Hochschule für Technik und Wirtschaft Dresden

Fachbereich Informatik/Mathematik
Diplomstudiengang - Allgemeine Informatik
Prof. Dr.-Ing. W. Nestler

DIPLOMARBEIT

Entwicklung einer Software für ein Olfaktometer

zum
Erlangen des akademischen Grades

Diplom-Informatiker (FH)

Dipl.-Inf. (FH)

Vorgelegt von:	Johannes Körner		
geboren am:	22. Juni 1986	in:	Pirna
Betreuer:	Prof. Dr. med. Thomas Hummel		
Betreuer Hochschule:	Prof. Dr.-Ing. W. Nestler		

Selbständigkeitserklärung

Hiermit erkläre ich, dass ich die von mir am heutigen Tage dem Prüfungsauschuss des Fachbereichs Informatik/Mathematik eingereichte Diplomarbeit zum Thema

Entwicklung einer Software für ein Olfaktometer

vollkommen selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Dresden, den 12.03.2009

Unterschrift

Inhaltsverzeichnis

1 Überblick

Die Olfaktometrie ist die Wissenschaft, die sich mit der Erforschung des Geruchssinns beschäftig. Für das Testen des Geruchssinns des Menschens gibt es verschiedene Verfahren. Die einfachsten und preiswertesten Verfahren werten die Reaktionen von Menschen auf einwirkende Gerüche aus. Einige können mit Hilfe eines sogenannten Olfaktometers durchgeführt werden. Olfaktometer sind Geräte, mit denen definierte Gerüche ausgegeben werden können.

Diese Arbeit beschäftigt sich mit der Entwicklung einer Software für ein solches Olfaktometer.

Die Hauptarbeit bestand aus der Ermittlung der Anforderungen und der Modellierung der Software. Bei der Implementierung wurden neue Technologien des .NET Frameworks einbezogen.

Folgende Gesichtspunkte sind in der Arbeit untersucht:

1. Eignung der Language Integrated Query (LINQ)- Technologie als Lösung für objektrelationales Mapping in kleinen Projekten.
2. Eignung der Windows Presentation Foundation als GUI- Lösung für .NET-Applikationen.

2 Einleitung

2.1 Ausgangssituation und Zielsetzung

Die folgende Arbeit bezieht sich auf die Entwicklung einer Software für ein sogenanntes Olfaktometer für die Hals-Nasen-Ohren (HNO)-Klinik des Universitätsklinikum Dresden. Bei dem Olfaktometer handelt es sich um ein Gerät, mit dessen Hilfe einem Probanden Gerüche präsentiert werden können. Damit kann das Riechvermögen (olfaktorische Wahrnehmung) eines Probanden getesten werden. Das Riechvermögen gibt an, wie gut eine Person Gerüche wahrnehmen kann. Dabei werden dem Probanden mehrere Geruchsproben präsentiert. Als Geruchsprobe wird die Ausgabe eines bestimmten Geruches vom Olfaktometer bezeichnet. Der Proband muss versuchen diese Geruchsprobe zu identifizieren, d. h. er muss herausfinden, um welchen Geruch es sich handelt.

Es soll eine Software entwickelt werden, die einem Probanden ermöglicht ohne fremde Hilfe einen Riechtest am Olfaktometer durchzuführen.

2.2 Zielgruppen

Für die zu entwicklende Software gibt es verschiedene Zielgruppen. Personen gehören zu einer Zielgruppe, wenn diese das System nutzen oder deren Arbeit vom System beeinflusst wird.

- Auftraggeber
- Arzt
- Wissenschaftler
- Krankenschwester
- Proband

Im Folgenden werden die Interessen der einzelnen Zielgruppen am System analysiert.

2.2.1 Auftraggeber

Auftraggeber für die Entwicklung der Software ist die Hals-Nasen-Ohren Klinik des Universitätsklinikums Dresden. Genauer handelte es sich um die Abteilung „Riechen und Schmecken“ unter der Leitung von Prof. Dr med. Thomas Hummel. Die Abteilung wurde 1998 gegründet und es entstand ein Funktionsbereich, in dem sowohl an der Beratung von Patienten mit Störungen des Riech- und Schmeckempfindens als auch an der Forschung in diesem Bereich gearbeitet wird. Der Auftraggeber will das System zur Unterstützung der Mitarbeiter in der Abteilung einsetzen.

2.2.2 Arzt

Der Arzt soll vom System die Informationen erhalten, die ihm sagen, ob bei einem Patienten ein Verdacht auf eingeschränktes Riechvermögen (bezeichnet als funktionelle Anosmie) besteht. Außerdem möchte er verschiedene Tests eines bestimmten Patienten miteinander vergleichen, um z. B. herauszufinden, welche Gerüche der Patient immer falsch bzw. immer richtig erkennt.

2.2.3 Wissenschaftler

Der Wissenschaftler kann anhand der erfassten Testdaten aller Probanden Zusammenhänge zwischen dem Riechvermögen und dem Alter oder dem Geschlecht des Probanden untersuchen. Für ihn ist es wichtig Daten zu erhalten, die miteinander vergleichbar sind und mit denen statistische Auswertungen vorgenommen werden können.

2.2.4 Krankenschwester

Die Krankenschwester ist bisher für die Durchführung von Riechtests verantwortlich. Auswertungsdokumente von Riechtests werden von ihr ausgedruckt und an den behandelnden Arzt übergeben. Mit Verwendung des Olfaktometers soll die Zeit eingespart werden, die sie benötigt, um den Identifikationstest mit einem Patienten durchzuführen. Das Auswertungsdokument des Tests soll vom Patienten selbst ausgedruckt werden können. Dadurch hat die Krankenschwester weniger bzw. idealerweise keine Arbeit mehr damit.

2.2.5 Proband

Als Proband wird die Person bezeichnet, welche einen Riechtest am System durchführen möchte. Es wird zwischen Patienten und Besuchern im Deutschen Hygienemuseum Dresden unterschieden. Ein Patient soll den Riechtest auf Anweisung des Arztes hin durchführen, damit der Arzt eine Diagnose stellen kann. Ein Besucher des Hygienemuseums hingegen wird aus eigenem Interesse einen Riechtest am System durchführen. Sowohl vom Patienten als auch vom Besucher des Hygienemuseums werden die erfassten Testdaten gespeichert. Aus datenschutzrechtlichen Gründen werden bei Besuchern des Hygienemuseums keine Namen und Adressen gespeichert.

Die Software soll möglichst einfach durch den Probanden bedienbar sein. Außerdem sollen abgeschlossene Riechtests aus datenschutzrechtlichen Gründen nicht von Probanden einsehbar sein.

2.3 Motivation

Ein Olfaktometer ist an der Uniklinik vorhanden. Dieses ist bisher mangels geeigneter Software nicht für die Durchführung von Riechtests nutzbar. Mit Hilfe einer geeigneten Software könnten Patienten alleine Riechtests mit dem Olfaktomter durchführen. Dadurch kann die Schwester Zeit sparen, da sie den Patienten nicht betreuen muss. Außerdem wird es möglich das Olfaktometer im Hygienemuseum auszustellen. Die durch eine Software gespeicherten Daten können für Vergleiche und Auswertungen zur Verfügung stehen.

3 Definition und Ermittlung des Riechvermögens

3.1 Das Riechvermögen

Das Riechvermögen (auch olfaktorische Wahrnehmung) bezeichnet die Fähigkeit zur Wahrnehmung von Gerüchen. Die Geruchsrezeptoren des Menschen sind in der Nase lokalisiert und schon bei der Geburt vollständig ausgereift. Die Wahrnehmung eines Geruches ist von folgenden Faktoren abhängig:

- Dauer der Geruchseinwirkung
- Luftfluss durch die Nase
- Konzentration des Geruchs
- Überdeckung durch andere Gerüche
- Temperatur
- Luftfeuchtigkeit

Nicht alle beschriebenen Faktoren können bei einem Test des Riechvermögens berücksichtigt werden.

3.2 Test des Riechvermögens

Beim Testen der olfaktorischen Wahrnehmung wird zwischen subjektiven und objektiven Verfahren unterschieden.

Bei objektiven Verfahren werden bei Reizaufnahme die Hirnströme gemessen und ausgewertet. Auf objektive Verfahren soll hier nicht näher eingegangen werden. Die Ergebnisse von objektiven Verfahren können nicht durch den Probanden beeinflusst werden.

Im Gegensatz dazu sind subjektive Verfahren auf die Mitarbeit des Probanden angewiesen. Sie zeichnen sich durch schnelle Durchführbarkeit und differenzierte Beurteilungsmöglichkeiten aus. Subjektive Verfahren basieren auf der Interpretation von Reaktionen eines Probanden auf olfaktorische Reize. Dabei ist zu beachten, dass ein bestimmter Geruch nur erkannt werden kann, wenn dieser einem Probanden auch bekannt ist.

Für das Testen der Riechfunktion mit subjektiven Verfahren gibt es folgende Methoden:

- Test mit Sniffin' Sticks
- Test mit Olfaktometer

Diese Verfahren werden als „Screening Tests“ bezeichnet, d. h. es können noch keine umfassenden Diagnosen erstellt werden. Sie sollen lediglich ermitteln, ob bei einem Patienten möglicherweise eine Riechstörung vorliegt oder nicht. Nur Patienten, bei denen ein Verdacht auf eine Riechtstörung vorliegt, werden weitergehend untersucht. Vorteil der subjektiven Verfahren ist, dass diese relativ schnell und günstig durchgeführt werden können.

An der Uniklinik werden derzeit Riechtests mit Sniffin' Sticks durchgeführt. Durch die zu entwickelnde Software sollen Riechtests mit dem Olfaktometer ermöglicht werden.

3.3 Ablauf eines Riechtests mit Sniffin' Sticks

Ein Riechtest mit Sniffin' Sticks besteht aus drei Teiltests, dem Identifikations-, Diskriminations- und Schwellentest.

Testart	Erklärung
Identifikationstest	Der Proband bekommt einen Sniffin' Stick und soll den präsentierten Geruch zu einer bestimmten Frucht, einem Gewürz oder sonstigem Objekt zuordnen. Er kann dabei aus vier vorgegebenen Antwortmöglichkeiten (1 x richtig + 3 x falsch) wählen.
Diskriminationstest	Der Proband hat die Augen verbunden und bekommt Sniffin' Sticks präsentiert, wobei zwei Sticks den gleichen Geruch haben und ein Stick einen anderen Geruch hat. Der einzelne Stick soll herausgefunden werden.
Schwellentest	Der Proband hat die Augen verbunden und bekommt drei Sniffin' Sticks präsentiert, wobei nur einer einen Geruch hat. Dieser muss ermittelt werden. Über mehrere Proben ändert sich die Konzentration des Geruchs.

Tab. 3.1: Teiltests mit Sniffin' Sticks

Jeder Teiltest besteht aus mehreren Proben (an der Uniklinik 16). Als Geruchsprobe wird der abgeschlossene Ablauf der Präsentation von einem oder mehreren Sniffin' Sticks bezeichnet, zu dem der Proband genau eine Antwort geben kann. Der Test wird von einer Schwester durchgeführt. Dabei trägt die Schwester zunächst die Stammdaten des Nutzers in ein vorhandenes Softwaresystem (OLAF) ein. Danach beginnt der Test und das System führt mithilfe eines Wizards über alle drei Teiltests. Zu jeder Probe werden dem Patienten, abhängig vom jeweiligen Teiltest, zwischen einem und drei Sniffin' Sticks präsentiert. Die Schwester hält die jeweiligen Sniffin' Sticks vor die Nase des Probanden. Daraufhin muss der Patient eine Antwort auf die jeweilige Frage geben. Der Patient darf sich nicht der Antwort enthalten. Die Schwester überträgt dann die Antwort des Patienten in das System. Nachdem alle drei Teilstests abgeschlossen wurden, präsentiert das System ein Auswertungsdokument.

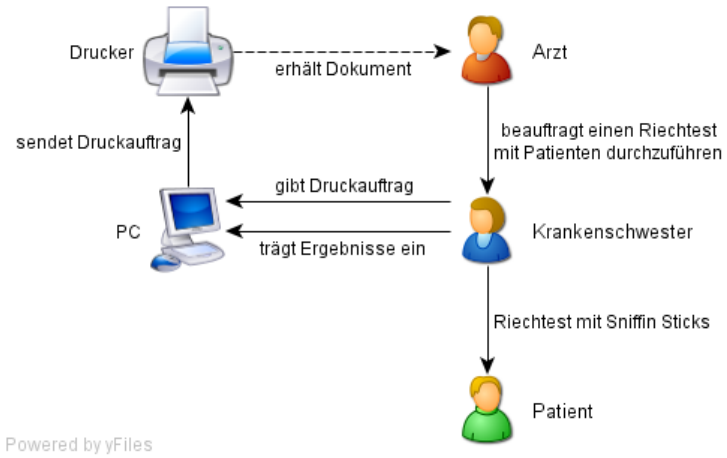


Abb. 3.1: Ablauf „Riechtest mit Sniffin' Sticks“

Im Anhang A.3 ist ein beispielhaftes Auswertungsdokument für einen Riechtest mit Sniffin' Sticks, welches das System OLAF erstellt, zu finden.

3.4 Das Olfaktometer

Die HNO-Klinik besitzt ein Olfaktometer der Firma „ScentCommunication“ mit der Bezeichnung „aerome® ScentController 2x6 (AB 60005)“ (Handbuch im Anhang A.2). Das Gerät kann mit Hilfe von austauschbaren Kartuschen verschiedene Düfte ausgeben.

Das Olfaktometer hat folgenden Bestandteile:

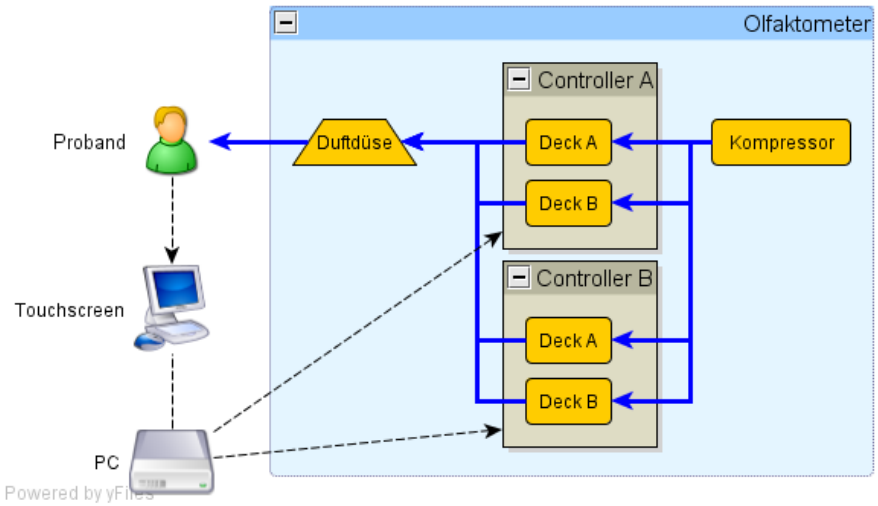


Abb. 3.2: Schematischer Aufbau eines Olfaktometers

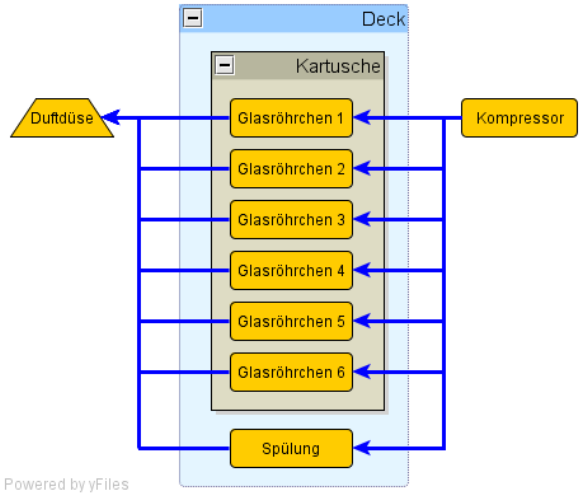


Abb. 3.3: Ein Deck im Detail

1. **Controller** Der Controller (aerome ScentController AB60004) ist für das Schalten der Ventile zuständig. Es sind zwei Controller vorhanden, die jeweils über eine serielle Schnittstelle mit einem PC kommunizieren können.

2. **Deck** Das Deck nimmt die Duftkartuschen auf. Zu jedem Controller gehören zwei Decks (Deck A und B).

3. **Ventil** Die Ventile sind Bestandteile des Decks und dienen der Steuerung des Luftstromes durch das Olfaktometer. Für jede der sechs Duftröhren einer Kartusche kann ein einzelnes Ventil geöffnet oder geschlossen werden. Zusätzlich gibt es ein Ventil für die Spülung.

4. **Touchscreen** Der Touchscreen steht für die Bedienung des Systems zur Verfügung.

5. **Duftdüse** Die Duftdüse ist der Endpunkt, an dem der Luftstrom das Gerät in Richtung des Probanden verlässt. Wird gerade kein Duft ausgegeben, wird die Duftdüse „gespühlt“ d. h. es wird permanent Luft durch die Düse geblasen um ausgegebene Düfte aus dem Schlauchsystem herauszuspülen.

6. **Kompressor** Der Kompressor ist dafür zuständig, Luft durch die Kartuschen fließen zu lassen und dadurch das Ausströmen der Düfte aus der Duftdüse am Touchscreen zu ermöglichen.

Mit dem verwendeten Olfaktometer kann man Identifikationstests und Diskriminationstests durchführen. Im Gegensatz zum Riechtest mit Sniffin’ Sticks kann man mit dem Olfaktometer keinen Schwellenwerttest durchführen, da es nicht möglich ist, die Konzentration von Gerüchen zu regulieren. Aus diesem Grund kann es den herkömmlichen Riechtest mit Sniffin’ Sticks auch nicht vollständig ersetzen.

Ähnlich zum Ablauf eines Riechtests mit Sniffin' Sticks soll auch der Vorgang beim Olfaktometer sein. Nach dem Einschalten des Olfaktometers erzeugt der Kompressor Druckluft, welche aus der Spüldüse austritt. Die Luft kann durch Spüldüsen oder durch ein Duftröhrchen geleitet werden. Dies geschieht mit Hilfe von Ventilen, die über den Controller geöffnet oder geschlossen werden können. Im Grundzustand sind alle vier Spülventile der Decks geöffnet und es fließt ein beständiger geruchsneutraler Luftstrom durch die Duftdüse. Diesen Zustand nennt man „Spülung“ da damit gewährleistet wird, dass Restpartikel von vorangegangenen Duftausgaben restlos herausgespült werden. Bevor ein Duft ausgegeben werden kann, müssen alle Spüldüsen geschlossen werden, da sonst der Luftstrom durch das gewählte Duftröhrchen und damit die Duftkonzentration an der Duftdüse zu gering wäre. Aus dem gleichen Grund ist es nicht sinnvoll, obwohl technisch möglich, mehrere Duftventile gleichzeitig zu öffnen. Um Gerüche vom Olfaktometer ausgeben zu können, müssen Duftkartuschen in dem verwendeten Deck eingelegt sein. Die Duftkartuschen enthalten jeweils 6 Glasröhrchen mit bestimmten Düften. Diese Glasröhrchen haben definierte Positionen in der Kartusche und können so gezielt über die Ventile angesprochen werden. Dem Controller werden über die serielle Schnittstelle Nachrichten übertragen, mit denen die Geruchsausgabe gesteuert werden kann. Dies erfolgt über das Ansprechen von Ventilen, die entweder geöffnet oder geschlossen werden können.

Nach einer gewissen Anzahl von Duftausgaben lässt die Konzentration der Düfte in den Duftröhrchen nach. Mitunter will man auch neue Duftsätze verwenden. Dann ist es möglich Kartuschen zu tauschen. Über einen Hebel an der Vorderseite des Decks kann eine Verriegelung gelöst werden. Dabei werden die von beiden Seiten in die Duftröhren der Kartuschen eingestochenen Kanülen herausgezogen und eine Klappe geöffnet. Die Kanülen sind für den Luftfluss durch die Duftröhren verantwortlich. Hinter der Klappe befindet sich die Kartusche, welche man nun herausnehmen kann. Bei neuen Kartuschen sind beide Enden der Duftröhren durch einen Gummiüberzug verschlossen. Nach dem Einlegen der neuen Kartusche werden durch Betätigung des Hebels die Kanülen durch den Gummiverschluss in die Duftröhren gestochen.

3.5 Auswertung und Diagnose

Die Auswertung und Diagnose eines Riechtests erfolgt anhand von Vergleichsdaten aus bereits durchgeführten Studien. Dabei werden Grenzwerte für die Anzahl von richtig beantworteten Proben festgelegt. Diese Grenzwerte sind abhängig von der Anzahl der präsentierten Proben, dem Alter und dem Geschlecht des Probanden. Am

Universitätsklinikum Dresden werden üblicherweise 16 Proben präsentiert. Tabelle 3.2 enthält Grenzwerte für den Identifikationstest bei einer Probenanzahl von 16.

Alter	Anzahl richtiger Antworten (16 Proben)	
	weiblich	männlich
5- 15	9	10
16- 35	11	11
36- 55	12	11
>56	9	9

Tab. 3.2: Richtige Antworten in Abhängigkeit von Alter und Geschlecht bei 16 Proben

Probanden die gleich oder mehr richtig beantwortete Proben haben, haben ein normales Riechvermögen. Bei darunter liegenden Werten besteht Verdacht auf eine Riechstörung. Wie bereits in Abschnitt 3.2 erwähnt, handelt es sich dabei nicht um eine endgültige Diagnose, sondern nur um eine erste Vermutung, auf welche weitere Untersuchungen folgen sollten.

Zu beachten ist, dass die Kriterien „Überdeckung durch andere Gerüche“, „Temperatur“ und „Luftfeuchtigkeit“ nicht berücksichtigt werden und damit keinen Einfluss auf die Auswertung von Riechtests haben. Dadurch könnten Ergebnisse verfälscht werden.

3.6 Kritik

Sniffin' Sticks Da Riechtests bereits manuell mit Hilfe der Sniffin' Sticks durchgeführt werden, stellt sich die Frage, warum man auf ein computergestütztes System umstellen will. Im Folgenden werden Kritikpunkte am bisherigen Verfahren beschrieben. Das bisherige Verfahren ist ineffektiv weil:

- eine Schwester dem Probanden die Sniffin' Sticks präsentieren muss,
- die Schwester die Antworten des Probanden manuell in das bisherige System eingeben muss,
- die Ergebnisse im bisherigen System nicht direkt miteinander vergleichbar sind

Eine Schwester muss für Durchführung eines Richtests ca. 20 Minuten vor Ort sein. Wenn der Proband das System selbst bedienen könnte, hätte die Schwester mehr Zeit für andere Arbeiten.

Außerdem wird sich von dem neuen System erhofft, dass möglicht viele Menschen verschiedener Altersgruppen einen Riechtest daran durchführen. Aus den daraus erfassten Daten möchte man genauere Normwerte für die Durchführung solcher Tests erhalten.

Olfaktometer Ein Riechtest mit dem Olfaktometer kann nicht den kompletten Umfang eines Riechtests mit Sniffin' Sticks anbieten, da kein Schwellenwerttest möglich ist. Der Proband schätzt sein eigenes Riechvermögen und den Luftfluss durch die Nase ein, jedoch werden diese Informationen nicht bei der Auswertung der Tests berücksichtigt.

Beide Bei beiden Verfahren wird nicht auf die Faktoren „Temperatur“ und „Luftfeuchte“ Rücksicht genommen, wodurch Ergebnisse verfälscht werden können.

4 Zielsetzung und fachliche Anforderungen

4.1 Kurzbeschreibung

Das zu entwickelnde Softwaresystem für das gegebene Olfaktometer soll im Zielzustand folgende Funktionen erfüllen:

1. Es soll dem Probanden ermöglicht werden, selbstständig einen Riechtest durchzuführen (Abschnitt 4.2.1.1).
2. Es soll dem Arzt oder Wissenschaftler ermöglicht werden, abgeschlossene Riechtests einzusehen und Riechtestdaten nach Alter und Geschlecht auswerten zu können (Abschnitt 4.2.1.2).
3. Es soll dem Administrator ermöglicht werden, Einstellungen am System vorzunehmen (Abschnitt 4.2.1.3).

Anhand dieser allgemeinen Ziele lassen sich detaillierte Anforderungen ableiten.

4.2 Anforderungen an die Software

Zu Beginn der Entwicklung einer Software ist es nötig herauszufinden, welche Anforderungen gestellt werden. Es wird zwischen zwei Gruppen von Anforderungen unterschieden:

- funktionale Anforderungen
- nichtfunktionale Anforderungen

Funktionale Anforderungen beschreiben, was das System leisten soll.

Nichtfunktionale Anforderungen legen fest, was für Eigenschaften das System haben soll.

4.2.1 Funktionale Anforderungen

4.2.1.1 Riechtest

Das Sytem soll dem Probanden ermöglichen, selbständig einen Riechtest durchzuführen. Dazu sollen folgende Schritte eingehalten werden:

1.1 **Eingabe der Stammdaten**

1.1.1 Der Nutzer muss dem System folgende Stammdaten mitteilen können:

- Geschlecht
- Geburtsdatum
- eigene Einschätzung des Riechvermögens
- eigene Einschätzung des Luftflusses in der Nase

1.1.2 Die Einschätzung des eigenen Riechvermögens und des Luftflusses durch die Nase soll der Proband aus folgenden vier Stufen auswählen können:

- sehr gut
- gut
- schlecht
- sehr schlecht

1.1.3 Optional (siehe Abschnitt 4.2.1.3) sollen folgende Daten erfasst werden können:

- Name
- Vorname
- Anschrift (Straße, Hausnummer, Postleitzahl, Wohnort, Staat)

Diese optionalen Daten sollen erfasst werden, um den Riechtest eines Probanden wiederfinden zu können und, sollte dieser Proband schon mehrere Riechtests durchgeführt haben, diese Tests miteinander vergleichen zu können.

1.1.4 Bevor der eigentliche Test gestartet wird, soll geprüft werden, ob die erforderlichen Daten vom Probanden angegeben wurden. Es müssen alle geforderten Felder außer der Anschrift ausgefüllt werden. Sind die Angaben unvollständig, soll folgende Warnung angezeigt werden: „Sie haben vergessen Ihr [...] anzugeben!“.

1.2 **Präsentation von Geruchsproben**

1.2.1 Das System soll dem Probanden nacheinander 16 Geruchsproben präsentieren.

1.2.2 Randbedingungen für die Geruchsausgabe:

1.2.2.1 Zu jeder Probe existiert genau eine richtige Antwort und drei falsche Antwortmöglichkeiten.

1.2.2.2 Der Proband bekommt die vier Antwortmöglichkeiten präsentiert.

1.2.2.3 Der Proband muss die Geruchsausgabe initiieren.

1.2.2.4 Nach der Initiierung soll das System den Geruch drei Sekunden lang ausgegeben.

1.2.2.5 Das System muss dem Probanden ermöglichen eine der vier Antwortmöglichkeiten auszuwählen.

1.2.2.6 Der Proband darf eine Antwort erst geben, wenn der Duft bereits ausgegeben wurde.

1.2.2.7 Das System muss sich nach einer Geruchsausgabe mindestens fünf Sekunden im Spülzustand befinden, bevor eine weitere Geruchsausgabe stattfinden kann.

