berechnungen ausführen, aber die Benutzerfreundlichkeit leidet, in dem keine Hilfe oder Anleitung zur Aufgabe vorhanden ist. Das bedeutet, bei einer solchen Umsetzung muss sich der Programmierer eindeutige Namen einfallen lassen, die der Nutzer versteht und zuordnen kann. Ein Nachlesen in Büchern ist auf der Baustelle ungünstig, wegen den Witterungsverhältnissen und der Mobilität. Aus diesem Grund bietet sich der PDA an, denn dieser kann innerhalb der Anwendung wichtige Inhalte auflisten. Das Programm führt den Nutzer durch die Aufgabe und gibt

9 PIM (englisch personal information manager) = Produkt-Infomations-Management

Hilfestellung bei Fragen zur Berechnung. Werte aus Tabellenbücher werden angeboten, die nur ausgewählt werden müssen. Dadurch ist ein Buch unnötig und schränkt die Aktivitäten des Benutzers auf der Baustelle nicht ein.

Die Nutzung des Programms kann auch im Büro oder in der Hochschule stattfinden. Im Lernbereich der Ausbildung von Bauingenieuren und Bauinformatikern können Studenten mit der Software PDA-SB arbeiten oder eigene Berechnungen für das Programm anfertigen. Die Erstellung von neuen Berechnungsmodulen sollte auch Personen ohne Programmierkenntnisse möglich sein. Umsetzbar ist das mit Hilfe von vorgefertigte Strukturen oder durch das Zusammensetzen von Berechnungsteilen. Eine Anleitung zur Nutzung der Aufgabe wird von Fachleuten geschrieben. Damit ist die Anwendung für andere Benutzer besser zu verstehen. Um solch ein Programm zu realisieren, dass flexibel auf die Anforderungen der Programmierer reagiert, bedarf es einem modularen Aufbau und eine Trennung zwischen der Berechnung und dem Hauptprogramm.

2.3 Untersuchung vorhandener Software

Im folgenden Kapitel wird vorhandene Software untersucht. Die Analyse beginnt mit den PDA-Programmen. Es wird geprüft, welche Aufgaben umgesetzt wurden. Das zeigt die Leitungsfähigkeit eines PDA. Die Vor- und Nachteile werden ermittelt. Anschließend werden Programme für den PC analysiert. Dabei ist die Vorgehensweise der Berechnungen interessant. Der Aufbau von möglichen Aufgaben, die ggf. noch nicht auf einem PDA umgesetzt sind, ist zu untersuchen. Das Kapitel endet mit dem Vergleich zwischen den Programmen für den PDA und dem PC.

2.3.1 Programme für den PDA

Im Internet sind überwiegend kleine Anwendungen für einen PDA zu finden, die von Bauingenieuren geschrieben sind. Diese Programme ergeben kein einheitliches Ganzes, so dass diese völlig losgelöst voneinander genutzt werden können.

Daraus entstehen einige Nachteile. Zum Beispiel ist das die fehlende Kommunikation zwischen den Programmen, d.h die Programme können keine Daten untereinander austauschen. Daher gibt es keine gemeinsame einheitliche Speicherung. Das schränkt die Nutzung und die Weiterverarbeitung in anderen Programmen ein. Des Weiteren fehlt eine komplette Auflistung der Programme nach Anwendungsgebieten. Wenn der Benutzer eine Berechnung benötigt, muss diese im Internet gesucht werden. Das nimmt viel Zeit in Anspruch. Darüber hinaus gibt es keine Sicherheit auf Richtigkeit oder Vollständigkeit der Aufgaben.

Der große Vorteil dieser Programme ist die kostenlose Nutzung. Diese sind oft als Freeware-Version oder als Open-Source-Produkt vorhanden. Die eigene Anpassung ist mit Kenntnissen aus der Programmierung möglich.

Einige wenige Komplettprogramme von Softwareanbietern sind vorhanden. Beispielhaft sind das Staticus 2.5 von CreativeByte[10] und CS-PALM von CSI GmbH[11]. Beide Firmen bieten große Programme mit einer umfassenden Bibliothek an, die allerdings kostenpflichtig sind. Beide Programme sind für das Betriebssystem PALM OS geschrieben. Wegen geringer Nachfrage wurde Staticus 2.5 eingestellt, währenddessen es demnächst für CS-PALM eine Version auf Windows Mobile geben soll. Dennoch können durchaus auch alte Produktinformationen hilfreich sein, um den Aufbau und die Struktur zu erkennen. Ebenfalls zeigen sich die Funktionen, welche der Bauingenieur gebrauchen könnte. Bilder veranschaulichen das Gesamtkonzept und das Design.

Staticus 2.5

Das Hauptmenü in Abbildung 7 von Staticus 2.5 zeigt eine Auswahl von Aufgaben. In Abbildung 8 ist die Eingabemaske für die Berechnung des Ausnutzungsgrads eines Holzbalkens dargestellt. Die Eingabe erfolgt mit dem „Stift" über die Tastatur. Optionen können über DropDown-Menüs gewählt werden. Der *Hilfe*-Button führt zu einer Skizze. Biegespannungs-, Schubspannungs- und Verformungsnachweise werden bei der Berechnung

10 http://www.staticus.de
11 http://csi.gmbh.de

berücksichtigt. Nach dem Klick auf den Button *Berechnen* erfolgt die Anzeige der Ergebnisse in einer Informationsbox, siehe Abbildung 9. Bei der Rahmenberechnung in Abbildung 10 sind die verschiedene Rahmenskizzen zu sehen, die die Aufgaben definieren. Unterschieden wird nach den Belastungen. Für Stahlberechnungen stehen verschiedene Profildaten standardmäßig zur Verfügung. Das können beispielsweise mittelbreite I-Träger mit den Reihen I und IPE oder breite I-Träger HEA, HEB und HEM sein. Die Ermittlung der Schnittgrößen und die Berechnungen, die das Tabellenbuch teilweise ersetzen, sind vorhanden. Umrechnungen der Einheiten sind möglich. [Staticus 06]

Abbildung 7: Staticus 2.5 Hauptmenü für die Berechnungen

Abbildung 8: Staticus 2.5 Eingabe Holzbalken

Abbildung 9: Staticus 2.5 Ergebnis Holzbalken

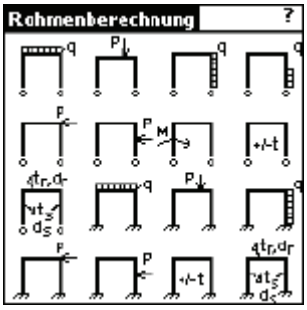


Abbildung 10: Staticus 2.5 Rahmen-berechnung

2.3.2 Programme für den PC

Es gibt eine große Auswahl an PC-Programmen, die Bauingenieuraufgaben berechnen. Beispielhaft sind das folgende Programme: ANSYS[12], ALLPLAN[13], BauStatik[14] und MicroFe[15]. Die letzten beiden gehören zu der Softwarereihe von mb AEC GmbH. Es gibt ebenfalls kostenlose Programme, die meistens kleine Teilgebiete abdecken, wie Inca2[16].

BauStatik

BauStatik von mb AEC Software GmbH ist mit über 90 Einzelprogrammen eines der umfangreichsten Statik-Systeme. Mit der Version 2006 hat sich die Bearbeitung von statischen Berechnungen geändert. Es gibt keine nach außen sichtbare Trennung zwischen Eingabe, Verarbeitung und Ausgabe, denn die Bearbeitung findet nun dokument-orientiert statt. Damit werden Änderungen in der Eingabe sofort in der Ausgabe sichtbar. Die Seitenausgabe wird durch ein gewähltes Layout dargestellt, siehe Abbildung 11. [mb AEC Software 09]

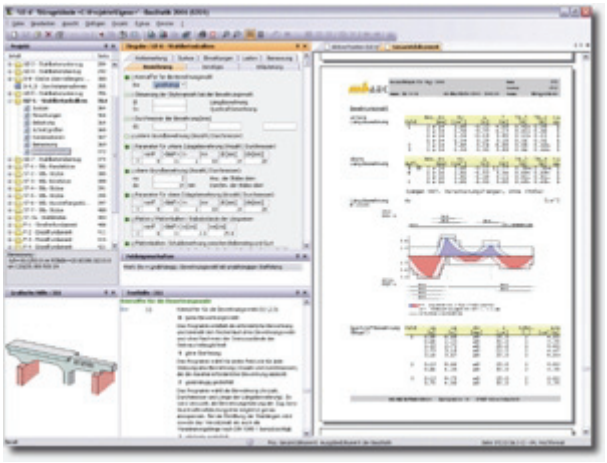


Abbildung 11: Design von BauStatik

12 Analysis System, http://www.ansys.com
13 http://www.nemetschek.de/
14 http://www.mbaec.de/baustatik/produktinfo.html
15 http://www.mbaec.de/microfe/produktinfo.html
16 http://www.u-pfeiffer.de/

ANSYS

„Der Name ANSYS steht für das kompletteste Angebot an High-End FEM[17]-Simulationstechnologie auf dem Markt, genutzt von der größten Anwender-Community weltweit."[18] Dieses Softwarepaket ist nicht nur für den Bauingenieurbereich interessant, sondern für jegliche Bereiche in denen FEM-Berechnungen und Darstellungen von Bedeutung sind. Im Bauwesen ist die Mechanik ein bedeutender Bereich, welcher FEM-Lösungen verwendet. Konstruktionen von aufwendigen Gebilden können direkt im Programm erstellt oder über die Schnittstellen eingelesen werden. Weitere Anwendungsgebiete sind die Automobilindustrie, die Luft- und Raumfahrt, die Schifffahrtsindustrie und der Maschinenbau.

ANSYS besitzt eine grafische Ein- und Ausgabe. Manuelle Texteingaben oder Korrekturen sind jederzeit möglich. Die Ergebnisse werden zusätzlich in Tabellen angezeigt. Die Berechnungen erfolgen im Hintergrund und für den Benutzer nicht sichtbar. Einsichten in den Quellcode sind nicht gegeben, so dass die Grundlage der Berechnungsphilosophie im Dunkeln bleibt. Allerdings ist die Methode der Finiten-Elemente bekannt.

Durch das enorme Leistungsangebot, welches ANSYS zur Verfügung stellt, ist die Anwendung auf einem PDA zum aktuellen Zeitpunkt nicht denkbar. Die benötigte Rechenleistung und Komplexität der möglichen Anwendungen übersteigen die Fähigkeiten eines Handrechners. Vereinfachte Modelle wären allerdings praktisch für die Handhabung von Bauingenieuranwendungen. In der Statik finden FEM-Lösungen ihren Platz. Zum Beispiel wird das Modell für ebene und räumliche Fachwerke verwendet. In anderen Bereichen wie Massivbau oder Stahlbau werden einfache Berechnungen bevorzugt.

2.3.3 Vergleich PC und PDA Programme

Im Bauingenieurbereich sind viele Softwarelösungen bereits vorhanden. Einige spezialisieren sich auf einzelne Teilgebiete, so zum Beispiel die Programme aus der Bodenmechanik. Andere dagegen sind so komplex, dass sie mehrere

17 FEM (englisch finite element method) = numerisches Verfahren zur Näherungslösung
18 http://www.cadfem.de/ANSYS.1105.0.html

Ingenieurbereiche abdecken, wie das Softwareprogramm ANSYS. Diese große Auswahl gibt es allerdings nur bei den PC-Anwendungen. Für einen PDA dagegen gibt es weniger und vor allem eher kleinere Softwarelösungen, die keine Daten untereinander austauschen können. Die wenigen Komplettsysteme sind mit Lizenzkosten verbunden. Um eine kostenlose Variante zu schaffen, wird das Programm PDA-SB erstellt.

Eine wichtige Gemeinsamkeit zwischen den Programmen ist auffällig. Sie nutzen eine interne Teilung zwischen der Eingabe, der Berechnung und der Ausgabe. Oft ist diese für den Nutzer nicht sichtbar. Nach außen wird ein Datenblatt oder eine Skizze gezeigt. Solche dokument-orientierten Programme gibt es in verschiedenen Bereichen. Damit ist eine Vorschau auf die Druckversion gegeben. Die internen Berechnungen sind nicht einsehbar. Es ist zu vermuten, dass FEM und Bauaufgaben genutzt werden.

Im Anschluss werden die Betrachtungsergebnisse der Programme zusammengefasst, welche den Kreis zu der eigentlichen Entwicklungsaufgabe eines PDA-Softwareprogramms schließen.

2.4 Zusammenfassung der Analyse

Die Zusammenfassung der Analyse bildet die Grundlage für den Entwurf. Es ergeben sich Aufgaben, welche das PDA-Programm können sollte.

Die Berechnungen aus dem konstruktiven Ingenieurbau müssen umgesetzt werden. Zur Vereinfachung der Arbeitsschritte sollen Tabellenwerte wie die Druckfestigkeitsklassen vom Programm vorgegeben werden. Beschreibungen der Formeln und Erklärung der Symbole / Formelzeichen sollen durch Aufforderung einsehbar sein. Durch die häufige Benutzung einer Berechnung ist eine ständige Hilfe lästig. Vorgefertigte Konstruktionen vereinfachen die Auswahl für Einsteiger.

Der PDA besitzt teilweise eine hohe Leistungsfähigkeit. Damit sind komplexe Aufgaben wie einfache Finite Elemente Methoden umsetzbar. Da das Programm PDA-SB auf möglichst viele Geräten nutzbar sein soll, müssen die Berechnungen einfach bleiben. Es gibt auch alte Modelle auf dem Markt, denen wenig Arbeitsspeicher zur Verfügung steht. Das Programm soll unabhängig vom Betriebssystem funktionieren. Die große Auswahl an Anschlüssen eines PDA bieten viele Varianten zum nachträglichen Hinzufügen von Berechnungsaufgaben. So ist ein Datentransport übers Internet genauso möglich wie per Kabel.

Aus dem Vergleich der Programme ergibt sich eine Struktur für Berechnungsaufgaben. Diese ist die Teilung zwischen der Eingabe, der Berechnung und der Ausgabe. Die Ergebnisse sollen ähnlich eines Dokuments angezeigt werden. Direkt vom PDA muss nicht gedruckt werden können. Aber es soll dennoch möglich sein, die Resultate auf dem PC anzeigen zu lassen und dort zu drucken. Realisierbar ist der Datentransport durch einen geeigneten Export.

Wenn die einzelnen Berechnungen Daten austauschen sollen, muss eine gemeinsame Datenstruktur gefunden werden. Das ist ein großer Vorteil gegenüber den vielen kleinen Programme, die es für einen PDA gibt. Denn diese haben keine geeignete Schnittstelle zum Datenaustausch. Bei vielen Berechnungen ist es angebracht, die Aufgaben zu verwalten. Eine Sortierung nach Anwendungsgebieten ermöglicht eine schnelle Suche.

Um nicht zwei Softwarelösungen zu erstellen, werden beide verbunden. Das Hauptprogramm bildet den Rahmen, verwaltet und steuert die Berechnungen. Es bietet zugleich Funktionen über die gemeinsamen Datensätze. Beispielsweise ist das der Import, der Export und die Speicherung. Die eigentliche Berechnung wird in einem Modul stattfinden. Die Aufgabe umfasst die Eingabe, die Berechnung, die Ausgabe und eine separate Hilfe. Damit beide Teile (Rahmen und Modul) miteinander Daten austauschen können, müssen Schnittstellen definiert werden. Dabei soll die Steuerung des Moduls vom Rahmen erledigt werden.

3 Programmentwurf

Der Entwurf des Programms beginnt mit dem Aufbau. Es wird nicht ein klassisches Modell verwendet, sondern eine Trennung des Rahmenprogramms von den Berechnungen. Die Aufgaben sind als Module aufgebaut. Es folgen genauere Betrachtungen der Daten innerhalb des Systems. Dabei ist die Struktur der Werte notwendig, um diese zwischen den Aufgaben auszutauschen. Verschiedene Varianten der Speicherung sind möglich. Eine davon wird in einem Datenbankmodell beschrieben. Die gemeinsamen Speicherung der Daten in einer Datenbank erlaubt es Berechnungsketten zu erstellen. Zuletzt wird ein passendes Design vorgestellt, dass die Oberfläche für den Benutzer leicht verständlich und zugleich ansprechend gestaltet.

3.1 Programmaufbau

Typischer Weise wird ein Programm nach dem EVA-Prinzip aufgebaut. Das beschreibt die Reihenfolge (**E**ingabe, **V**erarbeitung, **A**usgabe) einer Programmierung. Abgeleitet ist dieses Prinzip aus den Anfängen der Computerzeit und bezieht sich auf die Hardware. Die Eingabe steht für die Peripheriegeräte wie Tastatur oder Maus. Der Prozessor mit der Rechenleistung steht für die Verarbeitung. Für die Ausgabe stehen Geräte wie Monitor oder Drucker. Gleichbedeutend gilt das für die Software. [Lex_calsky 09][IT_Wissen 09]

Diese Grundlage ist auch die Basis für das zu entwickelnde Programm, nur mit dem Unterschied, dass zwei Ebenen betrachtet werden. Die Eine umfasst das Rahmenmodul, welches ein starres System ist, und zum Zweiten die Berechnungen, welche dynamisch aufgebaut sind. Der Rahmen ermöglicht Module nach dem EVA-Prinzip einzuladen.

Ohne die Auslagerung der Berechnungen vom Hauptprogramm ist eine spätere Änderung oder Erweiterung der Funktionen kompliziert und bedeutet den Umbau des Programms. Anhand des folgenden Beispiels wird das deutlich: Multiplikation

zweier vom Nutzer eingegeben Variablen. In der unten stehenden Abbildung 12 ist diese Aufgabe in einem Programmablaufplan einer PC-Anwendung gezeigt.

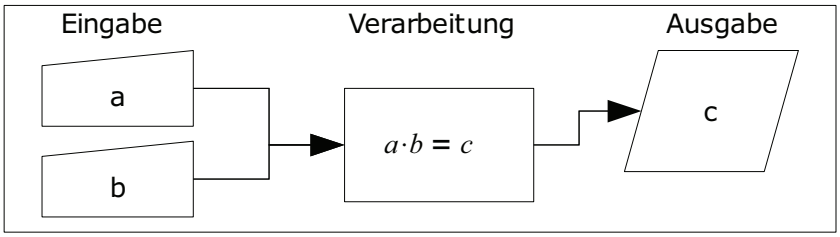


Abbildung 12: Programmablaufplan einer Multiplikation

Das Softwareprodukt kann zwei Variablen miteinander multiplizieren. Wenn der Nutzer eine weitere Funktion, wie z.B. die Addition, hinzufügen möchte, muss das Programm umgeschrieben bzw. erweitert werden. Der Quellcode muss angepasst werden. Nach dem Übersetzen kann das Programm wieder verwendet werden.

