Planung und Durchführung eines Software Engineering Projektes am Beispiel eines Internetportals aus dem Bereich Sport

Wissenschaftliche Arbeit im Fach Informatik

Autor: Thomas Hahn

Studiengang Lehramt an Gymnasien

Semester: Wintersemester 2009/2010

Betreuerin: Prof. Dr. Barbara Paech

Lehrstuhl für Softwaresysteme

Universität Heidelberg

Erklärung

Ich erkläre, dass ich die Arbeit selbständig angefertigt und nur die angegebenen Hilfsmittel benutzt habe. Alle Stellen, die dem Wortlaut oder dem Sinn nach anderen Werken, gegebenenfalls auch elektronischen Medien, entnommen sind, sind von mir durch Angabe der Quelle als Entlehnung kenntlich gemacht. Entlehnungen aus dem Internet sind durch Ausdruck belegt.

Schifferstadt den 31.03.2010

Thomas Hahn

Inhaltsverzeichnis

Abstract

Die vorliegende Arbeit thematisiert und erschließt den an der Universität Heidelberg gelehrten Software Engineering Prozess (TRAIN Prozess) im praktischen Einsatz. Nach einer Einführung erfolgt eine detaillierte Durchführung des Prozesses anhand eines Internet Portalprojektes aus dem Bereich Sport. Als Resultat entsteht eine multi-Tier Anwendungskomponente, die das Registrierungsmanagement für Sportveranstaltungen bereitstellt. Nach einer kritischen Betrachtung des durchgeführten Prozesses werden Vorschläge zur Modifikation von TRAIN für den schulischen Einsatz erarbeitet. Darauf aufbauend wird der Prozess in einer angepassten Version anhand einem thematisch verwandten aber technisch völlig unterschiedlichen Projekt erneut bis zur Implementierungsphase durchlaufen. Das Ergebnis dient als Grundlage für weitergehende Vorschläge, die besonders den Einsatz von Szenarien und systematischen Test berücksichtigen. Das Ergebnis stützt die These, dass ein umfassendes Software Engineering an allgemeinbildenden Gymnasien realisierbar ist.

1. Problembeschreibung und Motivation

Am Lehrstuhl für Software Systeme der Universität Heidelberg wurde zu Lehrzwecken ein praxisnaher Prozess entwickelt (TRAIN[1]-Prozess), mit dessen Hilfe Studenten strukturiertes Vorgehen bei der Softwareentwicklung vermittelt werden soll. Dabei wird der Schwerpunkt auf Anforderungsanalyse, Testen, Rationale und Inspektion gelegt. Das Lehrkonzept ist dabei für eine umfangreiche und intensive Vermittlung innerhalb eines Semesters ausgelegt. Ziel der Arbeit ist es, zu evaluieren wie sich Elemente des TRAIN Prozesses in einem Kleinprojekt, wie es sich typischerweise auch in schulnahen Szenarien wiederfindet, einsetzen und unter Berücksichtigung des engen schulischen Zeitrahmens in den Unterricht übertragen lassen. Um ein möglichst umfassendes Verständnis des zugrunde gelegten Software Engineering Prozesses (TRAIN Prozess) zu erhalten wird zunächst an einer Anwendungskomponente der komplette Prozess durchlaufen. Im Anschluss erfolgt eine Untersuchung des Prozesses im Hinblick auf die Anwendbarkeit der Teilschritte in einem schulischen Kontext. Als Resultat dieser Untersuchung wird ein Vorschlag erarbeitet, wie der Prozess oder einzelne Elemente in der Schule eingesetzt werden können. Dies wird dann anhand einer weiteren Anwendungskomponente durchgeführt und erläutert. Die Erfahrungen die während dieser Durchführung gesammelt werden, dienen als Grundlage für eine abschließende Betrachtung der Einsatzmöglichkeiten im Unterricht und einer weiteren Optimierung des Prozessvorschlags. Als Ausgangspunkt für das Rekonstruieren des Software-Engineering Prozesses dient eine Beispielanwendung aus dem Bereich Sport, die im Verlauf der Arbeit entwickelt wird. Die der Arbeit zugrunde liegende Problemstellung wird dabei Folgende sein.

In der Metropolregion Rhein-Neckar wird seit 2005 der „BASF Triathlon-Cup Rhein-Neckar“[2] veranstaltet. Die Veranstaltungsreihe[3], bei der sowohl die besten deutschen Triathleten, aber auch Freizeitsportler antreten, besteht aus mehreren organisatorisch unabhängigen Einzelveranstaltungen, unter Anderem dem „HeidelbergMan“. Seit 2008 wird parallel dazu ein Schüler- und Jugendcup veranstaltet. Die teilnehmenden Triathlonveranstaltungen variieren in Anzahl und Ausrichtung, jede der Veranstaltungen kann verschiedene Altersklassen umfassen, daher ist es möglich, dass eine Veranstaltung entweder am Schüler-und Jugendcup, dem Erwachsenencup oder beiden Cupwertungen beteiligt ist. Dabei müssen nicht notwendigerweise alle Altersklassen gestartet werden[4]. Um die Anmeldung der Teilnehmer können sich die Veranstaltungen selbst kümmern oder diese Aufgabe an einen beliebigen Zeitnahmedienstleister delegieren[5]. Abhängig von dem gewählten Verfahren, liefern die Veranstaltungen die Anmeldedaten an den Dienstleister weiter und erhalten von diesem die Ergebnisse, oder sie erhalten vom Zeitnahmedienstleister sowohl Anmeldedaten als auch Ergebnisse. Nach den einzelnen Veranstaltungen wird in einem veranstaltungsübergreifenden Prozess auf Basis einer Formel die Punktzahl der Veranstaltung, nach Wertungskategorie getrennt, manuell errechnet, die Gesamtpunktzahl händisch aktualisiert und auf der Internetseite des Cups publiziert.

---

1 TRAIN ist eine Abkürzung für „**T**esten, **Ra**tionalen und **In**spektion“.

2 Der Internetauftritt des BASF Triathlon-Cup Rhein-Neckar findet sich unter: http://www.rhein-neckar-triathlon-cup.de/.

3 Aufgrund der Komplexität der hier vorgestellten Zusammenhänge sei zur Begriffsbestimmung auf den Glossar verwiesen.

4 Vgl. Abbildung 1. In der genannten Reihenfolge Einzelveranstaltung1, Einzelveranstaltung3, Einzelveranstaltung2.

5 Vgl. Abbildung 1. In der genannten Reihenfolge Einzelveranstaltung1/Einzelveranstaltung3 und Einzelveranstaltung2.

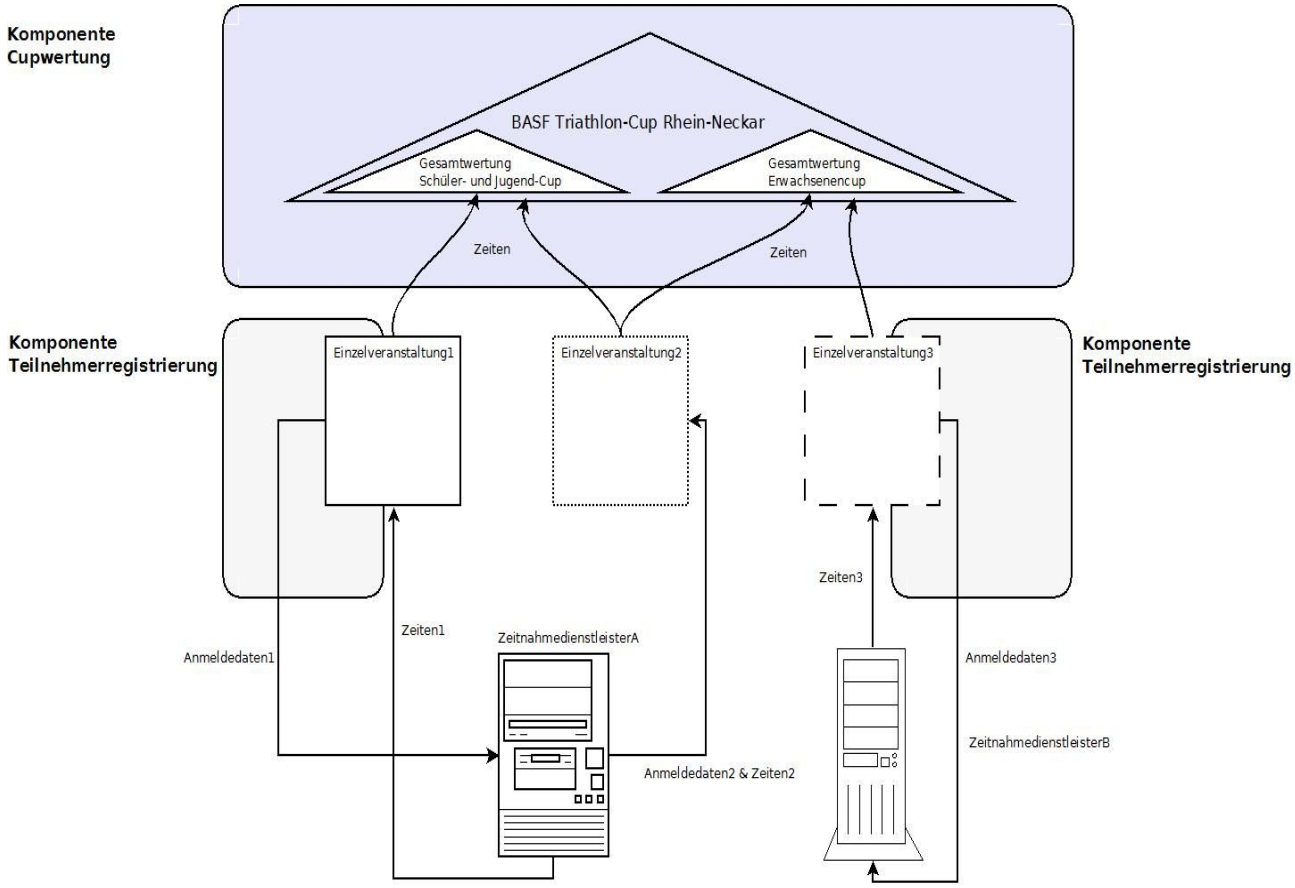


Abbildung 1: Problemstellung

Die Arbeitsabläufe bieten großes Potential zur Vereinfachung und Optimierung durch den Einsatz von IT-Systemen, so ist insbesondere der übergreifende Prozess zeitintensiv und durch viele manuelle Schritte fehleranfällig. So müssen immer wieder Korrekturen an den Ergebnissen und Zwischenergebnissen vorgenommen werden, was eine negative Außenwirkung gegenüber den betroffenen Sportlern hat. Auf der Ebene der Einzelveranstaltungen bieten die von den Zeitnahmedienstleistern zur Verfügung gestellten Registrierungsmöglichkeiten den Veranstaltern zwar eine Erleichterung für die Verwaltung des Wettkampfes, sind aber aufgrund ihrer generischen Konzeption komplex, umständlich zu bedienen und kaum an individuelle Anforderungen anpassbar. Eine papierbasierte oder nicht vernetzte Verwaltung der Anmeldungen scheidet aufgrund des hohen Aufwands für Vorbereitung, Durchführung und Nachbereitung des Wettkampfs ebenfalls aus[6].

Der Vorteil des beschriebenen Szenarios ist, dass sich zwei Teilanwendungen mit überschaubarem Umfang gut voneinander trennen lassen, was sie zu einem idealen Untersuchungsgegenstand macht. Im Verlauf dieser Arbeit soll die bestehende Vorgehensweise weiterentwickelt und mit Hilfe eines internetbasierten Systems unterstützt werden, was besonders der organisatorischen Unabhängigkeit der Einzelveranstaltungen Rechnung trägt. Es soll daher eine Anwendung entwickelt werden, die aus einer veranstaltungsspezifischen Komponente „**Teilnehmerregistrierung**“ sowie einer veranstaltungsübergreifenden Komponente “**Cupwertung**“ besteht (vgl. Abbildung 1). Durch die Teilapplikation „Teilnehmerregistrierung“ soll die Verwaltung einer Einzelveranstaltung erleichtert werden, während es der Teil „**Cupwertung**“ ermöglicht, die Daten des Zeitnahmedienstleiters, die von den Einzelveranstaltungen wei-

6 Die einzelnen Wettkämpfe umfassen mehrere hundert Teilnehmer.

tergemeldet werden, flexibel entgegenzunehmen, die Punktzahl der Einzelveranstaltungen zu errechnen, die Gesamtwertung entsprechend anzupassen und das Resultat auf der Internetseite des Cups zu veröffentlichen.
Im Folgenden wird ein Überblick über den zugrundegelegten Software Engineering Prozess (TRAIN) gegeben, bevor er dann am Beispiel der Komponente Teilnehmerregistrierung umgesetzt wird.

2. Software Engineering mit dem TRAIN Prozess

Der TRAIN-Prozess[7] besteht aus vier großen Arbeitsschritten. Auf eine ausführliche Requirements Engineering Phase, folgen Architekturdefinition, Feinentwurf und Implementierung, die im Wesentlichen sequentiell bearbeitet werden[8].

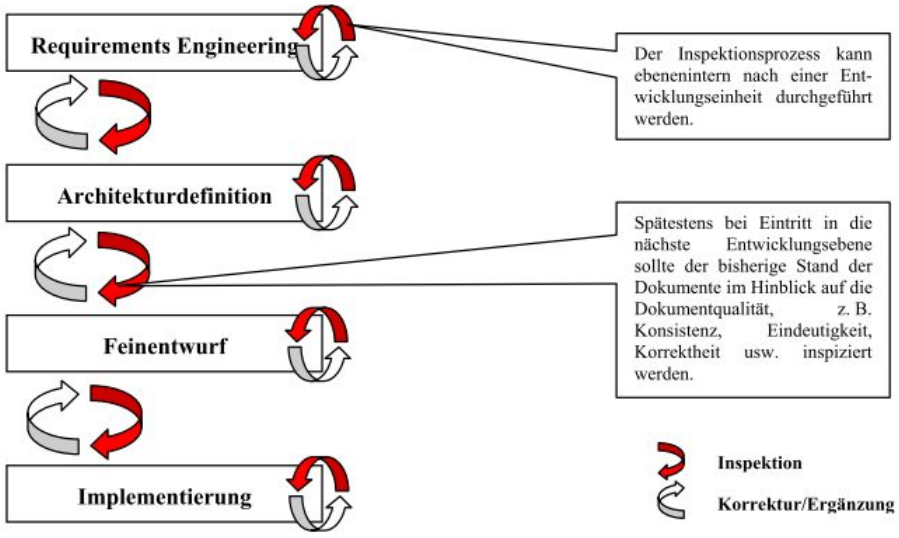


Abbildung 2: Phasen des TRAIN-Prozesses und Inspektionsintegration (aus: Häfele 2005: 8)

Requirements Engineering (Anforderungsanalyse)

Als Basis für die Strukturierung, der in der Anforderungsanalyse zu treffenden Entscheidungen (vgl. Abbildung 3), dient der TORE[9] Prozess. In diesem Prozess werden auf der *Aufgabenebene* die Rollen der handelnde Akteure sowie Aufgaben, die sich aus Geschäftsprozessen ergeben, betrachtet. Eine Betrachtung des zukünftigen IT-Systems wird dabei praktisch vollständig vermieden. Dies erfolgt erst auf der *Domänenebene.* Dort entscheidet sich, wie das gegenwärtige Vorgehen (Ist-Prozess) durch Einführung eines IT-Systems ändert (Soll-Prozess) und welche Aktivitäten vom System durchgeführt werden sollen (Systemverantwortlichkeiten), sowie welche Daten (Domänendaten) im System verwendet werden. Die *Interaktionsebene* beschreibt die Interaktion mit dem System. Dies beinhaltet die Strukturierung der Arbeitsbereiche, die Festlegung von Aktivitäten mit dem System und die ausgetauschten Daten (Interaktionsbeschreibung) sowie welche Systemfunktionen bereitgestellt werden sollen. Die unterste Analyseebene stellt schließlich die *Systemebene* dar, die sich in zwei Bereiche unterteilen lässt. Im ersten Bereich wird die graphische Benutzungsschnittstelle (GUI) definiert. Dies umfasst beispielsweise Überlegungen zu Bildschirmstruktur und Navigation. Der zweite Bereich thematisiert den Systemkern. Er hat eine Brückenfunktion von der Anforderungsanalyse zum objektorientierten Entwurf und ist die Basis, auf der Feinentwurf und teilweise auch Architekturdefinition

7 Eine ausführliche Darstellung des Prozesses an einer Beispielanwendung stellt Philipp Häfele in seiner Bachelorarbeit bereit (vgl. Häfele 2005). Eine Übersicht über den gesamten Prozess und den jeweiligen Artefakten, die dieser Arbeit entnommen sind, findet sich in Anhang A.

8 Ein streng sequentielles Vorgehen ist jedoch weder bei den Phasen des TRAIN-Prozesses, noch bei den Phasen des Requirements Engineering zwingend erforderlich.

9 TORE ist eine Abkürzung für Task Oriented Requirement Engineering (vgl. Paech/Kohler 2003).

aufsetzen.

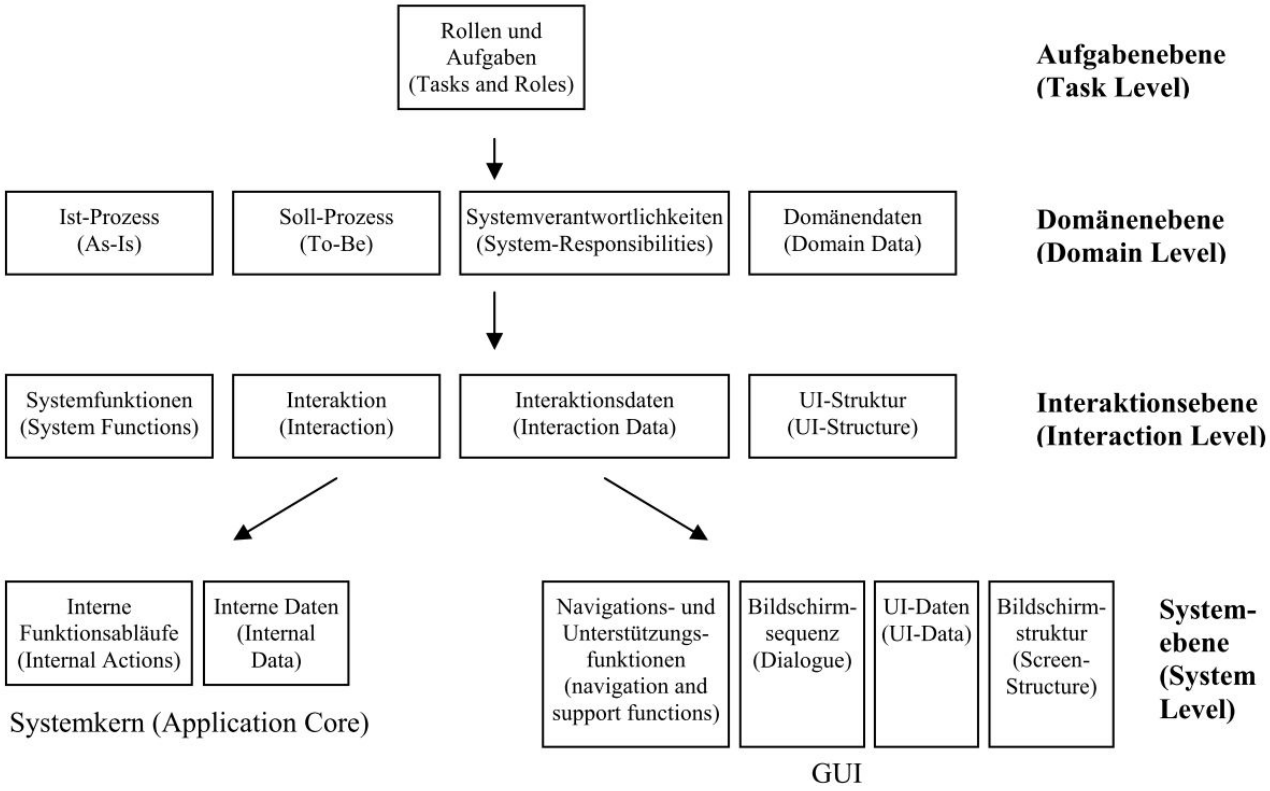


Abbildung 3: Phase1-Entscheidungsebenen der Anforderungsanalyse (aus: Häfele 2005: 11)

Auf allen Ebenen der Anforderungsanalyse erfolgt eine Berücksichtigung von Nicht-funktionalen Anforderungen. Hinzu kommen in TRAIN noch Elemente des GUI Designs, ein objektorientiertes und klassenbasiertes Vorgehen, eine Betonung der Dokumentierung von getroffenen Entscheidungen (Rationale), Inspektionen (vgl. Abbildung 2) der Spezifikationsdokumente sowie ausführliche Tests auf allen Ebenen (vgl. Borner et al. 2006: 2f). Bei der Konzeption und Durchführung der Tests wird in TRAIN eine enge Verzahnung von Entwicklungsprozess und Test auf Basis des W-Modells[10] gewählt, bei der die Testplanung und -spezifikation in der Planungsphase auf jeder Ebene integriert und in der Realisierungsphase auf jeder Ebene ausgeführt wird (vgl. Abbildung 4).

10 In Anlehnung an Spillner (vgl. Spillner 2002). Dort werden auch verschiedene andere Entwicklungsprozessmodelle verglichen, beispielsweise das Wasserfallmodell oder das V-Modell.