1.2.2.8 Das System muss dem Probanden ermöglichen den Geruch ein zweites Mal ausgeben zu lassen, aber kein weiteres Mal.

1.2.2.9 Das System muss den Probanden zwingen eine der vier Antworten zu wählen.

1.3 **Beurteilung nach Abschluss des Riechtests**

1.3.1 Nachdem der Proband den Riechtest abgeschlossen hat, soll das System ein Auswertungsdokument erzeugen.

1.3.2 Der Proband soll sich dieses Dokument ausdrucken lassen können.

1.3.3 Folgende Informationen sollen auf dem Dokument enthalten sein:

1.3.3.1 Übersicht über richtige/ falsche Antworten des Tests

1.3.3.2 Stammdaten des Probanden (Name, Vorname, Geschlecht, Alter, Riechvermögen, Luftfluss in der Nase)

1.3.3.3 Das System soll anhand des Riechtests und der voreingestellten Vergleichswerte eine Diagnose stellen (vgl. Tabelle 3.2). Wenn der Wert

des aktuellen Riechtests über dem Vergleichswert liegt soll die Diagnose „Normosmie“ gestellt werden, wenn es darunter liegt soll die Diagnose Verdacht auf „funktionelle Anosmie“ gestellt werden.

1.3.3.4 Grafik in der dargestellt ist, ob der Proband besser oder schlechter als andere Probanden im gleichen Alter ist

1.4 **Riechtest speichern**
Das System soll die Stammdaten des Probanden und den zugehörigen Riechtest in eine Datenbank speichern.

1.5 **Zeitbegrenzung**
Wenn länger als vier Minuten keine Eingabe am System vorgenommen wird, soll das System den Probanden fragen, ob der laufende Test abgebrochen werden soll. Dies gilt während der kompletten Dauer des Riechtests.

4.2.1.2 Auswertung

Es soll dem Administrator ermöglicht werden, die abgeschlossene Riechtests aller Probanden einzusehen, auszuwerten und miteinander zu vergleichen.

2.1 Es soll möglich sein, das Auswertungsdokument eines einzelnen Tests einzusehen.

2.2 Der Administrator soll mehrere Tests unter den Kriterien Alter und Geschlecht miteinander vergleichen können.

2.3 Es soll möglich sein, die Daten aller bisher abgeschlossenen Tests in eine MS Excel- Datei zu exportieren. Folgenden Informationen sollen dabei zu jedem Riechtest exportiert werden:

- Alter
- Geschlecht
- Einschätzung des Riechvermögens
- Einschätzung des Luftflusses in der Nase
- Anzahl richtiger Antworten
- zu jeder Probe
  - gegebene Antwort
  - war diese richtig oder falsch

4.2.1.3 Konfiguration

Der Administrator soll folgende Parameter des Geräts festlegen können:

3.1 ob Name, Vorname oder Anschrift vom Probanden erfasst werden sollen, um mehrere Tests dieses Probanden miteinander vergleichen zu können

3.2 wie die Reihenfolge der Geruchsproben sein soll, um verschieden Geruchsequenzen einstellen zu können

3.3 über welchen Port und an welchem Deck ein bestimmter Duft ausgegeben werden kann

3.4 Der Administrator soll Kartuschenbenschreibungen neu anlegen, bearbeiten und löschen können. Darin sollen folgende Informationen enthalten sein:

- ID der Kartusche (abgedruckt auf der Kartusche), um Kartuschen mit gleichen Düften zu identifizieren
- zu jeder der sechs Duftdüsen:
  - der tatsächliche Geruch
  - drei falsche Antwortmöglichkeiten
- zu jeder Antwortmöglichkeit soll optional ein entsprechendes Bild angegeben werden können

3.5 Außerdem soll der Administrator auch das Administratorenpasswort ändern können.

3.6 Ist zum Programmstart keine Konfigurationsdatei vorhanden, soll folgende Standardkonfiguration geladen werden:

- Passwort: *admin*
- Nutzerdatenerfassung: *keine Erfassung der optionalen Daten*
- Probenreihenfolge: *keine Proben ausgewählt*

4.2.2 Nichtfunktionale Anforderungen

4.1 **Anforderungen an Hard- und Software**

Das System soll in folgender Umgebung betrieben werden:

Hardware

Olfaktometer
Personal Computer inkl. 2 serielle Schnittstellen (EIA-232)
Bildschirm (evtl. Touchscreen)

Software

Microsoft Windows XP™/ Microsoft Windows Vista™
Microsoft .NET Framework 3.5
Microsoft SQL Server 2005 Express Edition™

4.2 Technische Anforderungen und Randbedingungen

4.2.1 Das Olfaktometer schaltet sich 20 Sekunden nach dem letzten Signal in den Grundzustand, d. h. eine Duftausgabe ist nicht länger als 20 Sekunden möglich.

4.2.2 Es muss immer mindestens ein Ventil des Olfaktometers geöffnet sein, damit kein Luftstau entsteht.

4.2.3 Der Controller gibt keine Rückmeldungen, ob ein Signal empfangen wurde oder nicht.

4.2.4 Zwischen dem Senden von Signalen muss mindestens eine Pause von 120 ms eingehalten werden.

4.3 Robustheit

Das System soll eventuell auftretende Fehler abfangen. Fehlerhafte Nutzereingaben sollen abgefangen und dem Nutzer signalisiert werden.

4.4 Sicherheit

Es darf dem Probanden nicht möglich sein Konfigurationen am System vorzunehmen, Daten anderer Probanden einzusehen oder das System zu beenden. Dies ist ausschließlich dem Administrator vorbehalten.

4.5 Bedienbarkeit

Das System soll mittels eines Touchscreens bedient werden können. Die Bedienung mittels Maus und Tastatur soll auch möglich sein.

4.6 GUI

Steuerelemente des GUI müssen groß genug sein um die Bedienung mittels Touchscreen zu ermöglichen. Aus diesem Grund muss auch eine Bildschirmtastatur zur Verfügung gestellt werden.

4.7 Benutzbarkeit

Das System soll vom Probanden intuitiv und ohne vorherige Schulung bedienbar sein.

4.8 **Mehrsprachigkeit**
Die grafische Nutzeroberfläche soll in den Sprachen „Deutsch“ und „Englisch“ zur Verfügung stehen.

4.9 **Mengengerüst**
Es ist mit maximal 10 000 durchgeführten Riechtests/ Probanden im Jahr zu rechnen. Bei 16 Proben sind das im Jahr 160 000 Probenergebnisse.

4.10 **Archivierung**
Die Archivierung von alten Daten soll manuell über ein Backup der Datenbank geregelt werden.

4.3 Abgrenzung

Das System gibt nur einen ersten Anhaltspunkt, ob bei einem Probanden eine Riechstörung vorliegt oder nicht. Für die genaue Ermittlung von Art und Ursache der Riechstörung sind weitere Untersuchungen nötig, welche das System nicht bieten kann (vgl. Abschnitt 3.2).

5 Vorgehen

5.1 Vorgehensmodelle

Für die Entwicklung von Software gibt es verschiedene Vorgehensmodelle. Diese beschreiben den Entwicklungsprozess in zeitlich aufeinanderfolgenden Phasen. Für jede Phase sind inhaltlich zusammenhängende Tätigkeiten definiert. Folgende prominente Vorgehensmodelle werden kurz erklärt:

- Wasserfallmodell
- V-Modell
- Rational Unified Process

Für weitere Vorgehensmodelle und detailiertere Informationen sei auf das Buch „Methodische objektorientierte Softwareentwicklung“[Winter, 2005] verwiesen.

Wasserfallmodell Das Wasserfall ist das älteste und einfachste der drei betrachteten Modelle. Es zeichnet sich aus durch eine klare Abgrenzung von Entwicklungsprozessen in einzelne Phasen.

Phase
Anforderungsermittlung
Entwurf
Implementierung
Test
Inbetriebnahme

Tab. 5.1: Phasen des Wasserfallmodells

Ein Wiederaufnahme von Aktivitäten aus früheren Phasen ist in diesem Modell nicht vorgesehen. Vorteil des Wasserfallmodells ist die Einfachheit und die klare

Abgrenzung der einzelnen Phasen. Es eignet sich aufgrund des relativ geringem Managementaufwands besonders für kleinere Projekte mit wenigen Entwicklern.

Vorraussetzung für das Gelingen eines Projektes nach diesem Modell ist die fehlerfreie und vollständige Durchführung der einzelnen Phasen. Genau das ist aber bei den meisten Projekten unrealistisch. Beispielsweise stellen sich einige Anforderungen an ein System erst im Laufe der fortlaufenden Entwicklung heraus. Die Reaktionsmöglichkeiten auf sich ändernde Anforderungen stellen sich jedoch als schwierig dar. Ein weiterer Kritikpunkt ist das Testen in der letzten Phase. Tests sollten während der kompletten Entwicklungszeit entworfen und durchgeführt werden, um Fehler möglichst frühzeitig zu entdecken. Je später ein Fehler entdeckt wird, desto schwieriger und teurer wird dessen Beseitigung. Eine mögliche Lösung für die angesprochenen Probleme wäre die Erweiterung des Wasserfallmodells um Iterationen, d. h. es kann in bereits abgeschlossenen Phasen zurückgekehrt werden.

V- Modell Bei dem V- Modell handelt es sich um eine Weiterentwicklung des Wasserfallmodells. Dabei werden zu den vorhanden Phasen zusätzliche Phasen für die Qualitätssicherung eingeplant. Zu jeder Phase gibt es zugehörige Tests zur Qualitätssicherung. Die Spezifikation der Testfälle erfolgt immer unmittelbar nach Fertigstellung der Bezugsphase. Dadurch wird diese Bezugsphase noch einmal überprüft und Fehler oder Widersprüche können schnell erkannt werden.

Rational Unified Process Dieses Modell wurde speziell auf die Belange der objektorientierten Softwareentwicklung zugeschnitten. Die Tätigkeiten werden in diesem Modell in Disziplinen unterteilt, welche den Phasen im Wasserfallmodell entsprechen:

Disziplin
Geschäftsprozessmodellierung
Anforderungsermittlung
Entwurf
Implementierung
Test
Inbetriebnahme

Tab. 5.2: Disziplinen des Rational Unified Process

Diese Disziplinen werden in unterschiedlichen Intensitäten auf die Phasen „Konzep-

tion“, „Ausarbeitung“, „Konstruktion“ und „Übergang in den Betrieb“ aufgeteilt. Ziel ist dabei immer eine ausführbare Zwischenversion des Systems zur Verfügung zu haben. Es handelt sich dabei um ein inkrementelles Vorgehen d. h. die Software wird so lange erweitert und verfeinert bis ein zufriedenstellendes Ergebnis erreicht wurde.

Phase
Konzeption
Ausarbeitung
Konstruktion
Übergang in den Betrieb

Tab. 5.3: Phasen des Rational Unified Process

Großer Vorteil dieses Verfahrens ist die gute Parallelisierbarkeit der Aktivitäten. Außerdem können auf bereits lauffähigen Prototypen Tests durchgeführt und eventuelle Fehler zeitig erkannt werden.

Entscheidung In diesem Projekt wurde aus folgenden Gründen das Wasserfallmodell gewählt:

- einfaches Modell
- lineares Vorgehen
- die Phasen sind klar abgegrenzt
- relativ kleines Projekt mit nur einem Entwickler, deshalb auch keine parallelen Aktivitäten notwendig

5.2 Analyse

Während der Entwicklung einer Software müssen Technologieentscheidungen getroffen werden. Diese Entscheidungen hängen von den vorher erhobenen nichtfunktionalen Anforderungen ab (vgl. 4.2.2). In diesem Kapitel ist beschrieben, welche Technologieentscheidungen getroffen wurden und warum die jeweilige Technologie gewählt wurde.

5.2.1 Betriebssystem

In den Anforderungen wurde Windows XP bzw. Windows Vista als Betriebssystem vorgegeben. Das ist darauf zurückzuführen, dass an der HNO-Klinik ausschließlich Windows Systeme im Einsatz sind und das Personal mit diesem System vertraut ist.

5.2.2 Entwicklung

5.2.2.1 Die Softwareplattform

Zu Beginn der Entwicklung muss man entscheiden, auf welche Softwareplattform man aufbauen möchte. Dabei stand die Wahl zwischen den zwei großen Softwareplattformen Java und .NET. Beide Plattformen basieren auf Verwendung einer Laufzeitumgebung („runtime environment“). Eine Laufzeitumgebung lässt Anwendungsprogramme theoretisch unabhängig von der verwendeten Hardware und dem Betriebssystem laufen. Sie stellt gewissermaßen eine zusätzliche Schicht zwischen Programm und Betriebssystem dar. Die Verwendung einer Laufzeitumgebung macht es nötig einen sogenannten Zwischencode (bytecode) zu erzeugen. Dieser Zwischencode wird dann von der Laufzeitumgebung in den jeweiligen Maschinencode umgewandelt und ausgeführt.

Die .NET Plattform, auch .NET Framework genannt, wurde von der Firma Microsoft entwickelt und besteht aus einer Laufzeitumgebung, der Common Language Runtime (CLR), und einer umfangreichen Sammlung von Klassenbibliotheken. Aufgrund der Verwendung einer Laufzeitumgebung ist theoretisch Plattformunabhängigkeit möglich, jedoch wird die Laufzeitumgebung von Microsoft nur für die eigenen Windowssysteme zur Verfügung gestellt. Für andere Betriebsysteme existieren Community-Projekte zur Entwicklung eigener Laufzeitumgebungen wie z.B. das Mono-Projekt und DotGNU. Die Implementierungen hängt zeitlich aber immer dem aktuellen Stand des .NET Frameworks hinterher. Ein Vorteil der .NET Technologie ist die Programmiersprachenunabhängigkeit. Man kann in verschiedensten Sprachen für die .NET Plattform entwickeln. Die gängigsten, weil von Microsoft dafür gefördert, sind Visual Basic .NET und C#.

Die Java Plattform wurde von der Firma Sun Microsystems entwickelt. Sie besteht ebenfalls aus einer Laufzeitumgebung und einer Sammlung von Klassenbibliotheken. Als Java wird gleichzeitig auch die zu verwendende Programmiersprache bezeichnet. Im Gegensatz zur .NET Plattform wird hier nur diese Sprache unterstützt. Vorteil gegebnüber der .NET Plattform ist die reale Betriebssystemunabhängigkeit.

Aus folgenden Gründen wurde die .NET Plattform gewählt:

- es ist keine Betriebssystemunabhängigkeit gefordert, die Software soll ausschließlich auf Windowssystemen laufen
- .NET Anwendungen sind für Windowssysteme optimiert, d. h. die Leistung des Betriebssystems wird besser genutzt
- Test aktueller .NET Technologien wie LINQ und WPF ist Teil dieser Arbeit

5.2.2.2 Die Programmiersprache

Als Programmiersprache wurde C# gewählt, da es neben Visual Basic .NET eine der Hauptsprachen für die .NET Plattform darstellt und damit die vollständige Funktionalität des .NET Frameworks zur Verfügung stellt.

5.2.2.3 Die Entwicklungsumgebung

Die Standardentwicklungsumgebung für die Entwicklung von .NET Anwendungen ist das Microsoft Visual Studio 2008™. Alternativ könnte man auch mit der freien Entwicklungsumgebung SharpDevelop arbeiten. Es wurde sich jedoch für das Visual Studio entschieden, da es einige Tools bereitstellt, die die Produktivität stark verbessern:

- ein komfortabler Debugger
- ein Designer für WPF Oberflächen
- ein Designer für das O/R Mapping
- Generator für Entwicklerdokumentation
- automatische Vervollständigung von Bezeichnern

5.2.3 Die grafische Benutzeroberfläche

Für die .NET Plattform gibt es zwei verschiedene GUI-Konzepte. Das ist zum einen das ältere „Windows Forms“ und zum anderen die relativ neue „Windows Presentation Foundation (WPF) “. Die WPF wurde entwickelt, um Windows Forms als GUI-Konzept für .NET abzulösen. WPF ist komplett in verwaltetem Code („managed code“) geschrieben, Windows Forms dagegen basiert nur auf einem Wrapper um die Windows API, welche aus nichtverwaltetem Code („unmanaged code“) besteht. Das bringt für die WPF den Vorteil eine bessere Plattformunabhängigkeit als Windows

Forms zu bieten. Ein technischer Vorteil der WPF ist, im Gegensatz zu Windows Forms, dass die Technik auf DirektX basiert und damit direkt auf die Grafikhardware (GPU und Grafikspeicher) aufsetzt und somit sowohl den Hauptprozessor als auch den Hauptspeicher entlastet. Zudem basiert das Konzept von WPF darauf, dass die Präsentation und die Geschäftslogik stark voneinander getrennt sind. Dies wird durch die Einführung der sogenannten Auszeichnungssprache eXtensible Application Markup Language (XAML) erreicht.

Weil die WPF einige Vorteile bringt und weil es in Zukunft wahrscheinlich Windows Forms als GUI-Lösung ersetzen wird, wurde die WPF zur Entwicklung der GUI gewählt.

5.2.4 Die Datenhaltung

Das System soll verschiedene Daten wie z. B. Nutzer- und Testdaten persistent speichern (vgl. Anforderung 1.4). Außerdem soll die Konfiguration des Systems gesichert werden können. Folgende drei Möglichkeiten der Datenhaltung wurden betrachtet und analysiert:

- in einem File im XML-Format
- in einem MS Excel™-File
- in einer relationalen Datenbank

XML ist eine Auszeichnungssprache und eignet sich gut zur Darstellung hierarchisch strukturierter Daten. Im Vergleich zu einer relationalen Datenbank lassen sich Beziehungen zwischen Objekten nur relativ umständlich realisieren. Für die Sicherung von Nutzer- und Testdaten eignet es sich nicht, da es wahrscheinlich viele Datensätze geben wird, wodurch die Verwendung eines einzelnen Files unangebracht ist.

Eine weitere Möglichkeit der Datenhaltung wäre ein MS Excel-Dokument. Excel-Dokumente verfügen über einfache Datenbankfunktionen, eignen sich aber auch nicht für die Sicherung von vielen Daten und bieten auch keine einfache Möglichkeit Daten relational darzustellen. Excel-Dokumente werden hauptsächlich für Berechnungen und weniger zur einfachen Datenhaltung verwendet.

Für die Datenhaltung wurden zwei verschiedene Wege gewählt. Wenn relativ große Datenmengen erwartet werden, wird aus folgenden Gründen in einer relationalen Datenbank gespeichert:

- es kann mit großen Datenmengen umgegangen werden
- Relationen lassen sich gut darstellen

Um die Konfiguration des Systems zu sichern wurde aus folgenden Gründen eine XML- Datei verwendet:

- nur wenige Objekte sollen gespeichert werden
- einfache Serialisierung möglich

Erfasste Testdaten sollten im Excel- Format exportiert werden können (Anforderung 2.3).

5.2.4.1 Das Datenbanksystem

Als Database Management System (DBMS) wurde der Microsoft SQL Sever 2005 gewählt. Es handelt sich dabei um ein etabliertes, in der Expressvariante kostenloses, System für das mit dem SQL Sever Management Studio ein sehr gutes Verwaltungstool zur Verfügung steht. Zudem lässt es sich gut mit dem verwendeten Visual Studio 2008 zusammen verwenden.

5.2.5 Die Datenbankanbindung

Für die Kommunikation mit der Datenbank gibt es die .NET-eigene Schnittstelle ADO.NET. Diese ermöglicht den Zugriff auf Datenbanken mit Hilfe von Treibern (Data Provider). Es sind Treiber für unterschiedliche DBMS vorhanden, wodurch der Zugriff auf Daten unabhängig vom zu Grunde liegenden DBMS erfolgen kann. Weil der Microsoft SQL Server verwendet wird, soll hier nur auf die Möglichkeiten des Zugriffes auf dieses System eingegangen werden. Eine Möglichkeit ist der Zugriff über OLE DB- oder ODBC. Sie stellen standardisierte Zugriffstechniken für DBMS dar. Nachteil ist, dass sowohl ein allgemeiner OLE DB- bzw. ODBC- Data Provider (*ODBC .NET Data Provider*/ *OLE DB .NET Data Provider*) sowie ein datenbankspezifischer Treiber (*SQL Server OLE DB Provider*/ *SQL Server ODBC Driver*) benötigt werden. Je mehr Komponenten bzw. Treiber zwischen Programm und Datenbank vorhanden sind, desto länger dauert die Verarbeitung. Deswegen gibt es für den Microsoft SQL Server auch einen eigenen Data Provider (*Microsoft SQL Server .NET Data Provider*), der für den direkten Zugriff auf den MS SQL Server optimiert wurde.

Da es für .NET seit November 2007 eine Technologie namens Language Integrated Query (LINQ) gibt, die die Implementation jeder Art von Datenzugriffen vereinfachen soll, lag es nahe, sich intensiver mit dieser Technologie auseinander zu setzen. LINQ stellt eine einheitliches Verfahren zur Verfügung, wie auf Datenquellen zugegriffen werden soll. Damit ist LINQ für objektrelationales Mapping verwendbar. Der Zugriff

ist jedoch nicht auf relationale Datenbanken beschränkt, sondern bezieht sich auf jegliche Art von Datenquellen. Voraussetzung ist dabei, dass für die gewünschte Datenquelle ein LINQ- Data Provider existiert. Mit *LINQ to SQL* existiert ein Data Provider für den MS SQL Server. Ein großer Vorteil von LINQ gegenüber den anderen Verfahren zum Datenbankzugriff ist die strenge Typisierung, welche bereits während der Kompilierung geprüft wird. In Abschnitt 6.1.2 werden die Konzepte von LINQ genauer erklärt.

Neben LINQ gibt es für objektrelationales Mapping auch Produkte von Drittanbietern wie z. B. *Invist* oder *ObjectMapper .NET*. Es wurde jedoch LINQ gewählt, da eine gute Integration in Visual Studio gegeben und die Technologie sehr gut dokumentiert ist.

5.2.6 Serielle Schnittstelle

Die serielle Schnittstelle dient zum Datenaustausch zwischen Computern und Peripheriegeräten, in unserem Fall zwischen dem PC und dem Controller des Olfaktometers. Die serielle Schnittstelle wird benötigt, um die in Abschnitt 4.2.1.1 behandelte Geruchsausgabe mit Hilfe des Controllers zu steuern. Für die Steuerung von seriellen Schnittstellen stellt die .NET Klassenbibliothek die Klasse `SerialPort` zur Verfügung. Der folgende Codeschnipsel zeigt die Verwendung der `SerialPort`- Klasse.

```
1  SerialPort serialPort = new SerialPort(
2          "COM1",
3            // Bezeichner der seriellen Schnittstelle im System
4          9600,
5            // Baudrate - Anzahl der übertragenen Bits pro
                Sekunde
6          Parity.None,
7            // zur Erkennung von fehlerhaften Signalen (
                deaktiviert)
8          8);
9            // Anzahl Datenbits pro Zeichen (ein Byte)
10
11 serialPort.Open();    // Port öffnen
12
13 if(serialPort.IsOpen)
14 {
15         serialPort.Write("Hello World!"); //Daten schicken
16 }
17
18 serialPort.Close();
```

Listing 5.1: Beispielhafte Verwendung der SerialPort- Klasse

5.2.7 Mehrsprachigkeit

Für die Software ist Mehrsprachigkeit gefordert. Konkret soll das Programm in Deutsch und in Englisch nutzbar sein. Um die Mehrsprachigkeit zu realisieren, wurde auf die „WPF Localisation Extensions“ zurückgegriffen. Diese ermöglichen es sowohl in XAML als auch im Quellcode auf lokalisierte Wörter und Wortgruppen zuzugreifen. Für jede Sprache wird eine eigene Ressourcendatei angelegt. Sie enthält die Wörter und Wortgruppen in der jeweiligen Sprache. Beim Kompilieren des Programms wird für jede Sprache eine sogenannte Satellitenassembly erstellt, welche die lokalisierten Strings enthält. Beim Laden einer Nutzeroberfläche werden die Verweise auf Texte, unter Berücksichtigung einer global verfügbaren Sprachdefinition, über die erstellten Satellitenassemblies aufgelöst. Der Nutzer kann die Sprache zwischen „Deutsch“ und „Englisch“ wechseln.

5.3 Modellierung

Die Modellierung ist eine der Hauptaktivitäten bei der Softwareentwicklung. Aufbauend auf die Problemanalyse und die Definition der Anforderungen wird hier die Grundlage für die Implementierung geschaffen. Dabei geht es um die Beschreibung der Architektur, der Komponenten und der Schnittstellen der Software. Als Ergebnis soll ein vollständiger und konstistenter Plan der zu erstellenden Software entstehen.

5.3.1 Komponentenmodell

Das Komponentenmodell dient zur Aufteilung eines Systems in Subsysteme. Der Vorteil dieses Vorgehens ist, dass Komponenten unabhängig voneinander entwickelt und getestet werden können. Damit die Kommunikation zwischen den Komponenten funktioniert, ist es erforderlich die Schnittstellen zwischen den Komponenten genau zu beschreiben.

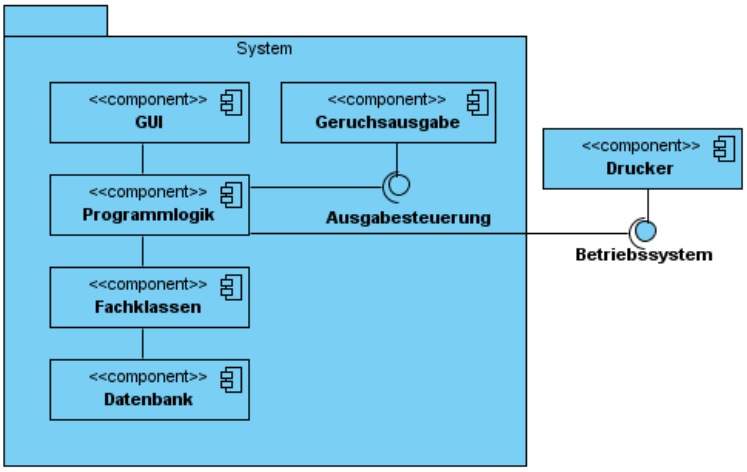


Abb. 5.1: Komponentendiagramm

GUI ist die Präsentationsschicht und definiert die Darstellung der Elemente auf dem Bildschirm.

Programmlogik definiert alle Vorgänge, die für den Programmablauf notwendig sind.

Fachklassen enthält die Klassen, die für die Datenobjekte benötigt werden.

Datenbank hält die zu sichernden Objektdaten.