Da das zu entwerfende Programm oft um Berechnungen erweitert wird, ist dieser Vorgang ungeeignet. Ein passenderes Konzept ist die Trennung der Berechnung vom Hauptprogramm. Das Rahmenprogramm bildet die Basis und lädt die Aufgabenmodule ein, um die Flexibilität zu gewährleisten.

Der Benutzer kann von außen seine Eingaben tätigen, welche eine Aktion beim System aufruft und ggf. Daten an die Berechnung schickt oder holt. Die Benutzung basiert auf eine gegenseitige Kommunikation, wobei nur das Rahmenprogramm bekannt ist. Um diesen Austausch von Daten und Aktionen zu realisieren, müssen Schnittstellen entworfen und umgesetzt werden. Die Speicherung der Daten sollte ebenfalls über das Hauptprogramm erfolgen. Die folgende Abbildung 13 zeigt schematisch den Aufbau der kompletten Softwarelösung.

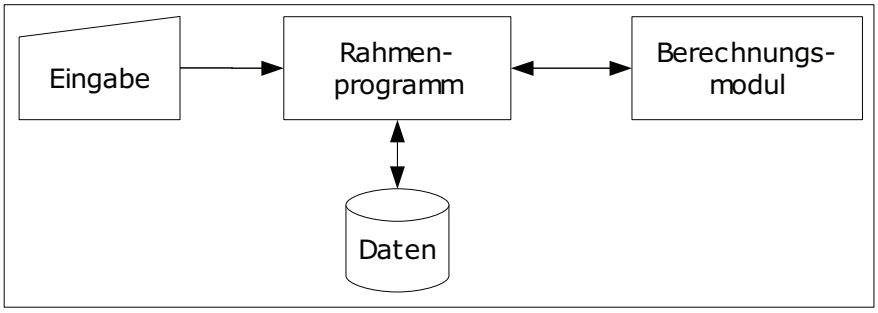


Abbildung 13: Aufbau des Programms

In der oben stehenden Abbildung 13 befindet sich das Rahmenprogramm im Zentrum. Der Nutzer hat die Möglichkeit die Eingabe zu tätigen. Diese Ansteuerung kann vieles sein, z.B. öffnen eines Berechnungsmoduls oder gespeicherte Daten auswählen. Das Modul soll keine Daten aus einer Datenbank oder anderer Datenspeicher lesen, sondern stellt eine Anfrage an das Hauptprogramm. Damit ist eine Unabhängigkeit vom Modul gegeben und die komplette Steuerung wird vom Rahmenprogramm erledigt. Wenn das nicht möglich ist, kann das Modul eigene Daten aufrufen, die dann allerdings nicht über das System für andere Module zugänglich sind. Es gibt eine Ausnahme. Diese ist dann gegeben, wenn die Berechnung die eingelesenen Daten zur Speicherung im System frei gibt. Dann kann eine andere Berechnung den Weg einer Berechnungskette folgen und die gespeicherten Daten einlesen.

Im Folgenden wird der Aufbau und die Funktionsweise der einzelnen Programmteile näher beleuchtet.

3.1.1 Hauptprogramm

Das Hauptprogramm bildet den Rahmen um die Berechnungsmodule, welche die wichtigsten Aufgaben umfassen. Diese sind das Modul laden, ausführen und schließen. Die nachstehende Abbildung 14 veranschaulicht dieses Prinzip.

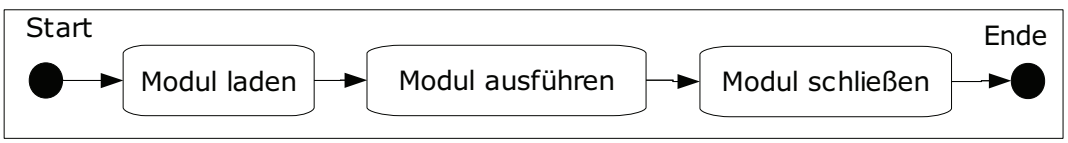


Abbildung 14: Hauptaufgaben des Rahmenprogramms

Während ein Berechnungsmodul ausgeführt wird, müssen weitere Aufgaben vom Rahmenprogramm übernommen werden. Nach dem Laden muss die Eingabemaske angezeigt werden. Wenn der Nutzer die Eingaben erledigt hat, muss die Berechnung gestartet werden. Nachdem die Ergebnisse vorliegen, müssen die Daten an die Ausgabe geschickt werden. Zwischendurch muss der Nutzer jederzeit die Möglichkeit haben, die Hilfe anzuschauen. Die folgende Abbildung 15 zeigt die Kommunikation zwischen Rahmenprogramm und Berechnungsmodul.

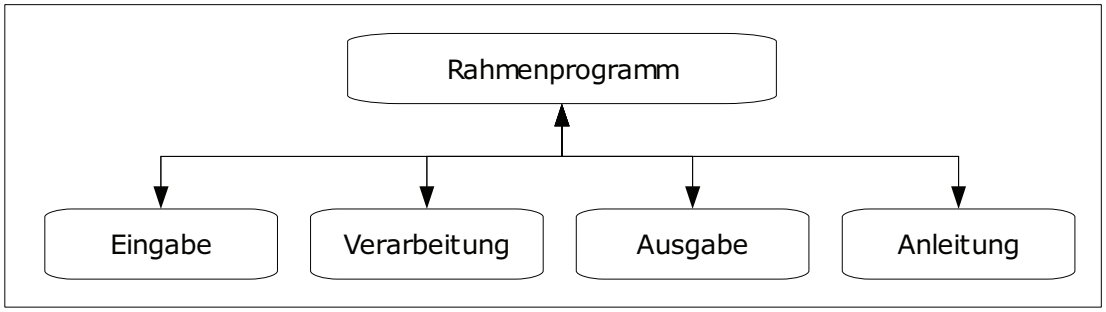


Abbildung 15: Kommunikation zwischen Rahmenprogramm und Modul

Des Weiteren soll das System die Daten bzw. Werte verwalten. Damit ist der Umgang mit den Daten gemeint, also z.B. die Datenspeicherung oder der Im- und Export. Detaillierte Beschreibungen zu dem Thema sind im Kapitel 3.2 Datennutzung im System zu finden.

Im Anschluss wird das Berechnungsmodul beschrieben, welches eine Aufgabe berechnet. Dieses wird vom Hauptprogramm geladen.

3.1.2 Berechnungsmodul

Aufgrund des EVA-Prinzips soll das Modul mindestens aus einem Eingabe-, Verarbeitungs- und Ausgabeteil bestehen. Hinzukommend ist eine Anleitung sinnvoll, die dem Benutzer selbst schwierige Aufgaben auf einfachem Wege erklärt. Das Berechnungsmodul steht in Kommunikation mit dem Rahmenprogramm, sprich in einem Datenaustausch, welches in Abbildung 15 bereits gezeigt wurde.

Ein typischer Ablauf ist die Ausführung einer Berechnung, welche der Benutzer durch öffnen eines Moduls erreicht. Standardmäßig wird nach dem Laden die Eingabemaske angezeigt. Der Benutzer hat dann die Möglichkeit die Werte einzutragen. Nach erfolgreichem Eintragen kann der Benutzer die Berechnung starten. Wenn diese fertig ist, wird automatisch die Ausgabe angezeigt. Danach kann der Nutzer erneut das Modul für eine weitere Berechnung verwenden, eine andere Aufgabe öffnen oder das Programm beenden. Der beschriebene Ablauf wird im Sequenzdiagramm in Abbildung 16 gezeigt.

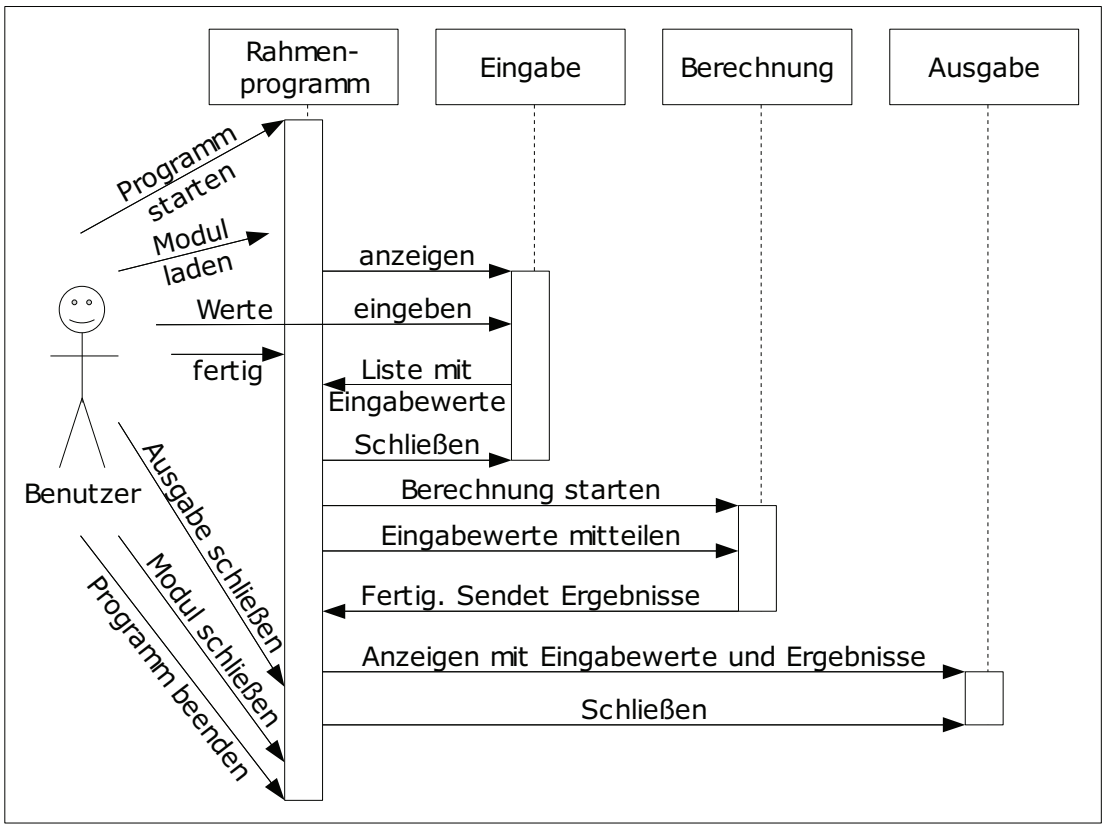


Abbildung 16: Sequenzdiagramm einer Berechnung

Ein weiterer typischer Ablauf ist die Nutzung der Hilfedatei des Berechnungsmoduls. Diese Datei kann eine Anleitung für die korrekte Benutzung der Berechnung sein. Es können Hinweise zum Ausfüllen der Eingabemaske, Berechnungshintergründe, Formeln, Theorien oder Erklärungen verankert sein. Der Aufruf der Hilfe wird im folgenden Sequenzdiagramm in Abbildung 17 demonstriert.

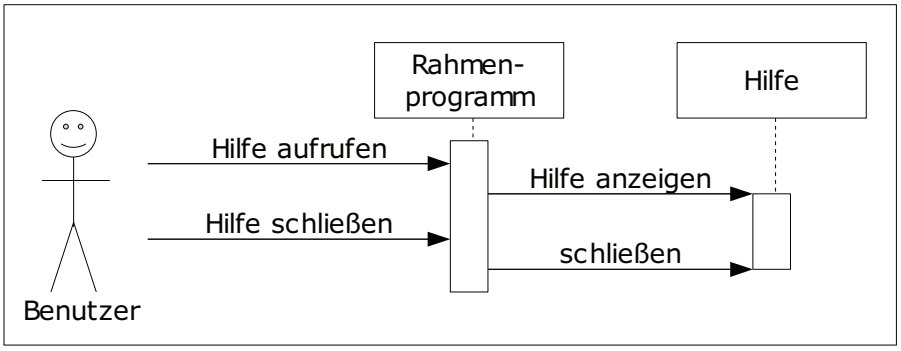


Abbildung 17: Sequenzdiagramm zur Verwendung der Hilfe

Schnittstellendefinition bilden den Rahmen eines Moduls und stellen zugleich die Kommunikation mit dem Hauptprogramm dar. Damit ein Teilgebiet keine Oberfläche benötigt, wird nur ein allgemeines Interface gegeben. Der Programmierer kann den weiteren Aufbau selbst bestimmen. Näheres wird im folgenden Kapitel beschrieben.

3.1.3 Schnittstellen

Als Schnittstellen werden die Interfaces bezeichnet, welche für das Rahmenprogramm bzw. für die Berechnungsmodule benötigt werden. Sie umfassen die Methoden und Funktionen, die zur Verwendung stehen. Damit wird der Datenaustausch zwischen den Klassen gewährleistet. Die Daten sollen vom Rahmenprogramm zum Berechnungsmodul geschickt werden. Wenn es ein „Okay" von der angesteuerten Klasse gibt, werden die Ergebnisse abgefragt. Ebenso soll das System die Möglichkeit haben, die Sichtbarkeit einzustellen und den Startbefehl zu geben. Diese grundlegenden Methoden und Variablen können für jede Berechnungsklasse abgeleitet werden. Für die Eingabe, die Berechnung, die Ausgabe und die Hilfe wird jeweils ein Interface gestaltet, welches nur Methoden und Variablen beinhaltet, die benötigt werden. Die Suche für den Ladevorgang läuft dann über die vorgefertigte Schnittstelle.

Die Standardmethoden sind in Abbildung 18 aufgelistet. Diese sind show() zum Anzeigen des Fensters, close() für das Schließen der Klasse und setSystem um dem Modul die Systemklasse mitzuteilen. Mit Hilfe der Objekte, die das Hauptprogramm zur Verfügung stellt, kann das Berechnungsmodul mit sich

selbst Daten austauschen. Das ist notwendig, um die Instanzen des Rahmenprogramm verwenden zu können.

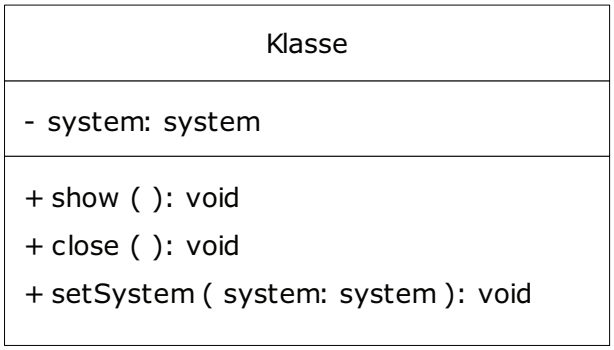


Abbildung 18: Methoden der Klasse

Darüber hinaus muss die Eingabe die Daten in einer Variable speichern. Entsprechende Methoden zum Abruf der Werte müssen vorhanden sein. Die Berechnung benötigt eine weitere Variable für die Ergebnisse, Methoden um diese Werte abzurufen und einen Startbefehl, z.B. run(). Die Ausgabe dagegen benötigt keine Startsequenz, sondern kann mit der show() Methode angezeigt werden. Die Eingabedaten und Ergebnislisten müssen verwaltet werden. Die Hilfedatei kommt mit den Standardmethoden aus. Aufgrund dieser Basis können entsprechende Interfaces geschrieben werden, welche im Kapitel 4.3 Schnittstellen erläutert werden.

Das Interface System beschreibt das Rahmenprogramm. Da das Hauptprogramm durch das Einladen eines Berechnungsmoduls die Klassen als Objekte anlegt, müssen die Instanzen vom System geholt werden. Damit können die Daten innerhalb der Aufgabe verwendet werden, ohne die Steuerung zu übernehmen. Die Eingabewerte und Ergebnisse stehen ständig dem Rahmenprogramm zur Verfügung. In Abbildung 19 ist die Schnittstelle abgebildet. Zu sehen sind die Methoden getInput(), getOutput, getCalculation() und getHelp(). Diese stehen für die Objekte der Aufgabe (Eingabe, Berechnung, Ausgabe und Hilfe).

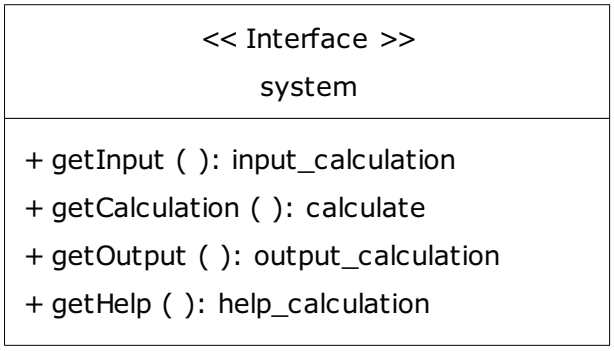


Abbildung 19: Schnittstelle für das Rahmenprogramm

Im Weiteren werden die Daten, die in einer Aufgabe verwendet werden, beschrieben.

3.2 Datennutzung im System

Im Programm muss die Struktur der Daten definiert werden. Auf dieser Basis kommunizieren die einzelnen Klassen miteinander und tauschen Werte aus. Um alle Datentypen zu erfassen, gibt es einen allgemeinen Datenaufbau. Näheres wird im folgenden Abschnitt erklärt. Im Weiteren müssen diese Daten bzw. Werte gespeichert werden. Es werden verschiedene Methoden der Speicherung erläutert. In einem Datenbankmodell wird eine Variante detaillierter beschrieben. Aufgrund der einheitlichen Datenstruktur und der gemeinsamen Speicherung sind Berechnungsketten möglich.

3.2.1 Datenstruktur

Für die Kommunikation zwischen dem Modul und dem Rahmenprogramm sowie innerhalb der Berechnungsmodule wird eine vordefinierte Datenstruktur benötigt. Diese Struktur leitet sich anhand der geforderten Aufgaben ab. Für eine Berechnung werden einfache Zahlen, Vektoren und Matrizen benötigt. Diese besitzen eine Bezeichnung, einen Namen, eine Einheit und entweder einen Wert, einen Vektor oder eine Matrix. Beispielhaft wird das in Abbildung 20 deutlich.