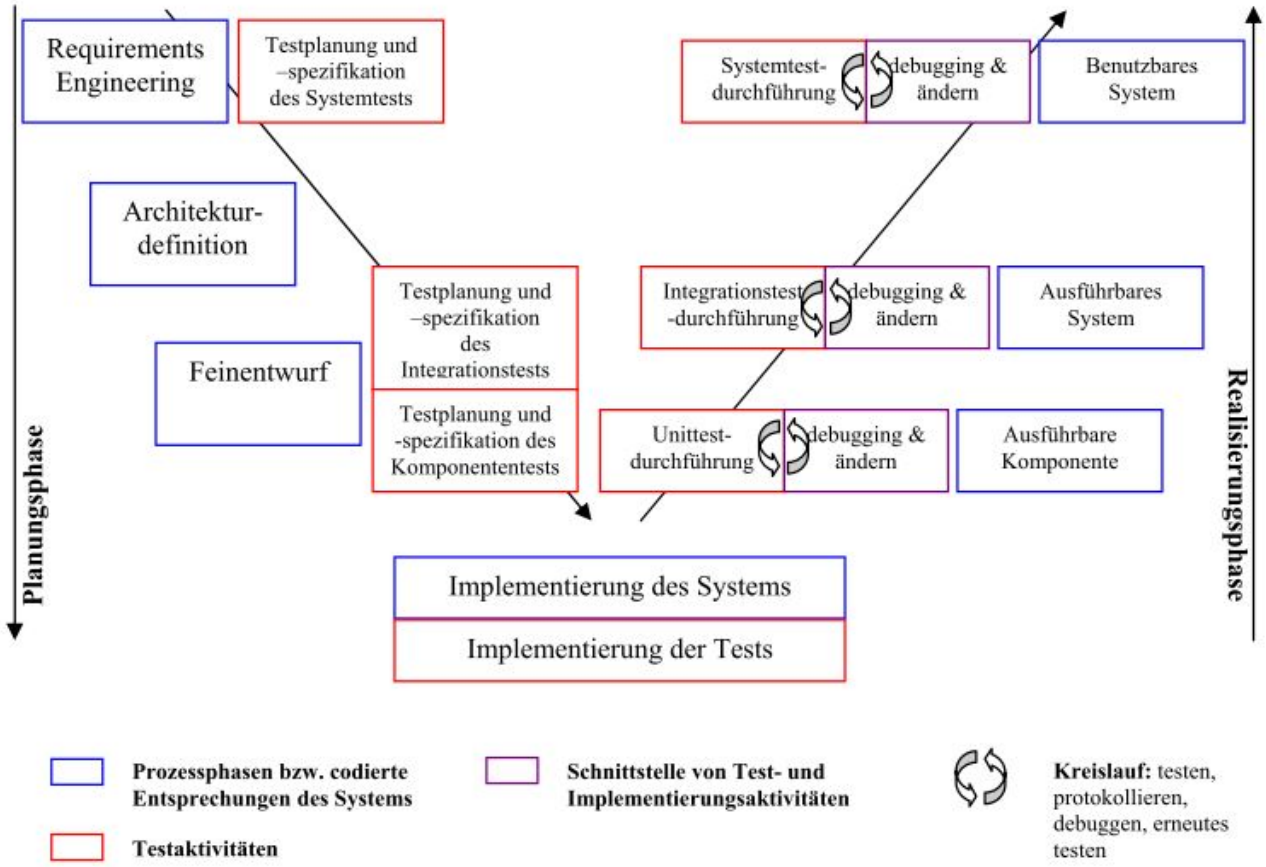


Abbildung 4: Testen im TRAIN-Prozess

3. Teilapplikation Teilnehmerregistrierung

Nachdem bisher der Software Engineering Prozess allgemein dargestellt wurde, erfolgt nun eine Umsetzung am Beispiel der Komponente Teilnehmerregistrierung[11]. *Die Komponente soll es Wettkampfteilnehmern ermöglichen sich unter der Angabe von verschiedenen Daten anzumelden. Wettkampfteilnehmer[12] sind Einzelstarter oder Staffeln die aus drei Startern bestehen. Bei Staffeln fungiert ein Teilnehmer als Staffelkontakt und führt die Registrierung durch. Die Daten werden beim Eintragen durch den Startlistenverwalter einer Plausibilitätsprüfung unterzogen. Sind alle Startplätze bereits vergeben, wird der Teilnehmer auf eine Warteliste gesetzt, die je nach Wettkampfart eine unterschiedliche Anzahl von Plätzen haben kann. Sind alle Wartelistenplätze vergeben, wird der Teilnehmer nicht mehr angenommen. Auf Anfrage kann eine Starterliste der bestätigten Starter in teilanonymisierter Form eingesehen werden. Sofern Änderungen an den Daten notwendig sind, muss der Startlistenverwalter kontaktiert werden oder unabhängig von einer Anfrage selbst tätig werden. Der Startlistenverwalter hat die Möglichkeit die Daten aller Teilnehmer zu bearbeiten oder zu entfernen. Auf Anfragen erfolgt eine Antwort durch den Startlistenverwalter.*

11 Die folgende Beschreibung ist zugleich die Beschreibung des Ist-Prozesses nach TRAIN. Eine darüber hinausgehende Darstellung erfolgt in Abschnitt 3.1.3.

12 Aus Vereinfachungsgründen wird im Folgenden einheitlich die männliche Form verwendet.

3.1 Anforderungsanalyse

Ausgehend von dieser grundlegenden Problembeschreibung wird eine Präzisierung vorgenommen. Hierzu werden im Rahmen des TRAIN Prozesses zunächst Aufgaben und die beteiligten Rollen definiert.

3.1.1 Aufgabenebene

Rollen

Die Teilnehmerregistrierung umfasst drei Rollen, den Starter, den Staffelkontakt und den Startlistenverwalter. Abbildung 5 zeigt exemplarisch die Rollenbeschreibung für den Starter[13].

Rollenbeschreibung (Actor)	
Name	**Starter**
Verantwortlichkeiten	Der Starter registriert sich mit seinen Daten für einen Start bei einer Einzelveranstaltung.
Erfolgskriterien	Starter hat sich registriert
Aufgaben	Sich registrieren Startliste einsehen Anfrage stellen
Kommunikationspartner	Startlistenverwalter
Innovationsgrad	niedrig, da Registrieren eine verbreitete Tätigkeit ist
Rollenprofil	
Wissen/Erfahrung/Fähigkeiten bzgl.	
Aufgaben	Gering-Mittel: Tätigkeit des Registrierens ist eine typische Aufgabe, wenngleich viele Teilnehmer zum ersten Mal einen Triathlon absolvieren und sich dafür registrieren.
Softwaresystem	Gering-Mittel, da die Teilnahme an einer Sportveranstaltung keinen beruflichen/privaten Umgang mit IT Systemen voraussetzt und somit der Grad der bisherigen Nutzung gering sein kann. Erfahrungsgemäß kommen auch Fremdanmeldungen vor, das lässt darauf schließen, dass Starter die zu geringe Kenntnisse aufweisen, die Registrierung delegieren.

Abbildung 5: Rollenbeschreibung Starter (vorläufig)

Auf Basis dieser Rollenbeschreibung ließen sich verschiedene Rolleninstanzen, also konkrete Nutzer, beschreiben. Rolleninstanzen dienen dazu den möglichen Nutzerkreis weiter zu konkretisieren und greifbarer zu machen[14], bringen in unserem Fall jedoch praktisch keine zusätzliche Informationen, da die verschiedenen Nutzertypen bereits implizit in der Rollenbeschreibung berücksichtigt wurden. In diesem Fall wäre es beispielsweise auch möglich, die Rolle „Staffelkontakt“ als Rolleninstanz der Rolle „Starter“ zu modellieren.

13 Alle weiteren Rollenbeschreibungen finden sich im Anhang B.
14 Solche Rolleninstanzen werden auch als „Personas“ bezeichnet.

Aufgaben

Aus der Problembeschreibung ergeben sich zunächst sechs, relativ abstrakte Tätigkeiten, die wie wir jedoch später sehen werden, zu kleinschrittig sind um Aufgaben zu sein

- Starter registrieren
- Staffel registrieren
- Starter bearbeiten/löschen
- Staffel bearbeiten/löschen
- Startliste einsehen
- Anfrage stellen/beantworten

In Abbildung 6 ist die Aufgabenbeschreibung für die Starterregistrierung dargestellt, die auch im weiteren Verlauf beibehalten wird.

Aufgabenbeschreibung (User Task)	
Name	**Starter registrieren**
Verantwortliche Rolle (Initiating Actor)	Starter
Beteiligte Rollen (Participating Actors)	Startlistenverwalter
Aufgabenbewertung	
Ziel (Goal)	Starter ist registriert oder abgewiesen
Eingriffsmöglichkeiten	Keine
Ursachen	Teilnahmewunsch an Triathlonwettkampf
Prioritäten	Hoch, da Datenqualität entscheidend für späteren manuellen Nachbearbeitungsaufwand
Aufgabendurchführung	
Durchführungsprofil (Häufigkeit, Kontinuität, Komplexität):	- Häufig für Gesamtsystem, 1x pro Starter - Abbruch/Unterbrechen durch Nutzer möglich - Niedrige Komplexität
Ausgangssituation (Vorbedingung)	Keine
Info-In	Starterdaten (manuell)
Info-Out	Bestätigung/neue Starterliste
Ressourcen (z. B. Arbeitsmittel etc.)	Anmeldeformular
Qualitätskriterien (Quality Constraints)	
1. Flexibilität	
2. Organisatorischer Zeitaufwand	
3. Datensparsamkeit	Nur die nötigsten Daten werden erfasst
4. Erlernbarkeit	
5. Verständlichkeit	
6. Subjektive Zufriedenheit	
7. Sicherheit	
8. Validierung/Anpassung der Daten	

Abbildung 6: Aufgabe „Starter registrieren“ (vorläufig)

Nicht-funktionale Anforderungen

Die Vorgehensweise soll auf jeden Fall an zukünftige Anforderungen anpassbar sein

(„Flexibilität“) und soll dazu beitragen den organisatorischen Zeitaufwand zu reduzieren. Da es sich bei den Aufgaben „Starter registrieren“ und “Staffel registrieren“ um Aufgaben handelt, die von einer heterogenen Endnutzergruppe des Systems durchzuführen sind, erscheint es außerdem sinnvoll, an dieser Stelle bereits Qualitätskriterien hinsichtlich der Benutzbarkeit festzulegen. Von den sechs Dimensionen von Benutzbarkeit nach Lauesen (vgl. Lauesen 2005: 9ff) sind in diesem Kontext „Erlernbarkeit“ („ease of learning“, „Verständlichkeit“ („understandability“) und „Subjektive Zufriedenheit“ („subjective satisfaction“) interessant, da die Nutzer das System selten benutzen werden und somit faktisch den Umgang immer wieder neu erlernen müssen. Gleichzeitig sollen die Endnutzer zufrieden mit dem gesamten Anmeldungsablauf sein.[15] Für die Aufgaben des Startlistenverwalters sind solche Anforderungen nur bedingt sinnvoll, da es sich bei dieser Rolle um einen einzelnen Expertennutzer handelt, der gut eingearbeitet werden kann.

Aus dem Umfeld, in dem die Applikation eingesetzt werden soll, ergeben sich weitere Nicht-funktionale Anforderungen. Aus den Vorschriften des Bundesdatenschutzgesetzes ergibt sich das Gebot der Datensparsamkeit[16], das somit als Qualitätskriterium aufgenommen werden muss[17]. Die Nutzung des Internets bei der Registrierung erzeugt die Notwendigkeit von Sicherungsmaßnahmen der Eingaben bei Übertragung sowie zum Schutz vor Angriffen auf das System. Aus Datensicht ist es zur Reduzierung von manueller Nacharbeit und Fehlervermeidung sinnvoll, die Datensätze nach der Eingabe zu validieren und gegebenenfalls aufzubereiten.

Inspektion und Rationale

Die Inspektion erfolgt auf Basis der Checkliste „Bereich Rollen, Aufgaben, NFR's, Rolleninstanzen“ (Abbildung 7).

Bei der Abarbeitung der Checkliste fällt bei Punkt 10 auf, dass die Aufgaben „Starter registrieren“ und „Staffel registrieren“ sowie „Starter bearbeiten/löschen“ und „Staffel bearbeiten/löschen“ sich so sehr ähneln, dass sie keinen bedeutenden eigenen Beitrag zur Spezifikation leisten. Sie werden daher im Folgenden unter „Registrieren“ und „Anmeldedaten Bearbeiten/Löschen“ zusammengefasst, was gleichzeitig bedeutet, dass „Staffelkontakt“ mit der Rolle „Starter“ ebenfalls verschmolzen werden kann.

Anders verhält es sich mit den Kontaktfunktionalitäten „Anfrage stellen/beantworten“. Diese leisten zwar prinzipiell einen Beitrag zur Spezifikation, sie können jedoch über verschieden Kommunikationswege bereitgestellt werden. Eine Erörterung der verschiedenen Möglichkeiten erfolgt auf der Domänenebene. Da die Funktionalität eng mit der Tätigkeit des Registrierens verknüpft ist, erfolgt keine Betrachtung als separate Aufgabe.

15 Eine Quantifizierung dieser Anforderungen erfolgt zu einem späteren Zeitpunkt auf der Interaktionsebene (vgl. Kapitel 3.1.3).

16 Bundesdatenschutzgesetz (BDSG), § 3a: „Datenvermeidung und Datensparsamkeit: Gestaltung und Auswahl von Datenverarbeitungssystemen haben sich an dem Ziel auszurichten, keine oder so wenig personenbezogene Daten wie möglich zu erheben, zu verarbeiten oder zu nutzen. Insbesondere ist von den Möglichkeiten der Anonymisierung und Pseudonymisierung Gebrauch zu machen, soweit dies möglich ist und der Aufwand in einem angemessenen Verhältnis zu dem angestrebten Schutzzweck steht“. Einen Kommentar zum BDSG stellt der Bundesbeauftragte für den Datenschutz bereit (vgl. Bundesbeauftragter für den Datenschutz 2007: S. 22/61).

17 Obwohl das BDSG sich auf Datenverarbeitungsanlagen bezieht, können die darin enthaltenen Grundsätze dennoch implizit auf papierbasierte Prozesse angewandt werden.

Checkliste - Bereich Rollen, Aufgaben, NFR's, Rolleninstanzen	
Stellen Sie sich vor, Sie sind ein Auftraggeber und möchten überprüfen, ob ihr Auftrag richtig erfasst worden ist. Überprüfen Sie anhand der Checkliste, wie ihr Auftrag im Dokument umgesetzt worden ist.	
Nr.	**Frage**
1.	**Vollständigkeit:** Sind alle Anforderungen aus der Problembeschreibung durch die Rollen, Aufgaben und NFR's vollständig umgesetzt?
2.	**Vollständigkeit:** Sind für alle Rollen, Aufgaben und NFR's die Textfelder mit sinnvollem Text gefüllt?
3.	**Konsistenz:** Sind die vorhandenen Rollen und Aufgaben konsistent zu der Problembeschreibung?
4.	**Korrektheit:** Sind alle beschriebenen Rollen und Aufgaben durch die Problembeschreibung gefordert?
5.	**Eindeutigkeit:** Sind die beschreibenden Texte einfach, eindeutig und verständlich formuliert?
6.	**Eindeutigkeit:** Sind alle Fachbegriffe im Glossar erläutert?
7.	**Korrektheit:** Ist jede NFR korrekt und sinnvoll verlinkt?
8.	**Prüfbarkeit:** Ist jedes NFR realistisch und eindeutig messbar?
9.	**Vollständigkeit:** Wirkt sich jede NFR auf mindestens ein Anforderungselement (Aufgabe, Use Case, Systemfunktion) aus?
10.	**Interne Konsistenz:** Leistet jede Rolle, jede Aufgabe und jede NFR einen wichtigen, neuen und einmaligen Beitrag zur Anforderungsspezifikation?
11.	**Vollständigkeit:** Ist jede Rolle mindestens für eine Aufgabe verantwortlich oder an mindestens einer Aufgabe beteiligt?
12.	**Vollständigkeit:** Besitz jede Aufgabe eine verantwortliche Rolle?
13.	**Vollständigkeit:** Ist jede Rolleninstanz ein Beispiel für einer Rolle?
14.	**Konsistenz:** Sind die Beschreibungen der Beziehungen zwischen Rollen und Aufgaben konsistent mit den Beschreibung im Nutzungsdiagramm?

Abbildung 7: Checkliste „Bereich Rollen, Aufgaben, NFR's, Rolleninstanzen"[18]

Zusätzlich zu den bereits beschriebenen Reduzierungen wird klar, dass „Startliste einsehen" ebenfalls eine Aktivität der Aufgabe „Registrieren" ist und als solche nicht getrennt betrachtet werden muss. Es verbleibt somit nach Bearbeitung der Aufgabe-nebene lediglich die Aufgabe „Registrieren".

18 Die Abbildung ist der Prozessbeschreibung entnommen (vgl. Häfele 2005: 119).

3.1.2 Domänenebene

Ist-Prozess

Der Ist-Prozess wurde bereits in Abschnitt 3 in seiner allgemeinen, systemunabhängigen Form dargestellt. Im Folgenden wird ergänzend beschrieben, wie der Ist-Prozess der Komponente „Teilnehmerregistrierung" für die Aufgabe „Registrieren" bei einer Delegierung an einen Zeitnahmedienstleister durchgeführt werden würde[19]. Bei genauerer Betrachtung dieses Vorgangs lassen sich dort vielfältige Schwachstellen isolieren.

Die Startseite des Dienstleisters ist ohne Bezug zur Veranstaltung und mit Fremdwerbung versehen, die Veranstaltungsdetails und -bedingungen sind schwer lesbar und in einer kleinen Scrollbox untergebracht. Die Veranstaltungsanmeldung ist zusammen mit anderen Veranstaltungen des Dienstleisters gelistet. Dies könnte auch als Vorteil gesehen werden, ist es für kleine Veranstaltungen ein Nachteil, da die wenigen Startplätze schnell von Teilnehmern ohne Verwurzelung im Landkreis besetzt werden und der Jedermanncharakter der Veranstaltung somit reduziert wird.

Aus der Perspektive des Datenschutzes ist problematisch, dass die Daten bei einem Zeitnahmedienstleiser gespeichert werden, und der Nutzung der Daten zu Zwecken des Dienstleisters nicht direkt widersprochen werden kann. Weiterhin sind die erfassten Daten zu umfangreich. Das Geburtsjahr sollte ausreichend sein, ein Geburtstag ist nicht erforderlich, da es auch andere Möglichkeiten gibt eine Person hinreichend eindeutig zu identifizieren, beispielsweise über eine Kombination von Name, Postleitzahl und Geburtsjahr, gegebenenfalls unter Hinzunahme einer Telefonnummer. Weiterhin ist zu kritisieren, dass das Geburtsjahr zusammen mit dem Klarnamen auf der Starterliste veröffentlicht wird.

Am Gravierendsten wirkt sich jedoch aus, dass bei einer Erstregistrierung 16 Bildschirme durchgearbeitet werden müssen. Selbst wenn bereits ein Benutzerkonto angelegt wurde, sind immer noch acht Bildschirme notwendig. Eine Fortschrittsanzeige existiert nicht, was den Benutzer über den genauen Fortgang im Unklaren lässt. Da bei der Delegation an einen Zeitnahmedienstleister keine Kontrolle über die Datenbank und das Datenmodell besteht, sind zukünftige Erweiterungen kaum möglich.

Die Funktionalität der zweiten Aufgabe „Anmeldedaten Bearbeiten/Löschen" ist auf dem System des Zeitnahmedienstleisters über eine Administrationsoberfläche gelöst und wird, wie zukünftig auch, von Expertennutzern durchgeführt. Dieser Prozess ist daher aus Benutzbarkeitsgesichtspunkten unkritisch und muss hier nicht genauer betrachtet werden, wichtig ist lediglich, dass der der Startlistenverwalter alle Werte aller Datenfelder bearbeiten kann. Außerdem muss ein Starter komplett anlegbar oder löschbar sein.

Soll-Prozess

Aufgabe „Registrieren"

Als Alternative zur manuellen Verwaltung kann diese Aufgabe an einen Zeitnahme-

19 Dabei wurde das Programm "DokuTool" testweise eingesetzt und der Programmablauf storyboardartig dargestellt, das Programm erwies sich jedoch als umständlich in der Handhabung und wareinem herkömmlichen Screenshot Werkzeug, wie etwa PicPick (http://www.picpick.org/) in Kombination mit einer Textverarbeitung unterlegen.

dienstleister delegiert werden, aufgrund der beschriebenen Einschränkungen kommt dies jedoch nicht in Betracht (vgl. Abbildung 11: Rationale Domänenebene). Daher wird ein möglichst idealer Prozess skizziert, der unabhängig vom Zeitnahmedienstleister durchführbar ist. In einem solchen Prozess kann der Starter auf der Startseite direkt seine Daten eingeben. Dabei kann er sich als Einzelstarter für eine Altersklasse oder als Staffelkontakt registrieren. Im Anschluss hat er die Möglichkeit seine Daten zu überprüfen. Nach der Bestätigung erhält er je nach Situation sofort eine Rückmeldung zu seinem Anmeldestand. Im Anschluss wird die Liste der bestätigten Teilnehmer angezeigt (Alternativ kann die aktuelle Startliste auch jederzeit direkt einsehen). Dies macht den Prozess einfacher, überschaubarer und nachvollziehbarer.

Auf Basis dieser Beschreibung lassen sich die Aktivitäten „Dateneingabe/Prüfung“ und „Startliste Einsehen“ unterscheiden.

Teilaufgabe *„Anmeldedaten Bearbeiten/Löschen“*

Der Startlistenverwalter hat über eine Administrationsoberfläche die Möglichkeit jeden Bestandteil der Datensätze zu ändern oder den Starter zu löschen, weiterhin ist er in der Lage, neue Datensätze anzulegen um Offlineanmeldungen ins System zu übertragen. Hier werden keine weiteren Unteraufgaben gebildet.