Geruchsausgabe definiert die Kommunikation mit dem Controller des Olfaktometers.

Drucker druckt Dokumente, die Steuerung erfolgt durch das Betriebssystem.

5.3.2 Schnittstellen

Schnittstellen sind Teile des Systems, die zur Kommunikation mit Personen oder anderen Systemen dienen. In diesem Softwaresystem gibt es Schnittstellen zu folgenden Personen oder externen Systemen:

1. Nutzer der Software (GUI)
2. Datenbank (O/R Mapper)
3. Controller des Olfaktometers
4. Drucker

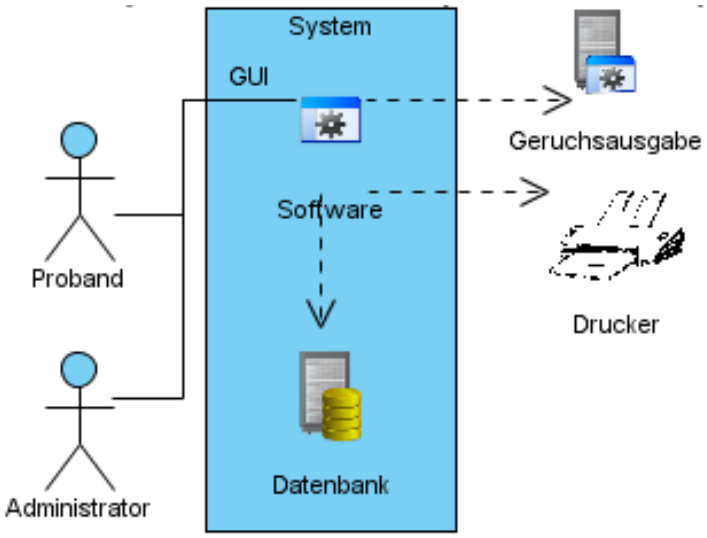


Abb. 5.2: Systemschnittstellen

5.3.2.1 Nutzerschnittstellen

Die Nutzerschnittstelle stellt die Interaktionsebene zwischen einem Menschen und einem Gerät dar. Es gibt in diesem System zwei Nutzergruppen, die unterschieden werden müssen, die Probanden und die Administratoren. Probanden führen am System Riechtests durch. Sie werden im Folgenden allgemein als „Nutzer“ bezeichnet. Administratoren sind laut Anforderungen der Arzt oder Wissenschaftler (vgl. Abschnitt 2.2). Sie können Riechtestdaten auswerten und das System konfigurieren. Beim Entwurf der Nutzerschnittstelle ist darauf zu achten eine einfache, fehlerresistente Bedienung zu gewährleisten.

Im konkreten Fall soll das GUI über einen Touchscreen gesteuert werden. Dabei ist zu beachten, dass Eingabefelder groß genug sein sollen und eine Bildschirmtastatur bei Texteingaben zur Verfügung stehen muss.

5.3.2.2 O/R Mapper

Die Objekte des Programms sollen persistent in einer relationalen Datenbank abgespeichert werden. Um dies zu gewährleisten, wird für die Kommunikation zwischen Programm und Datenbank ein O/R Mapper eingesetzt. Dabei werden die Objektdaten des Programms in die relationale Form der Datenbank gebracht. Der O/R Mapper generiert dabei die nötigen Anweisungen, um Daten in einer Datenbank abzulegen oder abzufragen.

Die Verwendung des gewählten O/R Mappers ist ab Kapitel 6.1.2.1 beschrieben.

5.3.2.3 Schnittstelle zu Controller des Olfaktometers

Um eine Geruchsausgabe anzustoßen, muss mit den Controllern des Olfaktometers kommuniziert werden. Jeder Controller ist jeweils über eine serielle Schnittstelle (EIA-232) mit dem Rechner verbunden. Signale werden blockweise als Hex- Code an den Controller übertragen. Die verfügbaren Befehle sind in Tabelle 5.4 dargestellt.

Befehl		Beschreibung
E0, E1, E2, E3		Aktivierung der LP-Steuerung
EE, EF		alle Ventile in Ruhestellung
1B		Steuerzeichen Blockanfang <ESC>
0D		Steuerzeichen Blockende <CR>
ScentDeck A	ScentDeck B	
40	50	Duftventil 1 auf
41	51	Duftventil 2 auf
42	52	Duftventil 3 auf
43	53	Duftventil 4 auf
44	54	Duftventil 5 auf
45	55	Duftventil 6 auf
A6	B6	Spülventil auf
C0	D0	Duftventil 1 zu
C1	D1	Duftventil 2 zu
C2	D2	Duftventil 3 zu
C3	D3	Duftventil 4 zu
C4	D4	Duftventil 5 zu
C5	D5	Duftventil 6 zu
26	36	Spülventil zu

Tab. 5.4: Befehlsübersicht für Controller

Ein Block besteht jeweils aus einem Start- Tag (`1B`), einem Befehlsteil und einem End-Tag (`OD`). Der Befehlsteil enthält die Symbole für die Ansteuerung der Ventile.

Die Ausgabe eines Duftes vom Olfaktometer ist bereits in Abschnitt 3.4 beschrieben. Folgende Tabelle zeigt beispielhafte Signale, die bei einer Geruchsausgabe durch das Duftventil 1 auf Deck A nötig sind:

Befehl	Beschreibung
`1B E0 E1 E2 E3 0D`	Initialisierungssignal
`1B EE EF 0D`	Alle Ventile in Ruhezustand, Duftventile geschlossen, Spülventile geöffnet
`1B 26 36 40 0D`	Spülventile beider Decks schließen, Duftventil 1 auf Deck A öffnen
`1B A6 C6 B0 0D`	Spülventile beider Decks öffnen, Duftventil 1 auf Deck A schließen

Tab. 5.5: Beispielhafte Signalfolge bei einer Geruchsausgabe

5.3.2.4 Druckerschnittstelle

Die Kommunikation mit dem Drucker ist einfach über einen Systemaufruf gelöst, welchen das WPF-Steuerelement *DocumentViewer* zur Verfügung stellt. Dabei wird ein Druck-Dialog geöffnet, mit welchem man einen Druckauftrag erteilen kann.

5.3.3 Klassendiagramme

5.3.3.1 Domänenmodell

Das Domänenmodell stellt alle beteiligten fachlichen Objekte und ihre Beziehungen untereinander dar. Die Eigenschaften sind erst im Datenmodell angegeben. Die eingekreisten Objekte haben 1-zu-1 Beziehungen untereinander und können so im Datenmodell zu einer Klasse zusammengefasst werden.

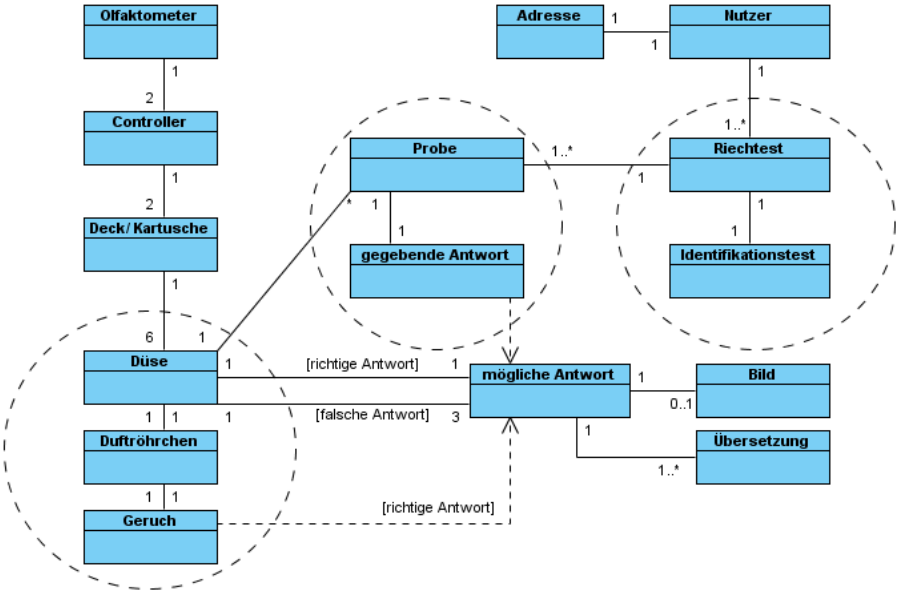


Abb. 5.3: Domänenmodell

5.3.3.2 Datenmodell

Das Datenmodell stellt die aus dem Domänenmodell extrahierten Klassen mit ihren Eigenschaften dar. Wie man sieht, wurden nicht alle fachlichen Objekte tatsächlich für das Datenmodell benötigt. Zusätzlich kamen Datenklassen für die Konfiguration hinzu.

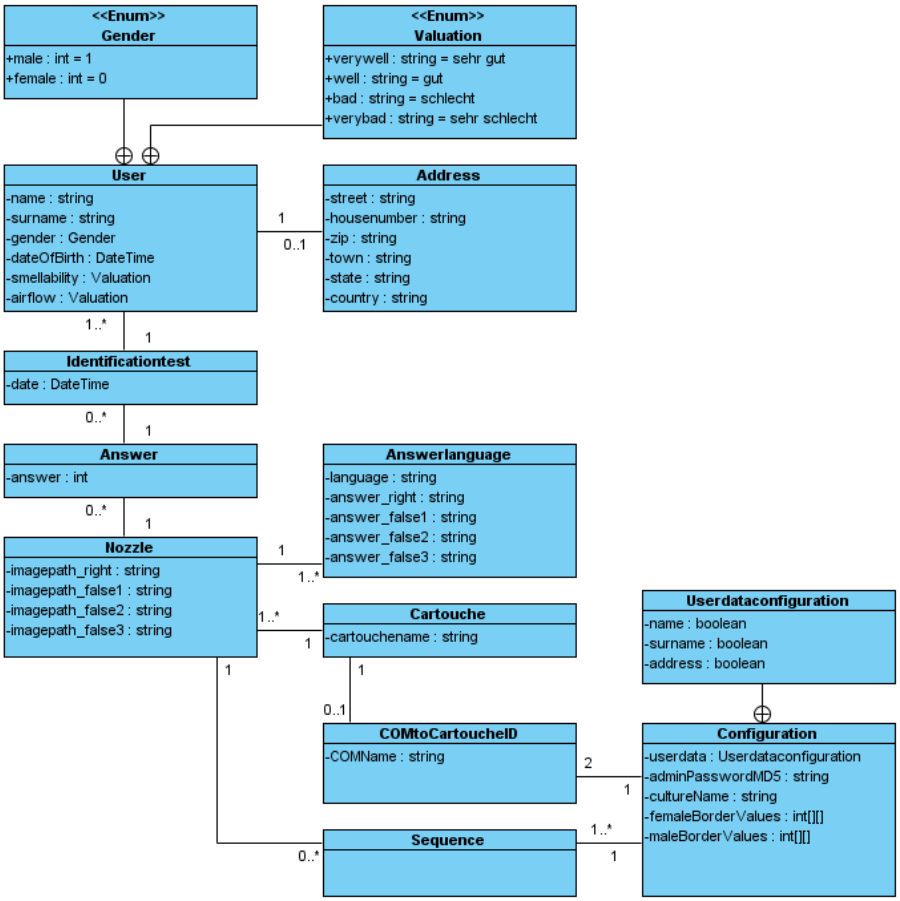


Abb. 5.4: Datenmodell

Folgende Tabelle zeigt die Verbindungen zwischen Domänen- und Datenmodell:

Domänenmodell	Datenmodell
Nutzer	User
Adresse	Address
Riechtest/ Identifikationstest	Identificationtest
Probe/ gegebene Antwort	Answer
Düse/ Duftröhrchen/ Geruch	Nozzle
mögliche Antwort/ Übersetzung	Answerlanguage
Kartusche	Cartouche

Tab. 5.6: Übertragung aus dem Domänen- in das Datenmodell

5.3.4 Anwendungsfälle

Anwendungsfälle sind „Funktionen des Anwendungssystems, die von den Akteuren zur Durchführung ihrer Aufgaben benötigt werden [...]“ [Winter, 2005, Seite 134]. Answendungsfalldiagramme bestehen aus Akteuren und Anwendungsfällen. Jeder Anwendungsfall definiert eine für einen Akteur sichtbare, abgeschlossene Teilfunktionalität des Systems.

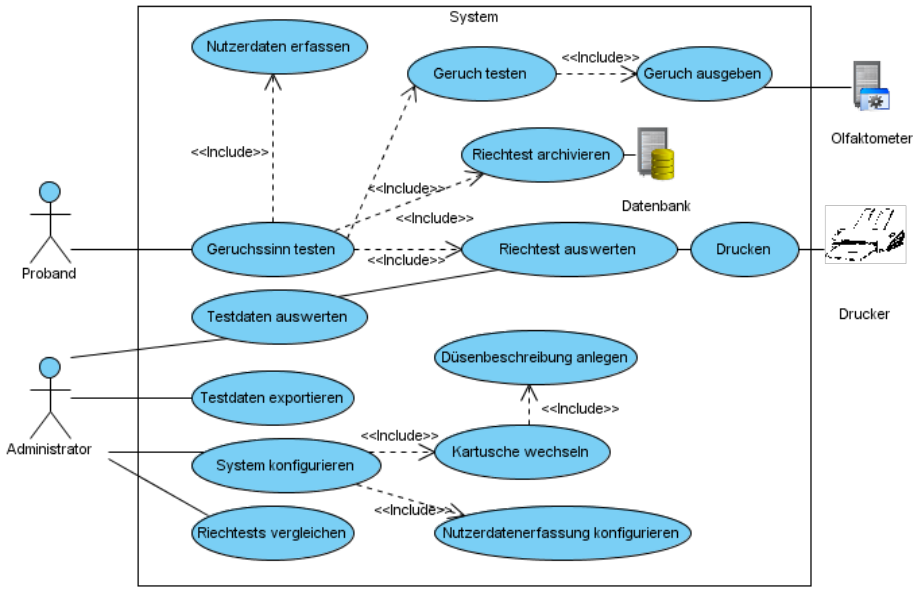


Abb. 5.5: Anwendungsfalldiagramm

Im Folgenden sind, falls vorhanden, zu jedem Anwendungsfall eine Kurzbeschreibung, die beteiligten Akteure, Vorbedingungen, Ergebnis bei Erfolg, Fehlerfälle, Auslöser und Vorgängerprozesse angegeben.

Für die meisten Anwendungsfälle wurden Aktivitätsdiagramme erstellt. Aktivitätsdiagramme dienen zur Beschreibung von Programmabläufen. Eine Aktivität besteht aus mehreren, in einer bestimmten Reihenfolge ablaufender Aktionen (abgerundete Kästen). Jede Aktivität hat einen Start- und einen Endknoten. Die einzelnen Aktionen sind über Kontrollflusskanten (schwarze Pfeile) miteinander verknüpft, um die Ablaufreihenfolge und -richtung zu erkennen. Jede Aktion hat mindestens eine eingehende und eine ausgehende Kontrollflusskante. Optional können Objekte (eckige Kästen) an Aktionen übergeben oder von Aktionen erzeugt werden. Die Verbindungen sind über grüne Kanten signalisiert. Für jede Aktion in den Diagrammen ist in der darunterliegenden Tabelle eine kurze Erklärung abgegeben. Es werden in den Diagrammen zwei Stereotypen verwendet. Der Stereotyp „«global»“ beschreibt die Gesamtheit der Objekte dieses Typs im System. Der Stereotyp „«actor»“ für

Aktionen bedeutet, dass die Aktion vom Nutzer ausgeführt wird.

5.3.4.1 Geruchssinn testen

Bezeichnung	Geruchssinn testen
Kurzbeschreibung	Der Proband teilt dem System seine Stammdaten mit. Das System präsentiert dem Proband Geruchsproben in vorkonfigurierter Reihenfolge. Der Proband teilt dem System seine jeweilige Antwort mit. Das System legt die Stammdaten und Testergebnisse in einer Datenbank ab. Das System erzeugt aus den Stammdaten des Probanden und den Testergebnissen ein Auswertungsdokument. Der Proband kann das Auswertungsdokument vom System drucken lassen.
Akteure	System, Proband
Vorbedingungen	• beide Controller angeschlossen • Probensequenz festgelegt • in Probensequenz verwendete Kartuschen vorhanden Sind diese Bedingungen nicht erfüllt, wird eine Fehlermeldung angezeigt und der Test wird nicht gestartet.
Erfolg	Der Proband erhält Informationen über den Zustand seines Riechvermögens.
Fehlerfälle	Ein Duftröhrchen einer Kartusche ist leer. Controller reagiert nicht auf Signale.
Auslöser	Proband möchte oder soll seinen Geruchssinn testen.
Vorgängerprozess	-

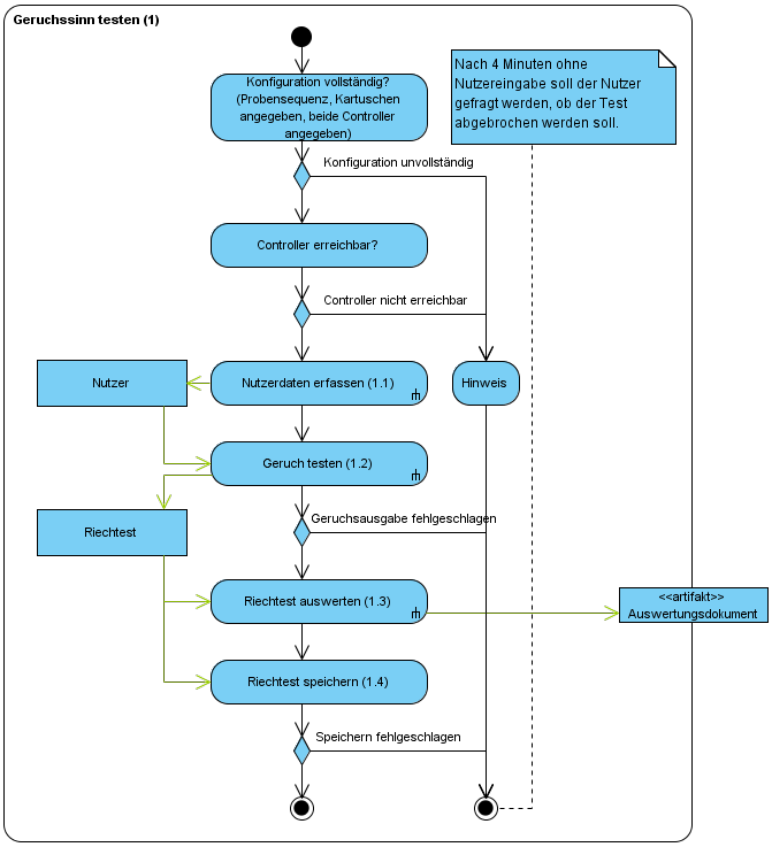


Abb. 5.6: Aktivitätsdiagramm „Geruchssinn testen“

Aktion	Beschreibung
Nutzerdaten erfassen	Der Proband teilt dem System seine Stammdaten mit und das System legt ein neues Nutzerobjekt an.
Geruch testen	Das System legt ein neues Riechtestobjekt an und testet das Riechvermögen des Nutzers.
Riechtest auswerten	Das System erzeugt ein Auswertungsdokument des abgeschlossenen Riechtests.
Riechtest speichern	Das System speichert das Riechtestobjekt in der Datenbank.

5.3.4.2 Nutzerdaten erfassen

Bezeichnung	Nutzerdaten erfassen
Kurzbeschreibung	Der Proband teilt dem System seine Stammdaten mit.
Akteure	Proband
Vorbedingungen	-
Erfolg	Das System kennt die Stammdaten des Probanden.
Fehlerfälle	Wenn Nutzereingaben unvollständig sind, soll eine Warnung angezeigt und der Test nicht gestartet werden.
Auslöser	Das System benötigt Daten über den Probanden um den Riechtest zuordnen und eine Auswertung erzeugen zu können.
Vorgängerprozess	Geruchssinn testen

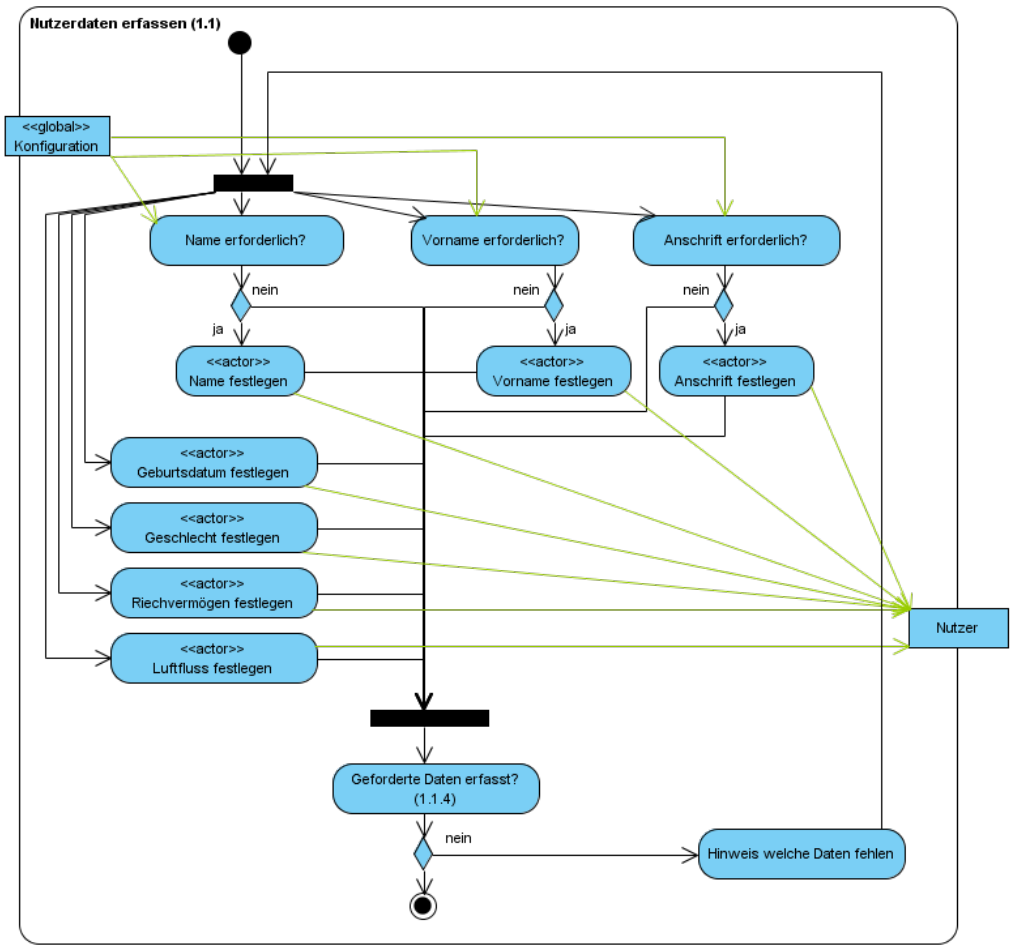


Abb. 5.7: Aktivitätsdiagramm „Nutzerdaten erfassen“

Aktion	Beschreibung
„Element“ erforderlich	Das System ermittelt aus der Konfiguration, ob das jeweilige Element vom Probanden abgefragt werden soll.
„Element“ festlegen	Der Proband teilt dem System den Wert für das jeweilige Element mit und legt es in einem neuen Nutzerobjekt ab.
Geforderte Daten erfasst?	Das System prüft, ob Name, Vorname, Geburtsdatum, Geschlecht, Riechvermögen und Luftfluss vom Probanden angegeben wurden. Falls nicht, wird der Riechtest nicht begonnen und der Proband wird aufgefordert die Daten zu ergänzen.

5.3.4.3 Geruch testen

Bezeichnung	Geruch testen
Kurzbeschreibung	Das System präsentiert dem Proband Geruchsproben in vorkonfigurierter Reihenfolge. Der Proband teilt dem System seine jeweilige Antwort mit.
Akteure	Proband
Vorbedingungen	Probensequenz festgelegt (geprüft durch Vorgängerprozess)
Erfolg	Der Proband hat zu jeder Geruchsprobe eine Antwort gegeben.
Fehlerfälle	• Ob ein verwendetes Duftröhrchen einer Kartusche leer ist, kann nicht durch das System erkannt werden. • Wenn der Proband keine Antwort gibt, wird der Test nach 4 Minuten ohne Eingaben abgebrochen.
Auslöser	Test des Geruchs ist Bestandteil des Riechtests
Vorgängerprozess	Geruchssinn testen

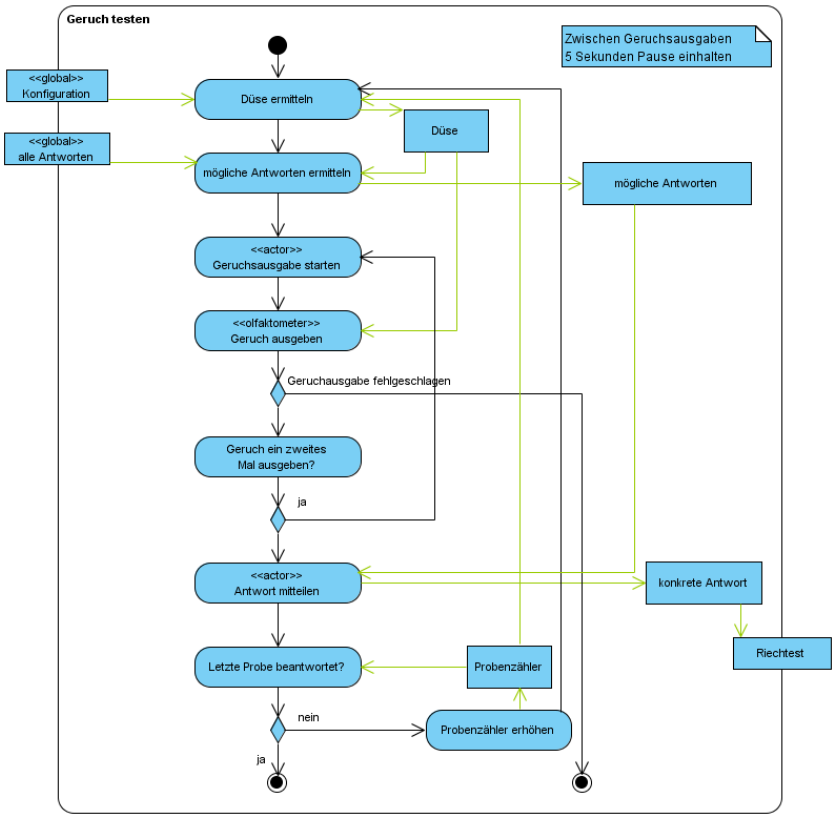


Abb. 5.8: Aktivitätsdiagramm „Geruch testen“

Aktion	Beschreibung
Düse ermitteln	Das System ermittelt anhand der Konfiguration und des Probenzählers, welche Düse geöffnet werden muss.
mögliche Antworten ermitteln	Das System ermittelt anhand der Düse alle für diese Probe möglichen Antworten (4 = 3 Falsche und 1 Richtige).
Geruchsausgabe starten	Der Nutzer teilt dem System mit, dass der Geruch ausgegeben werden soll.
Geruch ausgeben	Das System gibt den Geruch der aktuellen Probe aus.
Antwort mitteilen	Der Nutzer teilt dem System eine Antwort aus den möglichen Antworten mit.
Letzte Probe beantwortet?	Das System prüft, ob die letzte Probe abgeschlossen wurde, indem die Anzahl der Proben mit dem Probenzähler verglichen wird. Wenn ja, wird der Test abgeschlossen. Wenn nicht, wird der Probenzähler erhöht.
Probenzähler erhöhen	Das System erhöht den Probenzähler und lädt die nächste Probe.