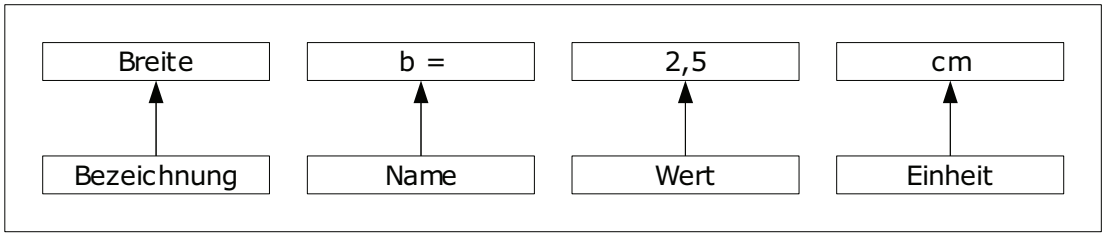


Abbildung 20: Bsp. Struktur eines Wertes

Die Zahlen, Vektoren und Matrizen werden mehrfach benötigt und sollen in einer Liste gespeichert werden. Um daraus einen bestimmten Wert zu finden, muss dieser nach einem Algorithmus gesucht werden. Es gibt Verschiedene, die eingesetzt werden können. Der Einfachste durchsucht linear die Liste. Dabei wird diese nach dem Datentyp sortiert. Weil das Rahmenprogramm die Typen aus den Berechnungsmodulen nicht kennt, ist die Datenstruktur allgemein gestaltet. Die Werte werden in Texten gespeichert. Damit können viele Möglichkeiten umgesetzt werden. Zum Beispiel wird eine Zahl in einen Text umgewandelt und kann bei Bedarf wieder zurück verwandelt werden. Das bedeutet, die Berechnung konvertiert die benötigten Werte in einen String und speichert diese in der Liste. Wenn ein Wert gebraucht wird, wird dieser aus dem Speicher gesucht und zurück in den Ausgangstypen konvertiert. Weil die Entscheidung des Datentyps bei der Aufgabe liegt, sind alle denkbaren Fälle möglich, die innerhalb der Speichergröße des vordefinierten Texts liegen. Damit sind beispielsweise Integer, Double, Datetime, Boolean oder selbstdefinierte Datentypen wie Point oder Line realisierbar.

In Abbildung 21 ist die Klasse *data* mit den Eigenschaften, Methoden und Konstruktoren abgebildet. Diese beschreibt das Schema der Struktur eines Wertes. Dabei sind die Eigenschaften *Name*, *Bezeichnung*, *Einheit* und *Wert* als String vorhanden. Der Konstruktor benötigt zwei Parameter, welche die Minimalanforderung an das Objekt stellen. Das sind Name und Zahl, z.B. a=2. Ohne diese Angaben kann das Programm nicht ordentlich arbeiten. Denn anhand des Namens ist die Suchfunktion standardmäßig aufgebaut. Dazu muss z.B. eine Zahl hinterlegt sein, weil ansonsten diese Variable nicht notwendig ist. Der Wert kann als String, Integer oder Double zurückgegeben werden. Falls andere Datentypen wie z.B. Boolean hinterlegt sind, muss dieser als Sting zur Konvertierung verwendet werden.

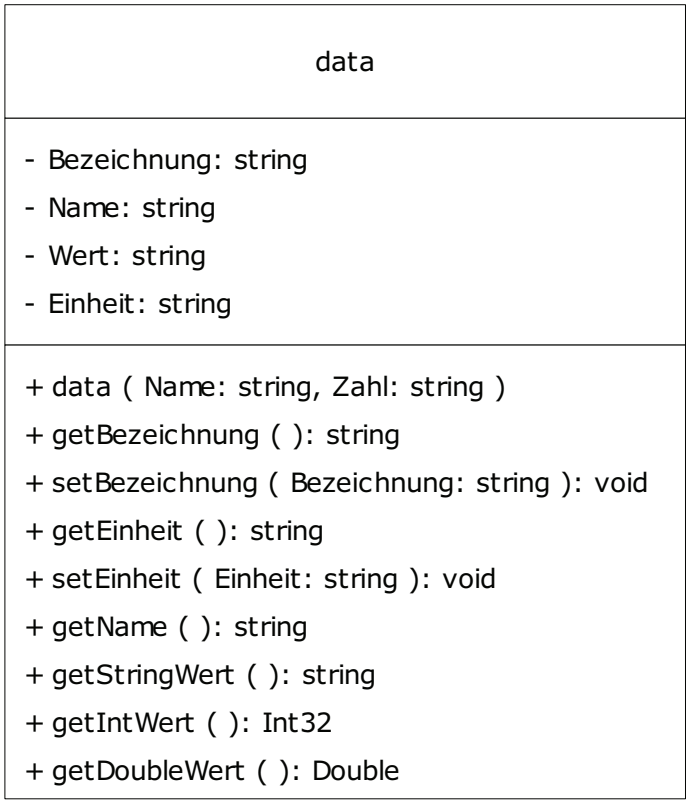


Abbildung 21: Klasse data

Es sollen auch Vektoren und Matrizen als Daten zur Verfügung stehen. Um dies zu realisieren, können die Klassen *dataVector* und *dataMatrix* eingebaut werden. Diese besitzen wie die Klasse *data* einen *Namen*, eine *Bezeichnung*, eine *Einheit* und einen *Wert*. Bei *dataVector* ist der *Wert* ein Vektor und bei *dataMatrix* ist dieser eine Matrix. Die Inhalte der Daten sind jeweils Texte. Näheres zu den beiden Klassen ist in Kapitel E Ergänzungen nachzulesen.

Um alle Werte sammeln zu können, bedarf es einer Liste. Diese Liste muss die neuen Klassen *data*, *dataVector* und *dataMatrix* verwalten können. Andere Datentypen müssen nicht berücksichtigt werden, denn das Hauptprogramm kennt die Daten innerhalb des Berechnungsmoduls nicht und geht prinzipiell von den vorgegebenen Klassen aus. Die Eingabewerte und Ergebnisse werden in internen Listen gespeichert. Die Klasse dataList ist solch eine Liste (siehe Abbildung 22).

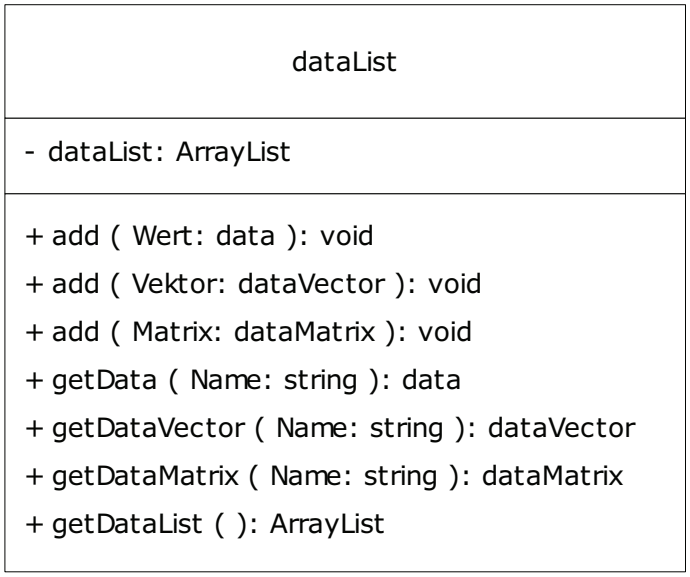


Abbildung 22: Klasse dataList

Die Klasse *dataList* speichert Objekte von *data*, *dataVector* und *dataMatrix*. Mit der Methode *add* können Werte übergeben werden, die in der Liste eingefügt werden. Ein gesuchtes Objekt wird über die *getData*-Methode ermittelt. Dabei ist der Name der Schlüssel, d.h. dieser muss eindeutig vergeben sein. Für die Unterscheidung zwischen Zahl, Vektor oder Matrix muss die entsprechende Methode aufgerufen werden. Die Suchmöglichkeiten beschränken sich nicht nur auf die vom Programm vorgegebene Suche, sondern es können innerhalb der Aufgabe eigene Suchalgorithmen umgesetzt werden. Dafür muss die interne Liste über die Methode *getDataList* abgefragt und verwendet werden.

3.2.2 Datenspeicherung

Das Programm soll relevante Daten wie z.B. Namen, Bezeichnungen und Werte der Aufgaben speichern können. Eine Berechnung kann Eingabedaten, Zwischenwerte und Ergebnisse umfassen. Diese können aus einer Liste von Zahlen, Vektoren und Matrizen bestehen. Die Speicherung kann auf verschiedene Varianten erfolgen.

Es ist denkbar, eine programmabhängige Binärdatei zu erzeugen, welche in Listen oder Bäumen strukturiert ist. Dazu wird ein programmeigenes Einlese- und Schreibtool benötigt. Von außen sind die Werte nicht lesbar und änderbar.

Jedes Modul erzeugt während der Berechnung temporäre Dateien, die nach der Verwendung gelöscht werden. Bei vielen Berechnungen wird der Speicher stark beansprucht.

Eine andere Idee umfasst die Speicherung in einer Textdatei (ASCII). Der Vorteil besteht darin, dass der Nutzer diese verstehen kann, ohne viel vom Programm wissen zu müssen. Eine Darstellung und eine weitere Verwendung der Daten ist sofort gegeben. Darüber hinaus ist eine Manipulation der Ergebnisse oder Eingaben möglich, die zur Unleserlichkeit oder zum Programmabsturz führen. Dieser Fall müsste bei der Umsetzung auf alle Fälle berücksichtigt werden.

Die Realisierung kann auch mit einer Datenbank erfolgen. Diese ist dann global und verwaltet alle Berechnungsmodule und dessen Ergebnisse. Die Daten sind vor unbefugten Zugriff geschützt. Damit ist eine Manipulation der Werte eingeschränkt. Der Nachteil einer Datenbank steckt in der Größe. Mit der begrenzten Speicherkapazität eines PDA können Probleme auftauchen. Allerdings gilt das nur für sehr viele komplizierte Anwendungen. Im Normalgebrauch stößt ein PDA nicht an diese Grenze. Zudem gibt es Lite-Versionen von Datenbanken, die eine geringe Größe besitzen. Nachteile sind in der Funktionalität gegeben, die für das zu entwickelnde Programm keine Rolle spielen.

Wenn eine Speicherung in einer Datenbank vorgesehen ist, besteht die Möglichkeit die Daten nicht nur aus der eigenen Aufgabe, sondern auch aus anderen Berechnungsmodulen zu nutzen. Das wird in Abbildung 23 dargestellt. Das Hauptprogramm läuft und führt das Berechnungsmodul 2 aus. Die Nutzereingabe kommuniziert mit dem Rahmenprogramm, welches sämtliche Aktivitäten steuert. Aus dem Speicher können Werte von anderen Berechnungsmodulen geholt werden, wie hier aus der Berechnung 1 stammen. Damit liegt eine Berechnungskette vor. Mehr dazu im Kapitel 3.2.4.

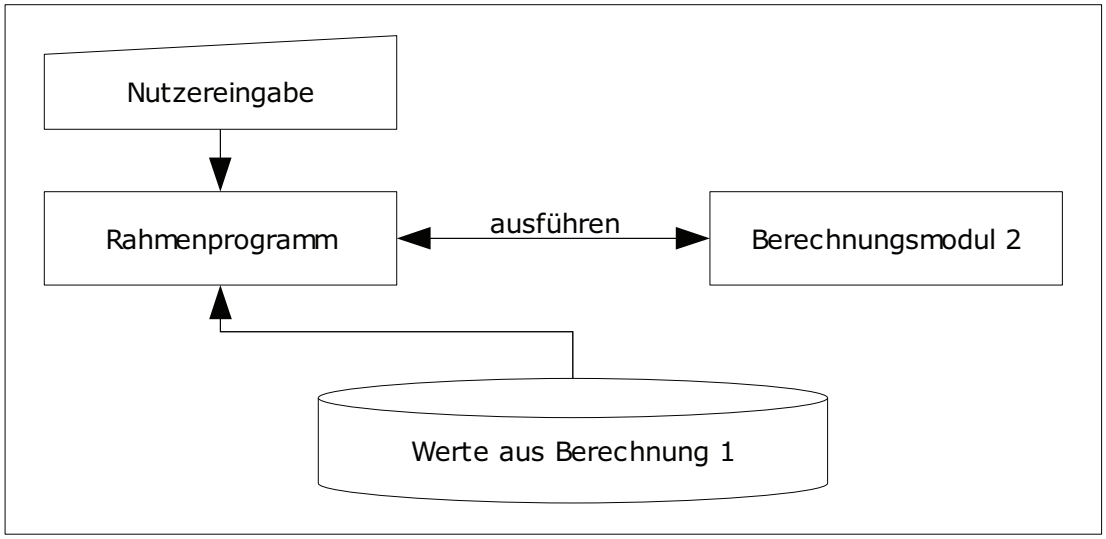


Abbildung 23: Schematische Datennutzung einer Berechnungskette

3.2.3 Datenbankaufbau

Für den weiteren Entwurf wird die Speicherung der Werte in einer Datenbank angenommen. In Abbildung 24 ist das ER-Modell für die Datenbank zu sehen. Es werden die Tabellen und dessen Beziehungen dargestellt.

Die Tabelle *Modul* beschreibt das Berechnungsmodul mit dem *Namen* und der Programmbibliothek (*DLL*[19]). Diese kann mehrere *Berechnungen* speichern. Diese bestehen aus einem *Namen* und einer *Beschreibung*. Dort kann der Nutzer eigene Informationen eintragen, um seine Aufgabe aus der Menge zu finden. Jede Berechnung kann Eingabedaten und Ergebnisse verwalten. Es können Daten der Klassen d*ata*, d*ataVector* und d*ataMatrix* gespeichert werden. Diese sind in Anlehnung zur d*ataList* entstanden und werden in dieser Form intern im Programm PDA-SB verwendet.

Data besteht lediglich aus einem Wert, welches in einer 1 zu 1 Beziehung zum *Datenelement* steht. Vektoren, die in der Tabelle *DataVector* beschrieben sind, besitzen eine Liste von Werten. Sichtbar ist das in der 1 zu n Beziehung zum *Datenelement*. Die Reihenfolge wird mit der Spalte *Position* in der Beziehungstabelle fest gehalten. Matrizen werden in der Tabelle *DataMatrix*

19 DLL ist eine Bibliothek von Programmdateien

gespeichert. Die einzelnen Elemente werden über die *Zeile* und die *Spalte* zu geordnet.

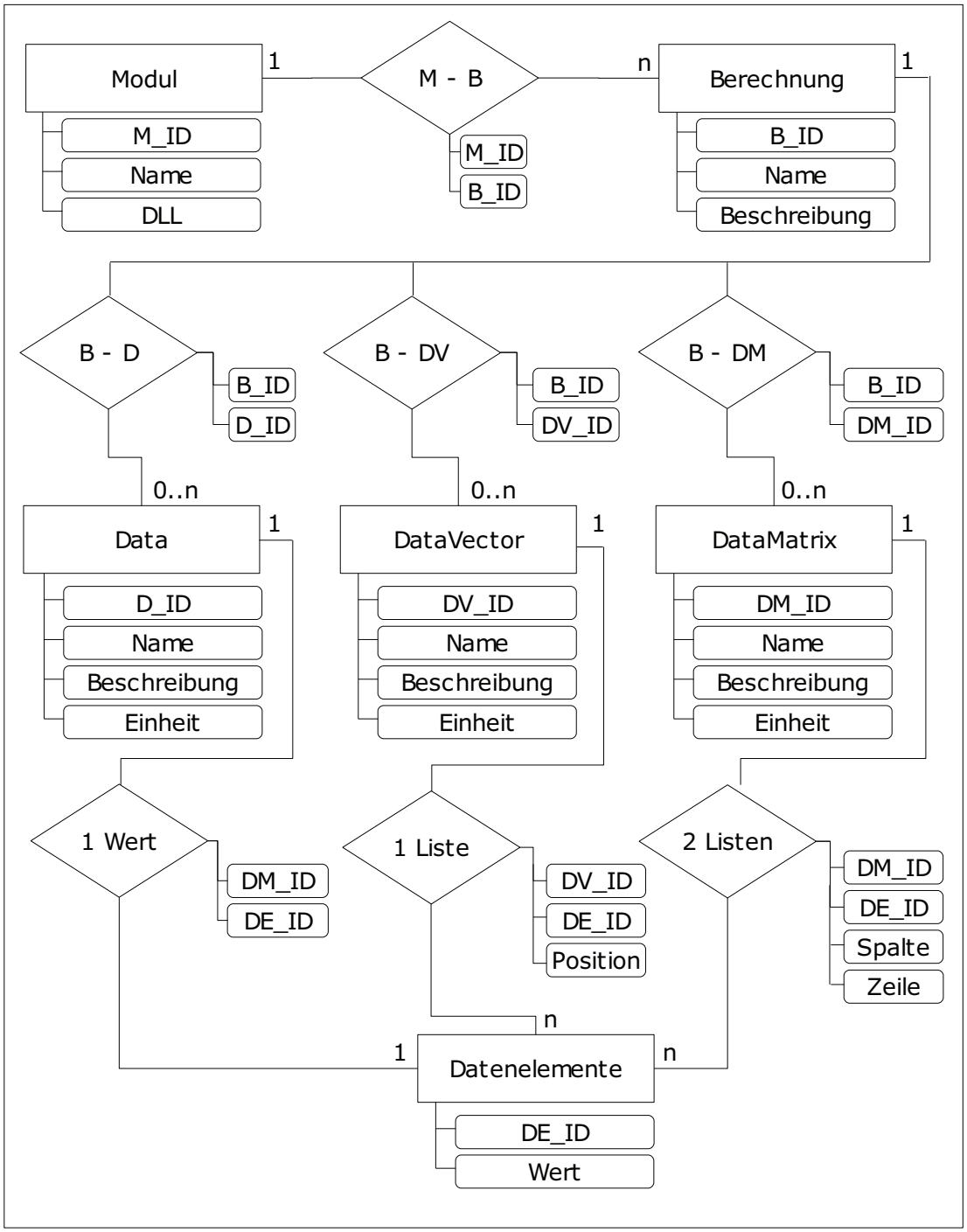


Abbildung 24: ER-Modell der Datenbank

Wenn die Zahlen, die Vektoren oder die Matrizen eine Einheit haben und diese umgerechnet werden soll, ist es sinnvoll das in einer separaten Tabelle zu

speichern. Es funktioniert, wenn der Umrechnungswert innerhalb der Einheiten bekannt ist. In der folgenden Abbildung 25 ist dazu ein ER-Modell abgebildet.