Der Soll-Prozess dient als Grundlage für alle weiteren Schritte auf dieser Ebene.

Systemverantwortlichkeiten

Auf Basis dieser Aktivitäten ergeben sich die folgenden Systemverantwortlichkeiten[20], die auch als Use Cases oder Systemfunktionen bezeichnet werden.

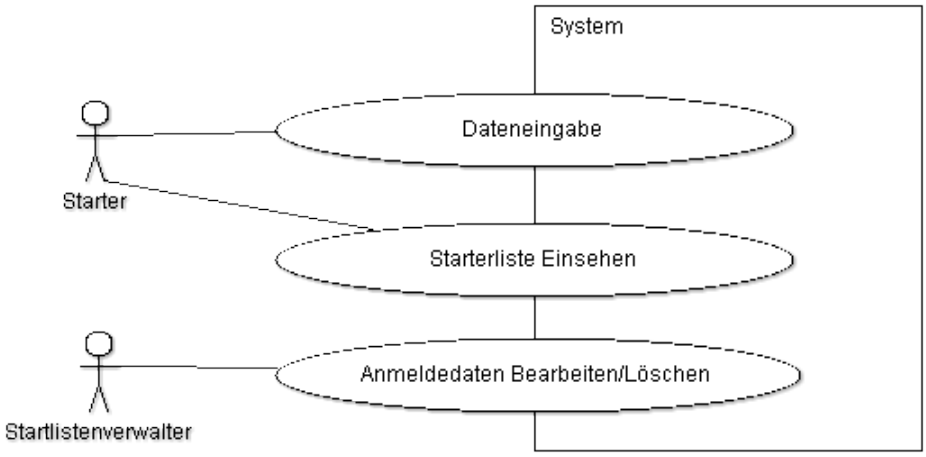


Abbildung 8: Systemverantwortlichkeiten

Domänendaten

Die Entität Teilnehmer[21] hat neben den offensichtlichen Merkmalen einer Person, wie Name, Geburtsjahr, Geschlecht und Kontaktdaten, auch Beziehungen zu einigen erläuterungsbedürftigen Merkmalen: Handelt es sich bei der Anmeldung um eine Staffelanmeldung, so hat er ein bis zwei weitere Staffelmitglieder. Außerdem hat er dann die Möglichkeit, zu einer eventuellen Erstanmeldung abweichende Angaben in den

20 Systemverantwortlichkeiten sind die Bestandteile für die zukünftig das System verantwortlich ist, die genaue Interaktion mit den Nutzern wird aber erst auf der Interaktionsebene festgelegt.

21 Um eine Verwechselung mit der Rolle Starter zu vermeiden, wird die Entität, die einen Starter im System repräsentiert, zukünftig „Teilnehmer“ genannt.

Feldern Verein/Teamname zu machen, erhält eine weiteren Registrierungszeitstempel, Starterstatus (Unbezahlt, Bezahlt oder Warteliste) und gegebenenfalls eine Startnummer mit Startgruppe.[22]

Für jede Anmeldung kann der Starter einen Kommentar hinterlassen, beispielsweise um Wünsche zur Startgruppe zu äußern oder Anmerkungen zur Zahlung oder ähnliches zu machen.

E-Mail kann auch leer sein, sofern der Teilnehmer durch den Startlistenverwalter manuell registriert wird.

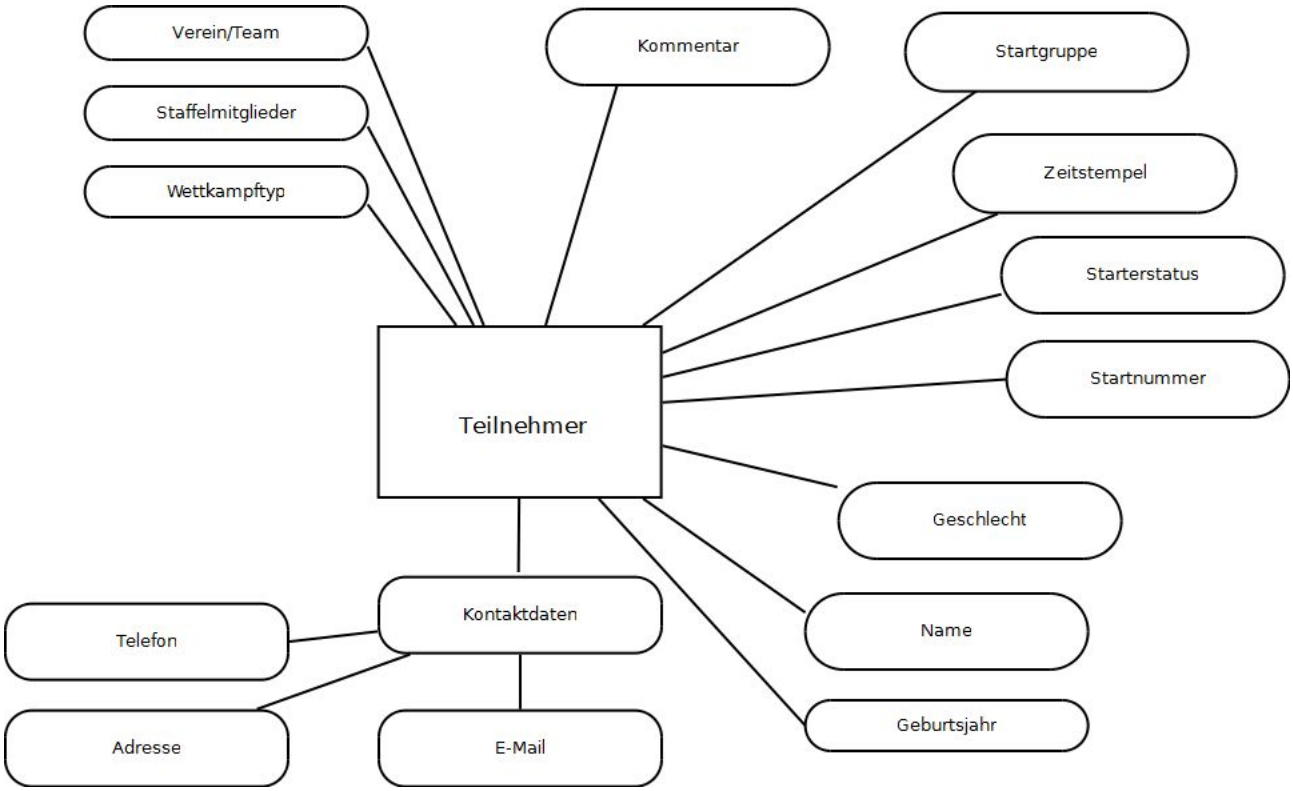


Abbildung 9: Domänendatendiagramm

Nicht-funktionale Anforderungen

Auf Basis der Analyse des Ist-Prozesses, wie er sich bei dem Zeitnahmedienstleister darstellt, lassen sich für die Aufgabe „Registrieren“ zwei zusätzliche Nicht-funktionale Anforderungen isolieren. Einerseits sollen alle Informationen gut lesbar sein („Lesbarkeit“) andererseits sollen möglichst wenige Bildschirme verwendet werden („Wenige Bildschirme“).

Systemtestplan

Im Systemtestplan[23] werden Testendekriterien, Testgrundlage und Testressourcen bestimmt.

Testendekriterien für den Systemtest

22 Die Vergabe des Status „Bezahlt“, der Startnummer und der Startgruppe erfolgt vor dem Wettkampf manuell durch den Starttlistenverwalter.

23 Der Systemtestplan orientiert sich an der Prozessbeschreibung von Häfele (vgl. Häfele 2005: 43f).

- Das System gilt als getestet, wenn alle Use Cases und alle Systemfunktionen getestet sind.
- Ein Use Case gilt als getestet, wenn alle Ausnahmen, mindestens ein gültiger Durchlauf und alle verwendeten Systemfunktionen getestet worden sind.
- Eine Systemfunktion gilt als getestet, wenn alle Exceptions und mindestens ein gültiger Ablauf getestet worden sind.

Grundlage für Systemtest

- Die Systemtests sollen in unserem Fall auf den Szenarien basieren, die auf der Interaktionsebene für jeden Use Case erstellt werden. Falls in den Szenarien nicht alle oben genannten Testendekriterien erfüllt sind, muss die Testspezifikation zusätzlich ergänzt werden, so dass diese erfüllt sind.

Systemtestressourcen

- Da ein Internetbasiertes System getestet werden soll, wird zum Test das Programm „Selenium“[24] eingesetzt.

Inspektion und Rationale

Wie bereits auf der Aufgabenebene festgestellt, muss eine Entscheidung zur Durchführung der Kommunikation bei Problemen oder Änderungswünschen getroffen werden. Die Kontaktfunktionalität kann dabei einfach über E-Mail bereitgestellt werden. Sie müssen daher nicht weiter betrachtet werden (vgl. Tabelle der Rationale, Abbildung 10). Dies ist insbesondere sinnvoll, da ein umfassender Prozess zunächst den Teilnehmern selbst die Möglichkeit geben sollte ihre Daten zu korrigieren oder die Anmeldung zurückzuziehen, bevor eine solche Kontaktfunktionalität in die Applikation integriert wird. Die Abbildung des Kontaktprozesses in der Applikation selbst (Möglichkeit 3) ergibt daher wenig Mehrwert. Bei einem E-Mail Formular entsteht der Vorteil, dass der Mailbetreff oder Mailempfänger je nach Sachbereich der Anfrage vorgegeben werden könnte. Der Mehraufwand hierfür scheint zum gegenwärtigen Zeitpunkt jedoch nicht rentabel, zumal diese Entscheidung jederzeit ohne Beeinflussung des restlichen Systems revidiert werden kann.

Optionen	Kriterien			
	Flexibilität	Prozesseffizienz	Geringe Kosten (x2)	Summe
Kontakt am Telefon/mündlich	-	--	++	1
Email (externes Programm)	**-**	**+/-**	**++**	**3**
Email Kontaktformular	+/-	+	+/-	1
Formular mit Applikations Anbindung	+/-	++	--	-1

Abbildung 10: Rationaletabelle zu „Anfrage stellen“ und „Anfrage beantworten“

Die wichtigste Rationale auf dieser Ebene ist jedoch, auf Basis der Betrachtung der Ist-Prozesse und der Qualitätskriterien von Aufgaben, die Entscheidung ein eigenes System zu entwickeln und keine Delegation an einen Zeitnahmedienstleister durchzuführen. Der Vorteil des Zeitnahmedienstleisters ist eine ausgebaute Datenverifikati-

24 Selenium kann unter http://seleniumhq.org/ bezogen werden.

on. Aus den Rationalen ergibt sich auch, dass sich die Systemumsetzung möglichst an der manuellen Anmeldung orientieren sollte, um von deren Stärken in der Handhabung profitieren zu können. Für die folgende Interaktionsebene bedeutet dies einen Fokus auf die Reduktion der Arbeitsbereiche und Sichten zu legen.

Optionen	Kriterien								Summe
	Flexibilität	Organisatorischer Zeitaufwand (x2)	Datensparsamkeit	Erlernbarkeit	Verständlichkeit	Subjektive Zufriedenheit	Sicherheit	Validierung/Anpassung der Daten	
Manuelle Anmeldung	-	- -	++	++	++	+	++	-	3
Delegation an Zeitnahmedienstleister	- -	+	+/-	+/-	+/-	+/-	+	++	3
Eigenes System	**+**	**+**	**+**	**+**	**+**	**+**	**+**	**+**	**9**

Abbildung 11: Rationale Domänenebene

Weiterhin wird die Entscheidnung getroffen auf eine umfassende Validierung der Eingabedaten (vgl. Abbildung 12) mit Hilfe einer Namensdatenbank oder ähnlichen Hilfsmitteln zu verzichten.

Optionen	Kriterien			
	Flexibilität	Prozesseffizienz	Geringe Kosten	Summe
Keine Überprüfung	-	- -	+	-2
Überprüfung durch Nutzer	+/-	+/-	+	1
Überprüfung durch Nutzer +Formatierung	**+**	**+**	**+/-**	**2**
Umfangreichere Validierung	+	++	--	1

Abbildung 12: Rationale für NFR: Validierung/Anpassung der Daten

Die vorgestellte Kombination aus Nutzervalidierung und Formatierung (Groß-/Kleinschreibung) wurde gewählt, da es im Hinblick auf eine Weiterverwendung der Daten flexibler ist, die Daten vorformatiert in die Datenbank zu schreiben, als sie beim auslesen umzuformatieren.

Bei der Inspektion wird geprüft, ob noch immer alle Bedingungen der Checkliste aus Abbildung 7 gegeben sind. Insbesondere, dass Aufgaben und Domänenebene Konsistent bearbeitet wurden.

3.1.3 Interaktionsebene

UI-Struktur

Es werden die in Abbildung 13 dargestellten Sichten entworfen und in die Arbeitsbereiche „Dateneingabe“ und „Starterliste Einsehen“ gruppiert. Dabei wurde auf wenige Schritte Wert gelegt, um die subjektive Zufriedenheit aber auch die Erlernbarkeit, und Verständlichkeit zu erhöhen.

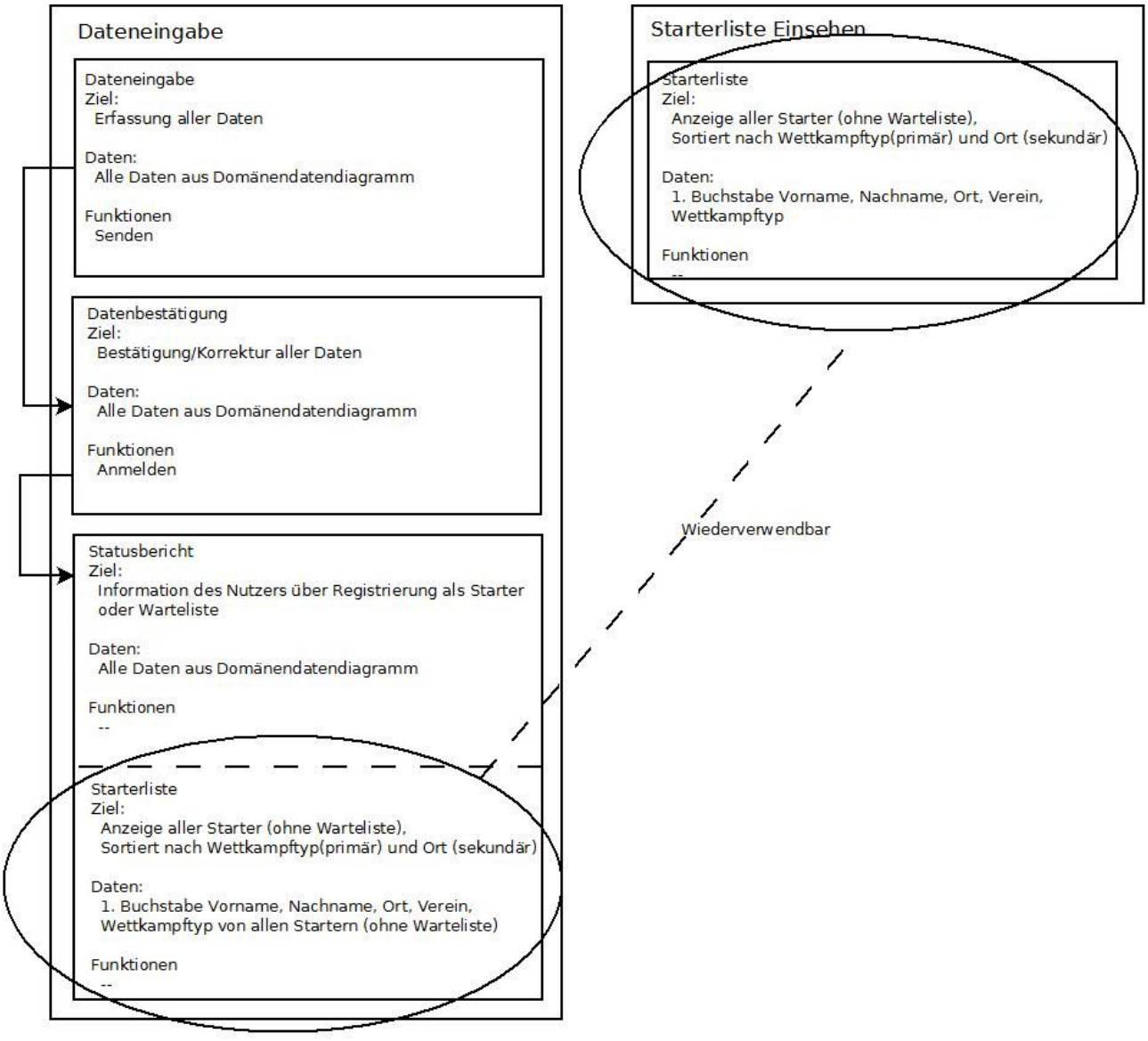


Abbildung 13: UI-Struktur

Aus Datenschutzüberlegungen erfolgt die Ausgabe des Geburtsjahrs in der Starterliste nicht und der Vorname wird auf den ersten Buchstaben reduziert. Wenn der Starter sich als Teilnehmer registriert hat, ist es naheliegend, dass er sofort die Starterliste einsehen will um seinen Eintrag oder seine Konkurrenten zu betrachten. Daher wird die Liste direkt an die Bestätigung angehängt, er kann somit die Starterliste einsehen, ohne aktiv dort hin navigieren zu müssen. So wird auch bei Dateneingabe und Datenbestätigung verfahren (oben werden die zu bestätigenden Daten angezeigt und unten erneut das Formular „Dateneingabe“, das die bereits ausgefüllten Felder

enthält), dies ist aber im Diagramm nur schwer darzustellen.

Interaktionsbeschreibung

Im Folgenden wird lediglich der Use Case[25] „Dateneingabe“, der aus der Aufgabe „Registrieren“, die in die Systemverantwortlichkeiten „Dateneingabe“ und „Starterliste Einsehen“ untergliedert wurde, hervorging, genauer ausgearbeitet. Der Grund dafür ist, dass „Starterliste Einsehen“ ein Bestandteil der Dateneingabe ist (vgl. Abbildung 14). Die Administrationsoberfläche bleibt, wie bereits bei dem Soll-Prozess geschehen, zunächst außer Acht. Ein entsprechender Use Case wird nachträglich erstellt, falls sich dies als notwendig erweisen sollte.

Interaktionsbeschreibung (Use Case)		
Name	Dateneingabe	
Verantwortliche Rolle (Initiating Actor)	Starter	
Beteiligte Rollen (Participating Actors)	Startlistenverwalter bei Problemen oder Nachfragen, ansonsten keine	
Ziel (Goal)	Teilnehmer ist erfolgreich im System registriert	
Vorbedingung (Preconditions)	Startseite der Anmeldung ist geöffnet Die Aktion „Anmeldung abschicken“ steht zur Verfügung	
Beschreibung (Flow Of Events)	Aktor	System
		0. Das System bietet nur die Wettkampftypen zur Anmeldung an, bei denen es freie Startplätze oder Wartelistenplätze gibt
	1. Füllt das Anmeldeformular aus	
		2. System zeigt die fehlenden Pflichtfelder sowie die eingegebenen Daten zur Überprüfung an, gefolgt vom Anmeldeformular System ermöglicht 2a)Änderungen der Daten im Anmeldeformular (wenn dem Nutzer ein Fehler aufgefallen ist oder Pflichtfelder fehlen) 2b)Anmelden
	3. - Falls Aktor 2a wählt weiter mit 2b - Falls Aktor 2b wählt weiter mit 4	
		4. System schreibt Teilnehmerdaten in Datenbank (Systemfunktion „Daten schreiben“)
		5. System zeigt an, ob der Eintrag erfolgreich war oder ein Wartelisteneintrag vorgenommen wurde, gefolgt von den gespeicherten Da-

25 Usecases werden in der Literatur auch häufig an anderen Stellen verwendet (vgl. Sommerville 2004: 154f, Häfele 2005: 47).

		ten und gefolgt von der Starterliste
Ausnahmefälle (Exceptions)	- Pflichtfelder nicht gefüllt (1.) - Änderung der Daten im Anmeldeformular (2a) - Daten bereits vorhanden (4.) - Datenbankfehler (4.) - Eintrag erfolgte nur in Warteliste (4.) (dann ist Teilnehmer nicht in Starterliste, aber trotzdem in der Datenbank)	
Nachbedingungen (Postconditions)	Teilnehmereintrag wird in Starterliste angezeigt	
Regeln (Rules)	Keine	
Qualitätsanforderung (Quality Constrains)	- Geringe Anzahl von Schichten - Geringe Navigation zwischen Schichten - Daten sind zusammenhängend gruppiert Geerbte: Qualitätskriterien der Aufgabe „Registrieren“ - Flexibilität - Organisatorischer Zeitaufwand/Datensparsamkeit - Erlernbarkeit - Verständlichkeit - Subjektive Zufriedenheit - Sicherheit - Validierung/Anpassung der Daten Domänenkriterien - Lesbarkeit	
Benutzte Funktionen (Used Services)	Daten in Datenbank eintragen	

Abbildung 14: Use Case „Dateneingabe“

Systemfunktionen

Aus dem Use Case ergibt sich im Wesentlichen nur eine größere Systemfunktion (Schritt 4 und 5 im Flow of Events).