5.3.4.4 Geruch ausgeben

Bezeichnung	Geruch ausgeben
Kurzbeschreibung	Das System lässt den zur aktuellen Probe gehörenden Geruch vom Olfaktometer ausgeben.
Akteure	System
Vorbedingungen	• beide Controller angeschlossen • den Duft enthaltende Kartuschen vorhanden (geprüft durch Vorgängerprozess)
Erfolg	Olfaktometer gibt den Geruch an der Duftdüse aus.
Fehlerfälle	• Ob angesprochenes Duftröhrchen leer ist, kann nur manuell festgestellt werden. • Sollte der Controller nicht erreichbar sein, wird eine Fehlermeldung angezeigt.
Auslöser	Geruch ausgeben ist Bestandteil von Geruch testen.
Vorgängerprozess	Geruch testen

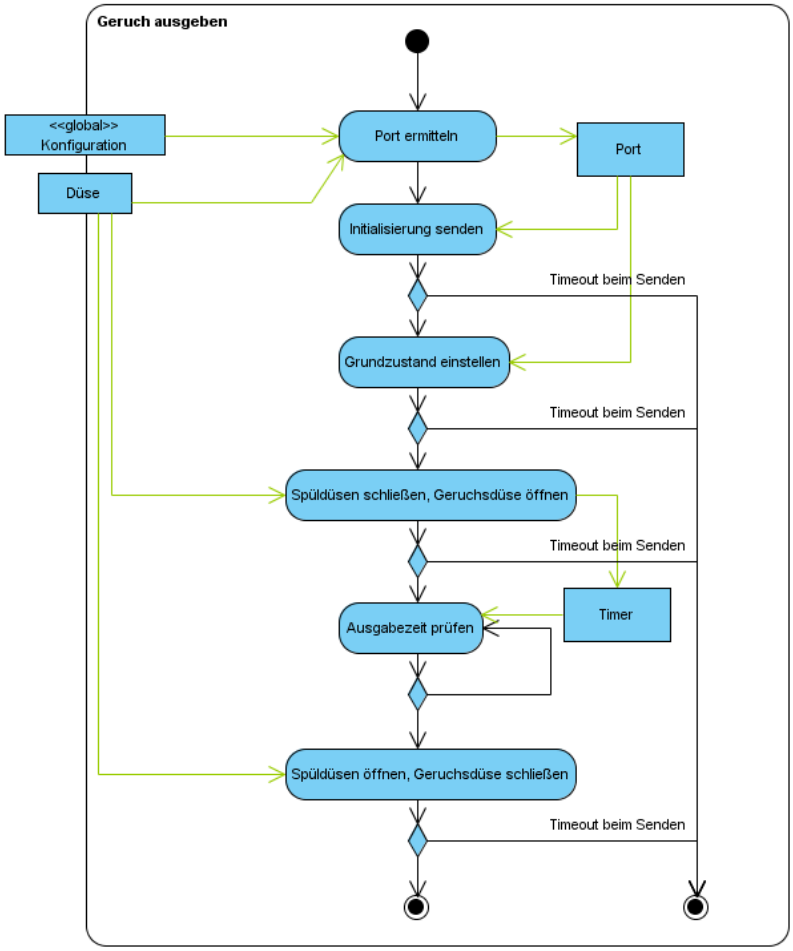


Abb. 5.9: Aktivitätsdiagramm „Geruch ausgeben“

Aktion	Beschreibung
Port ermitteln	Das System ermittelt anhand der Konfiguration welcher Port und damit welcher Controller angesprochen werden muss, um die gewünschte Düse zu öffnen.
Initialisierung senden	Das System sendet eine Initialisierungsnachricht an den entsprechenden Port. Damit wird der daran lauschende Controller empfangsbereit gestellt.
Grundzustand einstellen	Das System sendet ein Signal an den Controller, um den Grundzustand für alle Düsen herzustellen. Damit werden alle Spülventile geöffnet und alle Duftventile geschlossen.
Spüldüsen schließen, Geruchsdüse öffnen	Das System sendet das Signal zum Schließen der Spülventile und zum Öffnen des angegebenen Duftventils.
Ausgabezeit prüfen	Das System steuert die Zeit der Geruchsausgabe über einen Timer.
Spüldüsen öffnen, Geruchsdüse schließen	Das System öffnet alle Spülventile wieder und schließt das Duftventil.

5.3.4.5 Riechtest speichern

Bezeichnung	Riechtest speichern
Kurzbeschreibung	Das System speichert den abgeschlossenen Riechtest, um diesen auswerten zu können.
Akteure	System
Vorbedingungen	Der Riechtest wurde vom Probanden abgeschlossen.
Erfolg	Das System hat die Stammdaten des Probanden und den abgeschlossenen Riechtest auf der Datenbank gespeichert und die Daten stehen für Auswertungen zur Verfügung.
Fehlerfälle	-
Auslöser	Der Proband hat die letzte Probe beantwortet und damit den Riechtest abgeschlossen.
Vorgängerprozess	Geruchssinn testen

5.3.4.6 Riechtest auswerten

Bezeichnung	Riechtest auswerten
Kurzbeschreibung	Das System präsentiert dem Nutzer ein Auswertungsdokument, welches aus den Stammdaten des Probanden, den beim Riechtest ermittelten Daten und aus Vergleichsdaten für den Riechtest erzeugt wurde.
Akteure	System
Vorbedingungen	Das System hat den Riechtest in der Datenbank gespeichert.
Erfolg	Der Nutzer erhält Informationen über seinen Riechtest.
Fehlerfälle	-
Auslöser	Der Proband hat die letzte Probe beantwortet und damit den Riechtest abgeschlossen.
Vorgängerprozess	Geruchssinn testen

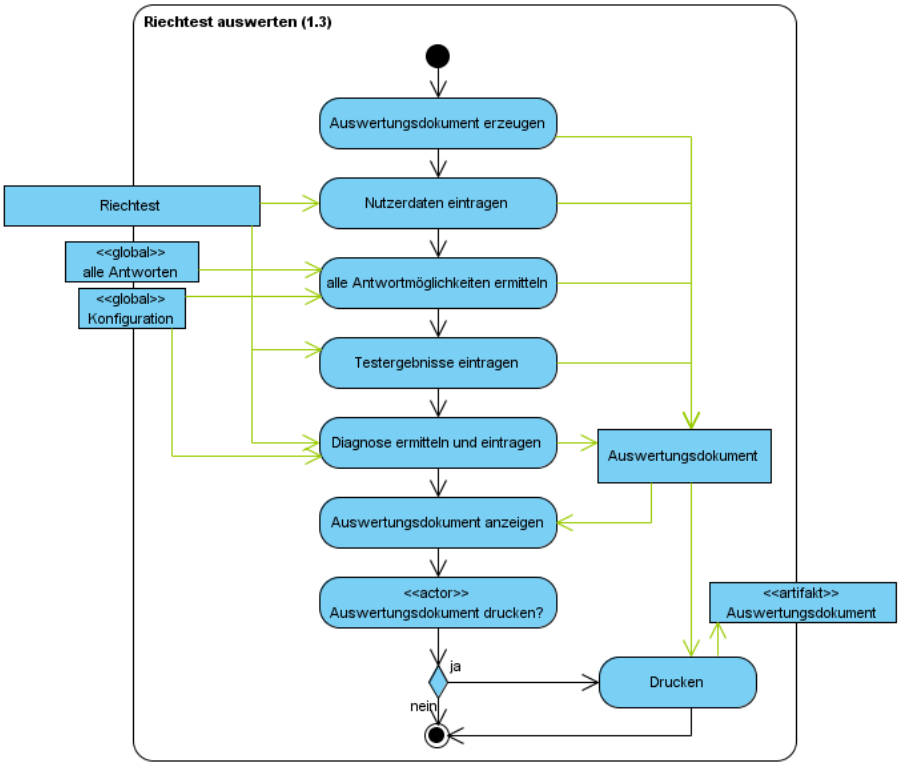


Abb. 5.10: Aktivitätsdiagramm „Riechtest auswerten“

Aktion	Beschreibung
Auswertungsdokument erzeugen	Das System erzeugt ein Template für das Auswertungsdokument.
Nutzerdaten eintragen	Das System trägt anhand des Riechtests die Nutzerdaten in das Dokument ein.
alle Antwortmöglichkeiten ermitteln	Das System ermittelt über die Konfiguration aus allen verfügbaren Antwortmöglichkeiten, welche Antwortmöglichkeiten für diesen Test relevant sind. Die relevanten Antwortmöglichkeiten werden in das Dokument eingetragen.
Testergebnisse eintragen	Das System markiert anhand des abgeschlossenen Riechtests die ausgewählten Antworten in allen Antwortmöglichkeiten.
Diagnose ermitteln und eintragen	Das System stellt anhand von in der Konfiguration vorhandenen Vergleichsdaten und den Testergebnissen eine Diagnose und trägt diese in das Dokument ein.
Auswertungsdokument anzeigen	Das System bringt das Auswertungsdokument zur Anzeige.
Auswertungsdokument drucken?	Der Nutzer kann dem System mitteilen, ob er das Auswertungsdokument drucken möchte.
Drucken	Das System erstellt einen Druckauftrag für das Auswertungsdokument.

5.3.4.7 Drucken

Bezeichnung	Drucken
Kurzbeschreibung	Das System druckt das erzeugte Auswertungsdokument.
Akteure	Drucker, Proband, Administrator
Vorbedingungen	• Auswertungsdokument wurde erzeugt • Drucker ist angeschlossen
Erfolg	Auswertungsdokument wird gedruckt
Fehlerfälle	-
Auslöser	Nutzer möchte Auswertungsdokument drucken
Vorgängerprozess	Riechtest auswerten

5.3.4.8 Testdaten auswerten

Bezeichnung	Testdaten auswerten
Kurzbeschreibung	Der Administrator wählt einen Riechtest aus allen bisher vorhanden Riechtests aus und lässt sich das Auswertungsdokument wie in Abschnitt 5.3.4.6 erzeugen.
Akteure	Administrator
Vorbedingungen	Es wurde mindestens ein Riechtest abgeschlossen.
Erfolg	Auswertungsdokument wurde erzeugt und eventuell gedruckt.
Fehlerfälle	-
Auslöser	Administrator möchte Auswertungsdokument für bereits abgeschlossenen Riechtest erzeugen lassen.
Vorgängerprozess	-

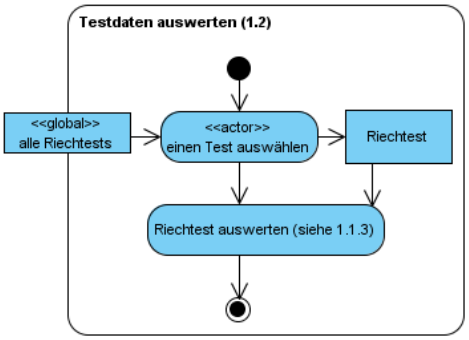


Abb. 5.11: Aktivitätsdiagramm „Testdaten auswerten“

Aktion	Beschreibung
einen Test auswählen	Der Administrator wählt aus der Liste mit allen vorhandenen Riechtests einen Test aus.
Riechtest auswerten	Das System erstellt für den ausgewählten Riechtest ein Auswertungsdokument und zeigt es an (vgl. Abschnitt 5.3.4.6).

5.3.4.9 Testdaten exportieren

Bezeichnung	Testdaten exportieren
Kurzbeschreibung	Das System soll alle bisher abgeschlossenen Tests in ein Excel- File exportieren können, damit diese für weiterführende Untersuchungen zur Verfügung zu stehen.
Akteure	System
Vorbedingungen	Es wurde mindestens ein Riechtest abgeschlossen.
Erfolg	Ein Excel- File wurde erstellt.
Fehlerfälle	-
Auslöser	Administrator möchte Riechtests exportieren
Vorgängerprozess	-

5.3.4.10 Kartusche wechseln

Bezeichnung	Kartusche wechseln
Kurzbeschreibung	Der Administrator teilt dem System mit, welche Kartusche an welchem Deck über welchen Port erreichbar ist. Zu unterscheiden ist der physikalische und der systemeigene Vorgang des Wechselns einer Kartusche. Wird eine Kartusche mit den gleichen Düften, d. h. der Kartuschentyp bleibt gleich, nur ausgetauscht, dann muss keine Änderung im System vorgenommen werden. Wird jedoch eine Kartusche eines anderen Typs eingelegt, dann muss das der Administrator dem System mitteilen.
Akteure	Administrator
Vorbedingungen	-
Erfolg	System weiß über die Position der Kartusche, wo sich ein bestimmter Duft befindet.
Fehlerfälle	-
Auslöser	Administrator möchte Kartusche wechseln und dies dem System mitteilen.
Vorgängerprozess	-

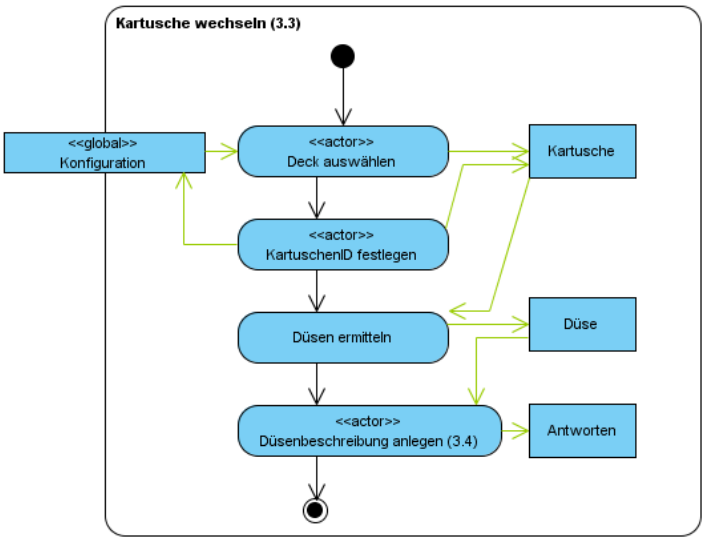


Abb. 5.12: Aktivitätsdiagramm „Kartusche wechseln“

Aktion	Beschreibung
Deck auswählen	Der Administrator bestimmt das Deck, in welches eine neue Kartusche eingelegt werden soll. Das System legt ein neues Kartuschenobjekt an.
KartuschenID festlegen	Der Administrator legt eine KartuschenID für die Kartusche fest. Es sollte die auf der Vorderseite der Kartusche angegebene Bezeichnung sein, um die Kartusche besser zuordnen zu können.
Düsen ermitteln	Das System legt für die Kartusche sechs Düsenobjekte an.
Düsenbeschreibung anlegen	Der Administrator legt für jede der sechs Düsen Antwortmöglichkeiten fest.

5.3.4.11 Düsenbeschreibung anlegen

Bezeichnung	Düsenbeschreibung anlegen
Kurzbeschreibung	Der Administrator muss eine Düsenbeschreibung anlegen, wenn eine neue Kartusche mit bisher unbekanntem Typ verwendet werden soll. Der Administrator legt für jede Düse dieser Kartusche die richtige und drei falsche Antwortmöglichkeiten fest. Optional kann er jeder Antwortmöglichkeit ein Bild zuordnen.
Akteure	Administrator
Vorbedingungen	Kartusche wurde angelegt
Erfolg	Antwortmöglichkeiten und Bilder wurden der Düse zugeordnet.
Fehlerfälle	-
Auslöser	Der Administrator möchte eine Kartusche verwenden, deren Typ bisher unbekannt war.
Vorgängerprozess	Kartusche wechseln

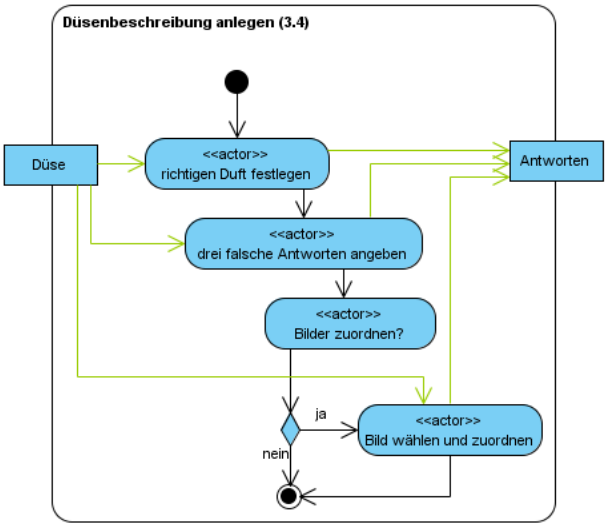


Abb. 5.13: Aktivitätsdiagramm „Düsenbeschreibung anlegen“

Aktion	Beschreibung
richtigen Duft festlegen	Der Administrator teilt dem System die Bezeichnung des richtigen Duftes mit.
drei falsche Antworten angeben	Der Administrator gibt drei weitere falsche Antwortenmöglichkeiten an.
Bilder zuordnen?	Der Administrator hat die Möglichkeit jeder möglichen Antwort ein Bild zuzuordnen.
Bild wählen und zuordnen	Der Administrator wählt ein Bild vom Filesystem und ordnet es einer möglichen Antwort zu.

5.3.4.12 Nutzerdatenerfassung konfigurieren

Bezeichnung	Nutzerdatenerfassung konfigurieren
Kurzbeschreibung	Der Administrator legt fest, ob Name, Vorname und Anschrift des Probanden erfasst werden sollen.
Akteure	Administrator
Vorbedingungen	-
Erfolg	In der Konfiguration wurde gesichert, welche Nutzerdaten erfasst werden sollen.
Fehlerfälle	-
Auslöser	Administrator möchte festlegen welche Nutzerdaten erfasst werden sollen.
Vorgängerprozess	(Kindprozess von „System konfigurieren“)

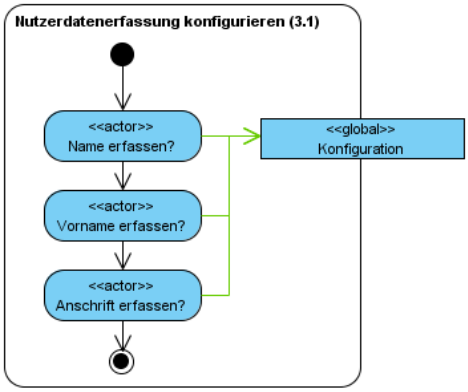


Abb. 5.14: Aktivitätsdiagramm „Nutzerdatenerfassung konfigurieren“

Aktion	Beschreibung
„Element“ erfassen?	Der Administrator legt fest, ob das entsprechende Element bei der Nutzerdatenerfassung mit erfasst werden soll oder nicht. Die Informationen werden in der Konfiguration abegelegt.

5.3.4.13 Riechtests vergleichen

Bezeichnung	Riechtests vergleichen
Kurzbeschreibung	Der Administrator kann mehrere Riechtests in einem Diagramm miteinander vergleichen. Er wählt aus der Liste aller vorhandenen Riechtests mehrere Tests aus. Das System erzeugt ein Auswertungsdiagramm und trägt die Ergebnisse (Anzahl der richtigen Antworten in Abhängigkeit vom Alter) der ausgewählten Tests ein. An den Markierungen für die Tests sollen, falls vorhanden, der Name und Vorname des Probanden sowie das Datum des Tests angezeigt werden.
Akteure	Administrator, System
Vorbedingungen	Mehrere Riechtests sind vorhanden.
Erfolg	Das System hat das Auswertungsdokument erzeugt und angezeigt.
Fehlerfälle	-
Auslöser	Administrator möchte mehrere Riechtests miteinander vergleichen.
Vorgängerprozess	-

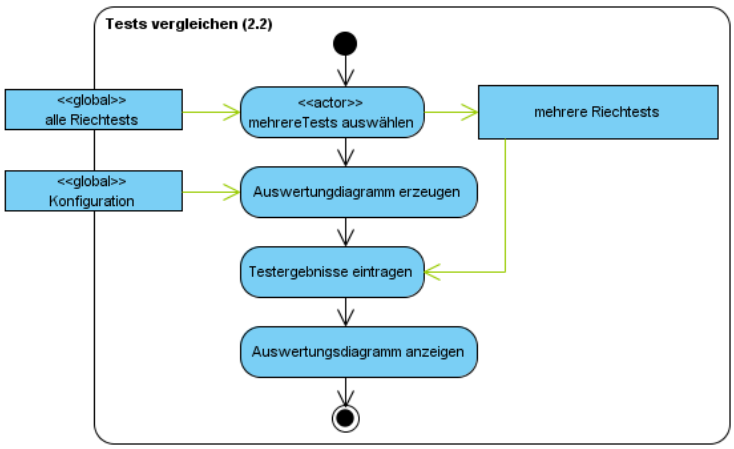


Abb. 5.15: Aktivitätsdiagramm „Riechtests vergleichen“

Aktion	Beschreibung
mehrere Tests auswählen	Der Administrator wählt aus allen Riechtests einen oder mehrere Riechtests aus.
Auswertungsdiagramm erzeugen	Das System erzeugt anhand der Daten, mit denen auch die Diagnose erstellt wird, ein Diagramm, in dem die Anzahl der richtigen Probenantworten in Abhängigkeit vom Alter dargestellt ist.
Testergebnisse eintragen	Das System trägt die Anzahl der richtigen Antworten in Abhängigkeit vom Alter für jeden Test in das Diagramm ein.
Auswertungsdiagramm anzeigen	Das System präsentiert dem Administrator das fertige Auswertungsdiagramm.

5.3.5 Maskenbeschreibungen

Tabelle 5.7 gibt eine Übersicht über die verwendeten Masken.

Bezeichnung	Aktionen
Anmeldemaske	Der Nutzer kann • die Sprache zwischen „Deutsch“ und „Englisch“ wechseln • den Riechtest starten Der Administrator kann zur Administrationsmaske wechseln (nach Passwortabfrage).
Nutzerdatenerfassung	Der Nutzer kann seine Stammdaten angeben und anschließend den Test starten.
Riechtest	Der Nutzer kann einen Geruch ausgeben lassen, eine Antwort auswählen und diese bestätigen.
Riechtestauswertung	Der Nutzer kann das Auswertungsdokument drucken und den Test beenden (zur Anmeldemaske zurückkehren)
Administrationsmaske	Der Administrator kann zu den Masken „Testdatenauswertung“, „Konfiguration“ und „Anmeldemaske“ wechseln oder das Programm beenden.
Testdatenauswertung	Der Administrator kann sich Auswertungsdokumente von abgeschlossenen Tests anzeigen lassen und mehrere Tests miteinander Vergleichen.
Konfiguration	Der Administrator kann die in Abschnitt 4.2.1.3 beschriebenen Einstellungen vornehmen.
Probensequenz	Der Administrator kann die Reihenfolge der Proben festlegen.

Tab. 5.7: Übersicht über Masken des Programms

Diagramm 5.16 zeigt, wie die entsprechenden Masken zu erreichen sind.

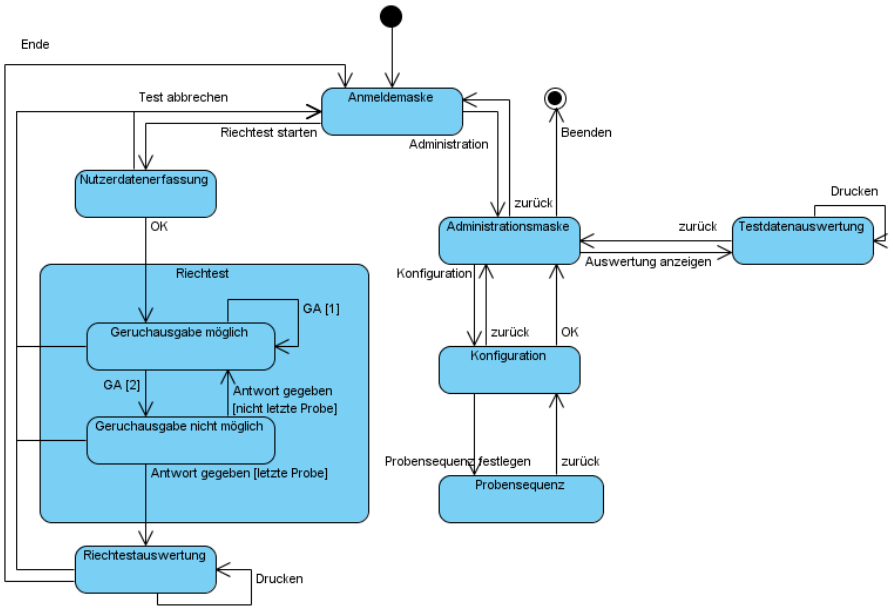


Abb. 5.16: Masken

6 Umsetzung

6.1 Verwendete Technologien

6.1.1 Die Windows Presentation Foundation

Im Abschnitt 5.2.3 wurde erklärt, was die WPF ist und warum diese als Technologie für die grafische Oberfläche verwendet wurde. Dieser Abschnitt befasst sich näher mit den Konzepten und einer beispielhaften Anwendung.

6.1.1.1 XAML

XAML ist eine auf XML basierende Beschreibungssprache für grafische Oberflächen in einer .NET Umgebung. Sie basiert auf dem statischen Beschreiben der Oberflächenelemente eines GUI. Der Vorteil gegenüber einem prozeduralem Ansatz ist die bessere Lesbarkeit des Codes und die einfachere Trennung von Logik und Anzeige. Für dynamische Oberflächen ist es alternativ möglich Teile des GUI in C#- Code zu implementieren. Sobald man Elemente verschachtelt, ist dieser Code jedoch wesentlich schwieriger zu lesen als der XAML- Code.

Eine XAML- Anwendung besteht immer aus einer XAML- Datei und einer sogenannten Codebehind- Datei. In der XAML- Datei wird das Aussehen und die Positionen der Steuerelemente bestimmt. In der Codebehind- Datei können Methoden und Ereignisse beschrieben werden. Die XAML- Datei wird zu einer BAML- Code und das Codebehind- File in CIL- Code kompilliert. Beide Teile werden wie im Bild 6.1 zusammen in einer Assembly abgelegt. BAML bildet XAML in einem binären Format ab.