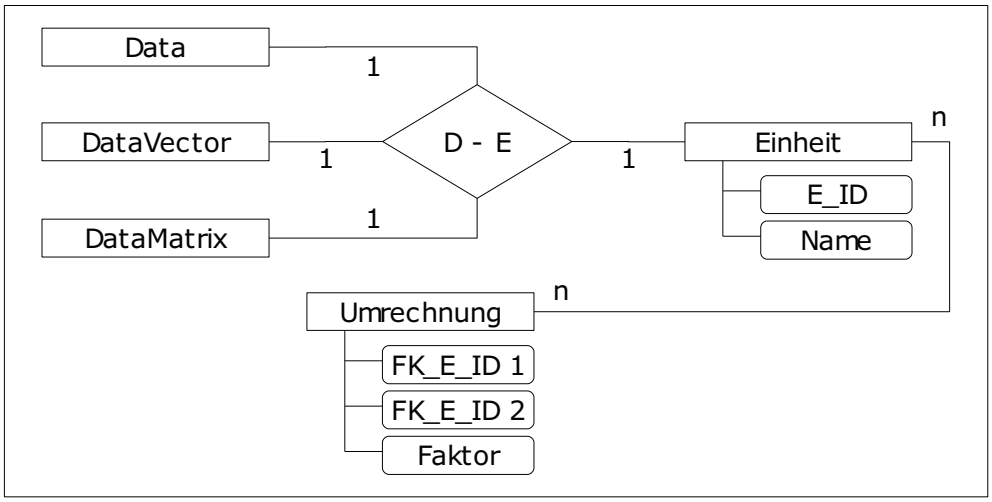


Abbildung 25: ER-Modell zu Daten, Einheit und Umrechnung

In Tabelle *Umrechnung* wird von der ersten Einheit zur Zweiten mit einem *Faktor* umgerechnet. Es sind die Fremdschlüssel zur Tabelle *Einheit* angegeben. Die erste Einheit ist mit *FK_E_ID 1* kenntlich gemacht. Analog ist die Zweite mit *FK_E_ID 2* beschrieben.

3.2.4 Berechnungskette

Die Berechnungskette ist eine Aneinanderreihung von Berechnungsmodulen. Es können beliebig viele hinter einander geschaltet werden. Dabei gibt der Benutzer zu jedem Modul Werte ein oder diese kommen von vorangegangenen Berechnungen. Eine schematische Darstellung solch einer Berechnungskette ist in Abbildung 26 zu sehen.

Der Nutzer gibt im Laufe der Berechnungskette vier Werte (na, nb, nc und nd) ein. Die Kette beginnt mit der ersten Berechnung Modul 1. Diese benötigt zwei Nutzereingabe (nc und nd). Mit Hilfe der Werte ermittelt die Aufgabe die Ergebnisse a, b, c und d. Das Modul 2 verwendet zwei Eingaben (a und c), die vom Modul 1 stammen. Die produzierten Daten x und y mit der Nutzereingabe nb sind die Startwerte für das Modul 3. Da Wiederholungen gleicher Aufgaben

möglich sind, endet die Berechnungskette mit dem Modul 1. Weil die gleichen Funktionen ausgeführt werden, werden zwei Werte benötigt. Diese stammen nicht nur vom Nutzer, sondern auch von der vorangegangenen Aufgabe.

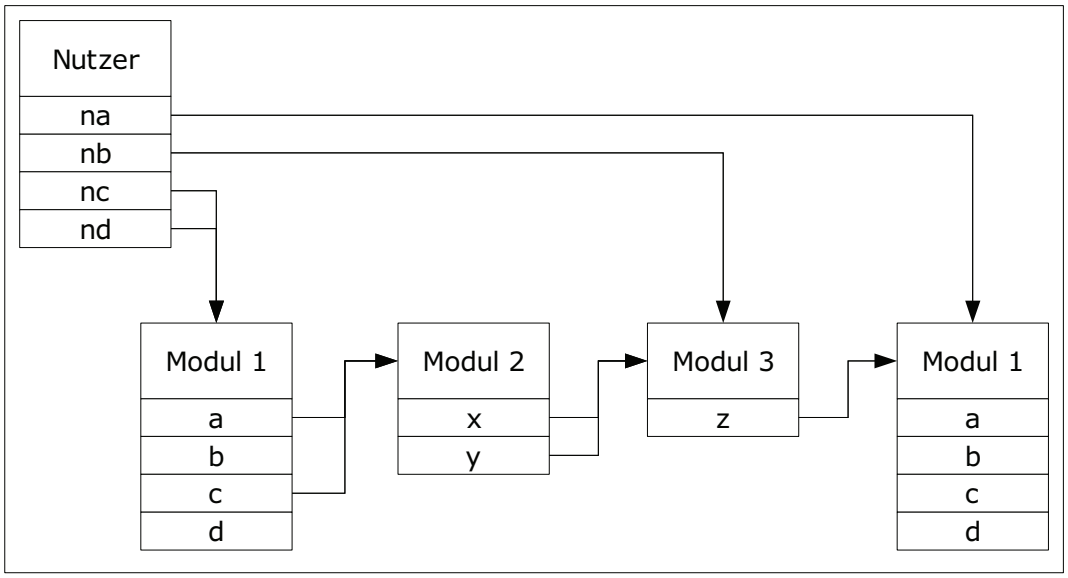


Abbildung 26: Berechnungskette

Mit einer Berechnungskette ist ein automatischer Ablauf möglich. Damit können Bauwerke über einen langen Zeitraum überwacht werden. Der entstehende Datenumfang bei einem Monitoring System[20] übersteigt die Auswertungsmöglichkeit eines PDA. Für dieses System ist ein schneller und leistungsfähiger Server praktischer, der das Ergebnis auch auf einem PDA anzeigen kann. Daher wird das Prinzip der automatischen Berechnungskette nur teilweise umgesetzt. Die folgende Abbildung 27 zeigt eine Berechnung mit gespeicherten Daten.

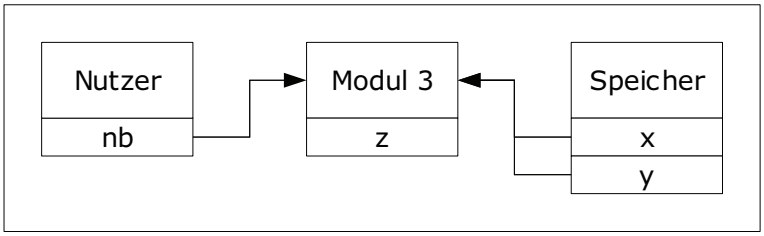


Abbildung 27: Berechnung mit gespeicherten Daten

Die Abbildung 27 zeigt ein Berechnungsmodul, das Daten aus dem Speicher nutzt. Das Schema basiert auf Abbildung 26. Das dortige Modul 3 ist hier das

20 Monitoring System = Überwachungssystem; hier für Dauerüberwachungen von Bauwerken

einzige Berechnungsmodul. Die Nutzereingabe von nb ist auf der linken Seite, während die Daten aus dem Speicher auf der Rechten zu sehen sind. Die gespeicherten Werte x und y können aus anderen Aufgaben, z.B. aus Modul 2, stammen. Damit ist eine kleine Berechnungskette entstanden.

3.3 Oberflächengestaltung

Die Oberfläche muss für den Nutzer ansprechend und einfach zu verstehen sein. Das Rahmenprogramm besitzt verschiedene Funktionen, die übersichtlich angeordnet werden müssen. Das ist in einem Menü möglich. Des Weiteren besitzt jedes Fenster einen Inhalt, z.B. einen Text für die Hilfefunktion. Wegen dem Menü und den Inhalt ergibt sich eine Zweiteilung des Fensters. In Abbildung 28 und 29 ist diese Unterteilung zu sehen.

Die Berechnungsmodule können variabel aufgebaut sein. Zum Entstehungszeitraum des Hauptprogramms ist nicht bekannt, welche Berechnungen erstellt werden und wie diese aussehen. Es ergibt sich dennoch eine Eigenschaft, die durch den PDA vorgegeben ist. Durch die begrenzte Bildschirmgröße können verschiedene Gestaltungsmöglichkeiten gewählt werden, um eine große Menge von Daten auf dem Monitor anzuzeigen. Es ist eine Scrollleiste möglich, wie die Abbildung 28 zeigt. Denkbar ist auch die Nutzung von Buttons, die durch die Menge von Daten führen. Dies ist in Abbildung 29 dargestellt.

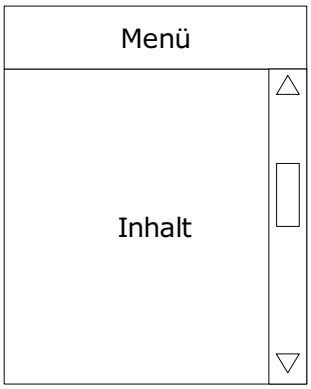


Abbildung 28: GUI 1

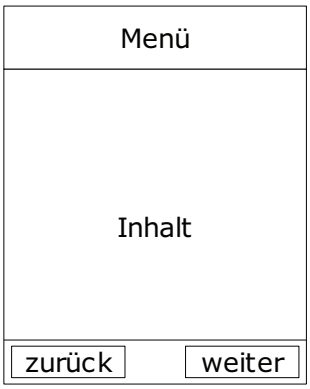


Abbildung 29: GUI 2

Bei einer großen Auswahl von Berechnungsmodulen ist eine themenbezogene Einordnung der Aufgaben sinnvoll. Die Einteilung nach Gebieten und die Gruppierung nach Anwendungsbereichen soll die Basis bilden. Beim Auswählen des Moduls für den Ladevorgang ist eine Ordnerstruktur, wie sie auf dem PC üblich ist, denkbar. Aussehen kann das wie in Abbildung 30.

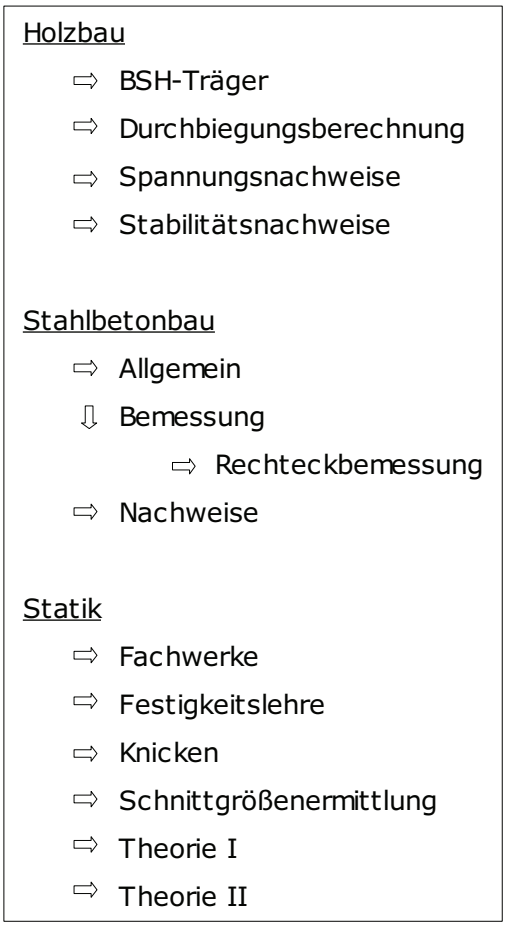


Abbildung 30: Beispielstruktur für die Gliederung von Berechnungen

Die Oberfläche des Berechnungsmoduls ist frei wählbar. Dennoch soll diese für den Benutzer ansprechend gestaltet werden.

Die Eingabemaske kann aus Textboxen oder Tabellen bestehen, indem die Zahlen, Vektoren und Matrizen eingetragen werden. Die Zuordnung der Werte kann in einer Skizze dargestellt werden. Das können beispielsweise die Namen wie Auflagerkräfte, Beanspruchungen und Geometrieangaben des statischen

Systems sein. Realisierbar sind Zeichnungen, die vom Nutzer angefertigt werden. Diese beschreiben zum Beispiel die Querschnittgeometrie eines Trägers.

Die Ausgabe und Präsentation der Ergebnisse kann in Textform geschehen. Des Weiteren können Kurven und Diagramme angezeigt werden. Um die Daten in vielen Varianten darzustellen, können verschiedene Layouts verwendet werden. Der Nutzer kann die Form der Ergebnisdarstellung selbst wählen.

Ein Beispiel des Layouts ist die Darstellung der Ergebnisse in der Form von Karteikarten. In der Abbildung 31 ist Gliederung in Text und Grafik zu sehen. Ein anderes Beispiel ist in Abbildung 32 verdeutlicht. Die Auswahl ist durch geordneten Links gegeben. Der Nutzer kann z.B. die Momentenlinie anklicken und erhält eine Grafik. Um das zusätzliche Menü platzsparend anzubringen, werden die Einträge innerhalb der Grafik und der Textausgaben erst sichtbar, wenn diese ausgewählt werden.

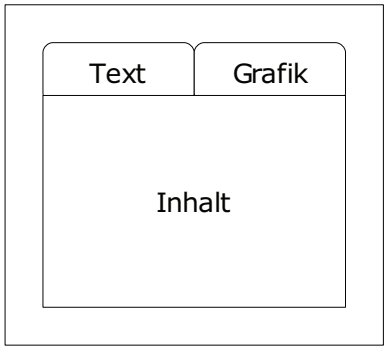


Abbildung 31: Layout 1 für Ausgabe

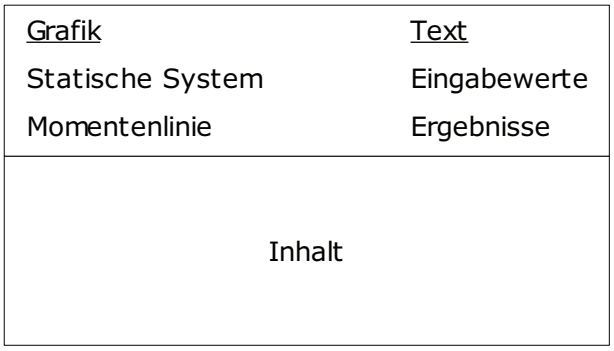


Abbildung 32: Layout 2 für die Ausgabe

Der Programmierer kann die Aufgaben und die Darstellungen selbst wählen und unterliegt keiner Grenze. Damit sind persönliche Freiheiten und Vorlieben umsetzbar.

4 Umsetzung

Die Umsetzung beschreibt den Aufbau des Prototypen. Angefangen wird mit der Beschreibung der Umgebung bezüglich der Programmieraufgabe. Dabei werden die Eigenschaften des zur Verfügung gestellten PDA, die Entwicklungsumgebung und die Programmiersprache beschrieben. Danach wird das Rahmenprogramm mit den Funktionen und den allgemeinen Abläufen erläutert. Im Anschluss folgen die Schnittstellendefinition, wobei System für das Hauptprogramm steht. Das Berechnungsmodul besitzt die Interfaces für die Eingabe, die Berechnung, die Ausgabe und die Anleitung. Zum Schluss ist die Aufgabe Rechteckbemessung umgesetzt. Damit wird der Aufbau und die Funktionsweise eines Moduls erläutert.

4.1 Umgebung für die Programmieraufgabe

Die Umgebung für die Programmieraufgabe ist die Grundlage für die Umsetzung. Angefangen wird mit der Beschreibung der Entwicklungsumgebung, welche zur Verfügung steht. Danach folgen nähere technische Informationen zum verwendeten PDA. Die Kommunikationssoftware zwischen dem PC und dem PDA wird genannt. Im Anschluss wird die Programmiersprache vorgestellt.

4.1.1 Entwicklungsumgebung

Die Entwicklungsumgebung von Microsoft Visual Studio 6 bietet verschiedene Pocket PC Programmiersprachen an. C-Sharp[21] und Java sind nur einige um sie zu nennen. Es wird eine schnelle und komfortable Variante der Softwareerstellung geboten. Anwendungen für ein Pocket PC können mit erheblicher Zeitersparnis generiert werden. Die Lizenzen sind zwar kostenpflichtig, dennoch besteht die Möglichkeit der Nutzung.

Als Alternative kann die Express Version von Microsoft Visual Studio oder SharpDevelop genutzt werden. Beide verursachen keine Lizenzkosten. SharpDevelop bietet beispielsweise eine attraktive grafische Benutzeroberfläche

21 Kurzform C#

mit einfacher Auswahl zwischen Applikation oder Programmbibliothek (DLL). Das kann in der angedachten Modulaufteilung sehr zum Vorteil sein, denn die Fenster können als Applikation gestalten werden. Bei fertiger Programmierung wird daraus lediglich eine DLL erzeugt. Das macht den Umgang für Einsteiger leichter.

Letztendlich können alle aufgezählten Programme verwendet werden, so dass es jedem Programmierer selbst überlassen ist, welches sein Favorit ist. Bei der Umsetzung des Prototypen wurde zu Beginn Microsoft Visual Studio 6 und zum Ende SharpDevelop verwendet.

4.1.2 Hardware und Software

Für die praktische Umsetzung steht ein Acer n50 Handheld mit den Eigenschaften, die in Tabelle 4 gezeigt werden, zur Verfügung.

Modell	Intel ® PXA272
Geschwindigkeit	312 MHz
Betriebssystem	Windows Mobile ™ 2003 Zweite Ausgabe
OS-Version	4.21
Arbeitsspeicher	60.29 MB

Tabelle 4: Geräteinformationen des verwendeten PDA

Wie aus der Tabelle 4 ersichtlich ist, ist das Gerät im Vergleich mit anderen Produkten (siehe Tabelle 1, Seite 21) im mittleren Segment einzuordnen. Die Leistung ist für das Programm PDA-SB ausreichend. Falls in den Tests höhere Anforderungen gestellt werden, wird ein anderes Gerät benötigt. Die Grenze ist zu notieren.

Zum Transport der Daten vom PC zum PDA und umgekehrt steht die Software ActiveSync zur Verfügung. Die Kabelverbindung läuft über die USB-Schnittstelle. Bei Bedarf stehen andere Programme zum Datenaustausch in Internet kostenlos bereit.

4.1.3 Programmiersprache

Mit C#[22] kann nah an der Hardware programmiert werden. Wegen der Objektorientierung kann eine Auslagerung der einzelnen Klassen erfolgen. Ableitungen und Vererbungen können im Nachgang zum Vorteil sein.

Java, als eine weitere Programmiersprache, kommt in Frage. Es fallen keine Lizenzgebühren an. Die objektorientierte Sprache ist wegen ihrer Portabilität praktisch. Ein Nachteil kann sich in der Geschwindigkeit während der Laufzeit des Programms ergeben. Das gilt allerdings nur für sehr komplexe Aufgaben. Diese benötigen einen schnellen Festplattenzugriff, der mit Java wegen der Fehlerabfrage eingeschränkt ist. Dennoch ist diese Sprache wegen der Unabhängigkeit interessant.

Die Programmiersprache für das Hauptprogramm und die Berechnungsmodule kann frei gewählt werden. Dabei müssen diese nicht unbedingt in derselben Sprache geschrieben sein, denn das Hauptprogramm soll unabhängig von der Programmiersprache arbeiten. Das funktioniert theoretisch über die Ansteuerung der Methoden innerhalb der Module über die Assembly-Informationen der DLL. In der Praxis muss das überprüft werden. Als Alternative zur Unabhängigkeit der Programmiersprache kann es ein Programm zur Erstellung von Berechnungsmodulen geben. Dieses kann dem Entwickler eine grafische Oberfläche zum Erstellen anbieten. Die Funktionen müssen einfach aufgebaut sein, wie z.B. eine Drag & Drop Variante. Weitere Möglichkeiten und Ausblicke befinden sich im Kapitel 6 Zusammenfassung.