Systemfunktionsbeschreibung	
Name	Daten Schreiben
Beschreibung (Description)	1a. System liest die Daten ein 1b. System entscheidet ob ein Wartelisteneintrag oder ein normaler Eintrag vorgenommen wird 2. System bereinigt Zeichenketten 3. System prüft ob die Daten schon in der Datenbank sind 4. System schreibt Daten in die Datenbank
Eingangsdaten	Alle Daten, die der Nutzer eingibt
Ausgangsdaten	Anmeldungsstatus: Warteliste oder Regulär
Ausnahmefälle	- Daten bereits vorhanden - Datenbankfehler
Regeln (Rules)	Keine
Vorbedingungen (Preconditions)	Alle Pflichtfelder der Dateneingabe sind gefüllt
Nachbedingungen (Postconditions)	Daten sind in der Datenbank oder Fehler wurde angezeigt
Qualitätsanforderungen (Quality Contraints)	Geerbte: Schritt 2 muss Angriffe auf die Datenbank erschweren (Aufgabenebene, Kriterium „7. Sicherheit“) Schritt 2 sollte zumindest den ersten Eintrag der Datenfelder

	zu einem Großbuchstaben umwandeln, wenn dies sinnvoll ist (Aufgabenebene, Kriterium „8. Validierung /Anpassung der Daten“)
Use Cases	Dateneingabe

Abbildung 15: Systemfunktion „Daten Schreiben“

Verfeinertes Datenmodell

Im verfeinerten Datenmodell werden nach dem Prozess den Häfele beschreibt (Häfele 2005: 48 und 59) die Sichten dem Domänendatenmodell zugeordnet. Eine entsprechende grafische Aufbereitung des Datenmodells macht in diesem Beispiel jedoch keinen Sinn, da praktisch alle Sichten alle Daten enthalten. Daher erfolgt hier nur eine grobe tabellarische Zuordnung.

- Dateneingabe: Alle außer Startgruppe, Starterstatus, Zeitstempel, Startnummer
- Datenbestätigung: Alle außer Startgruppe, Starterstatus, Zeitstempel, Startnummer
- Statusbericht: Alle außer Startgruppe, Zeitstempel, Startnummer
- Starterliste: Nur Name, Adresse (teilweise), Verein/Team, Wettkampftyp

Szenarien

Als Szenario wird für den Use Case eine Staffelanmeldung durchgespielt, da die Staffel der komplexeste Fall ist. Dabei wird auf eine fiktive Rolleninstanz der Rolle Starter mit dem Namen „Müller“ zurückgegriffen.

Szenario (Scenario)	
Name	Müllers Staffelanmeldung
Verantwortlicher Use Case (initiated Use Case)	Dateneingabe
Verantwortliche Rolleninstanz (Initiating Actor Instance)	Müller
Beschreibung (Flow Of Events)	Herr Müller möchte sich und seine Staffel der Freiwilligen Feuerwehr für den Staffeltriathlon anmelden. Hierfür öffnet er mit seinem Webbrowser (Firefox 3) die Informationsseite zum Triathlon. Direkt auf dieser Seite kann er seine Daten in ein Formular eingeben. Er gibt alle Pflichtfelder und zusätzlich „Freiwillige Feuerwehr“ im Feld Verein/Team sowie die Staffelstarter an und klickt auf den Button abschicken. In der folgenden Sicht „Datenbestätigung“ bemerkt er, dass er an Stelle seiner Festnetznummer seine Handynummer angeben möchte und nimmt diese Änderung vor. Danach klickt er erneut auf abschicken. Erfreut stellt er fest, dass er erfolgreich angemeldet wurde und sieht direkt in der Starterliste nach, welche Teams noch am Start sind, dabei bemerkt er einen Freund, der als Einzelstarter registriert ist. Daher ruft er im Anschluss die Startseite erneut auf und registriert sich als Einzelstarter, leider muss er hierfür nochmals seine Daten eingeben. Danach will er noch seine Frau anmelden, diese erhält jedoch nur noch einen Wartelistenplatz. Anschließend versucht er noch seinen erwachsenen Sohn anzumelden. Das System bietet ihm jedoch keine Anmeldemöglichkeit mehr an, da seine Frau den letzten Wartelistenplatz bekommen hat, und weist ihn mit einer Meldung darauf hin, dass es keine Startplätze mehr gibt.

Abbildung 16: Szenario „Müllers Staffelanmeldung“

Nicht-funktionale Anforderungen

Auf der Interaktionsebene lassen sich die neuen Kriterien „Geringe Anzahl von Schichten“ und „Geringe Navigation zwischen Schichten“ und „Daten sind zusammenhängend gruppiert“ hinzufügen. Außerdem werden in der Systemfunktion „Daten schreiben“ Anforderungen aus der Aufgabenebene erneut aufgegriffen und konkretisiert („Angriffe auf die Datenbank erschweren“ aus Kriterium „7. Sicherheit“ auf Aufgabenebene, sowie Umwandlung zu Großbuchstaben, wenn dies sinnvoll ist“ aus Kriterium „8. Validierung /Anpassung der Daten“ auf Aufgabenebene).

Systemtestspezifikation

Gemäß des Systemtestplans (vgl. Abschnitt 3.1.2) gilt das System durch das folgende Vorgehen als getestet, da das vorgestellte Szenario um alle möglichen Ausnahmen angereichert wurde.

Systemtestspezifikation			
Name	Systemtest Szenario „Müllers Staffelanmeldung“		
Typ (Type)	Starter		
Ziel (Goal)	Systemtest für „Registrieren“ Funktionalität (Szenario „Müllers Staffelanmeldung“)		
Vorbedingung (Preconditions)	Startseite der Anmeldung (Sicht „Dateneingabe“) ist geöffnet Die Aktion „Anmeldung abschicken“ steht zur Verfügung		
Beschreibung (Flow Of Events)	Input	Expected Output	Exceptions
	Pflichtfelder und Staffelstarter ausgefüllt, Button „Anmeldung abschicken“ geklickt	Sicht „Datenbestätigung“ wird angezeigt	- Pflichtfelder nicht ausgefüllt => Hinweis in Sicht Datenbestätigung und Korrekturmöglichkeit im nächsten Schritt
	Telefonnummer wird geändert, Button „Anmeldung abschicken“ geklickt	Sicht „Statusbericht“ wird mit der Meldung „Registrierung mit folgenden Daten erfolgreich:“ und den geänderten Daten angezeigt. Weiter unten findet sich der Name in der angezeigten Starterliste	- Daten bereits vorhanden Sicht „Statusbericht“ wird mit der Meldung „Sie haben sich bereits registriert“ angezeigt. Weiter unten findet sich der Name in der angezeigten Starterliste - Datenbankfehler Die Meldung „Es liegt ein technisches Problem vor, wir bitten dies zu entschuldigen! Bitte versuchen Sie es in ein paar Stunden noch einmal, oder kontaktieren Sie uns!“ wird angezeigt
	Navigation zur Startseite über Link „Zurück zur Anmeldung“ oder erneutes Eintippen der URL	Sicht „Dateneingabe wird angezeigt“	Navigation funktioniert nicht => Test fehlgeschlagen

	Pflichtfelder erneut ausgefüllt, Button „Anmeldung abschicken“ geklickt	Sicht „Datenbestätigung“ wird angezeigt	- Pflichtfelder nicht ausgefüllt => Hinweis in Sicht Datenbestätigung und Korrekturmöglichkeit im nächsten Schritt
	Button „Anmeldung“ abschicken“ geklickt	Sicht „Statusbericht“ wird mit der Meldung „Registrierung mit folgenden Daten erfolgreich:“ und den Daten angezeigt. Weiter unten findet sich der Name in der angezeigten Starterliste einmal mit Wettkampftyp „Staffel“ und einmal mit Wettkampftyp „Standard“	- Daten bereits vorhanden Sicht „Statusbericht“ wird mit der Meldung „Sie haben sich bereits registriert“ angezeigt. Weiter unten findet sich der Name in der angezeigten Starterliste einmal mit Wettkampftyp „Staffel“ und einmal mit Wettkampftyp „Standard“ - Datenbankfehler Die Meldung „Es liegt ein technisches Problem vor, wir bitten dies zu entschuldigen! Bitte versuchen Sie es in ein paar Stunden noch einmal, oder kontaktieren Sie uns!“ wird angezeigt - Eintrag erfolgte nur in Warteliste „Leider sind alle regulären Startplätze bereits belegt. Sie wurden mit den unten aufgeführten Daten auf die Warteliste eingetragen! Erfahrungsgemäß bestehen gute Chancen noch nachzurücken. Ihr Name ist noch nicht in der Starterliste zu sehen, wir benachrichtigen Sie, sobald Sie nachrücken können!“ Darunter werden die geänderten Daten angezeigt. Darunter wird die Starterliste ohne den Namen des Starters angezeigt (sofern zuvor auch Wartelisteneintrag erfolgte, ansonsten ist er mit dem Wettkampftyp „Staffel“ ein Mal vorhanden. [Um diesen Fall zu testen, muss noch geprüft werden, ob der Eintrag tatsächlich in der Datenbank ist!]
	Navigation zur Startseite über Link „Zurück zur Anmeldung“ oder Erneutes Eintippen der URL	Sicht „Dateneingabe wird angezeigt“	Navigation funktioniert nicht => Test fehlgeschlagen
	Button „Anmeldung abschicken“ geklickt	Eintrag erfolgte nur in Warteliste, Meldung: „Leider sind alle regulären Startplätze bereits	- Daten bereits vorhanden Sicht „Statusbericht“ wird mit der Meldung „Sie haben sich bereits registriert“ angezeigt. Weiter un-

		belegt. Sie wurden mit den unten aufgeführten Daten auf die Warteliste eingetragen! Erfahrungsgemäß bestehen gute Chancen noch nachzurücken. Ihr Name ist noch nicht in der Starterliste zu sehen, wir benachrichtigen Sie, sobald Sie nachrücken können!“ Darunter werden die geänderten Daten gefolgt von der Starterliste ohne den Namen des Starters angezeigt. [Um diesen Fall zu testen muss noch geprüft werden, ob der Eintrag tatsächlich in der Datenbank ist!]	ten findet sich der Name in der angezeigten Starterliste einmal mit Wettkampfty „Staffel“ und einmal mit Wettkampftyp „Standard“ - Datenbankfehler Die Meldung „Es liegt ein technisches Problem vor, wir bitten dies zu entschuldigen! Bitte versuchen Sie es in ein paar Stunden noch einmal, oder kontaktieren Sie uns!“ wird angezeigt
	Navigation zur Startseite über Link „Zurück zur Anmeldung“ oder erneutes Eintippen der URL	Sicht „Dateneingabe wird angezeigt“	Navigation funktioniert nicht => Test fehlgeschlagen
	Auswahl des Wettkampftyps „Standard“ ist nicht mehr möglich, eine Informationsmeldung weist darauf hin, dass der Standardtriathlon voll ist	Keine weiteren Aktionen des Nutzers	
Nachbedingungen (Postconditions)	In diesem Fall: Startseite wird angezeigt		
Testressourcen (Test Resources)	Selenium		

Abbildung 17: Systemteststpezifikation (Szenario „Müllers Staffelanmeldung“)

Usabilitytestspezifikation und -plan

Zu Usabilitytestspezifikation und -plan hält die Prozessbeschreibung von Häfele (Häfele 2005) keine Anweisungen und kein Template bereit, es wird lediglich seine Existenz erwähnt, daher erfolgt hier lediglich eine informelle Beschreibung.

Zu testen sind in unserem Fall die bereits auf der Aufgabenebene eingeführten Kriterien „Erlernbarkeit“, „Verständlichkeit“ und „Subjektive Zufriedenheit“. Da auf dieser Ebene noch keine Funktionalität und keine Benutzeroberfläche zur Verfügung steht, müssen die entsprechenden Tests mit Mock-ups durchgeführt werden, die entweder

papierbasiert sind, oder mit einem Zeichenprogramm erstellt werden. Für HTML-basierte Oberflächen bietet es sich auch an, diese einfach ohne Funktionalität und in simpelster Form vorab zu erstellen und den Benutzern vorzulegen. Da die Schnittstelle der hier besprochenen Anwendung bewusst einfach gehalten ist, wurde beschlossen, zuerst die Oberfläche komplett zu entwickeln, und im Anschluss die Elemente gegebenenfalls zu tauschen oder zu separieren, falls sich dies als notwendig erweisen sollte.

Zunächst müssen die Kriterien jedoch noch messbar gemacht werden. Lauesen diskutiert hierfür verschiedene Varianten. Auf Basis der von ihm bereitgestellten Entscheidungshilfe (vgl. Lauesen 2005: 39), erscheint es sinnvoll, für die ersten beiden Anforderungen das Zählen von Problemen (Problem Counts) einzusetzen, während die subjektive Zufriedenheit am Besten durch eine nachgelagerte Umfrage erfasst werden kann.

Diese Überlegungen führen zu folgenden Usability Zielen:

- Erlernbarkeit/Verständlichkeit: Maximal einer von zehn Nutzern stößt beim Abarbeiten der Aufgabe aus dem Szenario (das um die angezeigten Meldungen bereinigt wird, damit die Nutzer die Meldungen nicht vorab kennen) auf einen kritischen Fehler[26].
- Subjektive Zufriedenheit: 90% der befragten Nutzer sind zufrieden mit der Anmeldeprozedur (auf der Skala zufrieden - weniger zufrieden - nicht zufrieden).

Ergänzend könnte noch eine Punktzahl für Verständlichkeit[27] verwendet werden, worauf hier jedoch aus Zeitgründen verzichtet wird.

26 Hier wird das Verfahren von Lauesen angewandt: Fehlende Funktionalität, Abbruch der Aufgabe und sichtbare Genervtheit des Nutzers sind kritische Fehler (vgl. Lausesen 2005:13).

27 Dabei werden die Fehlermeldungen den Nutzern mit der Frage vorgelegt, was sie denken, was die Meldung bedeutet. Außerdem werden sie gefragt, was das System tun würde, wenn sie eine bestimmte Aktion vornehmen. Die Antworten werden vom Tester auf einer fünfstelligen Skala bewertet. Ziel wären dann 80-90% der Nutzer im Bereich der höchsten beiden Punktzahlen zu verorten (vgl. Lauesen 2005: 34f).

Inspektion und Rationale

Auf der Interaktionsebene wird im Wesentlichen die Entscheidung getroffen, die Sichten zusammenzufassen (vgl. Abbildung 18)

Optionen	Kriterien												Summe
	Geringe Anzahl von Schichten	Geringe Navigation zwischen Schichten	Daten sind zusammenhängend gruppiert	Flexibilität	Organisatorischer Zeitaufwand (x2)	Datensparsamkeit	Erlernbarkeit	Verständlichkeit	Subjektive Zufriedenheit	Sicherheit	Validierung/Anpassung der Daten	Lesbarkeit	
Starterliste separat „Dateneingabe“ auf Startseite	+	+						+/-	+/-				2
Starterliste separat, „Dateneingabe“ auf eigener Seite	-	-						-	-				-4
Starterliste bei „Statusbericht“ „Dateneingabe“ auf Startseite	**++**	**++**						**+**	**+**				**6**
Starterliste bei „Statusbericht“, „Dateneingabe“ auf eigener Seite	+	+						+/-	+/-				2

Abbildung 18: Rationale „Sichten“

Die Inspektion der Interaktionsebene erfolgt mittels der Checkliste „Bereich Use Cases“ (Abbildung 19).

Checkliste - Bereich Use Cases

Stellen Sie sich vor, Sie sind ein zukünftiger Benutzer des neu zu entwickelnden Systems und möchten überprüfen, ob die Arbeitprozesse und -abläufe für sie richtig umgesetzt worden sind. Überprüfen Sie anhand der Checkliste, ob die Aufgaben in den Use Cases korrekt umgesetzt worden sind.

Nr.	Frage
1.	**Eindeutigkeit:** Sind die beschreibenden Texte einfach, eindeutig und verständlich formuliert?
2.	**Vollständigkeit:** Sind alle Aufgaben durch Use Cases umgesetzt?
3.	**Vollständigkeit:** Sind die Aufgaben vollständig umgesetzt, d.h. dass alle Aktionen aus den Aufgaben sich im Use Case widerspiegeln?

Abbildung 19: Checkliste „Bereich Use Cases“ (Quelle Häfele 2005:119f)

3.1.4 Systemebene

Gui-Entwurf

Dem Entwurf der Benutzungsschnittstelle kommt im TRAIN Prozess nur eine untergeordnete Rolle zu. Nach der Prozessübersicht (vgl. Häfele 2005: 114) sind auf dieser Ebene die Artefakte „Dialog Text“, „Dialog Zustandsdiagramm“ und „Hilfsfunktionen“ zu erzeugen. Durch den erhöhten Detailgrad auf den bereits bearbeiteten Ebenen, insbesondere im Use Case, werden diese Aspekte jedoch hinreichend berücksichtigt. Daher wird dieser Prozesschritt hier nicht weiter ausgeführt, sondern den Ergebnissen der Usability Tests überlassen[28].

Bei der Betrachtung der benötigten Fenster ergibt sich jedoch aufgrund der Ausgabe des Ortes in der Starterliste die Notwendigkeit, das Attribut „Adresse“ im Domänendatendiagramm weiter aufzuspalten und PLZ und Ort getrennt zu erfassen.

Systemkern

Da der TRAIN-Prozess auf ein objektorientiertes Vorgehen ausgerichtet ist, kommt es auf dieser Ebene zu Umsetzungsschwierigkeiten, da in der gewählten Beispielanwendung Objektorientierung aufgrund der beschränkten Komplexität der Anwendung und des Datenmodells nicht notwendig und angebracht erscheint. Daher können auf dieser Ebene keine weiteren Präzisierungen vorgenommen werden. Entsprechend ergeben sich auch keine neuen Nicht-funktionalen Anforderungen und Rationale. Die Inspektion besteht aus einer Gesamtbetrachtung der bisher erstellten Dokumente.

28 Eine ausführlichere Betrachtung erfolgt in Kapitel 4.

3.2 Architekturdefinition

3.2.1 Entwurfsziele

Die Entwurfsziele für die Architektur ergeben sich in unserem Fall aus den bisher formulierten Nicht-funktionalen Anforderungen, insbesondere soll die Anwendung flexibel anpassbar sein, dazu bietet es sich an, die einzelnen Komponenten in kleinere Einheiten aufzuspalten, sowie Einstellungen weitestgehend in eine eigene Datei auszulagern. Die Administration muss einfach möglich sein, damit der Organisatorische Zeitaufwand gering bleibt. Aus dem Blickwinkel der Sicherheit müssen Angriffe auf die Datenbank erschwert sowie Verschlüsselung eingesetzt werden.

3.2.2 Architekturdokumentation

a) Soll-Architektur

Als grundlegende Architektur wird eine klassische Drei-Schicht-Architektur gewählt, die aus einem Client (Webbrowser), einem Webserver mit integriertem Application Server und einer Datenbankschicht zur persistenten Datenspeicherung besteht, denn nur eine Präsentation über das Internet ermöglicht es einem großen Nutzerkreis sich auf einfache Weise anzumelden. Die Geschäftslogik wird also auf dem Application Server ausgeführt.

Auf clientseitiges Skripting soll bei dem Entwurf verzichtet werden um eine größtmögliche Plattform- und Browserunabhängigkeit zu erreichen. Aus diesem Grund kommen auch Lösungen wie Java und Flash, die eine clientseitige Installation erfordern, nicht in Betracht, insbesondere da mit einem Nutzerkreis zu rechnen ist, der auch mit alten Computern, veralteten Browsern oder per Modem mit dem Internet verbunden sein kann.