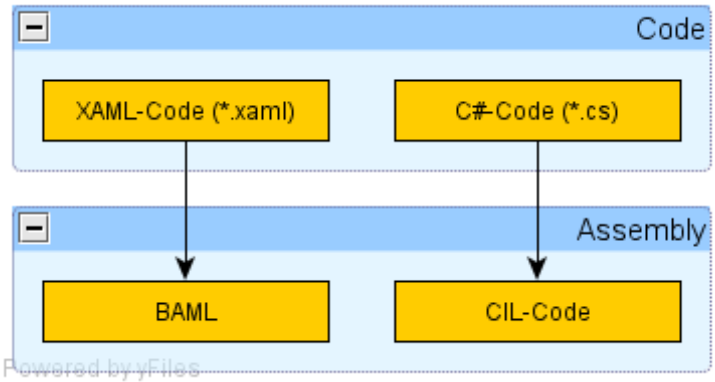


Abb. 6.1: Kompilierung von XAML und Codebehind-Datei zur Assembly

Folgendes Beispiel beschreibt ein Fenster mit einem Button darauf. Das Beispiel wurde sowohl in C# als auch in XAML implementiert. Beide Listings beschreiben exakt das gleiche Fenster.

```
class Window1 : Window
{
    public Window1()
    {
        Button button1 = new Button();
        button1.Height = 30;
        button1.Width = 100;
        button1.Content = "OK";

        Grid grid1 = new Grid();
        grid1.Children.Add(button1);

        this.SizeToContent = SizeToContent.WidthAndHeight;
        this.Content = grid1;
    }
}
```

Listing 6.1: WPF Beispiel in C#

```
<Window
    x:Class="wpf_example.Window1"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/
       presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Window1"
    SizeToContent="WidthAndHeight">
    <Grid>
        <Button Height="30" Width="100">OK</Button>
    </Grid>
</Window>
```

Listing 6.2: WPF Beispiel in XAML

Es folgen einige Erklärungen zum XAML- Code.

Zeile 2 Definiert den Namensraum und den Klassennamen und ist erforderlich, um die Verbindung zwischen Codebehind- und XAML- Datei herzustellen

Zeile 3/4 Stellt die Standardnamensräume für Steuerelemente zur Verfügung.

Zeilen 7-9 Das `Grid`- Element ist eine Layoutumgebung, in welche der `Button` eingebettet wurde.

6.1.1.2 Datenbindung (Data Binding)

Datenbindung bedeutet, dass zwei Eigenschaften aneinander gebunden werden. Bei der Entwicklung grafischer Oberflächen wird die Eigeschaft eines Datenobjekts an eine Eigenschaft eines grafischen Steuerelements gebunden. Durch diese Bindung können die Werte synchronisiert werden. Um in WPF Datenbindung zu ermöglichen, wurden sogenannte *Dependency Properties* (abhängige Eigenschaften) für grafische Elemente eingeführt. Die eigentliche Bindung erfolgt über das `Binding`- Objekt (`System.Windows.Data.Binding`), welches zwei Eigenschaften aneinander binden kann.

Um die Datenbindung zu veranschaulichen, folgt nun ein kleines Beispiel in XAML. In Listing 6.3 wird eine `TextBox` zur Eingabe eines Textes und ein `TextBlock` zur Anzeige des eingegebenen Textes erstellt. Nun kann, wie in Zeile 7 zu sehen, die `Text`- Eigenschaft des `TextBlocks` an die `Text`- Eigenschaft der `TextBox` gebunden werden.

```
1 <Window x:Class="DataBindingWPFElements.Window1"
2     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/
        presentation"
3     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4     Title="Window1" Width="200">
5     <StackPanel>
6         <TextBox Name="txtInput" />
7         <TextBlock Text="{Binding␣ElementName=txtInput,␣Path=
            Text}" />
8     </StackPanel>
9 </Window>
```

Listing 6.3: Bindung zweier WPF Elemente

Während der Eingabe eines Textes in die `TextBox` wird nun der Text gleichzeitig in dem `TextBlock` angezeigt:

Dieser Mechanismus wird dadurch möglich, dass die `Text`- Eigenschaft der `TextBox` eine *Dependency Property* ist, weshalb Änderungen weitergereicht werden.

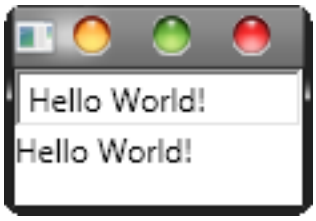


Abb. 6.2: Durch Datenbindung bewirkt die Änderung in TextBox sofortige Änderung in TextBlock

Datenbindung zu selbsterstellten Klassen Eigene Klassen haben keine *Dependency Properties*. Obwohl es die Möglichkeit gibt *Dependency Properties* in selbsterstellten Klassen zu implementieren, ist dieses Vorgehen nicht zu empfehlen, da es speziell für grafische Elemente und nicht für reine Datenklassen vorgesehen ist. Um Datenbindung trotzdem zu ermöglichen, muss die Schnittstelle `System.ComponentModel.INotifyPropertyChanged` implementiert werden. Die Schnittstelle definiert das `PropertyChanged`- Ereignis, welches Änderungen an einer Eigenschaft mitteilt.

```
public event PropertyChangedEventHandler PropertyChanged;
```

In der zu überwachenden Eigenschaft muss bei Änderung (`set`) das `PropertyChanged`-Ereignis ausgelöst werden. Dabei muss das Objekt und der Name der Eigenschaft als `String` übergeben werden. Wenn nun ein Binding- Objekt an diese Eigenschaft bindet, wird bei Änderung das Ereignis ausgelöst und leitet die Änderungsnachricht an das gebundene Element weiter.

```
1  public event PropertyChangedEventHandler PropertyChanged;
2
3  public bool BoolProperty
4  {
5          get { return _boolProperty; }
6          set {
7                  _boolProperty = value;
8                  if (PropertyChanged != null)
9                  {
10                         PropertyChanged(this, new System.
                               ComponentModel.
                               PropertyChangedEventArgs("
                               BoolProperty"));
11                 }
12         }
13 }
```

Listing 6.4: Implementierung der INotifyPropertyChanged- Schnittstelle

Bindung von Listen Für das Binden an Listen gibt es spezielle Listensteuerelemente. Diese sind von der Klasse `ItemsControl` abgeleitet, von welcher sie die `ItemsSource`-

Eigenschaft erben. Dieser Eigenschaft kann man nun eine Referenz auf eine Liste zuweisen, was bedeutet, dass kein `Binding`- Objekt benötigt wird. Das Listensteuerelement ruft nun für jedes einzelne Objekt in der Liste die `ToString`- Methode auf und nutzt den zurückgelieferten `String` für die Darstelllung des Objekts. Soll nicht die `ToString`- Methode genutzt werden, kann die Eigenschaft `DisplayMemberPath` gesetzt werden. Über diese kann man angeben, welche Eigenschaft des Objektes als Anzeigeelement verwendet werden soll. Dazu muss der Name der Eigenschaft als `String` zugewiesen werden.

Typumwandlung Wenn die Datentypen von aneinander gebundenen Eigenschaften unterschiedlich sind, ist es nötig, vor der Bindung eine Typenkonvertierung vorzunehmen.

Im folgenden Beispiel wird ein `DateTime`- Objekt an eine `TextBox` gebunden. Normalerweise würde die `TextBox` das Datum automatisch in einem langen Format mit Datum und Uhrzeit darstellen (es wird automatisch die `ToString`- Methode aufgerufen). Möchte man nun aber nur das Datum angezeigt haben, muss man einen Wertkonvertierer implementieren, welcher das `DateTime`- Objekt in einen `String` mit gewünschtem Format konvertiert. Dazu muss die Schnittstelle `IValueConverter` implementiert werden. Diese Schnittstelle fordert die Implementierung der zwei Methoden `Convert` und `ConvertBack`. Damit kann man die Typkonvertierung in beide Richtungen vornehmen. Um das Beispiel einfach zu halten, wird jedoch nur die Konvertierung in eine Richtung gezeigt, weshalb bei `ConvertBack` „null“ zurückgegeben wird.

```
1  [ValueConversion(typeof(DateTime), typeof(String))]
2  public class DateConverter : IValueConverter
3  {
4      public object Convert(object value, Type targetType,
5          object parameter, CultureInfo culture)
6      {
7          DateTime date = (DateTime)value;
8          return date.ToShortDateString();
9      }
10
11     public object ConvertBack(object value, Type targetType,
12         object parameter, CultureInfo culture)
13     {
14         return null;
15     }
16 }
```

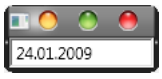
Listing 6.5: Implementierung eines eigenen DateTime- Konverters

In XAML wird, wie in folgendem Listing 6.6, von diesem Konverter ein Objekt erzeugt (Zeile 7) und bei der Bindung mit angegeben (Zeile 10).

```
1  <Window x:Class="IValueConverterExample.Window1"
2      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/
           presentation"
3      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4      xmlns:src="clr-namespace:IValueConverterExample"
5      Title="Window1" SizeToContent="WidthAndHeight">
6      <Window.Resources>
7          <src:DateConverter x:Key="dateConverter"/>
8      </Window.Resources>
9      <Grid>
10         <TextBox Name="txbText" Text="{Binding␣Path=.,␣
               Converter={StaticResource␣dateConverter}}"></
               TextBox>
11     </Grid>
12 </Window>
```

Listing 6.6: Verwendung des Konverters in XAML

In der Codebehind- Datei wird ein DateTime- Objekt als `DataContext` der `TextBox` festgelegt. Möchte die `TextBox` nun ihren Inhalt anzeigen, wird das `DateTime`- Objekt an die `Convert`- Methode des angegeben Konverters übergeben. Anschließend wird das zurückgelieferte Objekt von der `TextBox` zur Anzeige gebracht.

	Bindung an ein DateTime- Objekt ohne Konvertierung
	Bindung an ein DateTime- Objekt mit eigener Konvertierung

Tab. 6.1: Bindung an ein DateTime- Objekt mit und ohne Konvertierung

6.1.2 LINQ

LINQ ist die Abkürzung für Language Integrated Query und wird hier für das objektrelationale Mapping verwendet. Es ist ein in die Programmiersprache eingebetteter Abfragemechanismus für Daten. Die wesentlichen Bestandteile einer Software sind die Programmlogik und die von der Software zu verwaltenden Daten. Es ist also fast immer notwendig Code zu schreiben, der sich mit dem Zugriff auf Daten befasst. LINQ bietet dazu einen Lösungsansatz. Es ermöglicht den Zugriff auf die verschiedensten Datenquellen:

- relationale Datenbanken (LINQ to SQL)
- Arbeitsspeicher (LINQ to Objects)

- XML-Dokumente (LINQ to XML)

LINQ ermöglicht durch die Integration in die Sprache C# eine vom Compiler geprüfte, strenge Typisierung. Dadurch können bereits bei der Kompilierung eventuelle Fehler bei Datenabfragen erkannt werden. Mit LINQ werden zusätzliche Sprachelemente in C# integriert, welche im Folgenden kurz vorgestellt werden:

- Implizit typisierte lokale Variablen (Schlüsselwort `var`):

  ```
  var local = new String();
  ```

  Der Compiler leitet den Typ der Variablen implizit vom Initialisierungsausdruck der Variablen ab.

- Anonyme Typen

  ```
  var local = new { Name="'Mustermann"', Surname="'Max"'};
  ```

  Anonyme Typen erlauben es Daten in einem Objekt zu gruppieren, ohne dabei eine neue Klasse deklarieren zu müssen.

- Objektinitialisierer

  ```
  DateTime dt = new DateTime(){ Day = 26, Month = 1, Year = 2009 };
  ```

  Es ist möglich Eigenschaften eines Objektes direkt bei der Erzeugung zu initialisieren.

- Abfrageschlüsselwörter: from, where, select, orderby, groupby

  Die Abfrageschlüsselwörter sind an die Schlüsselwörter von SQL angelehnt.

Um das Mapping unter Verwendung von LINQ einzurichten, gibt es zwei verschiedene Möglichkeiten.

1. Manuelles Mapping
2. Verwendung des LINQ to SQL Designers

In den folgenden Abschnitten werden beide Verfahren vorgestellt.

6.1.2.1 Manuelles Mapping

Nachdem man das zu verwendende Datenmodell erstellt hat, können darauf aufbauend die Datenklassen implementiert werden. Außerdem muss ein Datenbankmodell entwickelt werden, welches die entsprechenden Objekte relational auf der Datenbank abbilden kann. Um die Kommunikation mit der Datenbank zu ermöglichen, muss:

1. das Mapping in den Datenklassen ergänzt werden

2. ein `DataContext`- Objekt erzeugt werden, welches sich um die Verbindung zur Datenbank kümmert

Folgendes Listing zeigt eine einfache Datenklasse mit grundlegendem Mapping:

```
1 [Table]
2 public class Users
3 {
4         [Column(IsPrimaryKey=true)]
5         public int id { get; set; }
6
7         [Column]
8         public int name { get; set; }
9 }
```

Listing 6.7: Einfache Datenklasse mit LINQ- Mapping

Die Attribute in eckigen Klammern (Zeile 1, 4, 7) stellen das Mapping dar. Auf der Datenbankseite muss eine Tabelle „Users“ mit den Spalten „id“ und „name“ vorhanden sein. Die „id“- Spalte muss als Primärschlüssel ausgewiesen sein.

Mit Hilfe des `DataContext`- Objekts kann nun eine Verbindung zur Datenbank hergestellt werden:

```
1 DataContext dataContext = new DataContext("Data␣Source=.\
    SQLEXPRESS;Initial␣Catalog=olfakt;Integrated␣Security=True
    ");
2
3 Table<Users> users = dataContext.GetTable<Users>();
4
5 var query =
6         from user
7         in users
8         orderby user.name
9         select user;
```

Listing 6.8: Verwendung des DataContext- Objekts

Dem `DataContext`- Objekt wird zunächst die Verbindungszeichenfolge übergeben. „Data Source“ bezeichnet die SQL- Server Instanz, „Initial Catalog“ bezeichnet die verwendete Datenbank und „Integrated Security“ gibt an, dass die Zugangsdaten des aktuell angemeldeten Windows- Users für die Authentifizierung auf dem SQL-Server verwendet werden. Über die Funktion `GetTable<[Tabellenname]>()` kann eine Referenz auf die angegebene Tabelle erzeugt werden. Erst wenn eine Abfrage auf diese Tabelle durchgeführt wird (Zeile 5), wird tatsächlich SQL- Code generiert und an den SQL- Server geschickt.

```
1 SELECT [t0].[id], [t0].[name]
2 FROM [dbo].[Users] AS [t0]
3 ORDER BY [t0].[name]
```

Listing 6.9: Generierter SQL- Code

Komplizierter wird das Mapping, wenn Beziehungen zwischen den Objekten abgebildet werden müssen. Eine deutlich einfachere Möglichkeit als das Mapping per Hand durchzuführen, bietet der LINQ to SQL Designer, welcher im nächsten Abschnitt vorgestellt wird.

6.1.2.2 Verwendung des LINQ to SQL Designers

Bei der Verwendung des LINQ to SQL Designers ist das Vorgehen dahingehend verändert, dass die Datenklassen und das Mapping nicht von Hand erstellt werden, sondern generiert werden. Es muss als Vorbereitung nur die Datenbank mit dem vollständigen Datenmodell vorliegen. Mit Hilfe dieser Informationen kann das Visual Studio die entsprechenden Datenklassen und das Mapping generieren. Damit spart man nicht nur die Arbeit, das Mapping mühsam per Hand zu erstellen, es werden auch Fehler ausgeschlossen, die durch ungünstige oder falsche Mappingkonfigurationen entstehen können. Bei erstmaligem Verwenden von LINQ-to-SQL ist es auch deutlich einfacher erste Ergebnisse zu erzielen.

Im Folgenden ist beispielhaft beschrieben wie der Visual Studio O/R- Designer zu verwenden ist.

1. Ein .NET Projekt erstellen und eine LINQ to SQL- Klasse hinzufügen.
2. Visual Studio erzeugt eine LINQtoSQL.dbml- Datei, welche folgende Dateien gruppiert:
   - *LINQtoSQL.cs*: generierte Klassen können hier mit eigenem Code erweitert werden
   - *LINQtoSQL.dbml.layout*: grafische Klassenansicht
   - *LINQtoSQL.designer.cs*: generierte Datenklassen mit Mapping
3. Tabellen können aus einer vorhandenen Datenbank im Server-Explorer mittels „Drag and Drop“ in den O/R-Designer „gezogen“ werden.

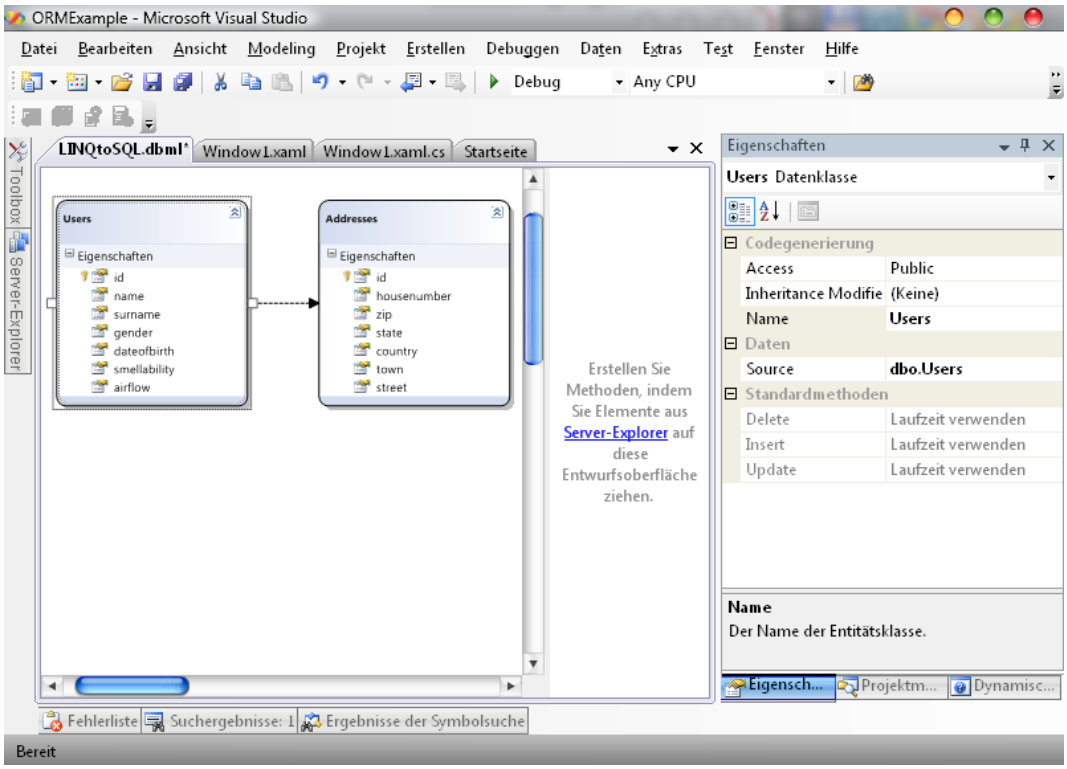


Abb. 6.3: Abbildung der Tabellen einer Datenbank im O/R- Designer

4. Der O/R- Designer bietet eine grafische Darstellung zum Datenmodell. Im Hintergrund wird das Mapping generiert. Folgender Codeausschnitt zeigt die generierten Mapping- Informationen für die Eigenschaft „id“ der Klasse „Users“.

```
[Column(Storage="_id",
AutoSync=AutoSync.OnInsert,
DbType="Int␣NOT␣NULL␣IDENTITY",
IsPrimaryKey=true,
IsDbGenerated=true)]
```

Storage gibt den Namen des Members der Klasse an.

AutoSync ist nötig, weil der Wert von der Datenbank generiert wird.

DbType spezifiziert den Typ in der Datenbank (nötig für `DataContext.CreateDatabase`-Methode).

IsPrimaryKey gibt die Primärschlüsselspalte der Tabelle an.

IsDbGenerated die „id“ wird von der Datenbank generiert.

Zusätzlich generiert Visual Studio eine Subklasse von `DataContext`, welche die Referenzen auf Tabellen als Eigenschaften zur Verfügung stellt. Für die Verbindung zur Datenbank schreibt Visual Studio die Verbindungszeichenfolge

in die Konfigurationsdatei „*app.config*“. Die `DataContext`- Klasse verwendet diese Verbindungszeichenfolge, so dass diese nicht immer explizit angegeben werden muss.

5. Folgendes Beispiel zeigt die Verwendung der generierten Klassen:

```
1  public partial class Window1 : Window
2  {
3      public Window1()
4      {
5          InitializeComponent();
6
7          LINQtoSQLDataContext dataContext = new
               LINQtoSQLDataContext();
8
9          var users = from u in dataContext.Users select u;
10
11         lstUsers.ItemsSource = users;
12         lstUsers.DisplayMemberPath = "name";
13     }
14 }
```

Listing 6.10: Generierte Klassen verwenden

- Zugriff funktioniert über Instanz der Klasse „[Name]DataContext“ (Zeile 7)
- „Users“ als Objekte von der Datenbank holen (Zeile 9)
- Bindung an eine WPF- ListBox (Zeile 11)
- Anzeige des Namens („name“) in der Liste (Zeile 12)

Wie bei dem Ausschnitt aus dem Mapping zu sehen, bildet der Mapping- Generator das Datenmodell der Datenbank detailgetreu nach. Dadurch bietet sich sogar die Möglichkeit, die Datenbank vom Programm aus erzeugen zu lassen. Das hat z. B. Vorteile bei späterer Installation der Software auf dem Zielsystem. Es muss als Vorbereitung dann nur ein entsprechender SQL- Server laufen, die Datenbank und das Datenmodell kann automatisch über die Methode `DataContext.CreateDatabase` erzeugt werden.

6.2 Implementierung

Nachdem im vorherigen Abschnitt noch einmal allgemein auf die verwendeten Technologien eingegangen wurde, werden in diesem Anschnitt programmspezifische Besonderheiten erläutert.

6.2.1 Das GUI

Die Entwicklung der grafischen Oberfläche mit WPF stellte sich als unkompliziert heraus. Sind die grundlegenden Konzepte, welche in Abschnitt 6.1.1 beschrieben wurden, verstanden und verinnerlicht, können schnell leistungsfähige Oberflächen entwickelt werden.

6.2.1.1 Bindung an LINQ-to-SQL Tabellen

Eine Besonderheit bei der Datenbindung stellte die Bindung an eine komplette LINQ-to-SQL Tabelle dar. Dies ist erforderlich, wenn eine Liste mit allen Datensätzen der Tabelle angezeigt werden soll. Sinn der Datenbindung ist das gegenseitige Benachrichtigen, wenn Änderungen an den Daten vorgenommen werden. Wenn eine Tabelle an eine Liste gebunden werden soll, möchte man in der Liste erkennen, ob neue Datensätze hinzugefügt oder gelöscht wurden. Das Problem bei LINQ-to-SQL besteht darin, dass die Klasse `Table` nicht die Schnittstelle `INotifyCollectionChanged` implementiert. Deshalb werden beim Hinzufügen oder Löschen von Datensätzen auf der Tabelle keine Änderungen an einer angebundenen Liste vorgenommen.

Wie die Änderungen auf einer Tabelle dennoch verfolgt werden, soll das folgende Beispiel zeigen. Auf der Konfigurationsmaske ist eine Liste mit allen verfügbaren Kartuschen vorhanden. Wird eine neue Kartuschenbeschreibung angelegt oder gelöscht, soll dies auch sofort in der Liste sichtbar werden. Um dies zu erreichen muss eine Hilfsliste vom Typ `ObservableCollection<Type T>` angelegt werden. Dort hinein kommen alle Datensätze aus der Kartuschen- Tabelle. Der Typ `ObservableCollection<Type T>` implementiert die Schnittstelle `INotifyCollectionChanged`, welche die Überwachung der enthaltenen Elemente übernimmt. An die WPF- Liste wird nun nicht die Tabelle gebunden, sondern die `ObservableCollection`. Werden nun Änderungen an den Datensätzen vorgenommen, müssen diese Änderungen sowohl der Tabelle (über `DataContext`) als auch der `ObservableCollection` mitgeteilt werden. Damit erhöht sich zwar der Aufwand bei der Implementierung, jedoch gibt es derzeit keine bessere Möglichkeit.

6.2.1.2 Benutzerdefinierte Steuerelemente

Grafische Elemente, die mehrfach verwendet werden, können am besten als benutzerdefinierte Steuerelemente implementiert werden. Dadurch wird es leichter diese Elemente wieder zu verwenden. Folgende benutzerdefinierte Steuerelemente wurden entwickelt:

- Auswertungsdokument: `ucEvaluation`
- Bildschirmtastatur: `ucKeyboard`
- Diagramm: `ucDiagramm`

6.2.2 O/R Mapping

Das O/R Mapping wurde, wie bereits erwähnt, mit Hilfe von LINQ und dem LINQ to SQL Designer realisiert.

Für das Erstellen des Datenbankschemas und die Erzeugung der Datenbank kann man verschiedene Wege nutzen. Für den MS SQL Server gibt es das grafische Tool *MS SQL Server Management Studio*, mit dem es möglich ist Datenbanken zu erzeugen und zu bearbeiten. Eine weitere Möglichkeit bietet das Visual Studio. Hier kann man direkt im LINQ to SQL Designer das Datenbankschema erstellen und zur Laufzeit die Datenbank erzeugen. Außerdem gibt es die Möglichkeit das Datenbankschema mit einem externen Tool zu entwickeln und von diesem ein SQL-Skript zur Erzeugung der Datenbank generieren zu lassen. Es wurde die letzte Variante gewählt und das grafische Tool *Power*Architect* verwendet. Bild 6.4 zeigt das Schema im *Power*Architect*.

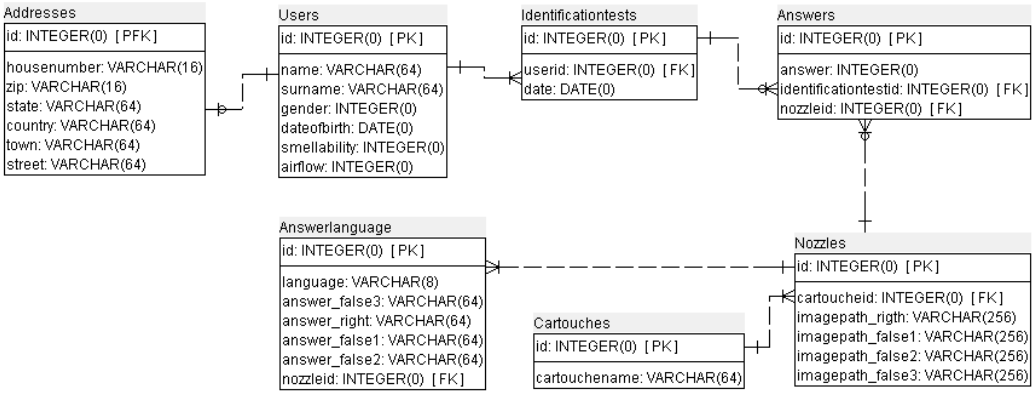


Abb. 6.4: Das Datenbankschema abgebildet durch den Power*Designer

Aus diesem Schema konnte ein SQL- Skript generiert werden, welches die Datenbank anlegt, die Tabellen erzeugt und die erforderlichen Attribute sowie Verbindungen setzt. Nachdem das Datenschema in die Datenbank eingespielt wurde, konnte mit Hilfe des LINQ to SQL Designers das Mapping hergestellt werden.