Für das Projekt fällt die Entscheidung auf C#, um eine möglichst schnelle Anwendung zu generieren. Letztendlich wird die Geschwindigkeit allerdings von den Modulen getragen, in der die Berechnungen eingelagert sind. Daher sollte C# auch in diesem Bereich eingesetzt werden.

22 C# ist das Kurzzeichen für CSharp, eine Programmiersprache

4.2 Hauptprogramm

Das Hauptprogramm besteht aus einem Fenster mit einem Menü. Zu Beginn wird eine Kurzhilfe mit der Bedienungsanleitung angezeigt. Damit kann ein neuer Benutzer mit den Funktionen des Programms arbeiten. Sämtliche Bedienmöglichkeiten stehen im Menü zur Verfügung. Der Prototyp benutzt selbsterklärende Begriffe aus dem Englischen, wobei eine Sprachunabhängigkeit nachträglich eingebaut werden kann.

Das Rahmenprogramm unterteilt sich in Anlehnung zum Menü in folgende Gliederung:

- Rahmenprogramm (System)
  Das System beinhaltet alle Steuerelemente, die sich auf das Hauptprogramm beziehen. Vom Berechnungsmodul laden über Modul schließen bis zum Beenden des Programms sind alle diese Elemente im Menüpunkt *System* zu finden.
- Berechnung (Calculation)
  Im Menüpunkt *Calculation* (Berechnung) ist die Eingabemaske, das Ausgabefenster, die Berechnung starten und eine Löschung der Wertelisten zu finden. Zu einer Aufgabe bzw. zum Berechnungsmodul gehört eine Hilfeinformation, die sich allerdings in dem Menüpunkt *Help* befindet.
- Hilfe (Help)
  In der Hilfe befindet sich eine allgemeine Anleitung zur Bedienung des Programms, damit jeder Nutzer jederzeit nachlesen kann, was wie erreicht wird. Des weiteren ist die Anleitung oder Hilfestellung zum geladenen Berechnungsmodul vorhanden. Information zu der aktuellen Version, dem Ersteller und eventuelle Anpassungen infolge eines Updates sind unter diesem Menüpunkt zu finden.
- Daten (Data)
  Der Menüpunkt *Data* umfasst die Koordination und das Handhaben von Daten und Werten, die während einer Berechnung erzeugt oder benötigt werden. Hier können die Daten gespeichert oder geladen werden. Ebenso können Werte im- oder exportiert werden. In diesem Bereich ist der Prototyp noch nicht vollständig umgesetzt.

Das Menü umfasst die Funktionen des Rahmenprogramms. Diese werden im Anschluss näher erläutert.

4.2.1 Erklärung der Menüfunktionen

Im Folgenden werden die Funktionen des Menüs von PDA-SB beschrieben. Die Umsetzung umfasst die Aufgaben des Hauptprogramms. Diese befinden sich im Menüpunkt *system*. Des Weiteren gibt es den Punkt *calculation*, welcher die Funktionen bezüglich des Berechnungsmoduls erläutert. In *help* sind die verschiedenen Hilfestellungen und Anleitungen zu finden. Nicht umgesetzt ist das Menü für die Datenspeicherung, den Im- und den Export.

Hauptaufgabe – system

Die Aufgaben des Hauptprogramms sind das Berechnungsmodul zu laden, zu schließen und das Programm zu beenden. Die Umsetzung der Funktionen wird im Folgenden beschrieben.

Berechnungsmodul laden (Load)

Die Ladefunktion basiert auf Reflection, da die Klassen und Objekte zum Erstellungszeitraum des Hauptprogramms noch nicht existieren und später an das Programm gebunden werden. Damit ist die nachträgliche Erweiterung von Berechnungsaufgaben gewährleistet. Aufgrund der Schnittstellendefinition des Systems, der Eingabe, der Ausgabe, der Berechnung und der Hilfe ist die Zuordnung möglich und die Objekte können anhand des Namens initiiert werden. Dazu wird eine Klasse mit den Eigenschaften des Interfaces gesucht und auf einem freigehaltenen Objekt geladen. Damit stehen die Funktionen und Methoden der Klasse zur Verfügung, welche für das Rahmenprogramm benötigt werden. Das Hauptprogramm kann damit die Steuerung der einzelnen Klassen und die Anzeige der Masken regeln. Mehr zum Thema von Reflection bietet das Buch C# in a nutshell. [C# 03]

Berechnungsmodul schließen (Close)

Das Schließen des Berechnungsmoduls ist einfach aufgebaut. Die Schnittstellenobjekte werden gelöscht. Ebenso werden die Werte, die innerhalb der vorangegangenen Berechnung entstanden sind, gelöscht.

Programm beenden (Exit)

Das Beenden des Programms ist notwendig, wenn das Programm wirklich geschlossen werden soll. Anwendungen auf einem PDA werden nicht über das Kreuz oben rechts, wie beim PC gewöhnlicher Weise, beendet, sondern in den Hintergrund geschoben. Durch die Philosophie, dass ein PDA in einen Ruhezustand geschaltet wird, wird das Prinzip genauso für Programme genutzt. Um diesen Vorgang zu Umgehen und das Programm nicht in den Hintergrund zu legen, sondern zu schließen, muss das Programm separat abgeschaltet werden. Das passiert nach dem Klicken auf den Menüeintrag *Exit*, der sich unter dem Menüpunkt *System* befindet. Nach dem Schließen gehen sämtliche nicht gespeicherte Werte verloren. Beim erneuten Öffnen des Programms wird der Ausgangszustand hergestellt.

Berechnung – calculation

Der Menüpunkt *calculation* beschreibt die Funktionen der Berechnung. Dabei kann zwischen der Eingabemaske und dem Ausgabefenster gewählt werden. Die Berechnung der Aufgabe kann mit dem Punkt *calculate* ausgeführt werden. Es können die Eingabewerte und Ergebnisse gelöscht werden. Weitere Beschreibungen sind im Anschluss zu finden.

Eingabe (Input)

Die Eingabemaske der Berechnungsaufgabe wird standardmäßig nach dem Laden des Moduls angezeigt. Wenn das Fenster nicht automatisch angezeigt wird, kann der Nutzer dies manuell über *Input* aufrufen. Ebenfalls können zu einem späteren Zeitpunkt die Eingabedaten eingesehen werden, falls diese nicht explizit in der Ausgabe angezeigt werden.

Berechnen (Calculate)

Calculate ist die Ausführung der Aufgabe, sprich die Berechnung. In den gewöhnlichen Anwendungsgebieten benötigt dieser Aufruf keine Anzeige, sondern soll die Aufgabe lösen. Dennoch kann der Programmierer ein Fenster gestalten. Das ist in dem Fall sinnvoll, wenn es sich um komplizierte und aufwendige Aufgaben handelt. Die Berechnung nimmt eine längere Zeit in Anspruch. Damit der Nutzer den aktuellen Stand kennt und weiß, dass das Programm noch rechnet und nicht abgestürzt ist, ist es angebracht einen Fortschrittsbalken anzuzeigen.

Ausgabe (Output)

Das Ausgabefenster ist die Anzeige der Ergebnisse. Eine günstige Gegenüberstellung mit den Eingabewerten erspart dem Nutzer unnötige Klicke. Nach der Datenanzeige können die Werte gespeichert oder exportiert werden. Auf einem PC sind die Daten mit anderen Programm sichtbar und druckbar. Diese Funktionen stehen erst mit der Umsetzung der Datenbank zur Verfügung.

Werte löschen (Clear)

Hier werden die Eingabewerte und Ergebnisse gelöscht. Damit ist eine Anwendung häufiger nutzbar, ohne jeden einzelnen Wert separat löschen zu müssen. Der Ausgangszustand des Berechnungsmoduls wird erreicht.

Hilfe – help

In diesem Menüpunkt befinden sich die Anleitungen zur Benutzung des Programms und zur Verwendung der Berechnungsaufgabe. Des Weiteren ist eine Information zum Hauptprogramm gegeben. Die folgende Unterteilung erläutert die Funktionen der Hilfe.

Allgemeine Anleitung (Common)

Die allgemeine Anleitung zum Programm ist ein Teil des Handbuches. Diese steht dem Nutzer jederzeit elektronisch zur Verfügung. Im Anhang befindet sich das ausführliche Handbuch.

Berechnungsanleitung oder Hilfe (Calculation)

Beim Laden eines Berechnungsmoduls wird die Hilfe der Aufgabe geladen, sofern diese vorhanden ist. Der Bereich ist für Erläuterungen und Hintergrundinformationen gedacht. Bilder können eingearbeitet werden. Für viele verschiedene Varianten der Hilfestellung ist hier ebenfalls eine Programmierung notwendig.

Informationen zum Programm (Info)

Als Informationen stehen die Version, der Autor und das Datum zur Verfügung. Das sind die Standardeinstellungen, die bei einem Update erweitert werden können. Änderungen oder Neuerungen werden eingetragen.

4.2.2 Allgemeine Routinen

Die allgemeinen Routinen beschreiben automatisierte Schritte, die das Programm beim Aufruf bestimmter Funktionen ausführt.

Nachdem das Berechnungsmodul geladen wurde, wird die Eingabemaske angezeigt. Wenn der Nutzer die Berechnung startet, werden zuerst die Eingabewerte geprüft. Falls diese korrekt sind, wird die Aufgabe ausgeführt. Ansonsten wird das Eingabefenster erneut angezeigt. Das Ausgabefenster wird nach fertiger Berechnung angezeigt. Von dort aus kann der Nutzer wieder die Eingabemaske anzeigen lassen, um Daten zu korrigieren oder zu ändern. Während das Eingabe- oder Ausgabefenster angezeigt wird, kann der Nutzer das Modul schließen. Solange die Berechnung ausgeführt wird, sind keine anderen Aktivitäten innerhalb des Programms möglich. Anschaulich wird das Schema im Zustandsdiagramm in Abbildung 33 präsentiert.

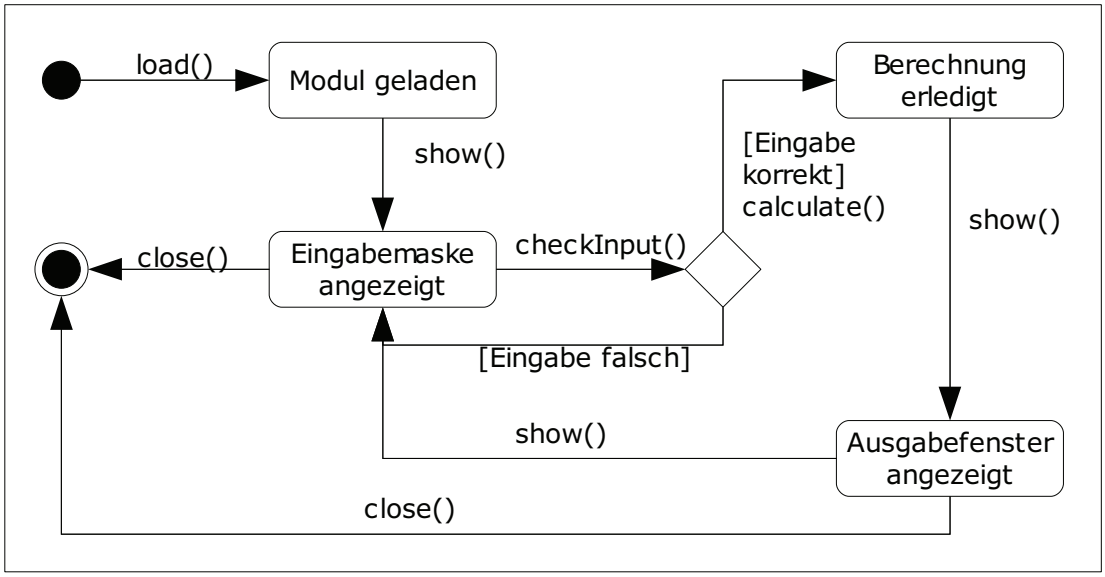


Abbildung 33: Zustandsdiagramm

4.3 Schnittstellen

Im Kapitel Schnittstellen werden die fünf erstellten Interfaces vorgestellt. Eines beschreibt das Hauptprogramm und wird näher im Abschnitt System erläutert. Die weiteren Vier beschreiben die Schnittstellen für das Berechnungsmodul. Das sind die Eingabe, die Berechnung, die Ausgabe und die Hilfe. Diese Interfaces sind umgesetzt und befinden sich in einer Programmbibliothek namens *PALM_SB.dll*.

4.3.1 System

Die Schnittstelle *system* ist für die Klassen des Berechnungsmoduls wichtig. Das Interface beinhaltet die Herausgabe der vier Objekte, in welche das Berechnungsmodul geteilt ist (Eingabe, Berechnung, Ausgabe und Hilfe). Diese sind für die interne Kommunikation notwendig. Die Schnittstelle ist in Abbildung 19 auf Seite 38 zu sehen. Die Methoden mit dem Namensbeginn *get* liefern die entsprechenden Objekte der Klassen. Das ist beispielsweise *getInput*, die das Objekt *input_calculation* liefert, welches die Eingabemaske der Aufgabe beschreibt. Implementiert wird das Interface *system* vom Rahmenprogramm und stellt damit die geforderten Objekte zur freien Verfügung.

4.3.2 Eingabe

Die Schnittstelle für die Eingabeklasse ist in Abbildung 35 zu sehen. Dabei ist *dataList* die interne Struktur für die Daten. Genaueres wurde im Kapitel 3.2.1 Datenstruktur beschrieben. Die boolsche Variable *visible* ist von außen lesbar und änderbar. Diese beschreibt die Sichtbarkeit der Eingabemaske. Es folgen zwei Funktionen, die zum Einen die Eingabedaten prüft und zum Anderen die Eingabeliste liefert. Die nächste Funktion übergibt bereits vorhandene Eingabewerte an die Eingabe. Das ist der Fall, wenn gespeicherte Daten oder Daten aus Berechnungsketten gegeben sind. Fenstereinstellungen sind möglich, die sich auf das Anzeigen (*show*) und das Schließen (*close*) beziehen. Zur Verwendung der anderen internen Berechnungsteile des Moduls, wie z.B. die Hilfe anzeigen oder eine Berechnung ausführen, wird die Systemklasse lokal gespeichert. Damit kann jederzeit die Systemschnittstelle angesteuert und das entsprechende Objekt abgefragt werden. Zuletzt beschreibt die Methode *clear*, das Leeren von Fensterelementen wie z.B. eine Textbox.

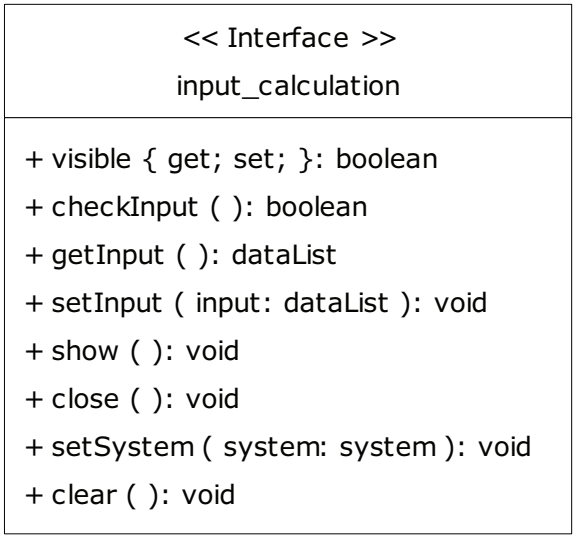


Abbildung 34: Interface input_calculation

4.3.3 Berechnung

In der Berechnungsschnittstelle gibt es zwei Listen des Typs *dataList*. Diese verwalten die Eingabewerte *input* und die Ergebnisse *result*. Die Sichtbarkeitsvariable *visible*, die den Zustand des Fensters beschreibt, kann verändert werden. Die Hauptmethoden sind *run* und *getResult*. Die Erste startet die Berechnung und die Zweite liefert die Ergebnisse. *Show* und *close* sind

wieder die Fenstersteuerungsfunktionen. Auch hier ist es vorgesehen, die Systemklasse zu speichern um z.B. direkt die Ausgabe aufzurufen. Einige selbstverständliche Aufgaben übernimmt allerdings bereits das Rahmenprogramm, so dass sich der Programmierer darum nicht kümmern muss. Näheres wird in Kapitel 4.2.2 Allgemeine Routinen erklärt. Zuletzt ist die *clear* Methode angeordnet, die die Eingabedaten und Ergebnisse löscht. Damit ist gewährleistet, dass beim erneuten Aufrufen der Berechnung die richtigen Ergebnisse vorhanden sind, nämlich die neu Ermittelten. Falls ein Fehler in der Aufgabe keine neuen Ergebnisse liefert, sind ohne den Löschvorgang alte Werte vorhanden. Die Ausgabe würde dann falsche Daten anzeigen. In Abbildung 35 ist die Schnittstelle der Berechnung zu sehen.

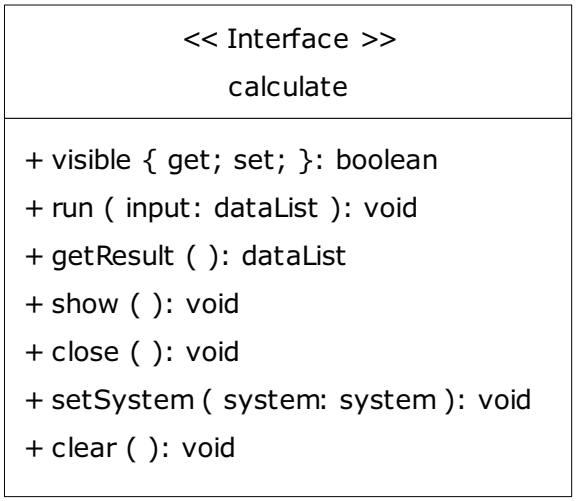


Abbildung 35: Interface calculate

4.3.4 Ausgabe

Die Ausgabe soll die Eingabedaten und die Ergebnisliste anzeigen, dass durch das Interface in Abbildung 36 ermöglicht wird. Die Standardeinstellungen wie *visible*, *show*, *close, clear* und *setSystem* sind vorhanden, wie in den anderen Schnittstellendefinitionen der Berechnungsmodule. Es gibt drei weitere Funktionen, die für den Datenaustausch zuständig sind. Es geht dabei um das Setzen der Eingabe- (*setInput*) und der Ergebnisdaten (*setResult*). Der Standardaufruf ist *fillOutput*, bei dem beide Listen übergeben werden.