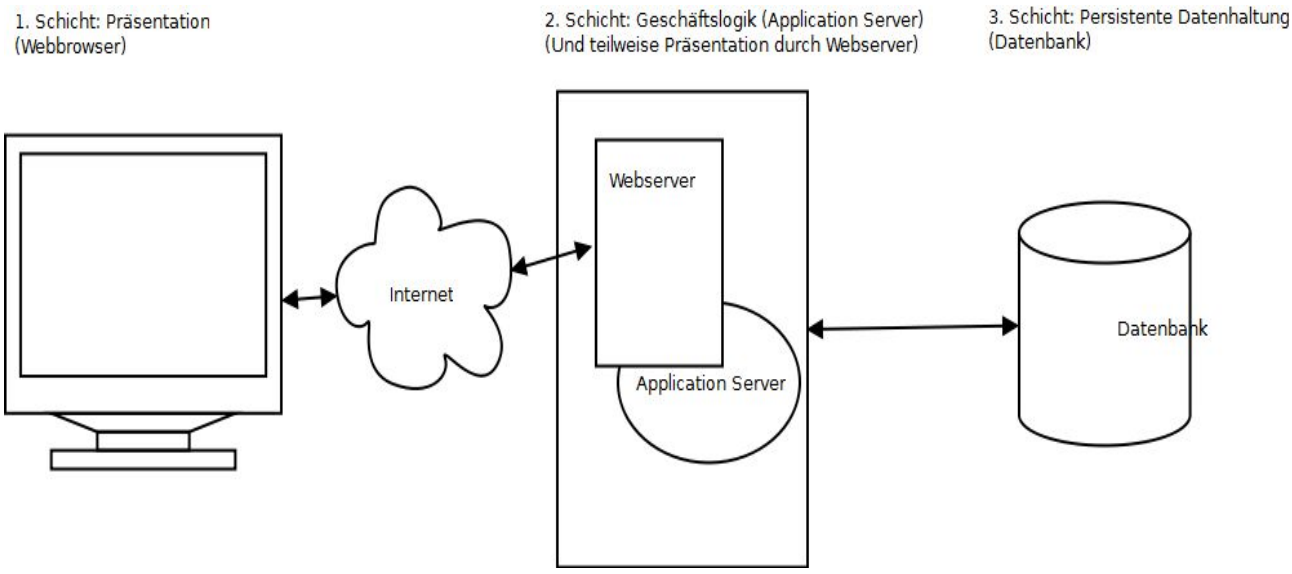
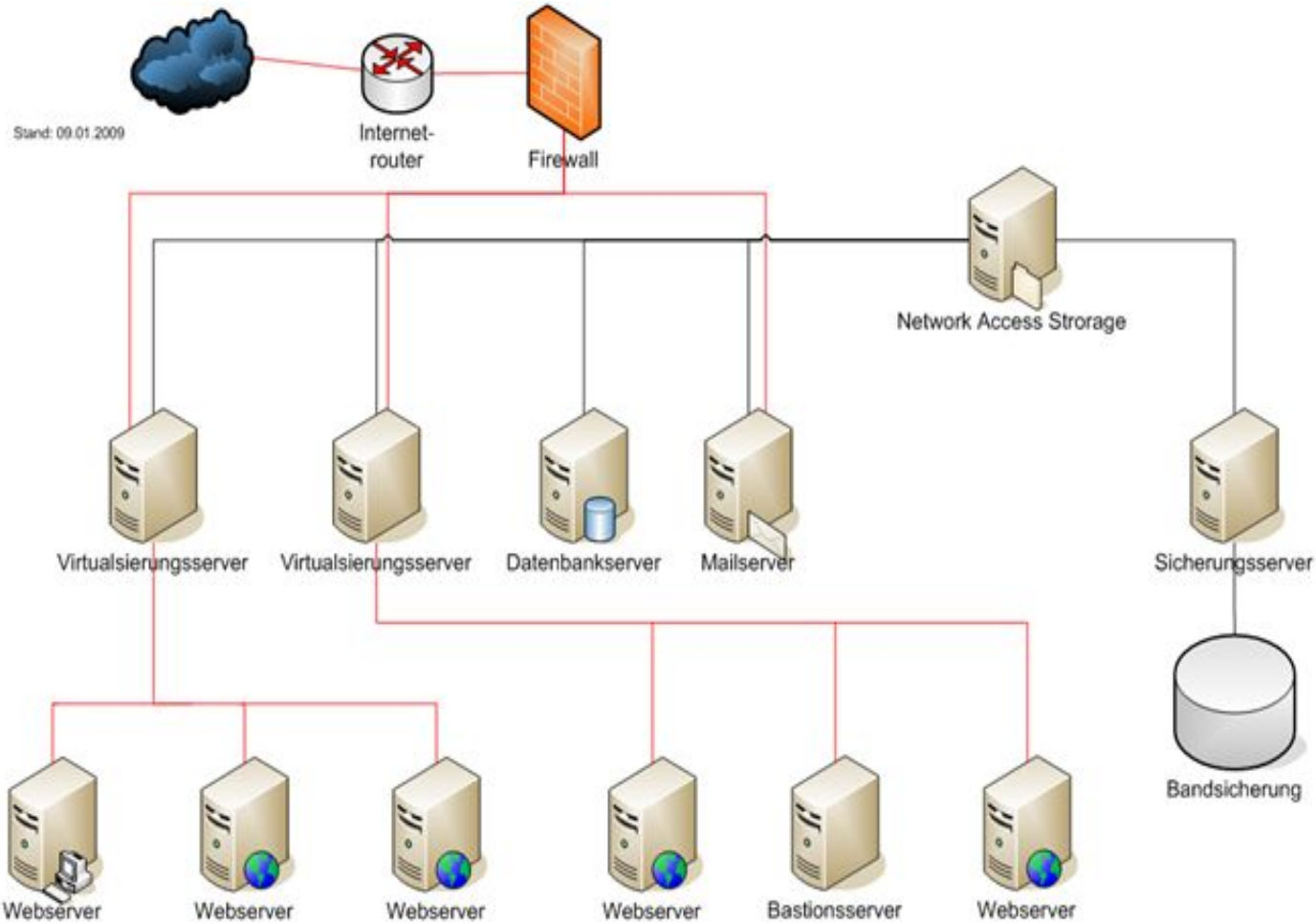


Abbildung 20: Architektur der Komponente „Teilnehmerregistrierung“

4.	**Korrektheit:** Sind alle beschriebenen Use Cases durch die Aufgabenbeschreibung gefordert?
5.	**Interne Konsistenz:** Leistet jeder Use Case einen wichtigen, neuen und einmaligen Beitrag zur Anforderungsspezifikation?
6.	**Korrektheit:** Sind die Use Cases korrekt und sinnvoll miteinander verlinkt?
7.	**Konsistenz:** Haben die Fachbegriffe in den Aufgaben und in den Use Cases die gleiche Bedeutung?
8.	**Eindeutigkeit:** Ist jeder verwendete Fachbegriff im Glossar erklärt?
9.	**Vollständigkeit:** Gibt es zu jeder Ausnahme eine entsprechende Beschreibung des Verhaltens, für den Fall, dass diese Ausnahme eintritt?
10.	**Prüfbarkeit:** Lassen sich aus den beschriebenen Use Cases sinnvolle Systemtestfälle ableiten?
11.	**Korrektheit:** Besitzt jeder Use Case bzw. einer seiner Unter - Use Cases mindestens einen Rollen- und einen Systemschritt?
12.	**Korrektheit:** Verwendet jeder Use Case bzw. einer seiner Unter – Use Cases mindestens eine Systemfunktion?
13.	**Konsistenz:** Sind die Beschreibungen der Beziehungen zwischen Aufgaben und Use Cases mit der Beschreibung aus dem Nutzungsdiagramm konsistent?

b) Ressourcen und Hardware-Software Mapping



Das System wird zunächst bei der Triathlonveranstaltung der DLRG Ortsgruppe Schifferstadt eingesetzt, daher können Applikation und die Datenbank beim Internetdienstleister der DLRG Bundesebene (TAL.de) für die Ortsgruppe kostenlos gehostet werden. Dort steht die in Abbildung 21 dargestellte Serverfarm bereit. Neben dem Content-Management-System (CMS) Typo3 wird dort ein PHP Applicationserver und ein MySQL Datenbankserver bereitgehalten. Somit sollen diese Technologien eingesetzt werden. Die genauen Hardwarespezifikationen sind nicht bekannt, da es sich ohnehin um eine Virtualisierungslösung handelt. Die Serverlast ist bei der Anwendung unkritisch, da sich nur sehr selten mehre Teilnehmer zur selben Zeit anmelden wollen.[29]

Abbildung 21: Serverfarm der DLRG Internetdienstleistungen (Quelle: DLRG.net)

c) Betriebskonzept

Für Betriebssicherheit, Backups und Updates sind die DLRG Internetdienstleistungen verantwortlich. Die für die Installation und Wartung der Anwendung benötigten Dateien können über einen FTP-Client hochgeladen werden. Ein Datenbankzugang steht bereit, der über ausreichen Rechte verfügt, eigene Tabellen anzulegen sowie Inhalte zu speichern und zu ändern.

29 Hiermit ist genau die selbe Minute gemeint. Natürlich kann es zu Beginn der Anmeldung zu einem kleinen Peak kommen, solange der Termin nicht lange vorher exakt bekannt gegeben wird, sind jedoch keine kritischen Lasten zu erwarten. Die Teilnehmerzahl von <1000 Teilnehmern sollte sich gut über den mehrwöchigen Anmeldezeitraum verteilen.

Hervorgehobene Aspekte

a) Datenverwaltung (Persistenz)

Da eine MySQL Datenbank bereitgestellt wird, ist dieses Datenbanksystem einzusetzen. Aktuell ist die Version 5.0.5 installiert. Da alle zu speichernden Daten in direkter Relation zum „Teilnehmer“ stehen, erscheint es nicht sinnvoll eine Normalisierung des Datenmodells vorzunehmen, obwohl es denkbar wäre, die Gruppierung der Staffeln in einer eigenen Tabelle vorzunehmen. Dies hat jedoch den gravierenden Nachteil, dass dann die Daten für jedes Staffelmitglied gefüllt werden müssten, damit diese auch selbst als Teilnehmer registriert sind. Bei der bereits bei der Anforderungsanalyse eingeführten Vorgehensweise bleiben die Staffelmitglieder jedoch lediglich namentlich erfasst. Entscheidend sind nur die Daten des anmeldenden Starters in seiner Funktion als Staffelkontakt. Alternativ könnten auch die Kontaktdaten oder die „administrativen“ Bestandteile des Datensatzes (Startnummer, Startgruppe, Starterstatus und Zeitstempel) ausgelagert werden, ein zusätzlicher Nutzen hiervon ist allerdings nicht erkennbar. Weiterhin gestaltet sich der Datenexport in ein CSV-Datei[30] einfacher, wenn nur eine Tabelle genutzt wird. Es wird daher lediglich eine einzige Tabelle (Starter) angelegt, die alle Datenfelder enthält. Bei komplexeren Systemen wird in der Regel eine Normalisierung des Datenbankschemas vorgenommen um Update-Anomalien[31] vorzubeugen und den Speicherbedarf zu reduzieren.

b) Zugangskontrolle und Sicherheit

Eine Authentifizierung ist nicht notwendig, die Zugangsdaten für die Datenbank sind in einer von außen nicht direkt zugreifbaren PHP-Datei mit der Endung „.inc“ hinterlegt. Die Endung bewirkt in Zusammenspiel mit der „.htaccess“ Konfigurationsdatei des Apache Webservers, dass ein Aufruf von Dateien mit der Endung „.inc“ nur möglich ist, nachdem diese von PHP vorverarbeitet wurde. Bei diesem Prozess wird von PHP lediglich der Teil des Dokuments an den Benutzer weitergereicht, den dessen Browser zur Darstellung benötigt, nicht aber interne Variablen oder Zugangsdaten zur Datenbank. Als weitere Sicherungsmaßnahme werden die eingegebenen Daten bereinigt um SQL-Injection Angriffe[32] zu erschweren. Die Daten werden bei der Übertragung vom Client zum Webserver per TLS oder SSL verschlüsselt.

c) Allgemeiner Kontrollfluss

Eine Besonderheit beim Kontrollfluss einer Anwendung, die auf HTML Formularen aufsetzt, ist es, dass die Formulare nach dem Absenden nicht mehr gefüllt sind. Es muss also ein Mechanismus genutzt werden, der die gerade gesendeten Daten in das Formular einträgt. Dies kann entweder clientseitig, beispielsweise mit Javascript, realisiert werden oder auf Seiten des Servers, durch die verwendete Skriptsprache (in diesem Falle PHP). Da auf clientseitiges Scripting verzichtet wird um eine größtmögliche Kompatibilität zu erreichen, müssen die gesendeten Daten durch Vorausfüllen der vom Webserver ausgelieferten Webseite durch PHP erneut zum Client über-

30 CSV steht für „Comma Seperated Values“ und bezeichnet ein Datenformat, bei dem die einzelnen Werte durch ein Trennzeichen unterteilt in einzelne Zeilen einer Textdatei gespeichert werden. Als Trennzeichen werden typischerweise Kommas oder Semikolons verwendet.

31 Eine Einführung in den Themenkomplex Normalisierung und Update-Anomalien geben Elmasri und Navathe (2002: 504ff).

32 Bei SQL-Injection Angriffen versucht ein Angreifer in Eingabefeldern SQL Statements auf die Datenbank abzusetzen. Eine genauere Beschreibung mit Beispielen findet sich bei Bachfeld (vgl. Bachfeld 2004).

tragen werden. Um die geforderte Funktionalität bereitstellen zu können, müssen zusätzlich Statusinformationen übertragen werden. Ein erfolgter „Submit“ führt auf Serverseite zur Prüfung der Pflichtfelder. Wenn die Pflichtfelder gefüllt sind, muss der Nutzer trotzdem die Daten erneut bestätigen. Hierfür muss die Nutzerbestätigung separat erfasst, zum Client übertragen und in einem versteckten Formularfeld vermerkt werden.

d) Komponentenbeschreibung

Aus Gründen der Wartbarkeit und Wiederverwendbarkeit erfolgt eine Aufteilung der logisch zusammenhängenden Codeblöcke in einzelne Dateien, die als Includes eingebunden werden. Bei der Aufteilung wird darauf geachtet, dass die Zugriffe auf die Datenbank (lesend: starterliste.inc, anmeldezahlLesen.inc; schreibend: standardeintrag.inc, wartelisteneintrag.inc) jeweils in einzelnen Komponenten liegen. Weiterhin wird so weit wie möglich die Logik (index.inc) von den darstellenden Elementen (fomular.inc, bedingungen.html, index.php) getrennt. In Ansätzen handelt es sich also um eine Model/View/Controller Separierung[33].

- index.php: Rahmenseite mit Seitenkopf
  - Beinhaltet index.inc

  => Dient der Integration in die übergeordnete Seitenstruktur und deren Layout

- config.inc: Allgemeine Konstanten und Konfigurationseinstellungen
  => Bündelung der Einstellungen, die von vielen Komponenten genutzt werden

- index.inc: Hauptlogik

- formular.inc: Formularfelder
  => Bündelung der Darstellungskomponente

- standardeintrag.inc: SQL Statement und Meldung für reguläres Eintragen
  => Bündelung des schreibenden Zugriffs zur Datenbank für den Standardfall

- wartelisteneintrag.inc: SQL Statement und Meldung für Eintragen auf Warteliste
  => Bündelung des schreibenden Zugriffs zur Datenbank für den Wartelistenfall

- anmeldezahlLesen.inc: Liest die Anzahl der Anmeldungen der einzelnen Wettkampftypen
  =>Bündelung des lesenden Zugriffs zur Datenbank, mit Einfluss auf die Auswahlmöglichkeiten

- starterliste.inc: Ausgabe der Starterliste (ohne Wartelisteneinträge)
  => Ausgeben der Starterliste, diese Funktionalität soll auch unabhängig von der Registrierungsprozedur nutzbar sein und bündelt den dafür nötigen lesenden Zugriff

- bedingungen.html: Enthält die Bedingungen für den Wettkampf als statisches HTML
  => Rein beschreibender Text, ohne Einfluss auf die Logik, der bei Bedarf angepasst wird

3.2.3 Rationale

PHP/MySQL sind kostenlose Open-Source Lösungen und verfügen über einen be-

33 Dabei werden die Daten (Model) von den Sichten (Views) und der Steuerungslogik (Controller) getrennt. Zur MVC Architektur und weiteren Architekturstilen (vgl. Dutoit/Brügge 2004: 273f). Genau genommen handelt es sich in unserem Fall um die Ansätze einer MVC Architektur, die auf einer 3-Schicht Architektur abgebildet wird.

achtlichen Verbreitungsgrad. Ein Zugriff auf die Datenbank ist über die standardisierte Datenbank Abfrage-und Beschreibungssprache SQL möglich. Generell ist die Speicherung von Informationen in Dateien und besonders in strukturierten XML Dateien immer eine Option bei der Datenspeicherung.

Obwohl PHP seit der Version 5 in größerem Umfang Objektorientierung unterstützt, wurde im behandelten Beispiel der traditionelle, prozedurale Entwicklungsansatz gewählt. Dies ermöglicht einerseits zu prüfen, ob der TRAIN Prozess auch außerhalb des objektorientierten Kontexts einsetzbar ist und erscheint andererseits, aufgrund der Strukturierung der zu lösenden Aufgabe, sinnvoll, da es bei dabei um den PHP-typischen Einsatz für die Verarbeitung von Formulardaten und einfachen Tabellen geht. Komplexe Objekte, denen in sinnvollem Umfang Attribute und Methoden zugeordnet werden können, sind nicht auszumachen. Zudem war es in dem beschränkten Zeitrahmen nicht möglich, diesen neuen PHP Entwicklungsansatz zu vertiefen.

Optionen	Kriterien			
	Verbreitung/ Hostingkosten	Benötigt Clientinstallation	Administrations-oberfläche	Summe
PHP/MySQL	**+**	**++**	**++**	**5**
PHP/XML	+	++	+/-	3
Java Applet/MySQL	+	--	-	-2
Andere serverseitige Sprache	-	++	?	1

Abbildung 22: Rationale

3.3 Feinentwurf

3.3.1 Implementierungsziel

Das wichtigste Implementierungsziel ist die Bündelung der Angaben und Einstellungen („Flexibilität“), die jährlich angepasst werden müssen. Dies erfolgt durch die Konstanten (config.inc) und die Auswahlhilfe für die Geburtsjahre. Weiterhin soll der Code wartbar sein, daher wird er in kleinere funktionale Einheiten zerlegt und nach Kodierungsstandards ausgerichtet. Im PHP Umfeld ist der dominante Standard hierfür der PEAR Coding Standard[34].

3.3.2 Beschreibung der Komponenten

Da nicht dem objektorientierten Vorgehen gefolgt wurde, werden die Schritte von TRAIN zur Bestimmung der Klassen, Attribute und Operationen hier übersprungen und es wird direkt zur Komponentenbeschreibung übergegangen. Im Einzelnen beinhalten die Komponenten die folgenden Funktionalitäten:

34 Die PEAR Pakete enthalten viele Erweiterungen und Hilfsfunktionen für PHP. Die PEAR Coding Standards können unter http://pear.php.net/manual/de/standards.php eingesehen werden. In unserem Beispiel gilt dieser Standard als erreicht, wenn ein Code Formatierungstool (http://thephppro.com/tools/beautify.php) durchlaufen wurde. Das Werkzeug prüft jedoch nicht die Namensvergabe. Abweichend vom Standard werden Konstanten in unserem Beispiel nicht in Großbuchstaben angegeben.

- index.php: Rahmenseite mit Seitenkopf
  - Zeigt ein Logo an und beinhaltet index.inc

- config.inc: Allgemeine Konstanten und Konfigurationseinstellungen für andere Dateien
  - Verbindungsdaten zum Datenbankserver (dbhost, dbuser, dbpass, dbname)
  - Tabellenname in Datenbank (tabellenname)
  - URL der Startseite (startseite)
  - Maximale Teilnehmerzahl (erwachsenenLimit, staffelLimit, jugendschuelerLimit)
  - Größen der Wartelisten (erwachsenenWartelistengroesse, staffelWartelistengroesse jugendschuelerWartelisten-groesse)
  - Beschreibung der Triathlontypen als Text (txtSchuelerC,txtSchuelerB,txtSchuelerA,txtJugendB, txtJugendA, txtStandard, txtStaffel)

- index.inc: Hauptlogik, vgl. Aktivitätsdiagramm, Abbildung 23
  - Verwendet entferneSlashes.inc, bedingungen.html, formular.inc, standardeintrag.inc, wartelisteneintrag.inc

- formular.inc: Formularfelder, Besonderheit: Immer bevor das Formular ausgeliefert wird, muss der aktuelle Teilnehmerstand gelesen werden. Ist die Warteliste bereits voll, wird der entsprechende Triathlontyp nicht mehr zur Auswahl angeboten (*Limit + *Wartelistengroesse), hierfür wird anmeldezahlLesen.inc genutzt. Verwendet auswahlhilfe.inc. Enthält verstecktes Formularfeld NutzerOk

- standardeintrag.inc: SQL Statement und Meldung für reguläres Eintragen
  - verwendet starterliste.inc

- wartelisteneintrag.inc: SQL Statement und Meldung für Eintragen auf Warteliste
  - verwendet starterliste.inc

- anmeldezahlLesen.inc: Liest die Anzahl der Anmeldungen der einzelnen Wettkampftypen

- starterliste.inc: Ausgabe der Starterliste (ohne Wartelisteneinträge)
  - Format: Erster Buchstabe des Vornamens. Nachname Verein Ort Triathlontyp, sortiert nach 1.Wettkampftyp, 2.Ort

- bedingungen.html: Enthält die Bedingungen für den Wettkampf als statisches HTML

Im Zuge der Entwicklung wurden weiterhin folgende Hilfskomponenten hinzugefügt:

- entferneSlashes.inc : Behandelt das PHP-spezifische „Magic Quotes“ Problem. Es werden je nach Einstellung von PHP automatisch Backslashes vor Anführungszeichen eingefügt, die dann hier entfernt werden müssen, da die Filterung der Eingaben durch Verwendung von htmlspecialchars() vor dem Anzeigen und mysqli_real_escape_string() vor dem Schreiben in die Datenbank explizit berücksichtigt wird.

- auswahlhilfe.inc: Stellt eine Mehrfachselektion mit Geburtsjahren zur Auswahl bereit, die Mehrfachselektion soll wie alle Formulardaten nach erstmaligem Senden vorausgefüllt werden. Vereinfachend wird angenommen, dass kein Teilnehmer existiert der vor 1931 und nach 2008 geboren ist.