Abb. 6.5: Das Datenbankschema abgebildet durch den O/R Mapper in Visual Studio

Das Mapping mit Hilfe des Designers ist sehr bequem. Es konnte nun ohne große Probleme eine Instanz der Klasse `dataDataContext` angelegt und über diese Instanz Daten in der Datenbank manipuliert werden.

Wenn die Datenbank vorhanden ist, funktioniert das Mapping hervorragend und ohne Probleme. Wenn auf einem Zielsystem die Datenbank nicht vorhanden ist, aber ein entsprechender Server läuft, soll die Datenbank und das Schema erzeugt werden. Dies realisiert die Klasse `DataContext` mit der Methode `CreateDatabase`. Die dafür nötigen Informationen werden aus der generierten Mapping- Datei über die in eckigen Klammern stehenden Attributen gewonnen. Diese Attribute stehen vor jeder Klasse und vor jeder Eigenschaft.

Das Erstellen der Datenbank funktionierte auch ohne Probleme, jedoch wurde die Löscheigenschaft `ON DELETE CASCADE` nicht bzw. falsch übernommen. Das Fehlverhalten wurde durch den Mapping- Generator erzeugt.

Um Relationen abzubilden, enthalten zwei generierten Klassen jeweils Eigenschaften, die Verweise auf die Vater- bzw. Kindelemente darstellen. Das Vaterobjekt hat eine Eigenschaft des Typs `EntitySet`, welches eine Liste ist, in der die Referenzen zu allen Kindobjekten eines bestimmten Typs enthalten sind. Die Kinder haben eine Eigenschaft des Typs `EntityRef`, welche das Vaterobjekt referenziert. Mit Hilfe des Mapper- Attributs `Association` vor der jeweiligen Eigenschaft wird signalisiert, dass es sich hier um eine Relation auf der Datenbank handelt. Um die Relation weiter zu spezifizieren, können verschiedene Eigenschaften angegeben werden, unter anderem die Eigenschaft `DeleteRule`. Wird hier der Wert „`CASCADE`“ angegeben, sollte beim Erstellen der Datenbank durch `CreateDatabase` die Relation mit der Löscheigen-

schaft `ON DELETE CASCADE` erzeugt werden. Da das `Association`- Attribut sowohl bei der Vater- als auch bei der Kindklasse definiert wird, ist es wichtig zu wissen, in welcher Klasse die `DeleteRule` angegeben werden muss. Der Generator erzeugt die Eigenschaft in der Kindklasse, was falsch ist. Dadurch wird das Löschverhalten nicht in eine erzeugte Datenbank übernommen.

Das Problem kann einfach umgangen werden, in dem die `DeleteRule` aus der Realtionsspezifikation der Kindklasse in die der Vaterklasse verschoben wird. Dabei ist jedoch Vorsicht geboten. Sobald etwas im LINQ to SQL Designer verändert wird, z. B. eine neue Tabelle hinzugefügt oder eine alte gelöscht, wird die entsprechende Codedatei neu generiert. Das hat zur Folge, dass die vorgenommenen Änderungen überschrieben werden. Korrekturen sollten an der generierten Datei also erst erfolgen, wenn keine Änderungen mehr im Designer vorgenommen werden.

Folgendes Beispiel zeigt die generierten Klassen `Cartouches` (Vater) und `Nozzles` (Kinder). Sobald ein Objekt von `Cartouches` gelöscht werden soll, sollen alle dazugehörigen `Nozzles` auch gelöscht werden. Listing 6.11 zeigt die betreffenden Ausschnitte aus den zwei generierten Klassen.

```
1  [Table(Name="dbo.Cartouches")]
2  public partial class Cartouches : INotifyPropertyChanging,
       INotifyPropertyChanged
3  {
4          /* ... */
5          private EntitySet<Nozzles> _Nozzles;
6          /* ... */
7          [Association(Name="Cartouches_Nozzles",
8          Storage="_Nozzles",
9          OtherKey="cartoucheid")]
10         public EntitySet<Nozzles> Nozzles
11         {
12                 get { /* ... */ }
13                 set { /* ... */ }
14         }
15     /* ... */
16 }
17
18 [Table(Name="dbo.Nozzles")]
19 public partial class Nozzles : INotifyPropertyChanging,
       INotifyPropertyChanged
20 {
21         /* ... */
22         private EntityRef<Cartouches> _Cartouches;
23     /* ... */
24         [Association(Name="Cartouches_Nozzles",
25         Storage="_Cartouches",
26         ThisKey="cartoucheid",
27         IsForeignKey=true,
28         DeleteOnNull=true,
29         DeleteRule="CASCADE")]
```

```
30          public Cartouches Cartouches
31          {
32                  get { /* ... */ }
33                  set { /* ... */ }
34          }
35          /* ... */
36  }
```

Listing 6.11: Fehlerhaftes Mapping (ON DELETE CASCADE)

Die Eigenschaft „`DeleteRule="CASCADE"`“ ist an der falschen Stelle gesetzt. Richtig müsste sie in die `Association` in Zeile 7. Versucht man aus einer damit erzeugten Datenbank eine Kartusche zu löschen, die bereits auf Düsen verweist, bekommt man folgenden Ausnahmefehler:

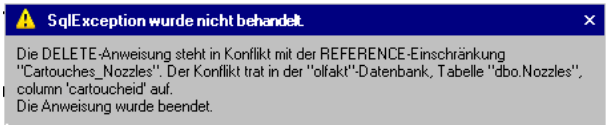


Abb. 6.6: Ausnahmefehler beim nicht kaskadierenden Löschen

Werden die Korrekturen an den entsprechenden Stellen vorgenommen und dann die Datenbank erzeugt, funktioniert der Mechanismus ohne Probleme.

6.2.3 Die serielle Schnittstelle

Die Kommunikation zu den Controllern über die serielle Schnittstelle wurde in der Klasse `OlfaktCommunicator` implementiert. Zur Initialisierung muss dem Konstruktor der Klasse einfach der Name des Ports angegeben werden, an dem der Controller angeschlossen ist. Die Klasse stellt vier grundlegende Funktionen zur Verfügung:

1. Duft ausgeben (`SpendFragrance`)
2. Spülventile schließen (`CloseFlush`)
3. Spülventile öffnen (`OpenFlush`)
4. Erreichbarkeit des Controllers prüfen (`CheckCommunication`)

Um einen Duft auszugeben, muss der Funktion `SpendFragrance` das Deck, das Ventil und die Dauer der gewünschten Geruchsausgabe übergeben werden. Spülventile öffnen und schließen sind als eigene Funktionen verfügbar, weil immer die Spülventile beider Controller geschlossen werden müssen. Die Funktion `CheckCommunication` prüft nur, ob Nachrichten, die auf dem Port gesendet wurden, auch ausgelesen werden. Es ist dabei nicht garantiert, dass ein Controller des Olfaktometers angeschlossen ist. Es könnte auch jedes beliebige andere Gerät sein.

6.2.4 Mehrsprachigkeit

Laut Anforderungen (vgl. Abschnitt 4.8) war für das Programm Mehrsprachigkeit gefordert. Bei Software wird diese Anforderung als Lokalisierung bezeichnet.

Um Lokalisierung zu realisieren, gibt es so genannte Resourcendateien. Für jede erforderliche Sprache wird eine eigene Resourcendatei angelegt. Eine Resourcendatei ist dabei als Standardresourcendatei definiert. Sollte in einer Resourcendatei eine bestimmte lokalisierte Zeichenfolge nicht gefunden wird, dann verwendet das Resourcensystem die Standardresource und verhindert somit einen eventuellen Ausnahmefehler. Eine lokalisierte Zeichenfolge (Resource) in einer Resourcendatei besteht immer aus einem Namen und dem dazugehörigen lokalisierten Wert:

	Name	Wert
	country	Staat
	date	Datum
	dateofbirth	Geburtsdatum
	dateoftest	Testdatum
	error	Fehler
	gender	Geschlecht
	gender_female	weiblich
	gender_male	männlich

Abb. 6.7: Ausschnitt einer Resourcendateiansicht in Visual Studio

Um eine Resourcendatei einer besimmten Sprache zuzuordnen, gibt eine Namenskonvention. Die im Programm verwendete Standard- Resourcendatei heißt *Language.resx* und enthält die deutschen Zeichenfolgen. Diese Resource wird in der Hauptassembly des Programms abgelegt. Die englische Resourcendatei musste dann *Language.en-GB.resx* benannt werden. Diese Erweiterung signalisiert dem Kompiler, dass diese Resource in eine Sattelitenassembly abgelegt werden soll. Diese Assembly wird im Ausgabeverzeichnis unter dem Pfad `.\en-GB\olfakt.resources.dll` abgelegt.

Der Zugriff auf die lokalisierten Strings erfolgt über den `ResourceManager`. Dem Konstruktor dieser Klasse muss der Bezeichner der Resource und die Assembly, welche die Resource enthält, übergeben werden. Der Bezeichner der Resource setzt sich zusammen aus dem *Namespace* und dem Namen der Resource. Die Assembly kann über die Funktion `System.Reflection.Assembly.GetExecutingAssembly` ermittelt werden. Um die lokalisierte Zeichenfolge zu erhalten, wird die Funktion `GetString` des `ResourceManager`s aufgerufen. Diese erwartet den Namen der Zeichenfolge und die aktuelle Sprache.

Folgendes Listing zeigt, wie die englische Übersetzung von „Datum“ geladen werden kann:

```
ResourceManager rm = new ResourceManager(
        "olfakt.language.Language",
        System.Reflection.Assembly.GetExecutingAssembly());
return rm.GetString(name, new CultureInfo("en-GB"));
```

Listing 6.12: Verwendung des ResourceManagers

Der `ResourceManager` versucht nun die Resource mit dem Namen „date“ in der Sattelitenassembly zu finden. Wird die Resource dort nicht gefunden, wird die Standardresource verwendet.

Die bisherigen Ausführungen beziehen sich nur auf die Resourcenverwendung in C#- Code. Um Resourcen auch in XAML zu verwenden, wird die Bibiliothek „LocalizeExtension“ benötigt. Diese ermöglicht die Bindung von GUI- Elementen an Resourcen. Hier ein kurzes Beispiel dazu:

```
<TextBlock Text="{
        LocText
        Key=date,
        Dict=Language,
        Assembly=olfakt}" />
```

Listing 6.13: Verwendung von Resourcen in XAML

Die `Text`- Eigenschaft wird an die „date“- Resource (Zeile 3) gebunden. `LocText` ermöglicht die Lokalisation von Zeichenfolgen. Über `Dict` wird die Resourcendatei und über `Assembly` die Assembly angegeben.

6.2.5 Fehlerbehandlung

Auftretende Fehler werden dem Nutzer über Mitteilungsdialoge signalisiert. Wenn ein Fehler in einer Klasse auftritt, wird in dieser Klasse der Eigenschaft `ErrorMessage` eine Fehlermeldung zugewiesen. Die Fehlermeldung wird vom aufrufenden Objekt bis zur GUI weitergeleitet und dort zur Anzeige gebracht.

Unterschieden wird zwischen Fehlern und Warnungen. Warnungen signalisieren, dass die wesentliche Funktionalität mit Einschränkungen gewährleistet ist. Fehler hingegen signalisieren, dass wesentliche Funktionen nicht verfügbar sind und darum der Riechtest nicht durchgeführt werden kann.

6.3 Verwendete Hilfsmittel

In diesem Abschnitt werden einige Hilfsmittel vorgestellt, die die Entwicklung vereinfachten.

6.3.1 Null-Modem Emulator (com0com)

Das Olfaktometer war nicht während der ganzen Zeit der Entwicklung verfügbar. Um dennoch die Kommunikation über sie serielle Schnittstelle zu testen, wurde mit Hilfe der Software „com0com“ eine virtuelle Schnittstelle eingerichtet. Das Programm stellt dabei zwei virtuelle COM- Ports zur Verfügung, auf denen von beiden Seiten gesendet und empfangen werden kann. Mit Hilfe des im nächsten Abschnitts vorgestellten Programms konnte dann ein am Port lauschender Controller simuliert werden.

6.3.2 COM- Empfänger

Um an eine virtuelle serielle Schnittstelle gesendete Nachrichten auf dem dazugehörigen Port zu empfangen, wurde der COM- Empfänger entwickelt. Zunächst kann der Port gewählt werden, an dem gelauscht wird. Alle dort ankommenden Nachrichten stellt das Programm als HEX- Blöcke in einer `TextBox` dar. So können erzeugte Nachrichten auf ihre Richtigkeit hin überprüft werden.

6.4 Tests

6.4.1 Unit Tests

Um die Richtigkeit einzelner Methoden zu testen, gibt es Unit Tests. Sie ermöglichen es fehlerhaftes Verhalten von Methoden zu erkennen. Dabei wird vor allem das Verhalten der Methoden bei Grenzwerten getestet. Unit Tests bieten sich bei Berechnungen an.

Für das Unit- Testing stellt das Visual Studio eine integrierte Testumgebung zur Verfügung. Dazu legt man in der Projektmappe ein Testprojekt an. Dort können per Hand oder mit Hilfe eines Wizzards Tests für ausgewählte Methoden beschrieben werden. Da die meisten Methoden nicht losgelöst vom System getestet werden können, wurden nur für einige wenige Methoden Unit- Tests geschrieben.

Die Testergebnisse sind im Visual Studio- Projekt enthalten. Das Projekt ist auf der beiligenden CD zu finden.

6.4.2 Systemtests

Mit dem Systemtest wird geprüft, ob das System die Anforderungen erfüllt. Dazu wurde zu einigen definierten Anforderung jeweils ein kurzer Test durchgeführt:

Anforderung	Test	Erfolg
1. Riechtest		
1.1.3	Erfassung von Name, Vorname und Anschrift laut Konfiguration erforderlich. Diese werden zu Beginn des Riechtests mit abgefragt.	ja
	Erfassung von Name, Vorname und Anschrift laut Konfiguration nicht erforderlich. Die Daten werden zu Beginn des Riechtests nicht abgefragt.	ja
1.1.4	Name wird nicht angegeben. Eine Warnung wird angezeit und der Riechtest wird nicht gestartet.	ja
1.2.2.6	Es kann noch keine Antwort gegeben werden, weil noch kein Duft ausgegeben wurde	ja
1.2.2.7	Zwischen zwei Duftausgaben wird eine Pause von 5 Sekunden eingehalten.	ja
1.2.2.8	Geruch kann ein zweites Mal ausgegeben werden, aber kein weiteres Mal.	ja
1.3.1	Ein Auswertungsdokument wird erzeugt und angezeigt.	ja
1.3.2	Das Dokument kann gedruckt werden.	ja
1.4	Prüfen, ob abgeschlossener Riechtest in der Auswertungstabelle angezeigt wird.	ja
1.5	Abbruch des Tests nach 4 Minuten ohne Nutzereingabe.	ja

Anforderung	Test	Erfolg
2. Auswertung		
2.1	Auswertungsdokument für einen abgeschlossenen Test wird erzeugt und angezeigt.	ja
2.2	Anzeige von zwei Tests im Auswertungsdiagramm.	ja
2.3	Export der bisher erfassten Tests in Excel- Datei.	ja

Anforderung	Test	Erfolg
3. Konfiguration		
3.2	Entspricht die Probenreihenfolge der Konfiguration?	ja
3.4	Kartuschenbeschreibung anlegen	ja
	Kartuschenbeschreibung bearbeiten	ja
	Kartuschenbeschreibung löschen	ja
3.5	Administrator Kennwort in „testkennwort“ geändert. Wird es beim Aufruf des Administrationsmenüs richtig abgefragt?	ja
3.6	Konfigurationsdatei wurde gelöscht. Wird die Konfigurationsdatei neu angelegt?	ja

Anforderung	Test	Erfolg
4. Nichtfunktionale Anforderungen		
4.4	Konfiguration und Einsicht von Riechtestdaten nur nach Passwortabfrage.	ja
4.5/ 4.6	Durchführung eines Riechtests mit Hilfe des Touchscreens und ohne Verwendung von Maus und Tastatur.	ja
4.8	Es kann zwischen den Sprachen „Deutsch“ und „Englisch“ ausgewählt werden.	ja

7 Fazit

7.1 Erreichter Stand

Mit der entwickelten Software ist es möglich, dass Probanden selbstständig einen Riechtest am Olfaktometer durchführen können. Damit kann teilweise Arbeitszeit der Schwester eingespart werden. Teilweise deshalb, weil der Riechtest nicht alle bisher verwendeten Teiltests abdecken kann.

Durch die einfache und selbstständige Bedienung durch den Probanden wird es möglich das System am Deutschen Hygienemuseum Dresden auszustellen.

Aufgrund der zentralen Datenhaltung und der Exportfunktion stehen erfasste Daten für weitergehende Untersuchungen zur Verfügung. Erfasste Riechtests können im System miteinander verglichen werden.

7.2 Bewertung der eingesetzten Technologien

Das Vorgehen nach dem Wasserfallmodell stellte sich als gute Wahl heraus. Weil es sich um relativ kleines Projekt handelte, wurde durch die Wahl dieses Verfahrens der Managementaufwand minimiert und man konnte sich auf die eigentliche Entwicklungsarbeit konzentrieren.

Der Einsatz der WPF als Technologie für die grafische Oberfläche erwies sich als gute Wahl. Die gute Trennung von Logik und Anzeige, die einfache Datenbindung und das ausgefeilte Layoutsystem machen die Entwicklung von Oberflächen komfortabel. Mit Hilfe des Visual Studio Designers wird die Arbeit mit WPF noch einfacher und es lassen sich sehr schnell Prototypen entwicklen.

LINQ-to-SQL als System für das objektrelationale Mapping zu verwenden, vereinfachte den Datenaustausch zwischen dem Programm und der Datenbank erheblich. Trotz des in Abschnitt 6.2.2 beschriebenen Bugs, eignet sich das System bei kleinen Projekten sehr gut. Bei größeren Projekten mit komplexeren Datenstrukturen kann

die Arbeit mit LINQ-to-SQL jedoch schnell unhandlich werden. Ein weiterer Nachteil von LINQ-to-SQL ist, dass es nicht mehr weiterentwickelt wird. Seit November 2008 wird von Microsoft das ADO.NET Entity Framework als O/R Mapping Lösung für .NET Anwendungen empfohlen (siehe [Schwichtenberg, 2008]).

7.3 Ausblick

An dieser Stelle sollen noch einige Ideen folgen, wie das System verbessert und weiterentwickelt werden könnte.

Die Nutzerdatenerfassung könnte an eine vorhandene, elektronische Patientenkartei angebunden werden.

Die Schwellenwerte für die Diagnose könnten nach einer bestimmten Anzahl von abgeschlossenen Tests dynamisch angepasst werden. Dabei sollen alle abgeschlossenen Riechtests in neuen Schwellenwerte einfließen.

Es könnten weitere statistische Auswertungen direkt im System vorgenommen werden. Einige Beispiel wären:

- welche Düfte wurden am meisten richtig und am meisten falsch beantwortet
- welche Düfte am häufigsten miteinander verwechselt werden

Literaturverzeichnis

[onm, 2008] (2008). *Riechprüfung*. onmeda.de. URL: http://www.onmeda.de/behandlung/verfahren/olfaktometrie.html?p=3 (abgerufen am 29.01.2009).

[Frischalowski, 2007] Frischalowski, D. (2007). *Windows Presentation Foundation*. Addison-Wesley.

[Hummel et al.,] Hummel, T., Müller, A., and Landis, B. *Riechstörungen*. URL: http://www.tu-dresden.de/medkhno/riechen_schmecken/art/neue_seite_2.htm (abgerufen am 29.01.2009).

[Marguerie et al., 2008] Marguerie, F., Eichert, S., and Wooley, J. (2008). *LINQ im Einsatz*. Hanser.

[Schwichtenberg, 2008] Schwichtenberg, H. (2008). *Wie geht es weiter mit LINQ-to-SQL und dem ADO.NET Entity Framework?* heise.de. URL: http://www.heise.de/developer/Wie-geht-es-weiter-mit-LINQ-to-SQL-und-dem-ADO-NET-Entity-Framework--/blog/artikel/118518 (abgerufen am 29.01.2009).

[Spillner and Linz, 2005] Spillner, A. and Linz, T. (2005). *Basiswissen Softwaretest*. dpunkt.verlag.

[Vasters et al., 2002] Vasters, Oellers, Javidi, Freiberger, and DePetrillo (2002). *.NET-Crashkurs*. Microsoft Press.

[Winter, 2005] Winter, M. (2005). *Methodische objektorientierte Softwareentwicklung*. dpunkt.verlag.

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

Glossar

Deck

Bestandteil des Olfaktometers, in welches man Kartuschen einlegen kann. Ein Controller kann zwei Decks ansprechen. 11

Duftkartusche

Batterie mit jeweils sechs Duftröhren zum Einsatz in ein Deck eines Olfaktometers. 11

funktionelle Anosmie

Fachbegriff für ein eingeschränktes Riechvermögen 4

Geruchsausgabe

Bezeichnet das zeitlich begrenzte Ausströmen eines Geruchs vom Olfaktometer 11, 33

Geruchsprobe

Bezeichnet die Untersuchung des Geruchssinns eines Probanden anhand einer Geruchsausgabe. Der Proband kann genau eine Antwort zu einer Geruchsprobe geben. 9

Hals-Nasen-Ohren Klinik

Medizinische Abteilung, die Erkrankungen der benannten Körperteile untersucht und behandelt. 3

Olfaktometer

Dient zur Messung von Geruchsemissionen mithilfe von Probanden, denen definierte Geruchsproben dargeboten werden 3, 8, 11, 30, 32

olfaktorische Wahrnehmung

Bezeichnet die Wahrnehmung von Gerüchen (Geruchssinn) 3, 7

Riechtest

Besteht aus einen oder mehreren Teilstests, die das Riechvermögen eines Probanden testen sollen. Als Ergebnis erfährt der Proband eine Diagnose zu seinem Riechvermögen. 3–5, 8, 11, 14, 16–18

Sniffin' Stick

Bezeichnung eines subjektiven Verfahrens zum Testen des Geruchssinns. Dabei sind die Gerüche in den namensgebenden Stiften enthalten. 8–11, 13, 14, 87, 89

Universitätsklinikum Dresden

Universitätsklinikum Carl Gustav Carus Dresden 3, 12

Abkürzungen

CLR

Common Language Runtime 26

DBMS

Database Management System 29

HNO

Hals-Nasen-Ohren 3, 10

LINQ

Language Integrated Query 1, 29, 66, 67

WPF

Windows Presentation Foundation 27, 28, 61, 63, 71, 83

XAML

eXtensible Application Markup Language 27, 61, 65, 78

A Anhang

A.1 Nutzerhandbuch

Nutzerdokumentation - Riechtestsoftware für Olfaktometer „aerome® ScentController 2x6 (AB 60005)“

Johannes Körner

10. März 2009

Inhaltsverzeichnis

1 Einführung

Die vorliegende Software ist in Verbindung mit dem „aerome® ScentController 2x6 (AB 60005)“ zu verwenden. Sie führt einen Probanden durch einen Riechtest und wertet die Testergebnisse aus. Nach einem Riechtest ermittelt die Software anhand von Vergleichsdaten, ob eine weitere Untersuchung des Geruchssinns beim Probanden erforderlich ist.

Das Programm bietet folgende Funktionen:

- Durchführung und Auswertung eines Riechtests
- Vergleich von mehreren Riechtests
- Export von Riechtestdaten

2 Installation

Für die Installation des Systems wird folgende Hard- und Software benötigt:

Hardware

Olfaktometer (aerome® ScentController (AB 60005))

Personal Computer inkl. 2 serielle Schnittstellen (EIA-232)

Bildschirm (optional Touchscreen)

Software

Microsoft Windows XP™/ Microsoft Windows Vista™

Microsoft .NET Framework 3.5

Microsoft SQL Server 2005 Express Edition™

Um das System in Betrieb zu nehmen, muss der SQL Server auf der lokalen Maschine unter der Adresse

„.\SQLEXPRESS“

erreichbar sein. Die Datenbank „olfakt“ wird beim ersten Start der Software automatisch angelegt.

Das Programmverzeichnis kann einfach von der CD an eine beliebige Stelle auf die Festplatte kopiert werden. Es sind keine weiteren Installationsschritte notwendig.

3 Funktionen

Die Software wurde für zwei Nutzergruppen entwickelt. Zum einen der normale Nutzer, welcher den eigentlichen Test durchführt, und zum andern der Administrator, der für die Datenverwaltung und Konfiguration des Systems verantwortlich ist.

3.1 Anmeldemaske

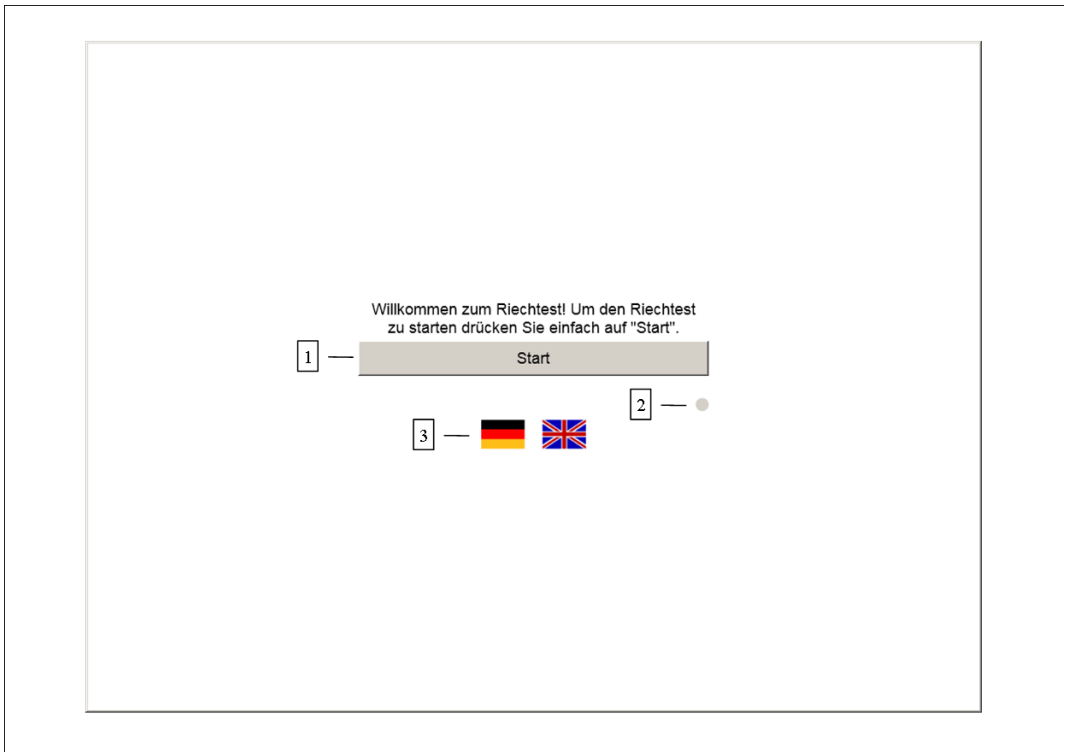


Abb. 3.1: Anmeldemaske

Nach dem Starten des Systems wird die Anmeldemaske angezeigt. Hier kann der Nutzer über „Start“ (1) den Riechtest starten. Bei Bedarf kann, durch Klick auf die jeweilige Flagge (3), zwischen den Sprachen „Deutsch“ und „Englisch“ gewechselt werden. Der Administrator erreicht das Adminstrationsmenü durch Klick auf (2) und nach einer Passwortabfrage.