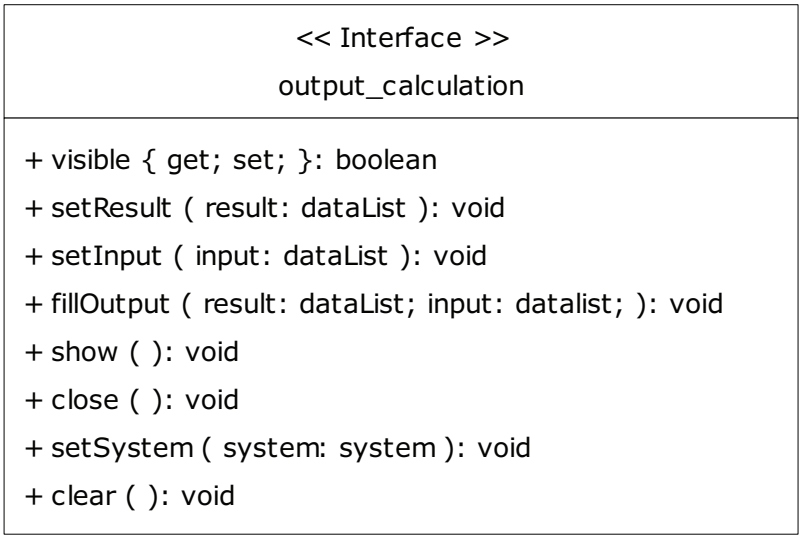


Abbildung 36: Interface output_calculation

4.3.5 Hilfe

Die Hilfeschnittstelle beinhaltet nur die Grundeinstellungen, wie *visible*, *show*, *close* und *setSystem*. Im Anschluss befindet sich in Abbildung 37 das Interface für die Hilfe.

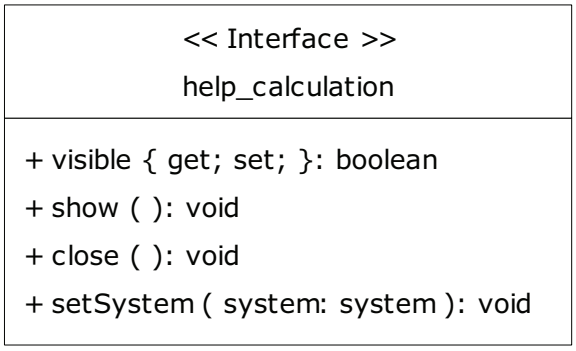


Abbildung 37: Interface help_calculation

4.4 Berechnungsmodul

Die Struktur des Berechnungsmoduls und dessen Funktionen werden anhand der in der Analysephase beschriebenen Beispielberechnung erläutert. Dabei handelt es sich um eine Rechteckbemessung mit einfacher Bewehrung. Die Rechengrundlage und Formeln sind bereits in Kapitel 2.1.4 Rechteckbemessung erklärt worden.

Zu Beginn wurde ein Standard – Muster oder Wizard erstellt. Dieses umfasst die Schnittstellen für das Berechnungsmodul und erzeugt ein Fenster zur Gestaltung der Eingabeoberfläche, der Ausgabe und der Hilfe. Die Berechnung der Aufgabe erfolgt im Hintergrund ohne ein bedienbares Fenster.

4.4.1 Hilfe

Das Berechnungsmodul kann eine Anleitung oder Hilfe besitzen. In dieser kann der Programmierer wichtige Hinweise für die Aufgabe, Erklärungen für Namensgebungen oder Hintergrundinformationen zu den Rechenschritten geben. Im Beispiel in Abbildung 38 wurde diese Anleitung kurz und knapp gehalten und weist den Nutzer hin, den Anweisungen auf dem Bildschirm zu folgen.

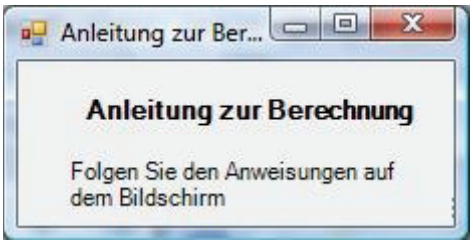


Abbildung 38: Hilfefenster

Die Anleitung kann sehr komplex gestaltet werden. Dabei können Bilder oder Videos genutzt werden. Das Fenster kann über das Menü geöffnet werden. Der Programmierer hat die Möglichkeit bei Eingabefelder einen Link zur Hilfe zu setzen. Dazu können Sprungmarken in der Datenliste mitgeschickt werden, die dann innerhalb der Hilfe eine bestimmte Funkion haben, z.B. zu einem Punkt springen oder eine Formelerläuterung anzeigen. Es gibt in der Umsetzung kaum Grenzen, so dass viele Varianten möglich sind.

4.4.2 Eingabeoberfläche

Die Eingabe des Nutzers erfolgt in einfachster Weise über Textboxen. Zur Erleichterung der Benutzung wurden die Betonklassen in einer externen Textdatei festgehalten. Diese Datei steht ebenfalls anderen Berechnungsmodulen zur Verfügung. Der Nutzer ist damit an die dortigen Namen gebunden und kann somit nichts falsch eingeben.

Da die Formeln für die Berechnung mit Normalbeton und einer Festigkeitsklasse von C ≤ 50/60 ausgelegt sind, umfasst die Textdatei nur die erforderlichen Bezeichnungen. Die Umrechnung des Betons in den Bemessungswert der Betondruckfestigkeit f_{cd} erfolgt automatisch. Des Weiteren wird eine Annahme getroffen, dass es sich um den verwendeten Stahl von BSt 500 S handelt. Daraus resultiert der Bemessungswert für die Stahlstreckgrenze f_{yd} von 43,48 kN/mm², welcher als Text angezeigt wird. Bei der Verwendung eines anderen Betonstahls muss eine Änderung im Quellcode erfolgen. Die Eintragungen der Beanspruchung M_{sd} muss in kNm erfolgen. Die Querschnittsgeometrie des Rechtecks mit der Breite b, der Höhe h und der Nutzhöhe d werden in cm eingetragen. Das Eingabefenster der Rechteckbemessung ist in Abbildung 39 zu sehen.

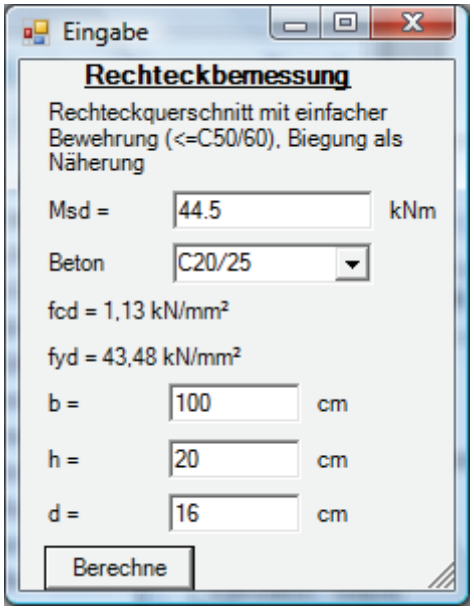


Abbildung 39: Eingabemaske Rechteckbemessung

4.4.3 Berechnung

Die Berechnung der Aufgabe ist relativ kurz, denn es werden nicht viele Formeln verwendet. Benötigt werden Multiplikation, Division und Wurzel ziehen, die als grundlegende mathematische Funktionen im Basispaket der Programmierung in C# vorhanden sind. In Abbildung 40 ist die Berechnung der Rechteckbemessung abgebildet.

```
public void run(dataList Werte) {
    //Eingabewerte auslesen
    double Msd = Werte.getData( "Msd" ).getDoubleWert();
    double fyd = Werte.getData( "fyd" ).getDoubleWert();
    double fcd = Werte.getData( "fcd" ).getDoubleWert();
    double b = Werte.getData( "b" ).getDoubleWert();
    double h = Werte.getData( "h" ).getDoubleWert();
    double d = Werte.getData( "d" ).getDoubleWert();

    //Berechnung ausführen
    double mcd = ( Msd * 100 ) / ( 0.95 * fcd * b * d * d );
    double ci = 1.25 * ( 1 – Math.Sqrt ( 1 – 2 * mcd ) );
    double vcd = 0.8 * ci;
    double As = 0; string Ergebnis;
    if ( ci <= 0.45 ) {
        As = ( 0.95 * vcd * fcd * b * d ) / fyd;
        Ergebnis = "Bemessung fertig";
    } else Ergebnis = "Querschnitt vergrößern oder doppelte Bewehrung";

    //Ergebnisse in Liste speichern
    ergebnis = new dataList();
    ergebnis.add(new data("mcd",mcd.ToString()));
    ergebnis.add(new data("ci",ci.ToString()));
    ergebnis.add(new data("vcd",vcd.ToString()));
    ergebnis.add(new data("Ergebnis",Ergebnis.ToString()));
    ergebnis.add(new data("As",As.ToString()));
    ergebnis.add(new data("Test",""+Msd*10));
}
```

Abbildung 40: Quellcode einer Rechteckbemessung

Wie in Abbildung 40 zu sehen ist, werden zu Beginn die Eingabewerte lokal gespeichert. Im Anschluss wird die Berechnung ausgeführt. Zum Ende werden die Ergebnisse in einer Liste gespeichert. Die Daten stehen innerhalb der Klasse zur Verfügung und können über die Schnittstelle *getResult()* abgefragt werden.

Die Berechnung erfolgt nach den Formeln aus Kapitel 2.1.4 Rechteckbemessung. Die Fläche des Betonstahls A_s wird als Ergebnis ermittelt. Wenn keine Lösung berechnet werden kann, weil die Eingaben nicht innerhalb der Grenzen liegen, wird das dem Benutzer mitgeteilt.

4.4.4 Ausgabefenster

Das Ausgabefenster ist bescheiden aufgebaut. Es besteht aus zwei Textboxen, die nicht veränderbar sind. Die Erste zeigt einen Text an, indem die Zwischen- und Endergebnisse in Worten stehen. Die zweite Textbox zeigt das Ergebnis in Zahlen, wobei die Fläche des Betonstahls As in cm²/m angegeben ist. Ein Beispielfenster der Ausgabe ist in Abbildung 41 dargestellt.

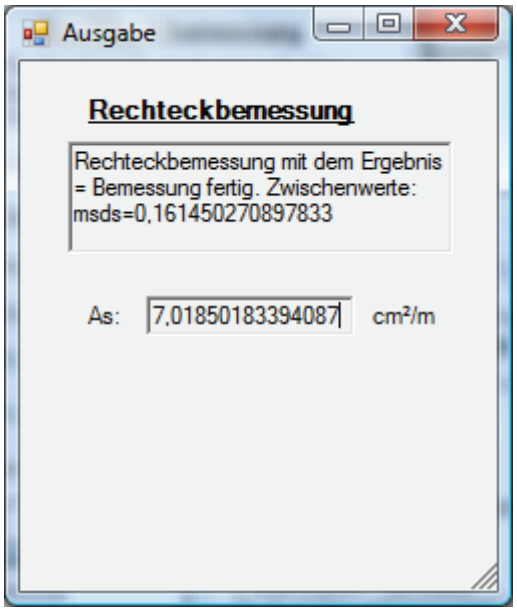


Abbildung 41: Ausgabemaske Rechteckbemessung

5 Test, Einsatz und Leistungsbewertung

5.1 Prüfung des Rahmenprogramms

5.1.1 Teststrategie

Die Hauptaufgabe des Programms PDA-SB ist die Ausführung von externen modularen Berechnungen für den konstruktiven Ingenieurbau. Daher soll der Ladevorgang dieser Aufgaben funktionieren. Die richtige Abarbeitung der Aufgabenschritte soll eingehalten werden. Es wird an dieser Stelle nicht das Berechnungsmodul in seiner Richtigkeit geprüft, sondern lediglich, ob die Ansteuerung der Schnittstellen korrekt funktioniert und an welchen Stellen Probleme oder Fehler auftreten. Es folgt die Ausführung und die Auswertung der gestellten Tests.

5.1.2 Ausführung und Auswertung

1. Das Programm PDA-SB wird gestartet.

→ Programm startet auf verschiedenen PC korrekt, wenn die Voraussetzungen, wie .Net Framework (2.0 oder höher), gegeben sind.

2. Alle Menüpunkte werden angewählt, um die Reaktionen darauf zu testen. Dabei ist noch kein Modul geladen.

→ Bei geschlossenem Modul werden modulbezogene Menüpunkte einfach ignoriert, so dass keine Fehlermeldung entsteht. Die restlichen Aufrufe erfolgen fehlerfrei.

3. Die Ladefunktion eines Berechnungsmoduls wird überprüft.

→ Das Laden funktioniert einwandfrei, wenn dieselbe PALM_SB.dll verwendet wird und die Module in dem entsprechenden Ordner untergebracht sind. Bei den mitgelieferten Aufgaben (Rechteckbemessung und Testberechnung) traten keine Probleme auf.

4. Nach dem korrekten Laden werden wieder alle Menüpunkte durcheinander gewählt, um mögliche Fehler aufzuspüren.

→ Die Menüpunkte sind normal verwendbar, solange der Quellcode der Berechnung korrekt umgesetzt worden ist. Bei den Beispielen sind keine Fehler aufgetreten.

5. Nach dem Schließen des Moduls soll dieses wirklich geschlossen sein.

→ Beim Schließen des Berechnungsmoduls wird das Modul „weggeschmissen" und als leer definiert. Ein erneutes Schließen wird ignoriert, so dass es keine Fehler zur Laufzeit entstehen. Im Test wurden nach dem Leeren der Objekte keine Fenster der Aufgabe geöffnet. Das lässt vermuten, dass das Modul erfolgreich geschlossen wurde.

6. Was passiert, wenn ein Modul geladen ist und ein anderes erneut geladen wird ohne das Vorherige zu schließen?

→ Wenn bei einem geöffneten Modul ein Neues geöffnet wird, werden als Erstes die Alten geschlossen und dann die neuen Objekte auf den vordefinierten Container gelegt. Das wurde als Schutzfunktion des Hauptprogramms eingebaut. Es wird nach dem Laden sofort die Eingabemaske angezeigt. Der Test bestätigt diese Theorie. Es wurde das Eingabefenster der neu geladenen Aufgabe angezeigt.

5.2 Prüfung des Berechnungsmoduls

5.2.1 Teststrategie

Das Modul Rechteckbemessung, welches umgesetzt wurde, ist nur für einen Teil der möglichen Anwendungsbereiche nutzbar. Es sind mehrere Grenzen vorgegeben. Innerhalb der ausführbaren Aufgabe soll das Modul stabil funktionieren. Eventuelle Fehler oder Verbesserungsvorschläge für nachfolgende Berechnungsmodule sollen aufgeschlüsselt werden.

5.2.2 Ausführung

Die Beispielberechnung Rechteckbemessung wird anhand einer geprüften Rechnung verglichen. Die Aufgabe liegt innerhalb der definierten Grenzen von C ≤ 50/60, einfacher Bewehrung und ohne Normalkraftbeanspruchung. Die zweite Aufgabe testet die Grenzen. Die Betonfestigkeitsklasse kann nicht außerhalb der Grenze gewählt werden, da dass vom Modul gesichert ist, anhand der externen Datei. Normalkräfte können nicht eingegeben werden und somit kann aus diesem Grund kein Fehler auftreten. Theoretisch soll das Modul darauf hinweisen, dass beim Grenzübertritt der „Querschnitt vergrößert oder doppelte Bewehrung" verwendet werden soll. Ob das funktioniert wird getestet.

Beispiel 1

Es ist ein Beispiel gegeben, welches innerhalb der Grenzen ein Ergebnis liefert. Die Aufgabe ist ein Balken mit einer Rechteckgeometrie und einfacher Bewehrung zu bemessen. Es wird der Betonstahl BSt 500 S verwendet. Die Betonfestigkeit von C 20/25 und die normale Luftfeuchtigkeit (umbauter Raum) wird angenommen.

1)Beanspruchung:

$$M_{sds} = \gamma_G \cdot M_{Gk} + \gamma_q \cdot M_{Qk} = 1{,}35 \cdot 15{,}6 + 1{,}5 \cdot 15{,}6 = 44{,}5\,kNm$$

2)Materialkennwerte:

Beton C 20/25 $\quad f_{cd} = \frac{0{,}85 \cdot 20}{1{,}5} = 11{,}33\,N/mm^2 = 1{,}13\,kN/cm^2$

Betonstahl $\quad f_{yd} = \frac{500}{1{,}15} = 434{,}8\,N/mm^2 = 43{,}48\,kN/cm^2$

3)Querschnittsgeometrie:

Breite $\quad b = 100\,cm$

Höhe $\quad h = 20\,cm$

Betondeckung $\quad c_{nom} = c_{min} + \Delta c = 20 + 15 = 35\,mm$

vorgewählter Bügeldurchmesser $d_s = 10\,mm$

$$d_1 = c_{nom} + \frac{d_s}{2} = 35 + 5 = 40\text{mm}$$

Nutzhöhe $$d = h - d_1 = 20 - 4 = 16\,cm$$

4)Bemessung mit dem ω – Verfahren:

$$\mu_{Sds} = \frac{M_{Sds}}{f_{cd} \cdot b \cdot d^2} = \frac{4450}{1{,}13 \cdot 100 \cdot 16^2} = 0{,}154$$

aus Tafel: $$\xi = 0{,}208 < 0{,}45 = \lim \xi$$

$$\omega = 0{,}1686$$

$$A_s = \frac{f_{cd}}{f_{yd}} \cdot \omega \cdot b \cdot d = \frac{1{,}13}{43{,}48} \cdot 0{,}1686 \cdot 100 \cdot 16 = 7{,}01\,cm^2/m$$

5)Bemessung mit dem Näherungsverfahren:

$$\mu_{Sds} = \frac{M_{Sds}}{\chi \cdot f_{cd} \cdot b \cdot d^2} = \frac{4450}{0{,}95 \cdot 1{,}13 \cdot 100 \cdot 16^2} = 0{,}162$$

$$\xi = 1{,}25 \cdot (1 - \sqrt{1 - 2 \cdot \mu_{Sds}}) = 1{,}25 \cdot (1 - \sqrt{1 - 2 \cdot 0{,}162}) = 0{,}222$$

$$\xi < 0{,}45 = \lim \xi$$

$$\nu_{cd} = 0{,}8 \cdot \xi = 0{,}8 \cdot \xi = 0{,}178$$

$$A_s = \frac{\nu_{cd} \cdot \chi \cdot f_{cd} \cdot b \cdot d}{f_{yd}} = \frac{0{,}178 \cdot 0{,}95 \cdot 1{,}13 \cdot 100 \cdot 16}{43{,}48} = 7{,}03\,cm^2/m$$

Beispiel 2

In diesem Beispiel soll die Grenze getestet werden. Als Ergebnis soll die Information „Querschnitt vergrößert oder doppelte Bewehrung" erscheinen. Das wird erreicht, indem der Querschnitt verkleinert wird. Beispielsweise ist die Höhe h = 15 cm. Die errechnete Nutzhöhe d beträgt 11 cm. Diese Werte ergeben Ergebnisse, die nicht innerhalb der Grenze liegen. Damit soll die Information angezeigt werden.