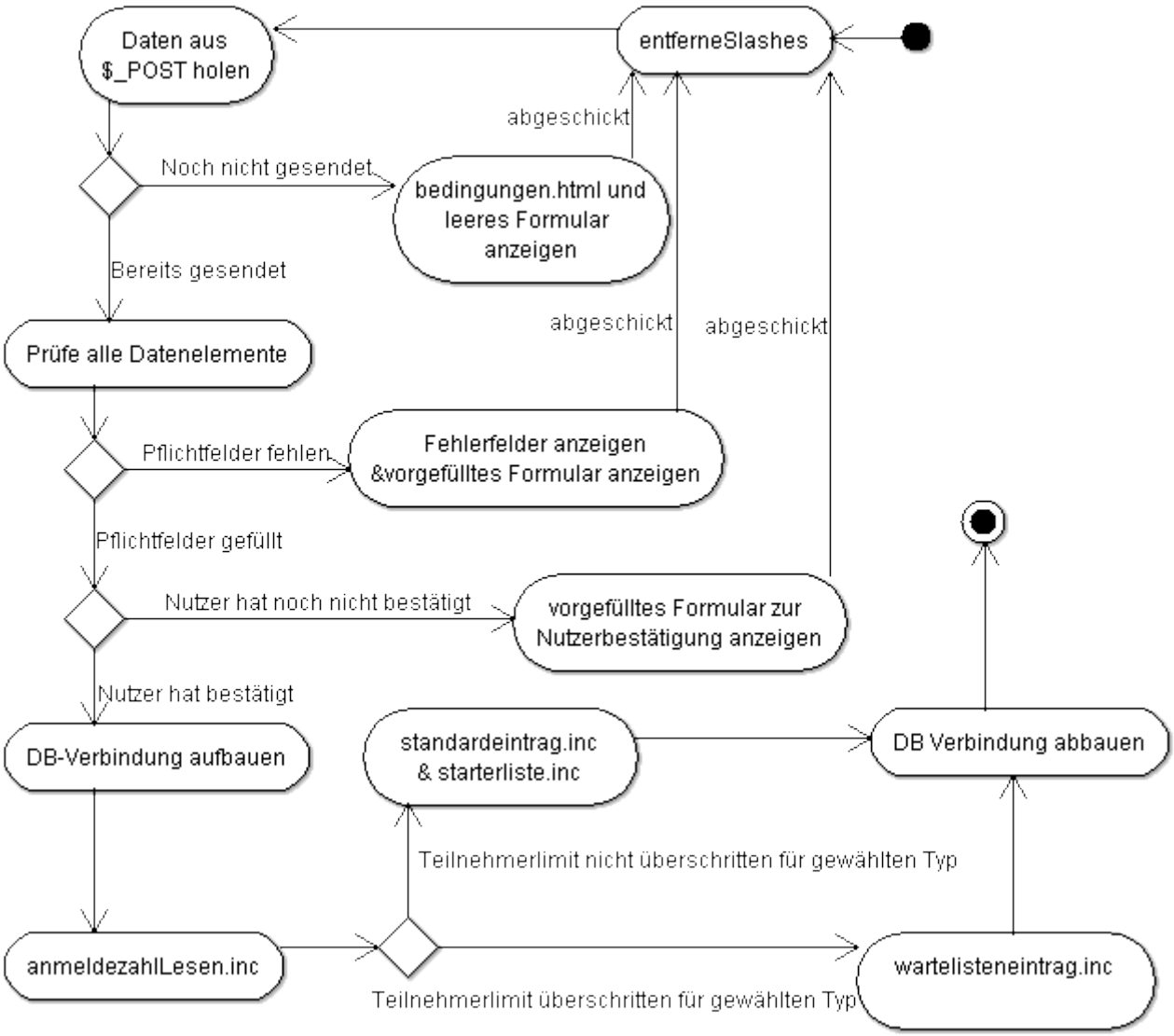


Abbildung 23: Aktivitätsdiagramm index.inc

Datenbank- und Eingabefelder

Formular (Feldlänge/Maximallänge)		Datenbank	
generiert		StarterID (PK)	mediumint(4), unsigned, autoinc
Nachname	40/50	Nachname	varchar(50)
Vorname	40/50	Vorname	varchar(50)
Geburtsjahr	4+(-01-01) = 6/6	Geburtsdatum	date
Geschlecht	1/1	Geschlecht	varchar(1)
Postleitzahl	5/5	Postleitzahl	varchar(5)
	40/50	Ort	varchar(50)
	40/50	Strasse	varchar(50)
	5/9	Hausnummer	varchar(9)
	40/50	Email	varchar(50)
	20/20	Telefon	varchar(20)
generiert	7-10	Wettkampftyp	varchar(10)
generiert	0/10	Starterstatus	varchar(10), NULL
	40/50	Verein	varchar(50), NULL
	200	Kommentar	varchar(200), NULL
	31/50	Starter1	varchar(50), NULL
	31/50	Starter2	varchar(50), NULL
	31/50	Starter3	varchar(50), NULL
generiert		Zeitstempel	varchar(20), NULL
zukünftige Nutzung		Startnummer	varchar(10), NULL
zukünftige Nutzung		Startgruppe	varchar(10), NULL

Abbildung 24: Formular und Datenbankfelder

3.3.3 Inspektion und Rationale

Die Inspektion überprüft die Konsistenz mit den vorhergehenden Phasen, Rationale fallen in unserem Fall keine weiteren an.

3.3.4 Integrations und Komponententest

Auf dieser Ebene werden die Spezifikationen für Integrations- und Komponententest erstellt. Aufgrund der begrenzten Komplexität unseres Systems kann der Integrationstest entfallen, da die Interaktion Client-Server-Datenbank durch den Systemtest mit abgedeckt wird. Das System gilt daher als getestet, wenn der Systemtest erfolgreich abgeschlossen wurde. Der Systemtest wird, wie bereits spezifiziert, im Blackboxverfahren[35] getestet. Für den Test genügt es ein Mitglied der jeweiligen Äquivalenzklasse, also einen typischen Vertreter der erwarteten Eingabedaten, zu testen. Eine Prüfung der Maximalwerte[36] erfolgt nicht, ebenso entfällt das Testen der ungülti-

35 Zu White- und Blackbox Testverfahren vgl. Spillner/Linz (2005: 109ff).

36 Im Allgemeinen erweisen sich Tests an den Grenzen der jeweiligen Äquivalenzklassen, zum Beispiel Maximal- oder Minimalwerte oder auch Übergangsfälle, ab denen etwas anderes

gen Äquivalenzkassen. Der Maximalwerttest ist nicht notwendig, da keine Zahlen verarbeitet werden, die eine Variable zum Überlaufen bringen könnten. Weiterhin sind alle Eingabefelder kürzer als die in der Datenbank zur Verfügung stehenden Längen der Zeichenketten. Das Testen auf ungültige Äquivalenzklassen ist ebenfalls nicht erforderlich, da alle Datenbankfelder als Zeichenkette variabler Länge deklariert werden, mit Ausnahme des Geburtsdatums, das als „Date“ deklariert, in unserem Fall aber fest mit dem Jahr aus der Auswahlhilfe gefüllt wird. Zusätzlich wird in einem Codereview überprüft, dass alle Eingabefelder nicht mehr Zeichen entgegennehmen, als in der Datenbank für das jeweilige Feld vorgesehen sind (vgl. Abbildung 24).

Der Systemtest muss noch um die folgenden Vorbedingungen ergänzt werden:

1) Datenbank ist geleert
2) Die folgenden Konstanten sind in include.inc gesetzt:
   - erwachsenenLimit = 1
   - erwachsenenWartelistengroesse = 1
   - staffelLimit = 1
   - jugendschuelerWartelistengroesse = 1
   - erwachsenenWartelistengroesse = 1
   - staffelWartelistengroesse = 1

Anders sieht dies für den Komponententest aus. Dort werden gewöhnlich die einzelnen Komponenten in einem Whitebox Verfahren getestet. Hierfür ist der Code bereits bekannt, und es wird versucht, möglichst alle Anweisungen, alle Zweige oder alle Bedingungen beim Testen zu „überdecken“, je nach dem, welches Kriterium zugrunde gelegt wird. In unserem Fall soll ein hybrides Vorgehen gewählt werden. Anhand des bereits spezifizierten (Blackbox) Systemtests erfolgt eine Whitebox Analyse, mit dem Ziel zu prüfen, welche Codeteile noch nicht überdeckt sind. Als Prüfkriterium soll eine Zweigüberdeckung von 100% erreicht werden. Mit der Zweigüberdeckung wird sichergestellt, dass während der Tests jede Möglichkeit des Programmflusses nach einer Bedingung mindestens einmal ausgeführt wird. Das Kriterium ist somit sehr streng, und kann auch selten auftretende Fälle erfassen.

3.3.5 Implementierung und Test

Der entwickelte Code findet sich in Anhang B. Bei der Durchführung des Komponententests[37] ergab sich, dass Abweichungen von den geforderten 100% Zweigüberdeckung sinnvoll sein können, denn nach diesem Kriterium müsste jedes einzelne Geburtsjahr getestet werden, ebenso wie jeder Triathlontyp. Auf den Test von entferneSlashes.inc wird ebenfalls verzichtet, da die Servereinstellung nicht geändert werden kann. Auf einen Test der Sicherheitsfunktionen wird verzichtet, da die dafür vorgesehene Standardfunktionalität von PHP verwendet wird (htmlspecialchars und mysqli_real_escape_string). Um alle übrigen Zweige zu überdecken, muss jeder Triathlontyp (Jugend/Schüler, Erwachsene, Staffel) einmal mit freien Plätzen, einmal mit freien Wartelistenplätzen und einmal ohne freie Plätze getestet werden. Der Testfall muss also zusätzlich in reduzierter Form für eine „junge Familie“ durchlaufen werden, die in den Schüler und Jugendklassen startet, sowie für eine Familie bei der Herr Müller für jedes Familienmitglied eine eigene Staffel anmeldet. Der Systemtest wird durchgeführt und mit dem Capture/Replay Werkzeug Selenium (vgl. Abbildung 25), das als Firefox Plug-in installiert wird, als Skript gespeichert. Die beschriebenen Erweiterungen des Systemtests werden im Skript angepasst.

passieren soll, als besonders ergiebig (vgl. ebd.: 111f).

37 Der Test erfolgt wie in 3.3.4 beschrieben durch ein Review des entwickelten Codes.

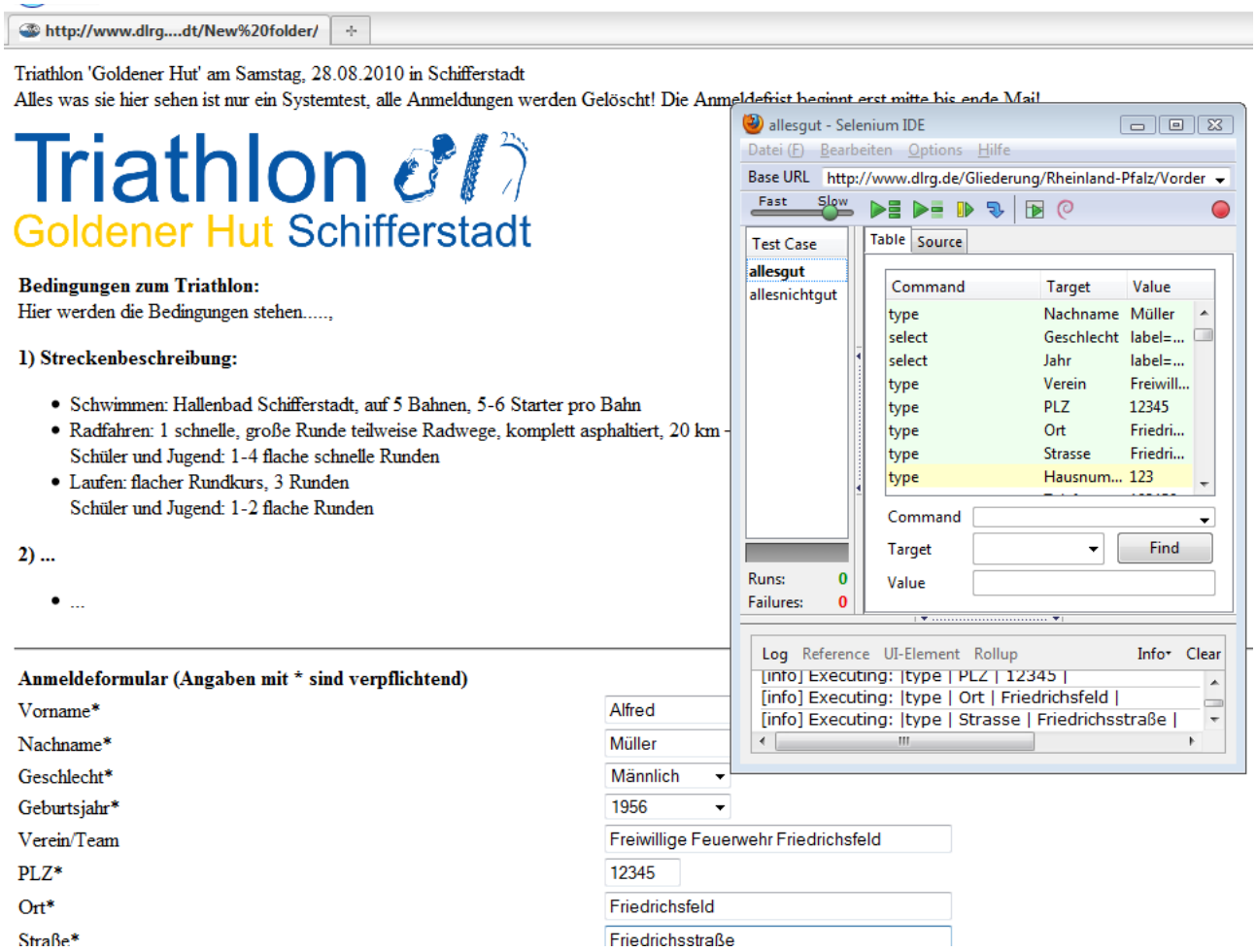


Abbildung 25: Selenium und Startseite der Komponente Teilnehmerregistrierung

3.4 Bewertung

Die hier vorgestellte Problemlösung ist noch keineswegs perfekt. Beim Test fiel auf, dass die Auswahlmöglichkeiten durch die Radiobuttons bis zur Durchführung eines erneuten Ladens der Internetseite zur Verfügung stehen. Hier besteht gegebenenfalls Nachbesserungsbedarf. Die Administrationsoberfläche, die von PHP MyAdmin bereitgestellt wird, ist zwar für die Zwecke ausreichend, und auch für Laien nach einer kurzen Einarbeitung verständlich, für das Ändern von vielen Datensätzen gleichzeitig ist sie jedoch weniger geeignet. Problematisch wird dies bei der Zuweisung von Startnummern, Startgruppen und dem Zahlungsstand. Hier sollte eine einfache, zweckgebundene Administrationsoberfläche nachgereicht werden. Besonders störend wirkt sich aus, dass der gewählte Hosting Dienstleister keine TLS/SSL gesicherte Verbindungen zulässt, so dass die Daten unverschlüsselt übertragen werden[38]. Da die Daten aber ohnehin so sparsam wie möglich erhoben werden, lässt sich dieser Effekt abfedern. Vor diesem Hintergrund kann darüber nachgedacht werden, Straße und Hausnummer nicht zu erfassen, da keine Notwendigkeit besteht die Starter

38 Dies ist jedoch weniger problematisch als es scheint, denn schließlich wird der Großteil aller E-Mails zumindest teilweise unverschlüsselt übertragen (vgl. Hochstätter 2008). Somit nimmt jeder dies billigend in Kauf. Dieser Tatbestand kann sich jedoch in einigen Jahren ändern, wenn mit dem De-Mail Projekt flächendeckend verschlüsselte E-Mail Kommunikation eingeführt wird.
Auf Nachfrage erklärte der IT Dienstleister, dass noch im laufenden Jahr Verschlüsselung für die DLRG Internetseiten eingeführt wird.

schriftlich zu benachrichtigen, sofern sie Zugriff auf eine E-Mail Adresse besitzen[39]. Daher erscheint es zukünftig auch sinnvoll, ein zweimaliges Eintippen der E-Mail Adresse zu verlangen und bei einer Abweichung den Nutzer zu benachrichtigen.

Entsprechende Änderungen sind aus Zeitgründen im Rahmen der vorliegende Arbeit nicht mehr zu realisieren und der Funktionsumfang musste auf einem definierten Stand festgeschrieben werden. Dies beeinträchtigt jedoch nicht die Einsetzbarkeit der Applikation. In einer weiterentwickelten Version sollten die Daten anhand einer Namensdatenbank besser auf Schreibfehler validiert werden. Gleichzeitig kann ein Abgleich des Felds „Geschlecht“ mit dem Vornamen erfolgen[40]. Eine Verbesserung der Datenqualität könnte möglicherweise durch eine Prüfung der einzelnen Felder, etwa von Postleitzahl oder E-Mail Adresse, auf Basis von regulären Ausdrücken, erreicht werden. Eine Prüfung, ob die gewählte Altersklasse (Triathlontyp) mit dem Geburtsjahr übereinstimmt, findet derzeit nicht statt. Gleiches gilt für den Staffeltriathlon. Dort wird nicht geprüft wird, ob die Staffelstarter angegeben wurden. Auch könnte der Ort auf Basis der Postleitzahl ermittelt werden. Die Auswahlhilfe für das Geburtsjahr lässt sich vermutlich eleganter realisieren.

Die visuelle Präsentation der Anwendung muss noch deutlich verbessert. Die Schriftgrößen und Platzierungen der einzelnen Meldungen müssen noch abgestimmt werden. Dies war jedoch von Beginn an bei der Definition des zu liefernden Ergebnisses so vorgesehen. Der „Abschicken“ Button zur Bestätigung der Daten durch den Nutzer ist möglicherweise nicht optimal positioniert, aber im Hinblick auf die Funktionalität, durchaus sinnvoll, da es den Nutzer zwingt, seine Daten tatsächlich nochmals zu betrachten. Zu testen bleibt, ob die Nutzer erkennen, dass sie nach dem ersten Bestätigungsschritt noch nicht angemeldet sind, da dies zu einem kritischen Benutzbarkeitsfehler führen kann. Der nächste wichtige Ausbauschritt der Anwendung wird sein, dass die Möglichkeit besteht, die verschiedenen Starterlisten, die bei der Wettkampfdurchführung benötigt werden, automatisch generieren zu lassen. Zwingend erforderlich ist aus rechtlicher Sicht außerdem ein Impressum sowie Hinweise zum Datenschutz. Aus konzeptioneller Sicht kann über die Verwendung von Sessions nachgedacht werden, damit die Informationen über die angemeldeten Teilnehmer nicht mehrfach aus der Datenbank ausgelesen werden müssen, sondern sessionspezifisch gespeichert werden können.

Obwohl der TRAIN-Prozess auf objektorientiertes Vorgehen ausgelegt ist, hat der Prozessdurchlauf gezeigt, dass er sich auch auf ein prozedurales Projekt anwenden lässt, sofern man Abstriche bei der Anforderungsanalyse des Systemkerns in Kauf nimmt.

Im folgenden Kapitel werden nun die grundsätzlichen Rahmenbedingungen für die Vermittlung der Themen Modellierung und Software Engineering an allgemeinbildenden Gymnasien untersucht (Kapitel 4.1). Auf dieser Grundlage und auf Basis der im Prozessverlauf gewonnenen Erfahrungen werden in einem zweiten Schritt Vorschläge zur Übernahme von Prozesselementen in den schulischen Unterricht entwickelt (Kapitel 4.2) .

39 Was bei praktisch allen Startern der Fall ist, zumindest wurden diese in der Vergangenheit so gut wie immer angegeben. Die wenigen Starter, die manuell vom Startlistenverwalter nachgetragen werden müssen, geben die Postadresse bei der Offlineanmeldung an und können separat berücksichtigt werden.

40 Jörg Michael hat hierzu Programm und Datenbank unter LGPL veröffentlicht (vgl. Michael 2007).

4. Didaktische Reduktion des TRAIN Prozesses

Als Ausgangspunkt für die Überarbeitung des TRAIN Prozesses werden zunächst die Lehrplanvorgaben untersucht, da diese den verpflichtenden Rahmen für die Unterrichtstätigkeit vorgeben. Im Anschluss wird eine allgemeine Bedingungsanalyse[41] erstellt werden.

4.1 Software Engineering im Lehrplan allgemeinbildender Gymnasien

Software Engineering gehört in Baden-Württemberg als fester Bestandteil zum Lehrplan für das berufliche Gymnasium[42].
Im Rahmen dieser Arbeit soll jedoch eine Konzentration auf das allgemeinbildende Gymnasium erfolgen, da dies den „Regelfall" der Sekundarstufe II darstellt. Zu diesem Zweck werden hier die Lehrpläne der Länder Rheinland-Pfalz und Baden-Württemberg genauer betrachtet. Eine Vermittlung von Software Engineering in der Mittelstufe erscheint wenig sinnvoll, da in dieser zunächst Grundlagen vermittelt werden müssen[43], daher erfolgt eine Beschränkung auf die Schuljahre oberhalb der 10. Jahrgangsstufe (Sekundarstufe II).
In Baden Württemberg führt das Schulfach Informatik außerhalb der beruflichen Gymnasien ein Schattendasein und steht lediglich als optionales Wahlfach[44] zur Verfügung. Dennoch wurden im Rahmen des Bildungsplans 2004 Standards für die Bildungsinhalte festgelegt (Ministerium für Kultus, Jugend und Sport 2004: 440ff). Dort sind die fachspezifischen Ziele als Leitideen „Information und Daten", „Algorithmen und Daten", „Informatik und Gesellschaft", „Wirkprinzipien von Informatik-Systemen" sowie „Problemlösen und Modellieren" strukturiert. Besonders die Zielsetzungen, die unter letztgenannter Leitidee explizit aufgeführt sind, unterstreichen die Bedeutung von Software-Engineering für den Unterricht. Im Einzelnen sind dies:

- Grundlegende Prinzipien beim Problemlösen
- Problem arbeitsteilig im Team lösen
- Problemlöseprozess strukturieren
- Basiskonzepte der objektorientierten Modellierung
- Reale Probleme in Objekte und Klassen abbilden
- Beziehungen zwischen Objekten beziehungsweise Klassen und die Kommunikation zwischen Objekten analysieren und beschreiben
- Eine Lösung dokumentieren, präsentieren und vertreten
- Ein Modell in einer Programmiersprache realisieren

Das Verständnis von Wirkprinzipien von Informatiksystemen hingegen wird eher indirekt durch Wiederholung und Vertiefung im Analyseprozess gefördert. Die anderen Leitideen sind in unserem Fall von untergeordneter Relevanz. Die zur Leitidee Problemlösen und Modellieren gehörenden Niveaukonkretisierungen sind wenig hilfreich, da sie nur einen sehr begrenzten Teilbereich abdecken (vgl.: Landesinstitut für

41 Das hier verwendete Schema ist Arnold 2004, S.73 entnommen .
42 So lässt sich beispielsweise für das Gymnasium mit wirtschaftlicher Ausrichtung (WG) im Fach Wirtschaftsinformatik eine umfangreiche Berücksichtigung feststellen. Im Lehrplan ist nicht nur vorgeschrieben, welche Prozesselemente und Entwurfstechniken zu vermitteln sind, sondern auch die Durchführung eines Entwicklungsprojekts ist als Bestandteil des Unterrichts vorgesehen.
43 Die Gesellschaft für Informatik (GI) hat für diese Klassenstufen einen Vorschlag für bundesweite Bildungsstandards verabschiedet, der sich mit dieser Bewertung deckt (vgl. GI 2008).
44 Der Wahlbereich wird in Baden -Württemberg mit Ausnahme von Fremdsprachen zweistündig unterrichtet.