3.2 Nutzer

3.2.1 Riechtest durchführen

Ein Riechtest besteht aus folgenden drei Abschnitten:

- Nutzerdatenerfassung
- Geruchsproben
- Auswertung des Tests

Diese Abschnitte werden nacheinander vom Probanden bearbeitet.

3.2.1.1 Nutzerdatenerfassung

Abb. 3.2: Nutzerdatenerfassung

Nachdem der Nutzer den Riechtest gestartet hat, werden zunächst seine Stammdaten abgefragt. Ob der Name, Vorname und die Anschrift bei der Eingabe gefordert sind, muss in der Konfiguration festgelegt werden. Folgenden Informationen werden immer abgefragt:

- Geburtsdatum

- Geschlecht
- eigene Einschätzung des Riechvermögens
- eigene Einschätzung des Luftflusses in der Nase

Auf der linken Seite des Bildschirms (1) ist eine Leiste, die anzeigt welche Probe gerade bearbeitet wird und welche schon abgeschlossen ist. Dabei verändern sich die Farben der Symbole:

Rot Die Probe wurde noch nicht bearbeitet.

Gelb Die Probe ist in Bearbeitung.

Grün Die Probe wurde abgeschlossen.

Der Administrator kann durch Klick auf (2) den Riechtest vorzeitig beenden. Zum Beenden ist die Eingabe des Administrator- Passwortes nötig.

Nach Klick auf „OK“ (3) wird die erste Geruchsprobe geladen.

Ist es erforderlich Texte einzugeben, kann die Bildschirmtastatur (4) genutzt werden.

3.2.1.2 Geruchsproben

Abb. 3.3: Geruchsprobenmaske

(1) Ausgabe des Geruches der aktuellen Probe. Während der Geruchsausgabe und 5 Sekunden danach ist der Button deaktiviert um eine zu schnelle wiederholte Ausgabe zu verhindern. Pro Probe sind maximal zwei Geruchsausgaben möglich. Danach wird der Button deaktiviert.

(2) Der Fortschrittsbalken füllt sich während der Geruchsausgabe.

(3) Die vier Antwortmöglichkeiten einer Probe. Es wird die Bezeichnung der Antwortmöglichkeit und wenn vorhanden ein Bild angezeigt. Die ausgewählte Antwort ist mit einem Rahmen hervorgehoben.

(4) Wenn eine Antwort gewählt wurde, wird nächste Probe geladen. Nach der letzten Probe wird die Auswertung geladen.

3.2.1.3 Auswertung

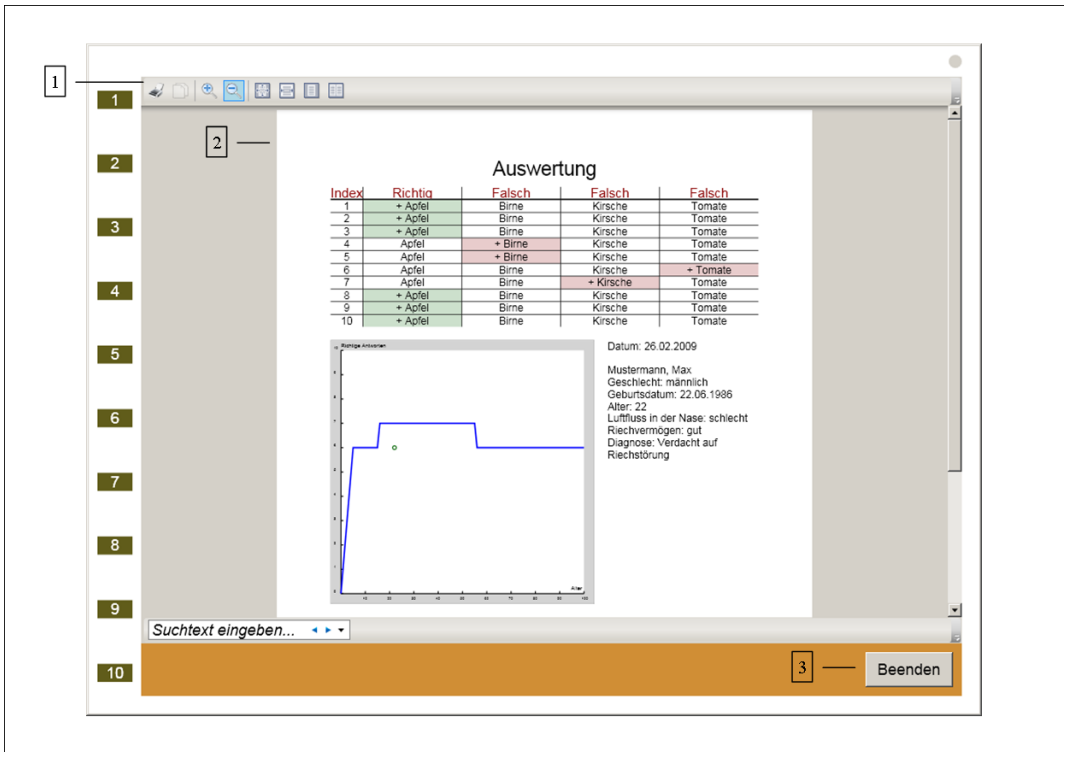


Abb. 3.4: Auswertung eines Riechtests

(1) Durch einen Klick auf das Drucksymbol kann das Auswertungsdokument gedruckt werden. Die anderen Symbole dienen zur Einstellung der Zoomstufe.

(2) Das Auswertungsdokument zeigt Informationen zum abgeschlossenen Riechtest an. In der Tabelle sind die Antwortmöglichkeiten der Proben aufgelistet und die gewählten Antworten mit einem „+“ markiert. Richtige Antworten sind

Grün unterlegt und falsche Rot. Der Graph zeigt Normwerte für die Anzahl der richtigen Antworten in Abhängigkeit vom Alter an. Das Ergebnis des aktuellen Tests ist darin eingetragen. Außerdem sind die Stammdaten des Nutzers, das Datum des Tests und die Diagnose vermerkt.

(3) Der aktuelle Riechtest wird beendent und die Anmeldemaske wird angezeigt.

Wird der „Verdacht auf Riechstörung“ diagnositiziert, sollte der Proband weitergehend untersucht werden.

3.3 Administrator

3.3.1 Administrationsmenü

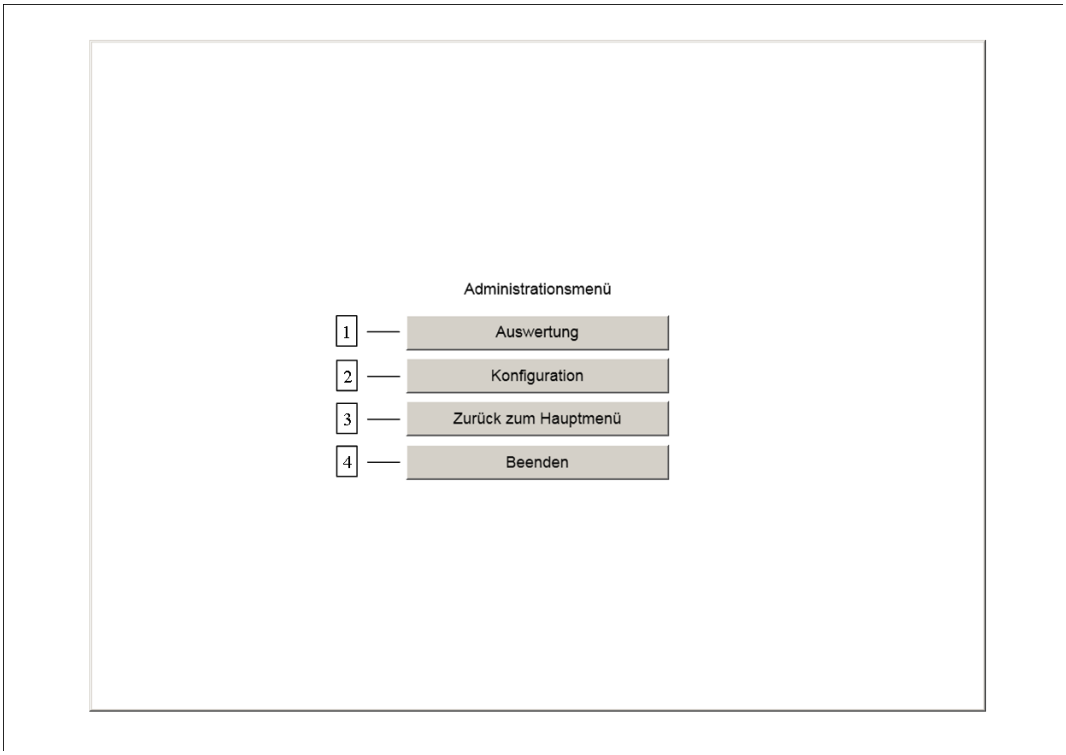


Abb. 3.5: Administrationsmenü

Das Administrationsmenü bietet dem Administrator folgende Möglichkeiten:

(1) Öffnet die Oberfläche zur Datenauswertung.

(2) Öffnet die Konfigurationsmaske.

(3) Führt zurück zur Anmeldemaske.

(4) Beendet das Programm.

3.3.2 Datenauswertung

Die folgenden Ansichten dienen dem Administrator zur Sichtung, Vergleich und Export von erfassten Riechtests.

3.3.2.1 Liste der Riechtests

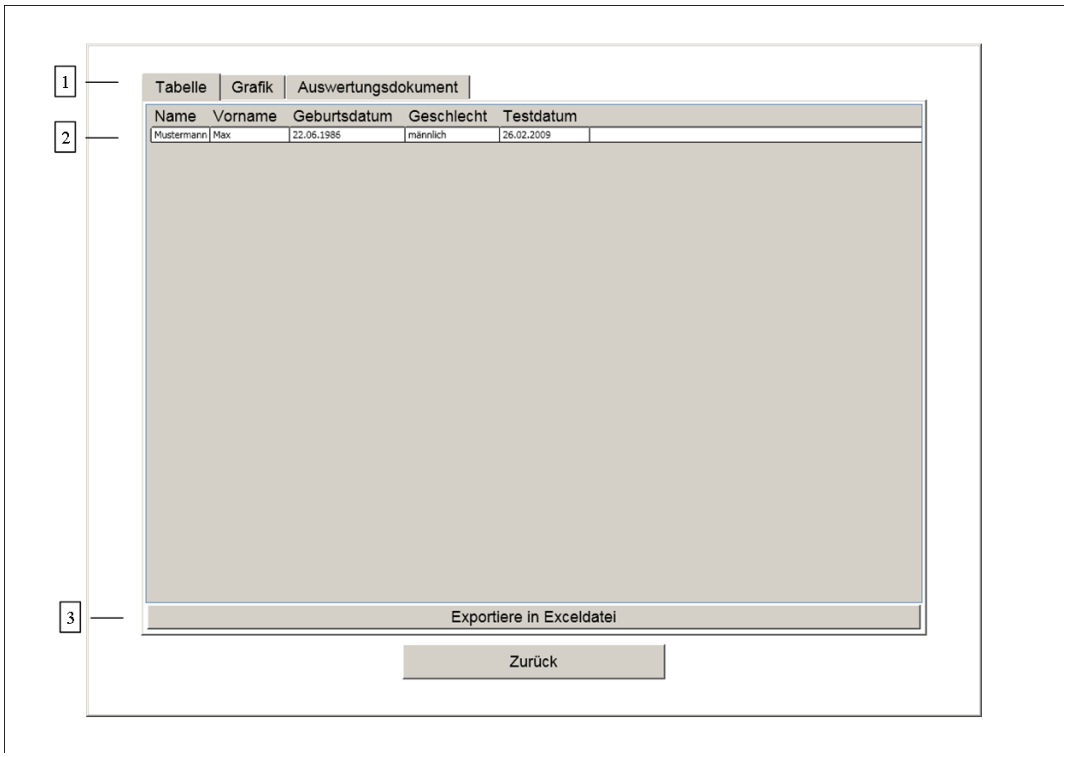


Abb. 3.6: Liste der Riechtests

(1) Tabreiter für die Ansichten „Liste der Riechtests“, „Grafik“ und „Auswertungsdokument“.

(2) Liste mit allen bisher abgeschlossenen Riechtests. Auswahl von Riechtests ändert die Werte in „Grafik“ und „Auswertungsdokument“.

(3) Exportiert alle vorhandenen Riechtests in eine Excel- Datei.

3.3.2.2 Grafik

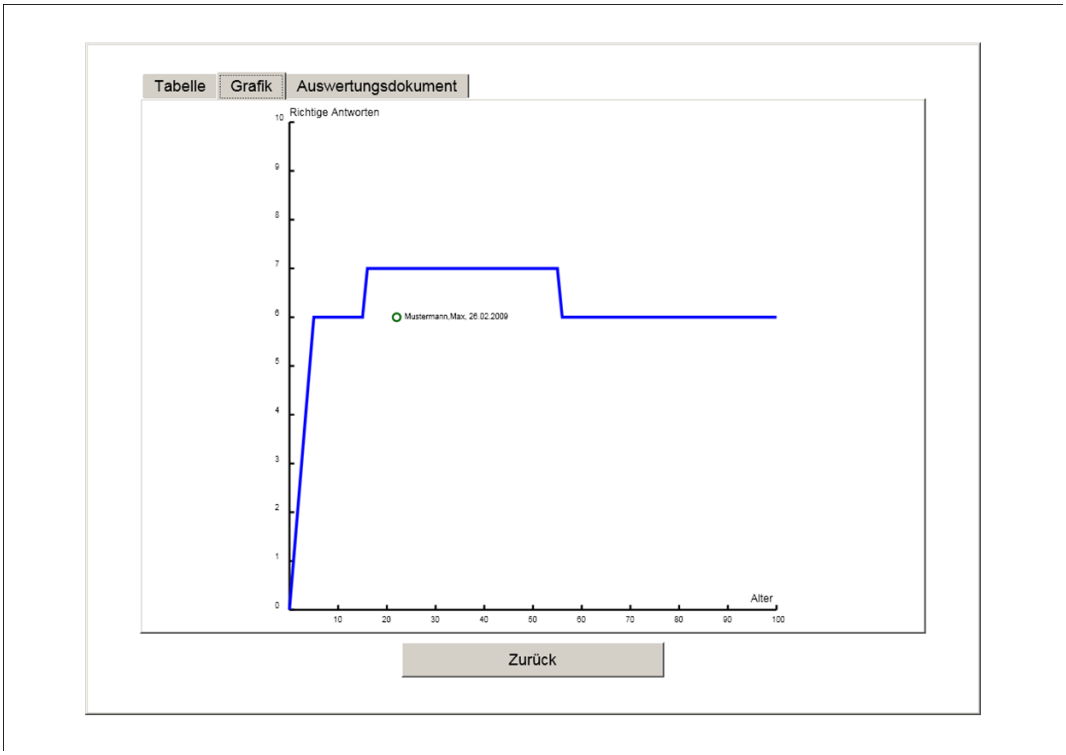


Abb. 3.7: Grafik

Zeigt den Graph für den in der „Liste der Riechtests“ ausgewählten Test an. Wenn mehrere Riechtest ausgewählt wurden, werden mehrere Eintragungen vorgenommen.

3.3.2.3 Auswertungsdokument

Zu dem in der „Liste der Riechtests“ ausgewählten Riechtest wird hier das Auswertungsdokument (vgl. Abschnitt 3.4) angezeigt.

3.3.3 Konfiguration

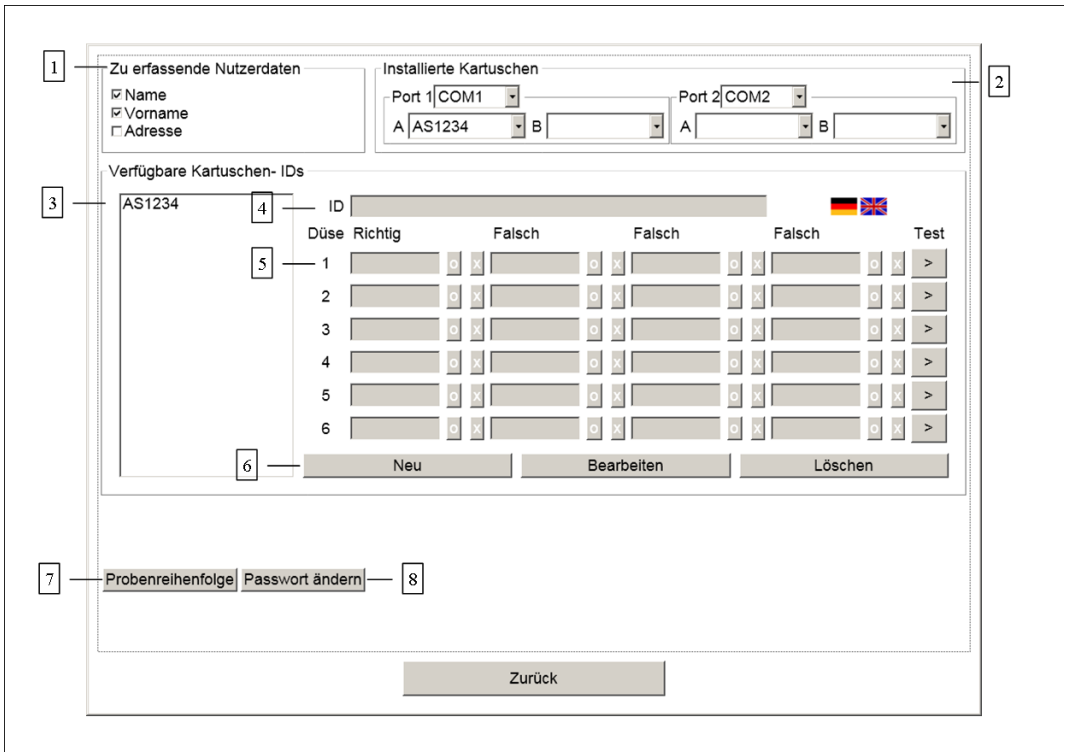


Abb. 3.8: Konfigurationsmaske

Auf dieser Maske kann der Administrator verschiedene Konfigurationen am System vornehmen.

(1) Bestimmt, welche Stammdaten vom Nutzer erfasst werden sollen.

(2) Legt fest, welcher Controller an welchem Port angeschlossen ist und was für Kartuschen an den jeweiligen Decks eingelegt sind.

(3) Zeigt die IDs aller bisher angelegter Kartuschenbeschreibungen an. Abhängig von der Auswahl werden die Daten zur gewählten Kartusche in (4) und (5) angezeigt.

(4) ID der in (3) ausgewählten Kartusche.

(5) Informationen zu den Antwortmöglichkeiten für jede der sechs Düsen der ausgewählten Kartusche. Folgende Informationen sind zu jeder Düse angezeigt:

- richtige Antwort
- 3 falsche Antworten
- zu jeder Antwort kann ein Bild zugeordnet werden (Button mit Kreis)

Ob ein Bild zugeordnet ist, wird durch die Farbe des Zuordnungsbuttons (o) bestimmt. Grau unterlegt bedeutet, dass ein Bild zugeordnet ist. Orange unterlegt bedeutet, dass kein Bild zugeordnet ist. Ein zugeordnetes Bild kann mit dem Löschbutton (X) wieder entfernt werden.

(6) **Neu** Neue Kartuschenbeschreibung anlegen.

Bearbeiten Vorhandene Kartuschenbeschreibung bearbeiten.

Löschen Vorhandene kartuschenbeschreibung löschen. (Ist nicht möglich, wenn bereits Test mit Gerüchen aus der Kartusche vorhanden ist)

(7) Hier kann festgelegt werden, in welcher Reihenfolge die Proben dem Nutzer angezeigt werden sollen.

(8) Hier kann das Administratorenkennwort geändert werden.

4 Fehler

Fehler	Problem	Lösung
Programm startet nicht	.NET Framework nicht vorhanden oder Version kleiner 3.5	aktuelles .NET Framwork installieren
Datenbank nicht erreichbar	SQL Server läuft nicht	SQL Server installieren oder Dienst starten
	SQL Server nicht erreichbar	prüfen ob SQL Server unter „.\SQLEXPRESS“ erreichbar ist (z. B. mit *SQL Server Management Studio*
	SQL Server verweigert Verbindung	SQL Server muss „Integrated Security“- Verbindungen zulassen und der angemeldete Windowsnutzer benötigt entsprechende Rechte
Riechtest startet nicht	mindestens ein Controller ist nicht angeschlossen	schließen Sie beide Controller an und stellen Sie die entsprechenden Ports ein (vgl. Abschnitt 3.3.3)
	Reihenfolge der Proben (Probensequenz) ist nicht eingestellt	stellen Sie die Reihenfolge der Proben ein (vgl. Abschnitt 3.3.3)
	verwendete Kartusche ist nicht vorhanden	stellen Sie sicher, dass alle in der Probensequenz verwendeten Kartuschen über einen Controller erreichbar sind

A.2 Handbuch „aerome® ScentController 2x6 (AB 60005)“

aerome® scent technology

Montageanweisung (AB60084) für den Einbau des

aerome® ScentController 2x6 (AB 60005) in ein ScentTechnology-System

Version II

Inhalt

Montageanweisung beachten

1 Zu Ihrer Sicherheit

Jede Handhabung bei der Montage setzt die genaue Kenntnis und Beachtung dieser Montageanweisung voraus.

Für den Betrieb ist die entsprechende Gebrauchsanweisung zu beachten.

Allgemeine Hinweise

- Montagearbeiten nur von Fachleuten ausführen lassen
- bei der Montage der Kartuschenaufnahme ist darauf zu achten, daß der Endanwender des ScentTechnology-Systems nicht von hinten in die Kartuschenaufnahme greifen kann.
- Für die Sicherheit, die Zulassung sowie die Normkonformität des ScentTechnology-Systems ist der Final Assembler verantwortlich!
- Beim Zusammenbau des ScentTechnology-Systems sind die geltenden nationalen Norm- und Sicherheitsvorschriften zu beachten.

Elektrozuleitungen verlegen

Es ist sicherzustellen, daß bei der Montage die geltenden nationalen Normen- und Sicherheitsbestimmungen eingehalten werden.

- Elektroarbeiten nur von Fachleuten ausführen lassen.
- Kabel nicht über scharfe Kanten legen.
- Netzleitungen nicht beschädigen.
- Das ScentTechnology-System, in das der Duftgenerator eingebaut wird, muß in seiner Gesamtheit ein Gerät der Schutzklasse 1 sein.
- Ein Berührungsschutz der elektrischen Anschlüsse muß durch den Einbau der Geräte gewährleistet sein.

Pneumatikleitungen verlegen

- Arbeiten nur von Fachleuten ausführen lassen.

- Die pneumatischen Leitungen aus Kunststoff müssen vor Knickung geschützt werden. Geknickte Leitungen müssen ausgetauscht werden.

- Die Duftleitung (Edelstahlrohr) ist dem jeweiligen System anzupassen. Der Biegeradius der Duftleitung muß minimal 20 mm betragen.

- Es ist darauf zu achten, daß durch das Öffnen und Schließen der Kartuschenaufnahme eine Hin- und Her-Bewegung des Rohranschlußes an der Kartuschenaufnahme möglich ist.

- Sind Kunststoffschläuche abzulängen, so ist hierzu ein spezieller Schlauchschneider oder ein sehr scharfes Messer zu verwenden. Dabei ist sicherzustellen, daß sich der Schlauch beim Ablängen nicht verformt.

- Bei Beschädigungen oder Verformungen des Schlauches ist ein neuer Schlauch zu verwenden.

2 Lieferumfang

im Lieferumfang des ScentController 2x6 (AB60005) sind folgende Komponenten enthalten:

2 x Kartuschenaufnahme	AB30922
E-Set 2x6	AB60040
Luftaufbereitung SC	AB31000
Drossel, vollst.	AB60076
Schlauch	1210165
Montageanweisung	AB60084
3 x Rohr	AB60003
Düse	AB30978
T-Verschraubung	AB60049

3 Verwendungszweck

Der ScentController ist Teil eines ScentTechnology- Systems.

ScentTechnology-Systeme dienen der szenen- oder ereignisgenauen Beduftung audiovisueller Wiedergaben am Point of Purchase (PoP), Point of Sales (PoS), Point of Entertainment (PoE) oder Point of Information (Pol).

4 Funktionsprinzip

Der ScentController verfügt über eine Luftversorgung (Kompressor mit Filter usw.) und zwei parallel geschaltete Kartuschenaufnahmen mit dem E-Set (Elektronik-Set). Durch das E-Set werden die Magnetventile der Kartuschenaufnahmen angesteuert.

Das E-Set ist weiterhin mit einer RS232-Schnittstelle (DSUB 9 Pol male) ausgestattet. Hierüber erfolgt die Kommunikation mit der entsprechenden Hardware (PC, CDi) zur Ansteuerung der Magnetventile.

Luftaufbereitung

Die Komponenten der Luftaufbereitung (1) sind in einem Gehäuse untergebracht. Die Luftaufbereitung besteht aus einem Kompressor, einem Lüfter, einem Sicherheitsventil und dem Filter.
Die Drossel (2) muß zwischen dem Filter und den Kartuschenaufnahmen plaziert werden.