5.2.3 Auswertung

Beispiel 1

Das Programm PDA-SB wurde mit dem Modul *Rechteckbemessung* gestartet. Die Eingabewerte wurden in die Maske eingetragen, siehe Abbildung 42. Anschließend wurde die Aufgabe berechnet. Das Ergebnis wird im Fenster in Abbildung 43 angezeigt.

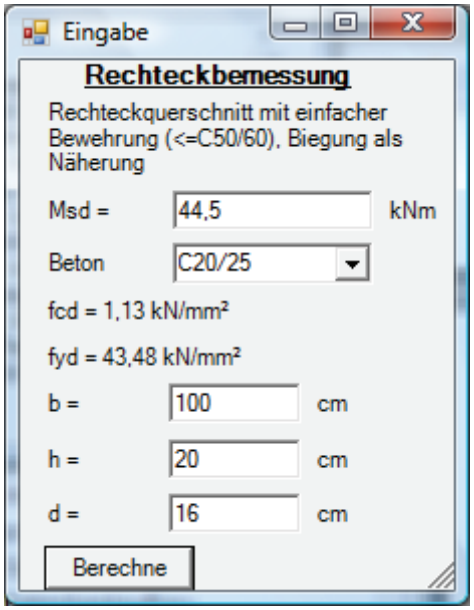


Abbildung 42: Eingabemaske Rechteckbemessung

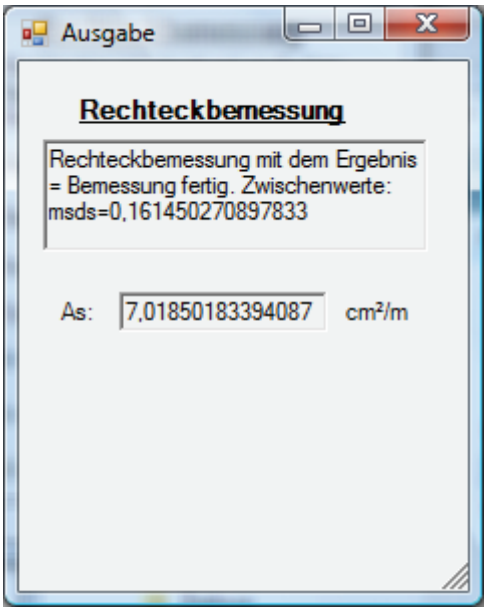


Abbildung 43: Ausgabemaske Rechteckbemessung

Das Ergebnis $\mu_{Sds} = 0{,}16145$ und $A_s = 7{,}0185\,cm^2/m$ sind nahe an den berechneten Werten. Damit ist repräsentativ die Gültigkeit auf richtiges Rechnen bestätigen.

Beispiel 2

Infolge der Verkleinerung des Querschnitts, bezogen auf das erste Beispiel, ergibt sich die Mitteilung, dass der „*Querschnitt vergrößert oder doppelte Bewehrung*" angenommen werden soll. Damit ist die gewünschte Reaktion eingetreten. Das Dialogfenster in Abbildung 44 zeigt diese Information.

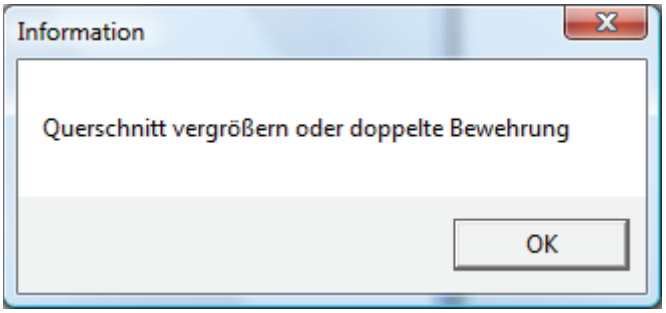


Abbildung 44: Information Rechteckbemessung

5.3 Leistungsbewertung

Die Hauptaufgaben wie Laden und Ansteuern eines Berechnungsmoduls funktionieren. Das Beispiel der Bemessung eines Rechteckquerschnitts läuft konstant ohne Fehler, zudem wird die Ausgabe schnell präsentiert. Die Wartezeit des Nutzers ist damit minimal.

Viele PDA mit dem Betriebssystem Windows Mobile können das Programm PDA-SB nutzen. Es ist ebenfalls möglich das System auf anderen Geräten, wie z.B. Handys oder Netbooks, zu verwenden. Eine Änderung der Programmiersprache in Java ist für die Erweiterung der Geräteauswahl denkbar, denn der Entwurf ist unabhängig von der Umsetzung.

Leider reichte die Zeit nicht aus, um die Datenspeicherung zu programmieren. Im Entwurf ist eine Beschreibung der Datenbankstruktur enthalten. Damit ist die Umsetzung auch im Nachfolgenden möglich. Bei erfolgreicher Implementation können mit dieser Funktion Berechnungsketten umgesetzt werden.

In der Summe führt der Prototyp die Aufgaben aus. Dennoch sind Verbesserungen möglich und würden das Programm PDA-SB ansprechender gestalten. Näheres ist im Kapitel 6 Zusammenfassung enthalten.

6 Zusammenfassung

Das Programm PDA-SB ist im Entwurf und als Prototyp vorhanden. Dabei sind Anwendungen aus dem konstruktiven Ingenieurbau umsetzbar. Das Beispiel einer Rechteckbemessung aus dem Stahlbetonbau demonstriert die Vorgehensweise, wie eine Berechnung aussehen kann und wie neue Aufgaben programmiert werden. Zeitlich war es nicht möglich die angedachte Datenbank zu implementieren. Dennoch ist aus dem Entwurf deutlich zu sehen, welche Möglichkeiten sich aufzeigen. Es ist nicht nur die Speicherung der Daten sondern dessen Nutzung in anderen Berechnungsmodulen angedacht. Daraus ergeben sich Aneinanderreihungen von Aufgaben. Was darunter zu verstehen ist, wurde im Kapitel 3.2.4 Berechnungskette geschildert.

Zur Erstellung einer Berechnung müssen die Schnittstellen für ein Modul benutzt werden. Aufgrund dieser Voraussetzung kann ein automatischer Wizard die Standardeinstellungen vornehmen. Diese Idee weiter gedacht, ergibt eine Automatisierung der Programmierung für das gesamte Berechnungsmodul. Es können mit Hilfe von Drag & Drop Funktionen die Oberflächen nicht nur grafisch gestaltet, sondern die Elemente auch mit mathematischen Formeln verknüpft werden. Das Datenblatt zur Umsetzung einer Aufgabe kann dokument-orientiert ablaufen. Daraus werden die Fenster erzeugt, welche später auf dem PDA Programm angezeigt werden. Es können vorgefertigte Programmteile eingebaut werden. Das ist sinnvoll, wenn z.B. eine Kurve angezeigt werden soll. Die Zeichnung kann unabhängig von den tatsächliche Daten funktionieren. Das Koordinatensystem muss sich dem echten Datenbestand anpassen und die Kurve einmalen, so dass das entstandene Bild auf dem PDA Bildschirm sichtbar ist. Es können auch mehrere Linien innerhalb einer Grafik angezeigt werden. Für jede Variante muss ein Tool entstehen, dass der Entwickler der Aufgabe benutzen kann. Dann benötigt dieser keine Programmierkenntnisse. Das automatisierte Programm bietet die Funktionen und Tools an. Diese können dann übernommen werden.

Literaturverzeichnis

KI Wiki 09: konstruktiver Ingenieurbau, 2009
http://de.wikipedia.org/wiki/Konstruktiver_Ingenieurbau

Schneider 02: Bautabellen für Ingenieure, 2002, Werner Verlag, 15. Auflage

Massivbau 06: Prof. Dr. - Ing. G. Bolle, Massivbau

PDA Wiki 08: Personal Digital Assistand, 22.02.2008
http://de.wikipedia.org/wiki/Personal_Digital_Assistant

Staticus 06: Staticus, Produktbeschreibung Staticus 2.5, 12.09.2006,
http://www.staticus.de

mb AEC Software 09: 2009, http://www.mbaec.de/baustatik/produktinfo.html

Lex_calsky 09: EVA-Prinzip,
http://lexikon.calsky.com/de/txt/e/ev/eva_prinzip.php

IT_Wissen 09: EVA-Prinzip, http://www.itwissen.info/definition/lexikon/EVA-Prinzip.html

C# 03: Peter Drayton, Ben Albahari, Ted Neward, C# in a nutshell, 2003,
O'reilly Verlag GmbH & Co. KG,

Glossar

C#	**C#** ist die Kurzform der Programmiersprache C-Sharp
CAD	**C**omputer **A**ided **D**esign: computer unterstütztes Design
DLL	**D**ynamic **L**ibrary **L**ink: dynamische Klassenbibliothek
FEM	**F**inite **E**lemente **M**ethode: numerisches Verfahren zur Näherungslösung
GUI	**G**raphical **U**ser **I**nterface: grafische Benutzeroberfläche
LCD	**L**iquid **C**rystal **D**isplay: Flüssigkristallbildschirm
PDA	**P**ersonal **D**igital **A**ssistant: ein kompakter tragbarer Computer
PIM	**P**ersonal **I**nformation **M**anager: Produkt-Informations-Management
RAM	**R**andom **A**ccess **M**emory: Arbeitsspeicher
ROM	**R**ead-**O**nly **M**emory: Festwertspeicher oder Nur-Lese-Speicher
SD	**S**ecure **D**igital Memory Card: sichere digitale Speicherkarte

Tabellenverzeichnis

Abbildungsverzeichnis

Anhang

A Benutzerhandbuch

A.1 Voraussetzungen

Um das Programm auf dem PC oder auf dem PDA auszuführen, wird .NET Framework (2.0 oder höher) benötigt.

Für den Transport vom PC zum PDA wird je nach Betriebssystem ein Programm benötigt. Die XP Versionen von Windows benutzen ActiveSync. Bei Windows Vista wird das Mobile-Gerätecenter verwendet, welches für 32 Bit oder 64 Bit Betriebssysteme vorhanden ist.

A.2 Installation

Das Programm PDA-SB liegt gepackt auf der CD. Diese ZIP Datei muss entpackt werden. Dann steht das Programm auf dem PC zur Verfügung.

Bei der Installation auf dem PDA, müssen die entpackten Dateien manuell über ein Transportprogramm wie ActiveSync auf den Handheld geschoben werden. Danach ist das Programm sofort einsetzbar.

In jedem Fall ist es wichtig, die Ordnerstruktur innerhalb des Arbeitsverzeichnisses zu erhalten. Dabei ist der Ordner *modul* sehr wichtig. Dieser sollte direkt im Arbeitsverzeichnis liegen. In Abbildung 45 ist die Anordnung zu sehen.

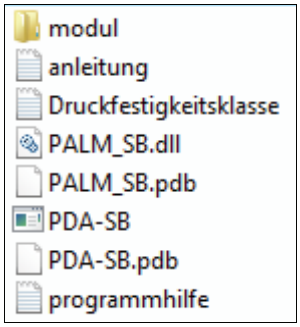


Abbildung 45: Arbeitsverzeichnis

A.3 Programm PDA-SB starten

Wenn die Voraussetzungen erfüllt sind und die Installation erfolgreich war, kann das Programm über die Anwendungsdatei *PDA-SB.exe* gestartet werden. Es sollte der Startbildschirm sichtbar werden.

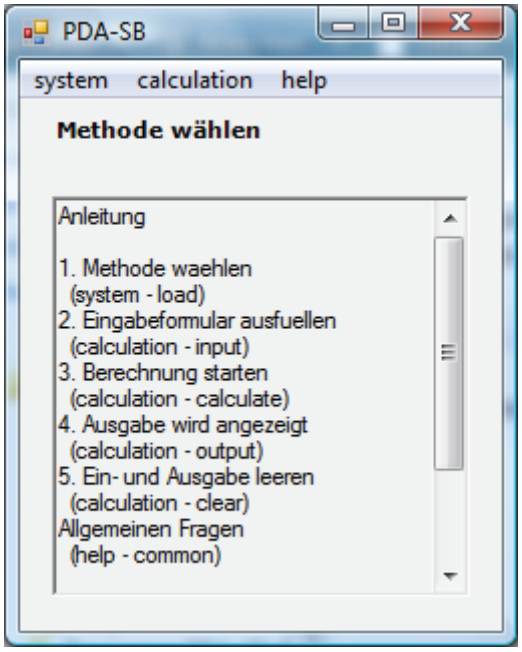


Abbildung 46: Startfenster von PDA-SB

A.4 Berechnung ausführen

Um eine Berechnung auszuführen, muss diese zur Laufzeit geladen werden. Die Erklärung erfolgt an einer Beispielberechnung, die auf eine bereits existierende

DLL Datei basiert. Wenn eine andere Aufgabe berechnet werden soll, die noch nicht gelistet ist, sollte das Kapitel A.6 (Berechnungsmodul hinzufügen) gelesen werden.

Das Modul für eine Berechnung wird über das Menü *system* und dann *load* gewählt. Es wird anhand der XML Datei *Modulliste* sämtliche frei geschalteten Berechnungen anzeigen. Veranschaulicht ist das in der folgende Abbildung 47 (Berechnungsmodul laden).

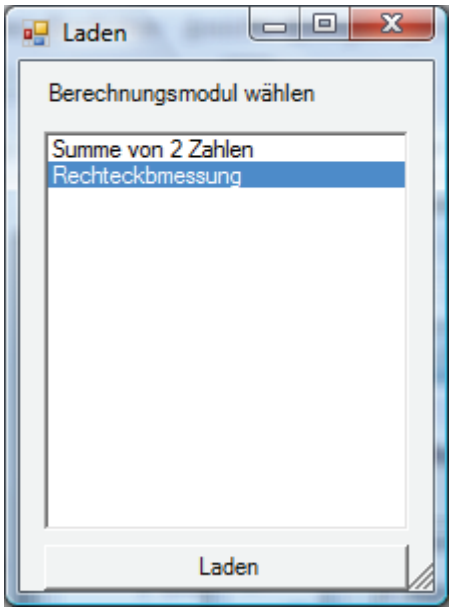


Abbildung 47:
Berechnungsmodul laden

Nach der Auswahl wird über den Button *Laden* das entsprechende Modul geladen. Im Anschluss wird bereits das Eingabefenster angezeigt. Dieses kann auch zu jedem späteren Zeitpunkt angesehen werden unter dem Menüpunkt *calculation* und dann *input*.

In unserem gewählten Beispiel handelt es sich um eine Bemessung für Rechteckquerschnitte mit einfacher Bewehrung ohne Normalkräfte, die mit einer Näherungsformel ermittelt wird. Das Eingabefenster sieht wie in Abbildung 48 (Eingabemaske Rechteckbemessung) aus.

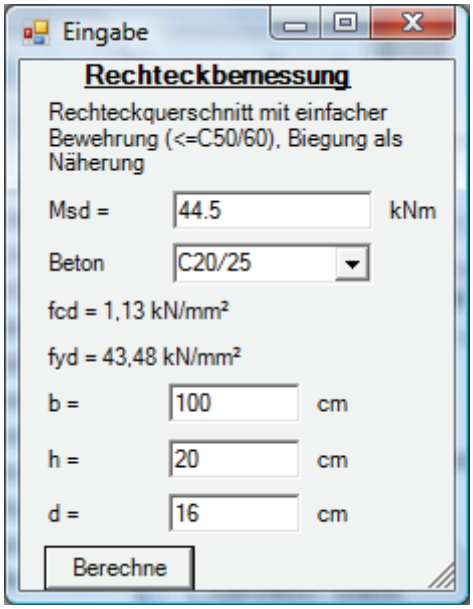


Abbildung 48: Eingabemaske Rechteckbemessung

Nach dem Klick auf *Berechne* erfolgt bei diesem Berechnungsmodul sofort die Berechnung. Im Anschluss wird die Ausgabe angezeigt. In anderen Fällen muss ggf. manuell die Berechnung über den Menüpunkt *calculation - calcute* angesteuert werden. Danach wird normalerweise die Ausgabe angezeigt. Diese kann dennoch jederzeit über den Menüpunkt *calculation – output* eingesehen werden.

Im gewählten Beispiel wird die Berechnung nach dem Klick auf *Berechne* ausgeführt. Sofort im Anschluss wird die Ausgabe, wie in Abbildung 49 (Ausgabemaske Rechteckbemessung) zu sehen ist, angezeigt.

Die Werte sind mit As = 7,02 cm²/m annehmbar. Im Vergleich mit dem ω-Verfahren beträgt As = 7,01 cm²/m, während mit der Näherungsformel und gerundeten Werten als Ergebnis As = 7,03 cm²/m ermittelt wird.

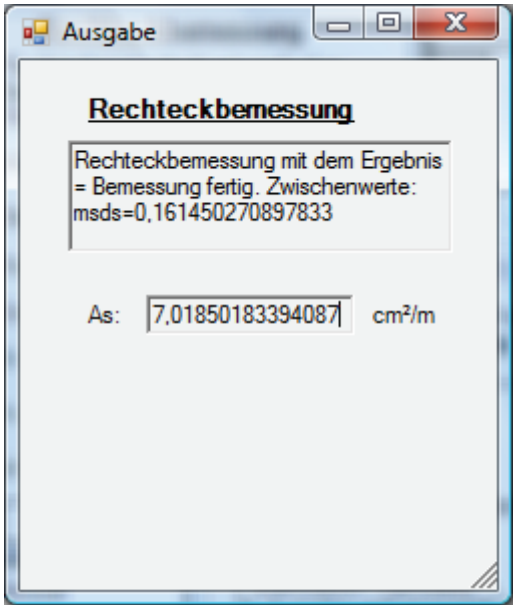


Abbildung 49: Ausgabemaske Rechteckbemessung

A.5 Anleitungen des Programms

Eine Anleitung zu dem Hauptprogramm befindet sich in kurzen Auszügen bereits auf dem Startfenster, während die ausführliche Beschreibung im Menüpunkt *help – common* nach zu lesen ist. Die Berechnungsanleitung, falls eine erstellt wurde, befindet sich im Menüpunkt *help - calculation*. Die Informationen zu der Version und Autor des PDA-SB Programms befindet sich unter *help – info*.

Zusammenfassung

Menüpunkte unter help	Beschreibung
common	Anleitung für das Hauptprogramm
calculation	Anleitung für die Berechnung
info	Allgemeine Informationen

Tabelle 5: Übersicht über Anleitungen im Programm PDA-SB

A.6 Berechnungsmodul hinzufügen

Die DLL Dateien der Berechnungsmodule müssen im Ordner *modul* eingefügt und in der dortigen XML Datei *Modulliste.xml* eingetragen werden.