Schulentwicklung 2003: 13f).
In Rheinland-Pfalz ist Informatik ein vollwertiges Unterrichtsfach, dass als Grund- oder Leistungskurs gewählt werden kann[45]. Entsprechend detailliert und umfangreich sind auch die Vorgaben im Lehrplan (Ministerium für Bildung, Wissenschaft, Jugend und Kultur 2008), in dem ebenfalls die Zielsetzungen „Wirkprinzipien von Informatiksystemen“ und „Modellierung von Informatiksystemen“ enthalten sind[46].
Als zu vermittelnde Kompetenz wird insbesondere „Software verantwortungsbewusst, systematisch und kooperativ entwickeln“ (Ministerium für Bildung, Wissenschaft, Jugend und Kultur Rheinland-Pfalz 2008: 38) vorgeschrieben.
Diese Kompetenz wird im Lehrplan in drei Teilziele untergliedert (vgl. Abbildung 26).

Software-Entwicklungsprozesse systematisch durchführen	
Verbindliche Inhalte	**Hinweise für eine mögliche Umsetzung**
Entwicklungsschritte: Anforderungsanalyse, Modellierung, Implementierung, Testen	➢ Die Vorgehensweise bei der Entwicklung von Software reflektieren und planen. ➢ Bei Entwicklungsprozessen exemplarisch einen der folgenden Schritte vertiefen: – Die Anforderungen ermitteln (z. B. über Fallstudien) und mit einem Pflichtenheft dokumentieren – Modellierungsansätze passend auswählen (zustandsbasiert, objektorientiert, algorithmisch) und das zu entwickelnde System mit Hilfe von Modellen systematisch konzipieren – Die entwickelten Modelle mit der gewählten Programmiersprache implementieren – Alle Funktionalitäten des entwickelten Systems gezielt und isoliert testen
Dokumentation	➢ Schritte exemplarisch dokumentieren (z.B. Pflichtenheft, Schnittstelle einer Klasse, kommentiertes Programm, Testprotokoll). ➢ Werkzeuge (z.B. HTML-Editor, CASE-Tool) bei der Dokumentation einsetzen.

45 Grundkurse werden in Rheinland-Pfalz dreistündig,, Leistungskurse fünfstündig erteilt.
46 In Anlehnung an das „Gesamtkonzept zur informatischen Bildung an allgemein bildenden Schulen“ der Gesellschaft für Informatik (GI 2000: 3).

Additum: Ein Software-Entwicklungs-Projekt organisieren	
Verbindliche Inhalte	**Hinweise für eine mögliche Umsetzung**
Organisationsbereiche: Projektauftrag, Aufgabenverteilung, Arbeitsplan, Absprachen	➢ Ein Projekt durchführen, bei dem eine gewisse Selbstständigkeit in Organisation und Durchführung erforderlich ist. ➢ Die Organisation des Software-Projektes gemeinsam besprechen und in Teilen von Schülerinnen und Schülern übernehmen lassen. ➢ Aufgabenverteilung und Rollenzuweisung klar vereinbaren. ➢ Rahmenbedingungen vorab klären (insbesondere Zeitvorgaben). ➢ Kommunikations- und Kooperationssysteme zum Austausch von Informationen einsetzen. ➢ Sowohl Zwischen- als auch Endergebnisse präsentieren und diskutieren.

Abbildung 26: Lehrplanvorgaben Rheinland-Pfalz (Grundkurs)
Quelle: Ministerium für Bildung, Wissenschaft, Jugend und Kultur Rheinland-Pfalz 2008: 39f

Die aufgeführten Lehrplaninhalte (vgl. Abbildung 26) sind dem Grundkurs entnommen. Der Unterschied zum Leistungskurs besteht darin, dass im Bereich „Qualitätsmerkmale für Software Kennen und beachten“ explizit eine Thematisierung der Qualitätskriterien aus DIN 66272 bzw. ISO/IEC 9126 vorgeschrieben ist[47]. Der Zweite Unterschied besteht darin, dass das Softwareprojekt nicht optional, sondern fest im Lehrplan vorgesehen ist. Bemerkenswert ist, dass im Lehrplan bewusst keine Vorschriften zur zeitlichen Einteilung für die Vermittlung der einzelnen Kompetenzen vorgegeben ist. Im besten denkbaren Fall handelt es sich vermutlich um eine Projektwoche, die aus 32 bis 40 Stunden und abschließender Ergebnispräsentation besteht. Im regulären Unterricht, der mit drei Stunden pro Woche (Grundkurs) erteilt wird, würde dies 11-13 Wochen bedeuten, was eher unrealistisch umfangreich erscheint. Im Leistungskurs (fünf Wochenstunden) entsprechen 40 Stunden einer Dauer von 8 Wochen, was ein realisierbarer Umfang zu sein scheint. Im schlechtesten Fall ist die Zeit nicht ausreichend um ein eigenes zusammenhängendes Projekt zu realisieren. Dann kann nur eine stückweise Vermittlung erfolgen.
Aus den Lehrplanvorgaben lässt sich jedoch folgern, dass ein Softwareentwicklungsprojekt entweder separat durchgeführt werden oder aufgrund des beschränkten Zeitrahmens im Grundkurs möglichst in der Vermittlung des Softwareentwicklungsprozesses einbezogen werden soll. Auch im Leistungskurs erscheint es sinnvoll, die im Projekt genutzten Prozesselemente bereits einzuführen, um sie in der Projektdurchführung aufzugreifen und selbständig anwenden zu lassen. Um ein genaueres Verständnis der Schülerzielgruppe zu erhalten wird im Folgenden eine Bedingungsanalyse erstellt, die sich aus eine Adressatenanalyse und einer Untersuchung der soziokulturellen Bedingungen zusammensetzt.

47 Im Lehrplan sind die aus diesen Standards hervorgehenden Qualitätskriterien Funktionalität, Zuverlässigeit, Benutzbarkeit, Effizienz, Änderbarkeit und Übertragbarkeit aufgeführt.

4.2 Bedingungsanalyse und Begründungszusammenhang

4.2.1 Adressatenanalyse: Lernfähigkeit und - bereitschaft

Die Schüler verfügen bereits über grundlegende JAVA Kenntnisse oder vergleichbare Fertigkeiten in der Verwendung einer anderen Programmiersprache. Weiterhin sind Computer- und Internetgrundkenntnisse vorhanden, denn im Anfangsunterricht erscheint eine umfassende Thematisierung von Software-Engineering nicht sinnvoll, da zunächst die grundlegenden Prinzipien und Sprachkonstrukte einer Programmiersprache vermittelt werden müssen. Die Schüler sind es gewohnt selbständig an separaten Computern Aufgaben zu bearbeiten. Sollte der Unterricht im freiwilligen Wahlbereich angesiedelt sein, ist damit zu rechnen, dass die Schüler zu einer lockeren Arbeitseinstellung neigen, die dann gegebenenfalls korrigiert werden muss.
Die Schüler haben Informatik als Fach gewählt, bringen also eine gewisse intrinsische Grundmotivation mit. Diese kann allerdings auch aus falschen Vorstellungen des Unterrichtsfachs Informatik resultieren. Zur Unterstützung der Lernbereitschaft ist es sinnvoll, eine Trennung zwischen Computerarbeitsplätzen und regulären Unterrichtstischen vorzunehmen, sowie die Internetverbindung der Schülerrechner zu trennen, falls dies möglich ist.

4.2.2 Analyse der soziokulturellen Bedingungen

Über die *Sozioökonomischen* und *Sozioökologischen* Voraussetzungen kann für den allgemeinen Fall keine Aussage getroffen werden. Im Bereich der *Soziokulturellen* Voraussetzungen kann jedoch davon ausgegangen werden, dass für Schüler der Sekundarstufe II Computer und Internet zum festen Bestandteil des alltäglichen Lebens gehören[48]. Dementsprechend ist von der Kenntnis allgemeiner Begrifflichkeiten und Vorgehensweisen in diesen Sachgebieten auszugehen und zu erwarten, dass die Schüler sich schnell in neue Werkzeuge einarbeiten können, die zur Unterstützung des Lernprozesses eingesetzt werden. Die Gesellschaft erwartet als *Ideologisch normbildende* Voraussetzungen von den Schülern ein (im Bildungsplan konkretisiertes) Beherrschen der grundlegenden Programmiertechniken und Konzepte. Aus Sicht der Unternehmen ist es begrüßenswert, wenn diese Techniken bereits in der Schule in systematisierter und wirtschaftsnaher Form eingesetzt werden. Dabei ist jedoch zu beachten, dass gerade beim Schulfach Informatik die Zeitspanne bis zum Eintritt ins Berufsleben besonders ins Gewicht fällt, da sich Technologien und Standards innerhalb von fünf bis zehn Jahren ab der elften Klassenstufe deutlich verändern können. Die grundlegende Fähigkeit Aufgaben zu strukturierten und bis zu einem definierten Zeitpunkt abzuschließen ist jedoch als allgemein erstrebenswerte Kompetenz anzusehen.

4.2.3 Begründungszusammenhang

Ausgehend von den Ergebnissen dieser grundlegenden Bedingungsanalyse ist nun in Anlehnung an das Perspektivenschema von Klafki[49] der Begründungszusammen-

48 Dieser Befund wird durch die Daten der ARD/ZDF-Onlinestudie 2007 erhärtet, so kann man, zumindest bei den 14- bis 29-jährigen feststellen, dass diese zu ca. 95% das Internet nutzen (ARD/ZDF 2007). Für die von uns betrachteten Schülergruppe eines Gymnasiums kann also berechtigt angenommen werden, dass der Wert bei 100% liegt.

49 Nach Jank/Meyer 1991, S. 236f, zusammenfassend Arnold/Pätzold 2002, S. 98f.

hang des Themas „Software Engineering“ näher zu betrachten. Die *Gegenwartsbedeutung* stellt sich hierbei wie folgt dar: Den Schülern ist bestenfalls unterbewusst klar, dass es Methoden zur strukturierten Konzeption und Entwicklung von Softwaresystemen gibt. Sie haben bereits erste Erfahrungen mit Struktogrammen, Programmablaufplänen[50] oder UML Elementen[51] gesammelt, vermutlich haben sie diese allerdings als lästiges Beiwerk zur Programmierung angesehen und nur in enger Verbindung zu kleineren Codeblöcken kennengelernt. Die Bedeutung dieser Konstrukte und deren Einbindung in einen übergeordneten Prozess sowie die damit verbundenen praktischen Implikationen sind den Schülern hingegen nahezu unbekannt. Die pädagogisch gewünschte *Zukunftsbedeutung* ergibt sich aus der Betrachtung des Lehrplans (vgl. Kapitel 4.1), während die *Exemplarische Bedeutung* am Besten durch nahe an der Lebenswirklichkeit der Schüler ansetzende Problemstellungen unterstützt wird. Zusätzlich bietet sich eine fächerübergreifende Zusammenarbeit, beispielsweise zum Thema Projektplanung, an. Dies schafft die Gelegenheit den Lerninhalt mit anderen Aspekten und Inhalten der Disziplin Informatik zu verknüpfen, insbesondere bei der Frage der Architekturdefinition mittels der konzeptionelle Fragen, Datenübertragung und Sicherheitsaspekte aufgegriffen werden können. Internetprojekte bieten zudem Ansatzpunkte zum Themenkomplex Datenschutz und anderen rechtlichen Fragen, die in den Bereich „Informatik und Gesellschaft“ hineinreichen.
Nicht vergessen werden sollte auch, dass das Schulfach Informatik wie kein weiteres Fach exemplarisch für die Klasse der praxis- und produktorientierten ingenieurwissenschaftlichen Studienfächer steht, und entsprechendes planvolle Vorgehen vermitteln kann, während in den Naturwissenschaften eher grundlegende Fragestellungen der Forschung thematisiert werden.

50 Struktogramme werden nach ihren Erfindern auch Nassi–Shneiderman (vgl. Nassi/Shneiderman 1973) Diagramme genannt, Programmablaufpläne sind in Deutschland normiert nach DIN 66001.

51 Insbesondere Klassendiagramme lassen sich natürlich schnell und problemlos im Anfangsunterricht einführen, wenn auf Beziehungen zwischen den Klassen zunächst verzichtet wird.

4.3 Anwendbarkeit des TRAIN Prozesses

Aus der Untersuchung der Rahmenbedingungen ergeben sich verschiedene Ansatzpunkte bei der Adaption des Prozesses. Aus den Lehrplanvorgaben lässt sich folgern, dass ein Softwareentwicklungsprojekt entweder separat durchgeführt werden, oder auf Grund des beschränkten Zeitrahmens im Grundkurs möglichst in der Vermittlung des Softwareentwicklungsprozesses einbezogen werden soll. Es sind also zwei grundlegende Fälle zu unterscheiden. Einerseits der *unterrichtsbegleitende Einsatz* einzelner Elemente und andererseits der umfangreichere *Einsatz in einer Projektarbeit*. Hier soll zunächst nur der Einsatz in einer Projektarbeit thematisiert, und ein reduzierter Prozess entwickelt werden, der dann bei der Entwicklung der Komponente Cupwertung getestet wird (Kapitel 5). In der abschließenden Bewertung (Kapitel 6) wird jedoch auch auf den unterrichtsbegleitenden Einsatz Bezug genommen.

Die Voraussetzungen der Schüler lassen erwarten, dass sie selbständiges Arbeiten gewohnt sind und über eine grundsätzlich positive Einstellung zum Fach besitzen. Durch die vorhandene Computererfahrung verfügen sie über ein grundlegendes Bewußtsein für mit IT erzielbare Ergebnisse und der prinzipiellen Fähigkeit sich in neue Benutzeroberflächen von Werkzeugen einzuarbeiten. Der Einsatz von unterschiedlichen unterstützenden Werkzeugen ist also legitim. Der Begründungszusammenhang verdeutlicht, dass es zentral ist, dass die Schüler die Nützlichkeit der Methoden erfahren und die Zusammenhänge zwischen den einzelnen Elementen von Modellierung und Prozess betont werden. Bei der Wahl der Aufgabenstellung sollte ein schülernahes Beispiel gewählt werden, das die Schüler kennen oder bereits eingesetzt haben und dessen Nutzen sich den Schülern erschließt. Synergieeffekte mit anderen Fächern sollten wenn möglich genutzt werden.

Zu diesen Rahmenbedingungen aus dem schulischen Umfeld müssen nun die Rahmenbedingungen, die dem Einsatz des TRAIN Prozesses im Universitären Umfeld zugrunde liegen, vergleichend betrachtet werden. Dort werden als Zeitbudget sechs Semesterwochenstunden[52] eingeplant, wovon zwei auf die Vorlesung und vier auf die praktischen Übungen entfallen (vgl. Paech et al. 2005: 56).

Der Betreuungsaufwand durch die Lehrkräfte ist während des gesamten Semesters relativ groß. Um den Aufwand für individuelle Rückmeldungen zu reduzieren, werden Gruppen von vier bis fünf Studenten zusammengefasst. Im Hinblick auf den notwendigen Zeitaufwand auf Seiten der Studenten zeigt sich, dass eine Mehrheit mindestens neun Stunden pro Woche zusätzlich aufwendet, um die Aufgaben zu bearbeiten (vgl. Borner et al. 2006: 7).
Betrachtet man das gegebene Zeitkontingent in der Schule (vgl. Kapitel 4.1), so ist klar, dass besonders die Zeit für die praktischen Übungen reduziert werden muss, selbst wenn davon ausgegangen wird, dass die verwendeten Modellierungstechniken bereits zuvor erarbeitet wurden und im Rahmen eines Projektes nur noch wiederholt werden müssen. In diesem Fall kann die theoretische Konzeptvermittlung geringer ausfallen. Ohnehin muss der Schulunterricht nicht dem Anspruch einer möglichst umfassenden Thematisierung von Software Engineering gerecht werden, der im universitären Umfeld als Maßstab gilt.
Gerade in der knapp bemessenen Zeit, die an der Schule zur Verfügung steht, müssen Modellbildungs- und Programmierphasen gegeneinander abgewogen werden. Dies ist auch bei der Umsetzung des TRAIN Prozesses zu beachten. In der Fachliteratur gibt es zu dieser Schwerpunktsetzung keine durchgängige Meinung, mitunter

52 Eine SWS entspricht dabei ungefähr 14 Schulstunden.

wird sogar zu einer Minimierung von Programmiertätigkeiten im Unterricht geraten[53]. Auf der anderen Seite wird unabhängig vom Fach Informatik gerade Handlungsorientierung gefordert, die Handlungsprodukte zum Ziel hat. Die Produkte sollen schüleraktiv Kopf- und Handarbeit in den Unterricht integrieren und an den (subjektiven) Schülerinteressen[54] orientiert sein[55].
Was sind also die entscheidenden Elemente des Prozesses, die diese Aspekte berücksichtigen und die gleichzeitig Raum für Kreativität lassen?[56] Die Essenz des TRAIN Prozesses ist es, Testen, Rationale und Inspektionen systematisch zu nutzen und mit einer gründlichen Anforderungsanalyse zu kombinieren.
Diese Elemente werden im Folgenden jeweils einzeln betrachtet und auf ihre Umsetzbarkeit untersucht. Im Anschluss werden die einzelnen Prozessebenen nochmals betrachtet.

53 Die entsprechende Argumentation legt Hubweser dar, der für eine Reduzierung der Programmiertätigkeit im allgemeinbildenden Informatikunterricht eintritt (vgl. Hubwieser 2004: 86ff). Auch für Sigrid Schubert und Andreas Schwill ist Modellierung das dominierende Thema gegenüber der Programmierung, der jedoch eine größere Rolle zugebilligt wird (Schubert/Schwill 2004: 85ff).

54 Subjektive Schülerinteressen sind die Interessen, die Schüler von sich aus verfolgen und anstreben (vgl. Meyer 1987b: 314).

55 Handlungsorientierung wird beispielsweise bei Janck und Meyer thematisiert (vgl. Meyer 1987a: 214ff und Janck/Meyer 1991: 314ff).

56 Kreativität kommt im Informatikunterricht häufig zu kurz, wie Ralf Romeike bei einer Analyse der Unterrichtsvorschläge in der didaktischen Fachzeitschrift „LOG IN“ feststellt (vgl. Romeike 2007).

4.3.1 Testen

Entscheidend ist, dass die Tätigkeit des Testens als fester und bedeutender Bestandteil der Softwareentwicklung wahrgenommen wird. Cristensen schlägt hierfür verschiedene Vorgehensweisen vor. So soll zunächst die Qualität der Programmteile und der Dokumentation bei der Bewertung ein höheres Gewicht bekommen als der implementierte Umfang der Funktionalität. Weiterhin sollen die Lerner einen konkreten Vorteil von ihren realisierten Tests haben. Dies kann durch eine iterative Aufgabenstellung realisiert werden, die es den Schülern vorschreibt, den eigenen Code über lange Zeit zu nutzen und ihn mehrfach zu modifizieren. Dabei führen funktionierende Regressionstests mit sehr hoher Wahrscheinlichkeit zum Entdecken von Fehlern und die Schüler erfahren, dass die Tests ihnen unmittelbar nützen (vgl. Cristensen 2008).
Da in der Regel noch keine gründlichen Programmierkenntnisse vorliegen, und der Programmiertätigkeit nur eine Hilfsfunktion zukommt, sollten die System-, Komponenten- und Integrationstests weitgehend automatisiert erstellt werden können. In Kapitel 3 wurde dies bereits durch einen GUI-basierten Testansatz und der Nutzung von Selenium realisiert. Gegebenenfalls kann hierfür noch ein besseres Werkzeug gefunden werden, dass auch Zweigüberdeckung unterstützt. Auf der anderen Seite bietet sich gerade beim manuellen Codereview die Möglichkeit, die Fähigkeit zum Lesen von Code weiter auszubauen. Zurecht wird kritisiert, dass in der Informatik häufig das Schreiben erlernt wird, bevor die Schüler und Studenten das Lesen beherrschen (vgl. Hartmann, Näf, Reichert 2007: 131f)[57]. Zusammen mit der Erkenntnis, dass ein Großteil der Kosten eines Softwaresystems in der Wartungsphase entstehen (vgl. ebd.), in der Lesen, Verstehen und Anpassen von vorhandenem Code die Hauptaufgaben sind, erscheint das Untersuchen des Codes einer anderen Gruppe äußerst sinnvoll. Es lässt die Schüler erfahren, wie wichtig eine ausführliche Codekommentierung unter dem Leitmotto „Warum, nicht Was!“ und die Verwendung von sprechenden Variablennamen ist, und dass die eigene, bereits geschriebene Kommentierung nicht ausreichend ist, um einem Außenstehenden die Funktionsweise des Programmteils zu vermitteln.
Gleichzeitig ermöglicht es Schülern, die verbreitete und effiziente Vorgehensweise des „Suchen-Lesen-Einfügen“ (vgl. ebd.) zu vermitteln, bei der bereits existente im Internet oder anderen Bibliotheken vorliegende Problemlösungen gefunden, verstanden und in ein eigenes Projekt integriert werden müssen[58].