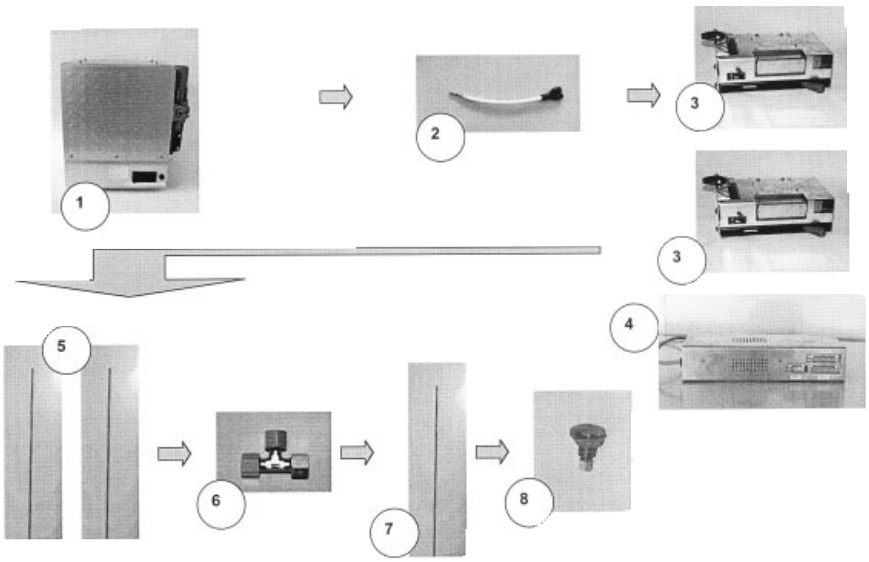
Der Filter dient zur Entfeuchtung und zur Filterung der komprimierten Luft. Er filtert die Luft von Fremdgerüchen aus der Umgebung. Fremdgerüche könnten einen Einfluß auf die Duftqualität haben. Durch die Entfeuchtung der angesaugten Luft sammeln sich Kondensatmengen im unteren Teil der Filtereinheit. Die Abgabe des Kondensats erfolgt beim Herunterfahren des ScentTechnology-Systems durch Drucklosschalten des Filters.

Durch die Drossel wird die Luftmenge (Volumen/Zeit.), der sog. "Air Flow" bestimmt. Der Air Flow beeinflußt die Duftwiedergabe. Der Air Flow darf nur durch von aerome autorisierten Personen verändert werden.

Kartuschenaufnahme und Ausgang aus PoI / PoS

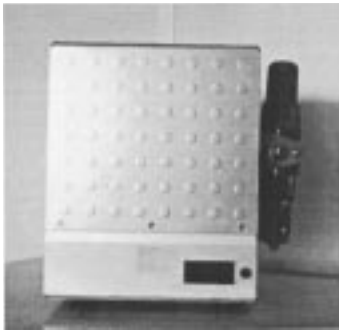
Nachdem die Luft systemspezifisch aufbereitet wurde, wird sie über eine Kunststoffleitung in die Kartuschenaufnahmen (3) geleitet. In der Kartuschenaufnahme wird die aufbereitete Luft mit dem Duft beaufschlagt. Die Ventile der Kartuschenaufnahme erhalten die Steuerimpulse vom E-Set (4), das mittels einer RS232 Schnittstelle angesteuert wird.
Am Ausgang der Kartuschenaufnahme befindet sich ein Verschraubungsanschluß zur Aufnahme der Duftleitung (5). Die Länge der Duftleitungen ist an das spezielle ScentTechnology-System anzupassen. Über ein T-Stück (6) werden die Duftleitungen (5) zusammengeführt und über eine dritte Leitung (7) mit der Düse (8) dicht verschraubt.

Nachfolgend ist eine Gesamtübersicht der einzelnen Bauteile dargestellt.

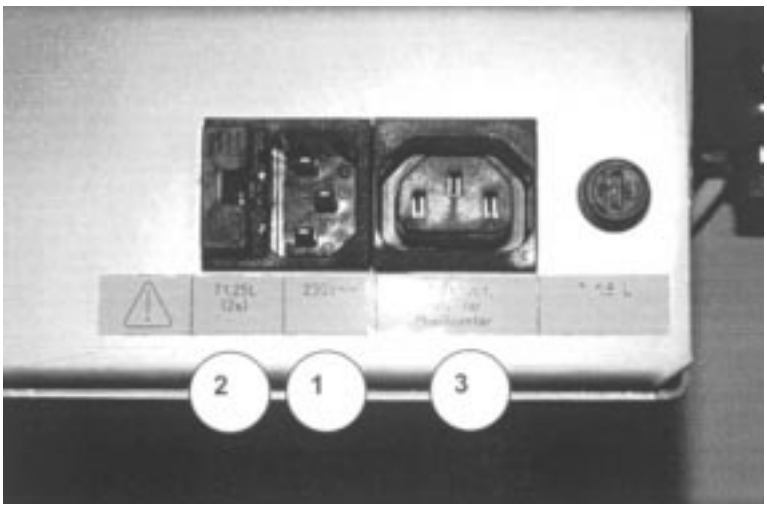


5. Montage

5.1. Luftaufbereitung



- Achtung! Vor Inbetriebnahme der Luftaufbereitung die eingestellte Spannung überprüfen (1)!

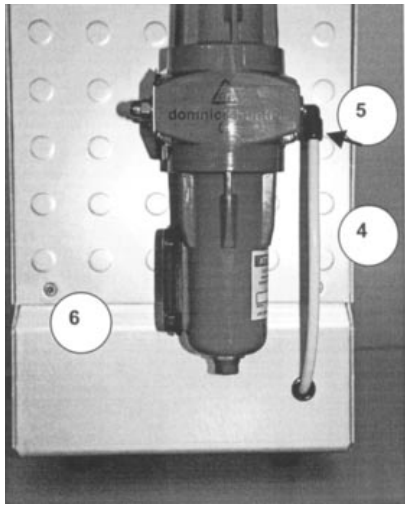


Aufkleber gegebenenfalls mit beigelegtem Aufkleber überkleben.

Sicherungen gem. Aufkleber wechseln (2).

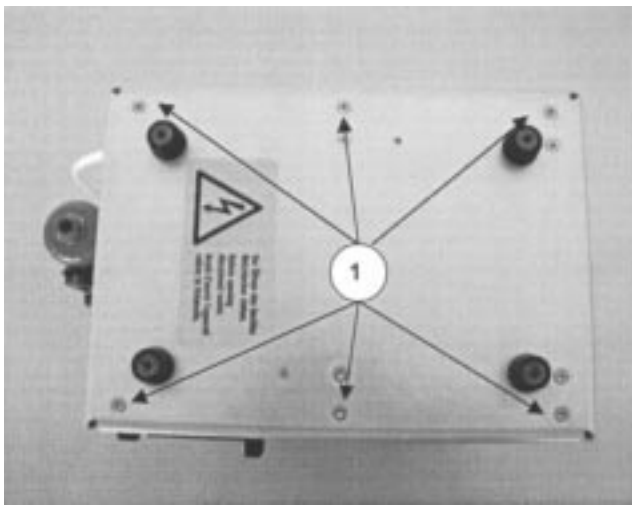
Spannung einstellen:

- Schlauch am Filter abziehen (4), dazu Druckring (5) auf den Einschraubanschluß drücken.



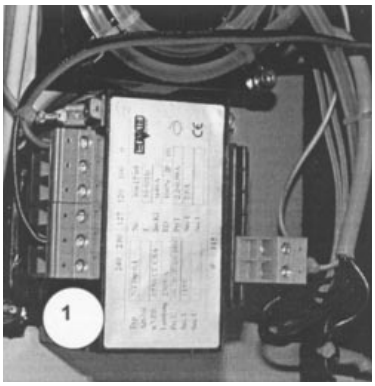
- 4 Schrauben (6) an den Seiten heraus schrauben.

- 6 Schrauben (1) am Bodenblech heraus schrauben.



- Bodenblech abziehen

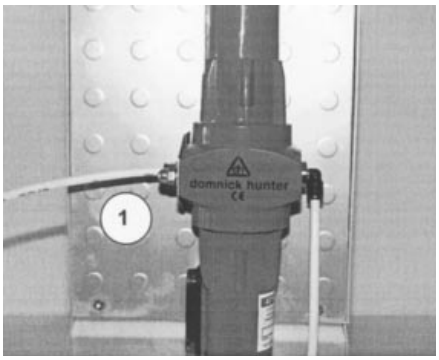
- Spannung entsprechend der Länderspannung an der Primärseite des Trafos (1) einstellen.
  Eingestellte Spannung und Schild am Gehäuse müssen übereinstimmen!



- Bodenblech wieder anschrauben, Schlauch an Filter anschließen.

- Die Luftaufbereitung in das ScentTechnology – System stellen

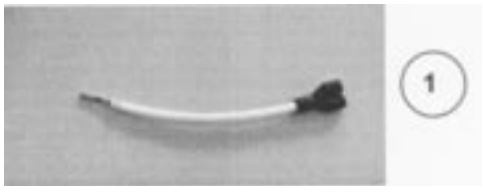
- Schlauch (1) von dem Filter zur Drossel führen



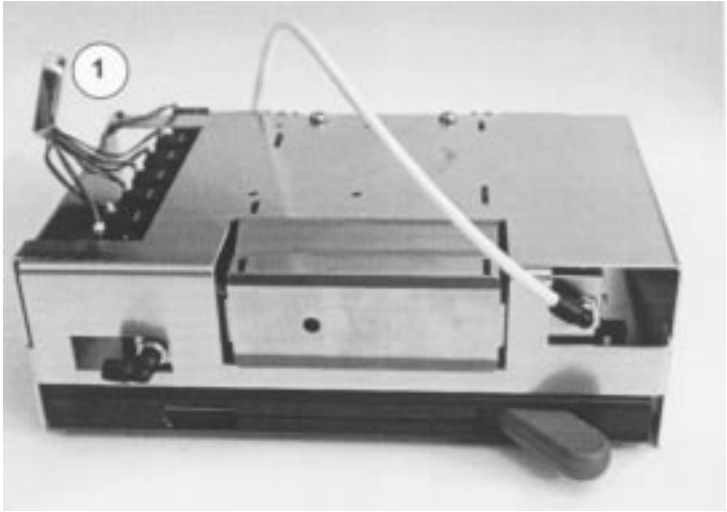
Achtung!

Die Drossel (1) muß in jedem Fall zwischen dem Filter und den Kartuschenaufnahmen montiert werden.

- Keine besondere Einbaulage.



5.2. Kartuschenaufnahme

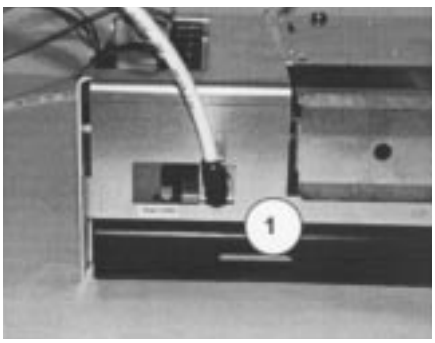


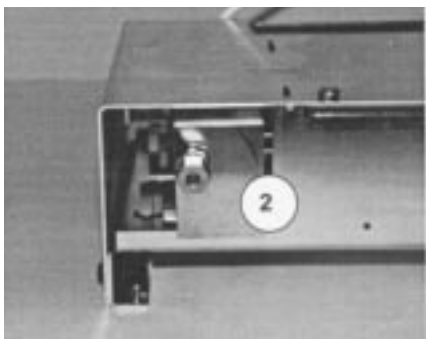
- Die Kartuschenaufnahmen in das ScentTechnology System montieren (Einbauskizze im Anhang). Waagerechten Einbau der Kartuschenaufnahmen beachten.

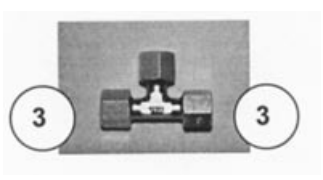
- Stecker (1) (25 Pol D-SUB) mit E-Set verbinden, auf Zuordnung achten (ScentDeck A bzw. B)

- Druckluftanschluß (Steckanschluß) (1) mit Zuleitung von der Drossel verbinden.

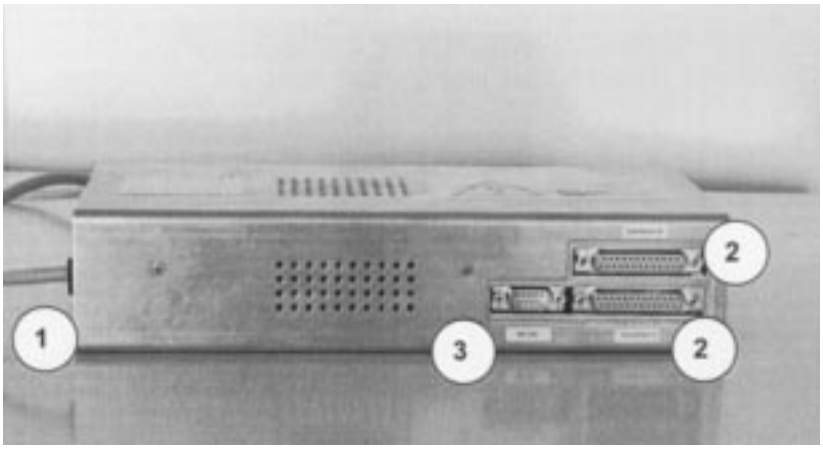
- Duftleitungen mit der Kartuschenaufnahme (2) dicht verschrauben, am T-Stück die beiden Duftleitungen zusammenführen (3)







5.3. E-Set 2x6:



- Das Netzkabel (1) mit einem landesspezifischen Stecker ausstatten.

- Montage in das ScentTechnology-System gemäß Maßblatt im Anhang. Lüftungsbohrungen nicht verdecken. Keine besondere Einbaulage erforderlich.

- Anschluß für Ventile der Kartuschenaufnahmen (25 pol. Sub-D- Stecker / female) (2). Auf Zuordnung achten (ScentDeck A bzw. B).

- RS 232 Schnittstelle (9 pol. Sub-D-Stecker / male) (3). Anschluß über abgeschirmtes Kabel!

Der Anschluß "fan" ist nur für den aerome-Lüfter AB30980 zu nutzen.

Achtung!

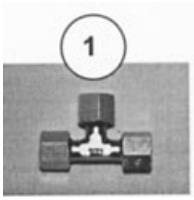
Da die Duftleitung die Bewegung des Einschraubanschlusses (ca. 30 mm) der Kartuschenaufnahme aufnehmen muß, ist auf eine hinreichende Bewegungsfreiheit zu achten.

5.4. Duftdüse und Rohr

- Bohrung (gem. Zeichnung im Anhang) in das Gehäuse des ScentTechnology Systems bohren (Ø 26,5 +0,5 / SW22). Die Düse mit der Mutter und der Scheibe fest mit dem ScentTechnology System verschrauben.

- Die Duftdüse muß so positioniert werden, daß sie im eingebauten Zustand auf die Nase des Benutzers gerichtet ist. Der Abstand zwischen Duftdüse und Nase soll nicht mehr als 600 mm betragen.

- Die Duftleitung (Øinnen 4 / Øaußen 6 / 980 lang) mit dem T-Stück (1) und der Duftdüse dicht verschrauben.



- Der Biegeradius der Duftleitung muß mindestens 20 mm betragen.

6. Schnittstelle RS 232 zur Ansteuerung der Kartuschenaufnahme

6.1. Beschreibung der Programmierung

Die folgenden Befehle müssen immer als Datenblock übertragen werden: Hex 1B xx xx 0D

- Initialisierung: Hex E0,E1,E2,E3
- z.B. Duftventil 1 öffnen: Hex 40
- zeitgleich Lüftungsventil schalten: Hex 26
- Duftzeit einstellen
- Duftventil 1 schließen: Hex C0
- zeitgleich Lüftungsventil schalten: Hex A6

Spätestens 20 sec. nach dem letzten Befehl werden die Ventile stromlos geschaltet. Das bedeutet, daß die Beduftung max. 20 sec. stattfinden kann.

Achtung!
Zum Beduften müssen beide Spülventile (V7) geschlossen sein!

6.2. Auflistung der Befehle und Funktionen:

- Die Befehle müssen immer in einem Datenblock übertragen werden.
- Die Blöcke werden durch ein Blockanfang- und ein Blockende-Zeichen voneinander getrennt.
- Ein Block darf aus maximal 20 Zeichen bestehen.
- Zwischen den Blöcken muß eine Pause von 120ms liegen.

Die folgenden Befehlscodes sind in Hex angegeben.

Befehl	Beschreibung
E0,E1,E2,E3	Aktivierung der LP-Steuerung
EE, EF	alle Ventile in Ruhestellung
1B	Steuerzeichen Blockanfang <ESC>
0D	Steuerzeichen Blockende <CR>

Befehl	Beschreibung
ScentDeck A	
40	Duftventil 1 an
41	Duftventil 2 an
42	Duftventil 3 an
43	Duftventil 4 an
44	Duftventil 5 an
45	Duftventil 6 an
26	Spülventil an (V7)
C0	Duftventil 1 aus
C1	Duftventil 2 aus
C2	Duftventil 3 aus
C3	Duftventil 4 aus
C4	Duftventil 5 aus
C5	Duftventil 6 aus
A6	Spülventil aus (V7)
ScentDeck B	
50	Duftventil 1 an
51	Duftventil 2 an
52	Duftventil 3 an
53	Duftventil 4 an
54	Duftventil 5 an
55	Duftventil 6 an
36	Spülventil an (V7)
D0	Duftventil 1 aus
D1	Duftventil 2 aus
D2	Duftventil 3 aus
D3	Duftventil 4 aus
D4	Duftventil 5 aus
D5	Duftventil 6 aus
B6	Spülventil aus (V7)

6.3. Beschreibung der Schnittstelle

1. Übertragungsformat:

Die Übertragungsgeschwindigkeit beträgt 9600 bit/s,1 Start-, 8 Daten-, 1 Stop-Bit, keine Parität.

2. Anschluß an der Leiterplatte:

Die Steuerelektronik der Kartuschenaufnahme ist mit einem 9-poligen SUB-D-Stecker (male) versehen. Die Pinbelegung des Steckers ist wie folgt:

2=TxD
3=RxD
5=DGND

Jede ScentCartridge darf nur einmal eingesetzt werden !

7 Inbetriebnahme des Gerätes

- Zum Einlegen der ScentCartridge grünen Hebel herausziehen (1) und nach links schwenken (2).

- ScentCartridge in die Kartuschenaufnahme einlegen.
- Beim Einlegen der Kartuschenaufnahme Pfeil auf ScentCartridge beachten.
- Hebel nach rechts schwenken (1) (schwerer als beim Öffnen, da ScentCartidge geöffnet wird), grünen Hebel wieder hineindrücken (2).

Bei Erstanschluß die mitgelieferte Reinigungskartusche einlegen und mit beliebiger Software einige Male betätigen !

Nur Orginal aerome ScentCartridges verwenden!

Achtung!
Nicht in geöffnete Klappe fassen – Verletzungsgefahr!

8 Hersteller

aerome® GmbH Scent Technology
Neuer Zollhof 1
D -40221 Düsseldorf
Germany

phone: +49. 211 / 944 88.0
fax: +49. 211 / 944 88.44
web: www.aerome.com
email: aerome@aerome.com

9 Technische Daten

Umgebungsbedingungen für den Betrieb:

Temperatur:	+10 bis +30°C
rel. Luftfeuchte:	20 bis 90% (keine Betauung !)
Luftdruck:	900 bis 1100hPa

Temperatur an der Duftkartusche während des Betriebes:	max. 35°C

Umgebungsbedingungen für die Lagerung:

Temperatur:	+5 bis +55°C Temperaturstürze sind zu vermeiden!
rel. Luftfeuchte:	20 bis 90% (keine Betauung)
Luftdruck:	900 bis 1100hPa

Gewicht :	ca. 24 kg

Netzanschluß :

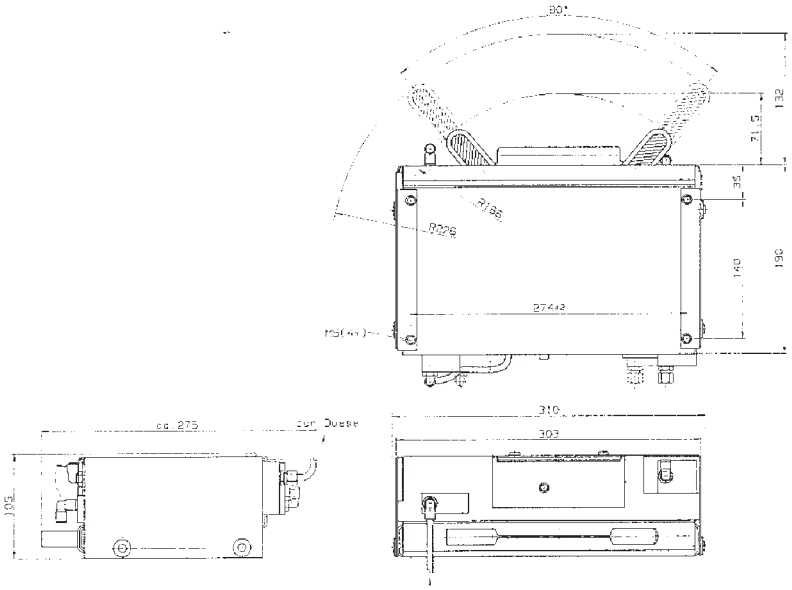
Kompressor:	115 V ; 50 / 60 Hz
E-Set:	100 – 250 V; 50-60Hz

Leistungsaufnahme :	ca. 220 W

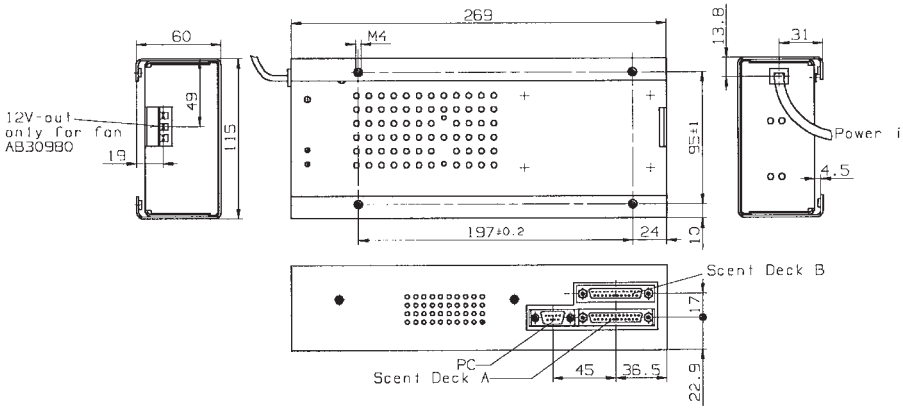
10 Installationsmaße

Installation dimensions in mm

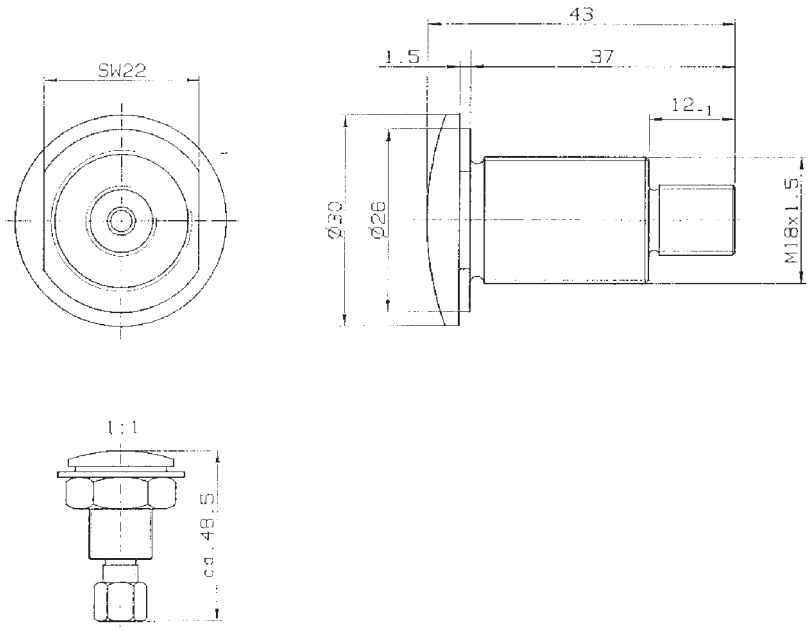
10.1 Kartuschenaufnahme AB30922



10.2 E-Set 2x6 AB60040



10.3 Düse AB30978



A.3 Auswertungsdokument „Olaf - Riechtest mit Sniffin' Sticks“

Olaf - Riechtest mit Sniffin' Sticks

Schwelle

	1	2	3	4	5	6	7
VS 1							
VS 2							
VS 3							
VS 4							
VS 5							
VS 6							
VS 7							
VS 8							
VS 9							
VS 10							
VS 11							
VS 12							
VS 13							
VS 14							
VS 15							
VS 16							
WdP	1	2	3	4	5	6	7

bei VS

Diskrimination

1	blau	grün	rot	0
2	blau	grün	rot	0
3	blau	grün	rot	0
4	blau	grün	rot	0
5	blau	grün	rot	0
6	blau	grün	rot	0
7	blau	grün	rot	0
8	blau	grün	rot	0
9	blau	grün	rot	0
10	blau	grün	rot	0
11	blau	grün	rot	0
12	blau	grün	rot	0
13	blau	grün	rot	0
14	blau	grün	rot	0
15	blau	grün	rot	0
16	blau	grün	rot	0

Identifikation

1	• Orange	Brombeere	Erdbeere	Ananas	1
2	Rauch	Schuhleder	Klebstoff	• Gras	0
3	Honig	Vanille	• Schokolade	Zimt	0
4	Schnittlauch	Pfefferminz	• Fichte	Zwiebel	0
5	Kokos	Banane	Walnuss	• Kirsche	0
6	Pfirsich	Apfel	Zitrone	• Grapefruit	0
7	• Lakritz	Gummibärchen	Kaugummi	Kekse	1
8	Senf	• Gummi	Menthol	Terpentin	0
9	Zwiebel	• Sauerkraut	Knoblauch	Möhren	0
10	Zigarette	Kaffee	• Wein	Kerzenrauch	0
11	Melone	• Pfirsich	Orange	Apfel	0
12	Gewürznelke	Pfeffer	Zimt	• Senf	0
13	Birne	Pflaume	• Pfirsich	Ananas	0
14	Kamille	• Himbeere	Rose	Kirsche	0
15	Anis	• Rum	Honig	Fichte	0
16	Brot	Fisch	• Käse	Schinken	0

2

Untersuchung am
21.01.09

Patient
Mustermann Max, *
22.06.1986
22 Jahre, männlich

Untersuchungsmodus
Seite: beidseits
Abschwellung: nein
Testbatterie: Identifikation

Ergebnis
Identifikation 2

V.a. pathologischen Befund

Bemerkung

Unterschrift
Untersucher

Schwellentestung: "Triple forced choice"-Test in Eingabelungstechnik
Diskriminationstest: 16 Riechproben;
"Triple forced choice" Test
Identifikationstest: 16 Riechproben, verbale Identifikation aus Listen mit je 4 Begriffen
SDI-Wert: Summe der Testergebnisse aus der Schwellen-, der Diskriminations- und Identifikationstestung;
Normosmie >30, Anosmie < 16

Zeichen- und Abkürzungserklärung:
VS = Verdünnungsstufe
WdP = Wendepunkt
x = erkannt; o = nicht erkannt
! = Wendepunkt
grau hinterlegt = korrekte Antwort
ausgefüllte Kreise = erhaltene Antwort
0/1 = Punktwert falsch/richtig

zuletzt geändert am 28. 01. 2009