Kurzanleitung

1. DLL einfügen in Ordner modul
2. Modulliste.xml anpassen

Beispiel

Nachdem die DLL in den Ordner *modul* eingefügt wurde, kann die Datei *Modulliste.xml* angepasst werden. Die Herangehensweise wird im Folgenden beschrieben. Ein Auszug aus der Liste in Abbildung 50 zu sehen.

```
<Modulliste>
    <modul>
        <Name>Rechteckbemessung</Name>
        <DllName>Bemessung.dll</DllName>
    </modul>
    <modul>
        <Name>Multiplikation von 2 Zahlen</Name>
        <DllName>Test.dll</DllName>
    </modul>
</Modulliste>
```

Abbildung 50: Auszug aus Modulliste.xml

In Abbildung 50 ist die Modulliste auszugsweise angezeigt. Dabei besteht das Modul aus einem *Namen* und einem *DllNamen*. Die Namen sind in der Auswahl zum Laden der Berechnung angezeigt. Nach dem Bestätigen des Nutzers wird die entsprechende DLL Datei eingeladen. Diese muss sich im Ordner *modul* befinden. Der *DllName* muss exakt eingetragen werden. Groß- und

Kleinschreibung ist zu beachten. Ansonsten treten Probleme beim Ladevorgang auf.

A.7 Berechnungsmodul entfernen

Damit ein Berechnungsmodul entfernt wird, ist die Löschung des Eintrages aus der Datei *Modulliste.xml* notwendig. Dort muss der Eintrag für das Modul mit dem Namen und dem DLL-Verweis gelöscht werden. Somit ist die Berechnung für das Programm PDA-SB nicht mehr sichtbar.

Um das Berechnungsmodul komplett vom PC oder vom PDA zu entfernen, muss die DLL Datei gelöscht werden. Doch Vorsicht, denn die Datei wird dann vollständig vom System entfernt.

B Entwicklerhandbuch

B.1 Voraussetzungen

Die Voraussetzungen für die Weiterentwicklung des Programms sind umfassender als die für die Benutzung von PDA-SB. Hier wird das .Net Framework SDK (2.0 oder höher) benötigt. Dieses ist in der Express Version von Microsoft Visual Studio vorhanden und wird mit installiert. Bei SharpDevelop als Entwicklungsumgebung muss dieses separat installiert werden.

Für den Transport kommt es auf das Betriebssystem des Entwickler PC an. Bei XP oder ähnlichen Windows Programmen funktioniert ActiveSync. Bei Windows Vista wird Windows Mobile Device Center (Gerätecenter) benötigt. Die Programme sind kostenlos im Internet verfügbar.

Microsoft Visual Studio ist kostenpflichtig, was bei der Einrichtung eines Entwickler PC zu bedenken ist. Die Alternativen sind SharpDevelop oder die Express Version von Visual Studio.

B.2 Installation und Programmstart

Die gepackte Datei wird in einem gewählten Ordner entpackt und kann sofort benutzt werden. Über *PDA-SB.exe* kann das Programm gestartet werden. Nähere Information befinden sich im Kapitel A Benutzerhandbuch.

Für die Veränderung des Hauptprogramms PDA-SB ist der Quellcode separat vom Programm gepackt. Die Erstellung erfolgte in Kombination von Microsoft Visual Studio und SharpDevlop. Daher ist es zu empfehlen, sich lediglich die C# Dateien anzuschauen bzw. zu verändern oder sein eigenes Projekt zu erstellen, um die Dateien zu verändern. Ansonsten können Fehler auftreten. Es kommt auf einen Versuch an.

Im Anschluss befindet sich eine ausführliche Anleitung, um Berechnungsmodule zu erstellen.

B.3 Berechnungsmodul erstellen

Das Berechnungsmodul wird losgelöst vom Rahmenprogramm entwickelt. Es werden Schnittstellen benötigt, die sich in der *PALM_SB.dll* befinden. Die DLL muss im Projekt als Referenz eingefügt werden, damit diese für die Programmierung zur Verfügung steht.

Die vier Klassen (Eingabe, Ausgabe, Berechnung und Hilfe) müssen die Schnittstellendefinitionen einladen und die Methoden füllen. Was im Detail passiert, entscheidet der Programmierer. Um einen Überblick zu erhalten, wird die Vorgehensweise an einem Beispiel erklärt.

Auf der CD befinden sich die DLL und der Quellcode für das Beispiel. Diese werden für die Erstellung einer neuen Berechnung verwendet. Die zu entpackende Datei heißt *Test* und umfasst ein SharpDevelop Projekt, dass bereits sämtliche Klassen implementiert und die *PALM_SB.dll* als Referenz eingebunden hat.

Die Eingabe, die Ausgabe und die Hilfe besitzen jeweils ein Fenster, das separat gestaltet werden kann. Die vom Hauptprogramm angesteuerte Eingabe ist die Klasse *input_calculation*. In dieser werden die Schnittstellen eingefügt und an die Fensterklasse *input* weitergeleitet. Das bedeutet, dass diese beiden Klassen eng miteinander kommunizieren. *Input.cs* hat einen Designer, mit dem über drag & drop Funktionen das Fenster gestaltet wird. Es können dort auch Dialoge zum Einlesen von externen Dateien verwendet werden. In einer anderen Berechnung wurden die Betonfestigkeitsklassen aus einer Textdatei gelesen. Der Nutzer kann daraus auswählen.

Wenn aus Dateien, die der Berechnung angefügt sind, gelesen werden soll und eine feste relative Pfadangabe erfolgt, sollte bedacht werden, dass die DLL

später im Unterordner *Modul* liegt. Falls die Datei (hier *Textdatei.txt*) im Order *Modul* liegt, muss der Pfad *„Modul\Textdatei.txt*" lauten. Ansonsten muss die Datei im Arbeitsverzeichnis des Hauptprogramms liegen. Dann ist der Pfad *„Textdatei.txt*".

Eingabe

Das Auslesen der Eingabewerte erfolgt nach einer Kontrolle, die der Programmierer selbst bestimmen kann. Mindestens sollte geprüft werden, ob überhaupt Daten eingetragen worden sind. Erweiterbar ist die Kontrolle mit Regulären Ausdrücken.

Nachdem die Kontrolle der Daten erfolgte, werden die Werte in die Eingabeliste geschrieben. Mehr dazu befindet sich im Kapitel 3.2.1 Datenstruktur. Solch eine Ausleseabfrage kann wie in Abbildung 51 aussehen. Die Methode *getInput* liefert die Eingabeliste als *dataList*. In der nachfolgenden Abbildung wird nur der Name und die dazugehörige Zahl gespeichert. Bei Bedarf können die Werte auch eine Bezeichnung und eine Einheit besitzen.

```
/// <summary>
/// Liefert die dataList, wenn die Werte in den Textboxen stehen.
/// Ansonsten wird null zurück gegeben.
/// </summary>
/// <returns>dataList mit den Eingabewerte</returns>

public dataList getInput() {
    dataList dl = null;
    if (checkInput()) {
        dl = new dataList();
        dl.add(new data("a",textBox1.Text));
        dl.add(new data("b",textBox2.Text));
    }
    return dl;
}
```

Abbildung 51: Beispielmethode um Eingabedaten auszulesen

Berechnung

Die eigentliche Berechnung erfolgt in der Klasse *calculate.cs*. Hier wird im Beispiel die Summe der beiden Zahlen a und b gebildet. Angenommen wird der Typ double. Der Quellcode dazu ist in Abbildung 52 zu sehen.

Die Methode *run* führt die Berechnung aus. Als Erstes werden die Eingabewerte aus der Liste gelesen. Mit den Werten *a* und *b* wird die Multiplikation ausgeführt. Das ist der eigentliche Teil der Berechnung. Danach werden die Ergebnisse (hier *c*) lokal gespeichert, die für die Ausgabe bereit stehen.

```
/// <summary> Führt die Berechnung aus. </summary>
/// <param name="Werte">Eingabewerteliste mit a und b</param>

public void run(dataList Werte) {
    //Werte aus Eingabe auslesen
    double a = Werte.getData("a").getDoubleWert();
    double b = Werte.getData("b").getDoubleWert();

    //Berechnung
    double c = a+b;

    //Ergebisliste erzeugen
    ergebnis = new dataList();
    ergebnis.add(new data("c",c.ToString()));
}
```

Abbildung 52: Berechnung einer Aufgabe

Ausgabe

Es folgt die Ausgabeklasse *output_calculation* mit der Fensterklasse *output.cs*. Dort erscheint das Ergebnis. Das Beispiel zeigt solch eine Ausgabemethode in Abbildung 53.

Die Hauptmethode dieser Klasse ist *fillOutput*, die zwei Parameter benötigt. Das sind zum Einen die Eingabedaten (*input*) und zum Anderen die Ergebnisse (*result*). Die Werte werden in Textboxen angezeigt.

```
/// <summary> Diese Methode füllt das Ausgabefenster. </summary>
/// <param name="result">Ergebnisliste mit c</param>
/// <param name="input">Eingabeliste mit a und b</param>

public void fillOutput(dataList result, dataList input) {
    //Eingabewerte anzeigen
    textBox1.Text = "Das Ergebnis von a + b = c."
      +Environment.NewLine+input.getData("a").getStringWert()
      +" + "+input.getData("b").getStringWert()+" = c";

    //Ausgabewerte anzeigen
    label2.Text = "c =";
    textBox2.Text = result.getData("c").getStringWert();
}
```

Abbildung 53: Ausgabefenster mit Werten füllen

Anleitung

Die Hilfe bzw. Anleitung zu der Berechnung befindet sich in *help_calculation.cs*, welche eine Fensterklasse erzeugt, die *help.cs* heißt. Das Fenster kann beliebig gestaltet werden. Wenn in der Eingabe bereits eine Sprungmarke für die Hilfe angelegt wurde, kann die richtige Stelle angezeigt werden. Ansonsten wird am Anfang gestartet. Im Beispiel steht lediglich ein einfacher Hinweis in einem Label, siehe Abbildung 54.

```
label1.Text = "Hier steht die Hilfe...\n\n";
label1.Text += "Folgen Sie den Anweisungen auf dem Bildschirm";
```

Abbildung 54: Hinweistext in der Anleitung

C Erklärung der Menüpunkte von PDA-SB

C.1 System

Load

Dieser Menüpunkt zeigt die Auswahl der auf dem PDA befindlichen Programmmodule. Es kann ein Modul ausgewählt werden. Dieses führt dann eine Berechnung aus.

Close

Ein Klick auf diesen Menüpunkt deaktiviert das Berechnungsmodul. Es steht dann nicht mehr zur Nutzung bereit und wird vom System gelöscht.

Exit

Dieser Menüpunkt beendet das Hauptprogramm. Es werden alle aktiven Fenster geschlossen.

C.2 Calculation

Input (Eingabe)

Hinter diesem Menüpunkt versteckt sich die Eingabemaske für die Berechnung. Konfigurationseinstellungen, falls diese erforderlich sind, werden von diesem Fenster bereitgestellt. Die Aufteilung des Bildschirmes ist frei wählbar. Lediglich die Schnittstellendefinition muss eingehalten werden.

Calculate (Berechne)

Mit diesem Menüpunkt wird die Berechnung manuell gestartet. Dazu muss die Eingabe korrekt ausgefüllt sein.

Output

Jede Berechnung besitzt mindestens eine Ausgabe. Es können auch mehrere Ausgaben geliefert werden. Die Aufteilung und Strukturierung ist dem jeweiligen Programmierer überlassen.

Clear

Bei diesem Aufruf werden die Berechnungsfelder geleert. Das wird erreicht, indem die Eingabe- und Ergebnislisten geleert werden. Das Modul muss darauf reagieren, dass die Listen leer sind und nicht alte Werte anzeigen. Alternativ kann die Methode *clear* verwendet werden.

C.3 Help

Common

Eine allgemeine Anleitung zur Verwendung des Programms mit nützlichen Hinweisen für den Umgang.

Calculation

Eine ausführliche Hilfe zum Berechnungsmodul. Bei einer Neuerstellung einer Berechnung, kann diese entsprechend gestaltet werden. Jederzeit kann die Anleitung außerhalb des Systems geändert werden. Eine gewisse Kenntnis über die Berechnung sollte allerdings gegeben sein, damit keine fehlerhafte oder falsche Hilfe zur Verfügung steht.

Info

Die Information enthält wichtige Angaben zum Programm. Details wie Versionsstand, Copyright, Autor und Datum.

D Daten auf der CD

Auf der CD sind einige Daten vorhanden. Die Beschreibungen erfolgen in einer Aufzählung, wobei sich diese auf die Ordnerstruktur in Abbildung 55 bezieht.

1. Die Diplomarbeit *DA_Preik* befindet sich als PDF Datei auf der CD.
2. Das ausführbare Programm *PDA-SB* liegt im gleichnamigen Ordner. Es ist im ZIP Format gepackt.
3. Der Quellcode des Hauptprogramms liegt im Ordner *PDA-SB_Quellcode*. Die C# Dateien sind gepackt (ZIP).
4. Die Schnittstellendefinition sind im Ordner *Schnittstellen*. Mit dem Quellcode (ZIP Format) kann eine Änderung vorgenommen werden. Dabei sind die Versionen zu berücksichtigen, denn ansonsten funktionieren ältere Berechnungsmodule nicht mehr. Die Programmbibliothek *Palm_SB* ist die DLL für die Schnittstellen, die einem Berechnungsmodul hinzugefügt wird.
5. Die Rechteckbemessung liegt im Ordner *Berechnung Rechteckbemessung*. Der Quellcode (ZIP Datei) und die Programmbibliothek (DLL) sind enthalten.
6. Die Berechnung Test, welche zwei Zahlen summiert, ist im Ordner *Berechnung Test*. Diese dient als Anleitung für die Erstellung weiterer Berechnungsmodule. Vorhanden ist die Programmbibliothek (DLL), welche das fertige Modul umfasst, und der Quellcode gepackt als ZIP Datei.

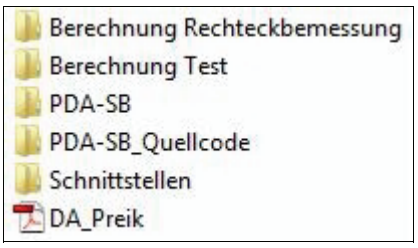


Abbildung 55: Ordnerstruktur auf der CD

E Ergänzungen

Im Weiteren folgen Klassen, die das Kapitels 3.2.1 Datenstruktur ergänzen. Es folgt die Klasse *dataVector*, welche Vektoren beschreiben (siehe Abbildung 56).

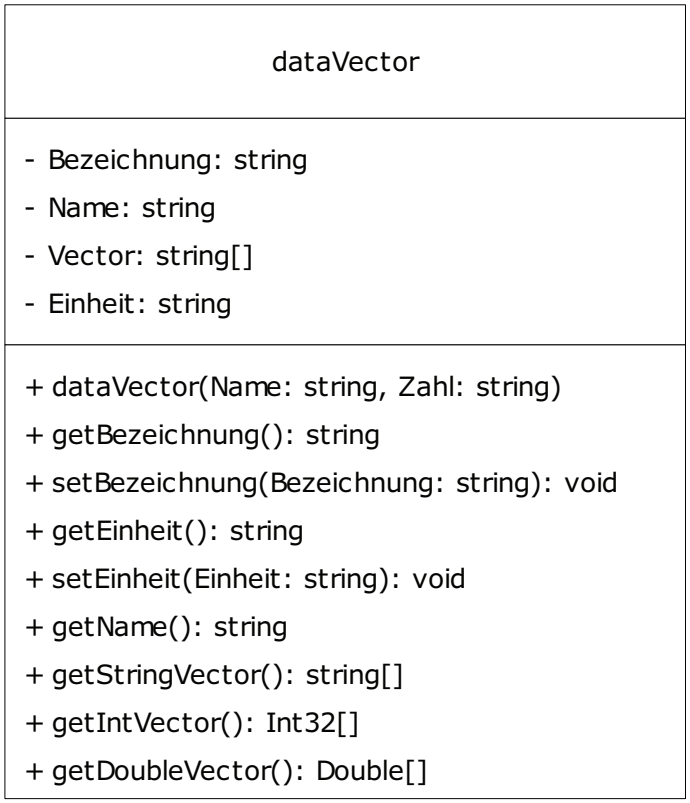


Abbildung 56: Klasse dataVector

Die Klasse *dataVector* (siehe Abbildung 56) besitzt einen Vektor um die Zahlen zu hinterlegen. Angepasst sind die *get*-Methoden. Ansonsten ist der Rest wie in der Klasse *data* (siehe Abbildung 21).

Des Weiteren gibt es die Klasse *dataMatrix*, die Matrizen beschreibt. In Abbildung 57 ist diese zu sehen.

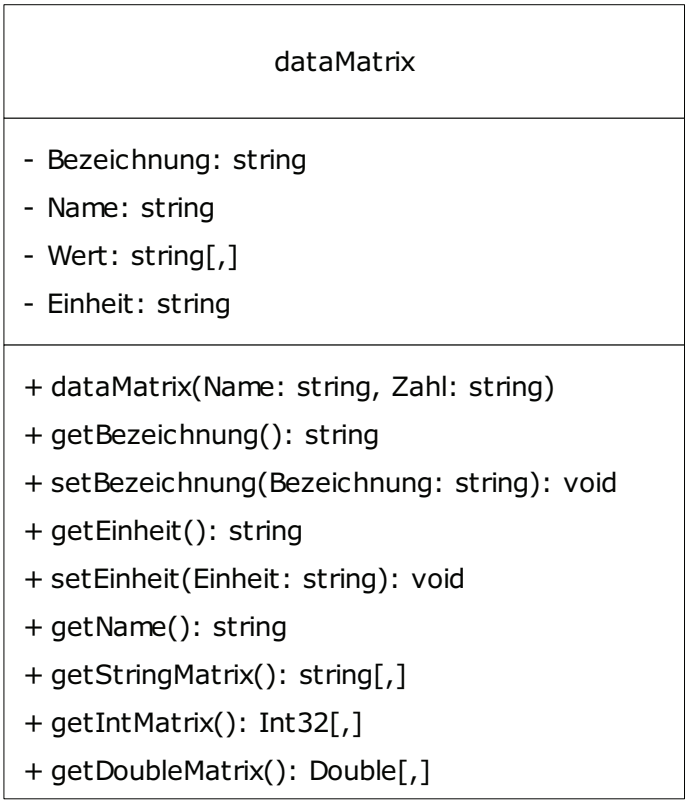


Abbildung 57: Klasse dataMatrix

In Abbildung 57 ist die Klasse *dataMatrix* zu sehen. Diese unterscheidet sich zum *dataVector* nur um die Matrix. Ebenfalls sind die *get*-Methoden angepasst.