4.3.2 Rationale

Rationale sind eine aufwandsarme Möglichkeit Designentscheidungen zu dokumentieren. Die Schüler sollten daher ermutigt werden diese zu nutzen, wobei die Anzahl der dokumentierten Rationale. Es ist jedoch davon auszugehen, dass sie sich meist nicht über Alternativen und deren Vor- und Nachteile bewusst sind, besonders wenn es um Architekturfragen geht. Dann ist eine Vorgabe von Optionen und möglichen Bewertungskriterien sinnvoll. Dies wird auch im universitären Einsatz so gehandhabt.

57 An der Universität Heidelberg wird auch aus diesem Grund zu Beginn eine vorhandenen Komponente von den Studenten anhand des Codes analysiert und anschließend erweitert.

58 Die fertigen Bausteine werden häufig als „Idiome“ oder Muster bezeichnet, wobei sie auf einer niedereren programmiersprachenspezifischen Ebene angesiedelt sind, als Entwurfsmuster.

4.3.3 Inspektionen

Inspektionen bieten eine gute Gelegenheit alternative Lösungswege nachzuvollziehen, wenn sie in der Schule von einer anderen Gruppe übernommen werden. Gleichzeitig bietet dies die Möglichkeit der Selbsteinschätzung („Comprehension Monitoring“) im Verhältnis zu den anderen Schülern, was eine positive Auswirkung auf den Lernprozess hat (vgl. Gudjons 2003: 141).

4.3.4 Gesamtprozess und Projektmanagement

Beim Einsatz von unterstützenden Werkzeugen ist im ganzen Prozess darauf zu achten, dass diese einfach gehalten sind[59] und möglichst in einer deutschen Übersetzung sowie als Open Source Software vorliegen[60], denn nur so ist gewährleistet, dass diese dauerhaft kostenlos verfügbar sind.
Besonders in einem derartig langen und komplexen Prozess, erscheint die Nutzung von Advance Organizern[61] sinnvoll um zu verhindern, dass sich die Schüler im Prozess und zwischen Fachbegriffen verlieren. Ein Wandplakat bietet sich hierfür an. Sofern bereits behandelte Elemente enthalten sind, können diese auch von den Schülern selbst gestaltet werden. Im Hinblick auf die Organizerfunktion, muss das Plakat jedoch in jedem Fall von der Lehrkraft ergänzt werden. Advance Organizer könnten auch hilfreich dabei sein, das Problem der unterschiedlichen Terminologien auf der konzeptionellen Ebene und in der Programmiersprache zu reduzieren[62].
Gerade wenn das Projekt unter Beteiligung anderer Fächer oder im Rahmen einer Projektwoche durchgeführt wird, kann die Projektmanagementkomponente ausgebaut werden. Hierfür bietet es sich an entweder OpenProj zu nutzen oder das webbasierte und auf Zusammenarbeit ausgelegte PHProjekt einzusetzen[63]. In jedem Fall sollte ein derartiges Werkzeug für die initiale Zeitplanung genutzt werden um ein Gantt Diagramm des zeitlichen Projektverlaufs zu erstellen.
Vocke und Woigt geben einige Beispiele für zusätzliche realistische Komponenten, die den Bezug zur Realität erhöhen können (vgl. Vocke/Woigt 2005). Im schulischen Umfeld muss jedoch genau abgewogen werden, ob derartige „Störungen“ des Ablaufs sinnvoll sind, und die Schüler nicht zu sehr vom eigentlichen Entwurfsprozess ablenken.

---

59 Börstler argumentiert hier, dass die Kognitive Belastung („Cognitive Load“) der Lernenden reduziert werden sollte, um eine Konzentration auf die wesentlichen Inhalte zu ermöglichen (vgl. Börstler 2007: 15)

60 Open Source Software kann nicht mehr „zurückgezogen“ werden und kann gegebenenfalls von einem neuen Entwicklerteam weiterentwickelt werden. Im Idealfall hat das genutzte Projekt natürlich bereits eine aktive Entwicklergemeinschaft.

61 Advance Organizer strukturieren bereits zu Beginn ein Thema vor und fassen erklärend Inhalte zusammen. Besonders wirksam sind Advance Organizer, wenn sie Analogien zu anderen Lebensbereichen oder Beispiele enthalten (vgl. Hartmann/Näf/Reichert 2007: 109ff und Gudjons 2003: 53f).

62 Beispielsweise „Attribut“ als Konzept und „Variable“ in einer konkreten Programmiersprache (vgl. Börstler 2007: 16).

63 Die Programme können unter openproj.org und phprojekt.com bezogen werden. OpenProj hat eine komplexere Oberfläche und ist an das weit verbreitete MS Project angelehnt. Die PHProjekt Oberfläche erscheint weniger komplex und bietet zusätzlich Unterstützung für Dateientausch und Kommunikation, damit ist es in der Lage als CVS Ersatz zu fungieren, was die zusätzliche Komplexität vermeidet und dennoch ähnliche grundlegende Fähigkeiten bereitstellt, damit die Schüler je nach Motivationslage auch außerhalb der Schule weiter arbeiten können.

4.3.5 Anforderungsanalyse

Das Hauptproblem bei der Anpassung des Prozesses an schulische Anforderungen ist die große Anzahl an Artefakten, die erstellt und aktuell gehalten werden müssen. Für kleinere Projekte, die nicht produktiv eingesetzt werden, und die somit auch eine weniger umfangreiche Dokumentation benötigen, erscheint der Aufwand im Verhältnis zum Nutzen zu groß. Der Glossar wird daher gestrichen. An dessen Stelle tritt eine ausführliche Dokumentation im Code. Außerdem werden nicht-funktionale Anforderungen auf allen Ebenen sehr sparsam eingesetzt. Für die nicht-funktionalen Anforderungen, die sich auf die Benutzbarkeit beziehen, ist ein möglicher Ansatz Usability Tests durchzuführen, bei denen die Benutzer bestimmte Aufgaben erledigen sollen. Dabei werden auftretende Probleme notiert und im Nachhinein nicht-funktionale Anforderungen isoliert. Dies bietet für die Schüler die Möglichkeit, genau solche fehlenden Qualitätskriterien zu entdecken. Dies ist im Hinblick auf den Lerneffekt positiv zu bewerten und verdeutlicht die Nützlichkeit von Tests.

Aufgaben und Domänenebene

Um die Prozessartefakte zu reduzieren werden Aufgaben- und Domänenebene zusammengefasst, da die Aufgaben ohnehin direkt in den Prozessen der Domänenebene aufgehen. Der Ist-Prozess wird hierbei nur genutzt, falls dies sinnvoll erscheint, denn bei der Programmierung eines Computerspiels oder eines Chatprogramms ist ein Prozess ohne Computereinsatz kaum vorstellbar und wenig hilfreich.

Um eine weitere Reduktion zu erreichen ist das Vorgehen im Folgenden vergleichbar zum „(strictly) Objects First“ Ansatz[64], bei dem zunächst nur Objekte sowie deren Interaktion und Ausprägungen festgehalten werden. In unserem Fall werden jedoch keine Objekte sondern die Artefakte, die konkrete Ausprägungen enthalten, übernommen und die anderen weitestgehend verworfen. Dies stärkt gleichzeitig den Bezug zur realen Umwelt der Schüler. Aus dem Soll-Prozess werden lediglich die jeweiligen Rolleninstanzen übernommen, die später als Grundlage für die Szenarien dienen. Die Rollen selbst werden nicht mehr separat beschrieben. Die Domänendaten und die Systemverantwortlichkeiten werden jedoch beibehalten, wobei Erstere möglicherweise auch zunächst als Ausprägungen erfasst werden könnten.

Interaktionsebene

Auf der Interaktionsebene ergibt sich erneut die Möglichkeit Ausprägungen als Grundlage für das weitere Vorgehen zu verwenden. Die Szenarien sind Ausprägungen der Interaktionsbeschreibungen. Somit können die Interaktionsbeschreibungen ohne größeren Informationsverlust entfallen. Eine Verfeinerung des Datenmodells, wie sie von Häfele (Häfele 2005: 59) beschrieben wird, erfolgt auf dieser Ebene nicht. Die Systemtestspezifikation wird wie bereits im ersten Prozessdurchlauf auf Basis der Szenarios erstellt.

Systemebene

Beim Design der Benutzungsschnittstelle kann auf Musterbibliotheken verwiesen werden. Naheliegender ist es jedoch, dass die Schüler zunächst eine Basisversion erstellen und diese dann kreativ ausbauen[65]. Da im ersten Durchgang der Systemkern übersprungen wurde, muss diese Komponente jetzt durchlaufen werden, bevor

64 Ira Diethelm hat ein umfassendes Konzept zum Einsatz im Unterricht vorgelegt (vgl. Diethelm 2007).

65 Hier bietet sich die Gelegenheit für kreatives Arbeiten, sofern keine guten Gründe dagegen sprechen.

eine Beurteilung vorgenommen werden kann. Die Phase der Anforderungsanalyse wird mit einer Inspektion durch eine andere Gruppe abgeschlossen.

4.3.6 Architekturdefinition und Feinentwurf

Die Architekturdefinition ist besonders heikel, da die Lehrkraft nicht zu viele Vorgaben machen sollte, die Schüler aber noch nicht über genügend eigene Erfahrung und Wissen verfügen. Eine Thematisierung von Alternativen bietet sich an, die die Schüler dann auf Basis von Rationale selbst auswählen können. Für den Feinentwurf gilt, dass dieser im Folgenden zunächst in seiner ursprünglichen Form durchlaufen werden muss, bevor Veränderungen vorgenommen werden.

4.3.7 Implementierung und Test

Bei der Implementierung sind Namenskonventionen zu befolgen. Im Idealfall werden diese auf wenige, wichtige Punkte reduziert, übersichtlich zusammengefasst und auf einer Konventionenkarte zusammengefasst, die jeder Schüler neben die Tastatur legen kann[66].

Um die Arbeit der Schüler zu vereinfachen und zu beschleunigen wird zur Formatierung des Codes ein Source-Code Formatierer eingesetzt. Zum Testen wurden bereits Vorschläge gemacht. In Anbetracht des knappen Zeitkontingents an der Schule muss das Testen so weit wie möglich automatisiert und reduziert werden, gleichzeitig muss es aber als wichtiger und für die Schüler nützlicher Bestandteil der Softwareentwicklung deutlich werden.

Für das wahrscheinliche Szenario, dass die Schüler in Kleingruppen unterschiedliche Teilfunktionalitäten für ein gemeinsames Programm entwerfen und implementieren, ist jede Gruppe für den Komponententest und den Test der jeweiligen Schnittstellen zu anderen Komponenten selbst verantwortlich. Um das zeitaufwändige Implementieren von eigenen Testklassen zu vermeiden, aber dennoch den Test zu systematisieren, soll ein Code-Coverage Tool eingesetzt werden, dies schafft anschaulich Problembewußtsein, dass bestimmte, farblich markierte Teile des Codes nicht ausgeführt wurden. Wenn die Komplexität des Anwendungsteils nicht zu groß ist, kann jetzt mit einem Debugging Werkzeug durch Änderungen von Variablenwerten ein Einsprung in diesen Codebereich erzwungen werden. Ein solches Vorgehen stärkt zugleich die Fähigkeiten in Code-Lesen und Fehlerbehebung.

Als zu Zielvorgabe für den Test wir das Kriterium (nahe) 100% Zweigüberdeckung verwendet, da es sich mit Hilfe eines Code-Coverage Tools gut prüfen lässt. Eine weitergehende Forderung von Bedingungsüberdeckung oder Pfadüberdeckung wird nicht verlangt. Die Nützlichkeit und Aussagekraft der Zweigüberdeckung kann erhöht werden, wenn eine Zerlegung („Schachtelung“) von Mehrfachbedingungen von den Schülern verlangt wird.[67]

Der auf der Interaktionsebene spezifizierte Systemtest sollte mit Hilfe eines Capture-

66 Eine solche Zusammenfassung könnte man auch gemeinsam mit den Schülern auf Basis einer Vorauswahl erstellen. Dieses Vorgehen ist aus motivationspsychologischer Sicht sinnvoll. Nach dem Origin-Pawn Konzept von DeCharms (1979) sind Schüler, die sich nicht nur als Spielball äußerer Einflüsse (Pawn) verstehen, sondern die sich im Gegensatz als Ursprung (Origin) und gestaltende Kraft einer Entwicklung oder Entscheidung erleben und empfinden, motivierter und engagierter.

67 Der Vorschlag zur Zerlegung ist Spillner (2005: 153) entnommen.

Replay Tools erstellt werden um den Systemtest auch bei Änderungen in den Anforderungen (vgl. Kapitel 4.3.1) gut nutzen und leicht anpassen zu können. Dafür ist es wichtig, dass bereits früh eine Rohfassung der Benutzungsschnittstelle bereitgestellt wird. Um den Prozess zu beschleunigen wird daher ein GUI-Builder eingesetzt. Eine Verfeinerung der Oberfläche durch die Schüler kann immer noch erfolgen, wenn die Funktionalität der Anwendung sichergestellt ist.

5. Teilapplikation Cupwertung

Bei der Entwicklung der Komponente Cupwertung werden die in Kapitel 4.3 angestellten Überlegungen zur Prozessänderung anhand eines konkreten Beispiels durchgespielt, um zu überprüfen, ob die getroffenen Entscheidungen sich in der Praxis konsistent umsetzen lassen. Gleichzeitig wird der beim ersten Prozessdurchlauf übersprungene objektorientierte Entwurfsteil genauer betrachtet und gegebenenfalls angepasst.

Die Anwendungskomponente Cupwertung, die im Folgenden entwickelt wird, ermöglicht es nach der Durchführung einer Veranstaltung deren Ergebnisdaten entgegenzunehmen und auf dieser Grundlage die Punktzahl für die Cupwertung zu errechnen. Anschließend werden die Daten mit den bereits vorhandenen Daten zusammengeführt und die Gesamtwertung des Cups neu berechnet. Die aktuelle Cupwertung wird dann auf der Internetseite des Cups veröffentlicht.

5.1 Anforderungsanalyse

5.1.1 Aufgaben und Domänenebene

Gemäß der Überlegungen aus Kapitel 4.3.5 beginnt die Analyse der Aufgabenstellung mit einer Betrachtung der Rolleninstanzen, die die Funktionen Anwendungsverwalter und Veranstaltungsleiter[68] wahrnehmen.

Rolleninstanz Veranstaltungsleiter
Der Veranstaltungsleiter Max Schuler ist für die Organisation einer Einzelveranstaltung verantwortlich. Er arbeitet als Verkäufer im Sportgeschäft „Sport & Co“. Die Veranstaltung in der Kleinstadt Raderstadt findet an einem Wochenende statt. An den Tagen vor der Veranstaltung und während dem Wettkampf ist viel Engagement und Zeitaufwand notwendig. Er ist nach dem Abbau sehr müde und will die organisatorischen Aufgaben, die sich aus der Cupteilnahme ergeben möglichst schnell abschließen.

Rolleninstanz Veranstaltungskoordinatorin
Die Veranstaltungskoordinatorin Sonja Berger hat neben dem ehrenamtlichen Engagement für den Triathloncup einen Anstrengenden Job als Abteilungsleiterin der Versicherungsfirma „Securitaria“. Sie hat wenig Freizeit und ist zudem auch noch in die Organisation einer Einzelveranstaltung in Heidelberg eingebunden. Sie ist dafür zuständig die Einstellungen des Systems zu den teilnehmenden Veranstaltungen sowie

68 Veranstaltungsleiter bezeichnet hierbei den Leiter einer Einzelveranstaltung, wie sie im Glossar beschrieben ist.

zum Wertungssystem vorzunehmen und gegebenenfalls Änderungen an den vorhandenen Daten vorzunehmen, wenn Fehler gemeldet werden.
Sie will die Einstellungen einmalig im Frühjahr vornehmen und danach möglichst keinen zusätzlichen administrativen Aufwand haben.

Soll Prozess Datenänderung
Sonja Berger hat die Möglichkeit jeden Datensatz zu ändern.

Soll Prozess Veranstaltungskonfiguration
Bei der Konfiguration legt Sonja Berger zu Jahresbeginn 2010 die Teilnehmenden Einzelveranstaltungen Raderstadt und Heidelberg sowie die Wettkampfkategorien, die von den einzelnen Veranstaltungen angeboten werden im System fest. Heidelberg bietet die Kategorien „Erwachsene“ und „Jugend A“ in der Cupwertung an, während Raderstadt die Kategorien „Jugend A“, „Schüler A“ anbietet. Die Wertungsregel wird für jeden Wettkampftyp wie folgt festgelegt:
In die Gesamtwertung kommt, wer mindestens 1 Rennen bestritten hat. Liegen 2 oder mehr Wertungen vor, werden die schlechtesten Wertungen gestrichen.
Zusätzlich zu den Veranstaltungsspezifischen Einstellungen werden in einer zweiten Datei die Konfigurationseinstellungen der die die technischen Rahmenbedingungen konfiguriert.

Soll Prozess Wertungsberechnung
Max Schuler öffnet die Anwendung und wählt zunächst seine Veranstaltung „Raderstadt“ sowie den Wettkampftyp „Jugend A“ aus.
Anschließend lädt er seine Ergebnisdatei „2010_Ergebnisse_Raderstadt_Jugend-A.csv“ in das Programm und wählt die von ihm benötigten Datenfelder und deren Reihenfolge im Verhältnis zum Datenformat, das im System vorliegt.
Das System zeigt dann die geladenen Daten an und berechnet gleichzeitig die Wertungspunkte für die geladenen Daten. Als Basis für die Punktvergabe dienen die jeweiligen Siegerzeiten (Frauen und Männer) der Einzelveranstaltungen. Die Siegerzeiten werden jeweils durch die Zeit des zu berechnenden Cup-Teilnehmers dividiert und mit 1000 multipliziert.
Im Anschluss lädt das System die Daten der vorangegangenen Einzelveranstaltung in Heidelberg und fügt die Daten zusammen. Dabei verschmilzt das System zunächst die eindeutig zuordenbaren Datensätze und addiert die Wertungspunkte nach den Wertungsregeln, die von Sonja Berger festgelegt wurden.
In einem zweiten Schritt prüft das System, ob es Datensätze gibt, die eine hohe Ähnlichkeit zu den bereits vorhandenen aufweisen. Ist dies der Fall, schlägt das System vor, die Datensätze zu kombinieren. Herr Schuler kann dies jeweils akzeptieren oder ablehnen. Wenn er ablehnt, werden die Datensätze als Daten von unterschiedlichen Startern behandelt, ansonsten werden sie als ein Starter behandelt und bei der Berechnung der Wertungspunkte kombiniert.
Alle weiteren Datensätze von Startern, die noch nicht im System erfasst sind, werden ebenfalls in die Berechnung der Gesamtpunktzahl einbezogen. Dabei wird angenommen, dass sie bei der vorhergehenden Veranstaltung in Heidelberg nicht teilgenommen haben (denn dann wären sie bereits verschmolzen oder es gäbe einen Kombinationsvorschlag für sie). Im Anschluss wird die errechnete Wertungstabelle angezeigt.

Nicht-funktionale Anforderungen
Die Anwendung soll dazu beitragen den organisatorischen Zeitaufwand zu reduzie-

ren (Prozesseffizienz). Alle Tätigkeiten in der Komponente Cupwertung werden von einzelnen Expertennutzern durchgeführt, die gut eingearbeitet werden können. Die Nutzung des Internets bei der Registrierung erzeugt die Notwendigkeit von Sicherungsmaßnahmen der Eingaben bei Übertragung (Sicherheit). Aus Datensicht ist es zur Reduzierung von manueller Nacharbeit und Fehlervermeidung sinnvoll, die Datensätze nach der Eingabe zu validieren und gegebenenfalls aufzubereiten.
Wie bei der Beschreibung der Rolleninstanzen bereits angedeutet, sind die Organisatoren stark beansprucht, daher sollen verschiedene Flüchtigkeitsfehler vermieden werden, gleichzeitig sollen weniger Fehler bei der Ermittlung der Gesamtwertung auftreten als bei dem bisher verwendeten Prozess.

- Organisatorischer Zeitaufwand (soll mindestens 30% schneller sein als der bisherige Prozess)
- Sicherheit (Passwörter nicht im Klartext übertragen)
- Validierung/Anpassung der Daten (Ähnliche Datensätze werden vorgeschlagen)
- Fehlertoleranz (mindestens 50% weniger Fehler in Gesamtwertung als im Jahr 2009)

Domänendaten

In unserem Fall ergeben sich die Domänendaten aus dem bisherigen Prozess. Die von den Einzelveranstaltungen weitergeleiteten Daten sollen nicht geändert werden. Daher stehen für die Zusammenführung der Daten die Felder Altersklasse (Wettkampftyp), Name (das Datenfeld besteht aus Name und Vorname) und Geschlecht, sowie optional „Verein“ zur Verfügung. Von jeder Veranstaltung werden die Zeit und die vom Programm zu errechnende Punktzahl weitergegeben. Außerdem existiert ein Feld für die Summe der Punkte und ein Feld für die Platzierung in der Gesamtwertung. Zusätzlich wird die Anzahl der Starts als eigenes Feld aufgenommen. Der Wettkampftyp ergibt sich aus den zu verarbeitenden Daten, da für jeden Wettkampftyp eine eigene Ergebnisdatei als Eingabe verwendet wird. Die Kontaktdaten werden nicht zentral gespeichert sondern verbleiben bei den Einzelveranstaltungen. Aus diesen Anforderungen ergibt sich das folgende Domänendatendiagramm.

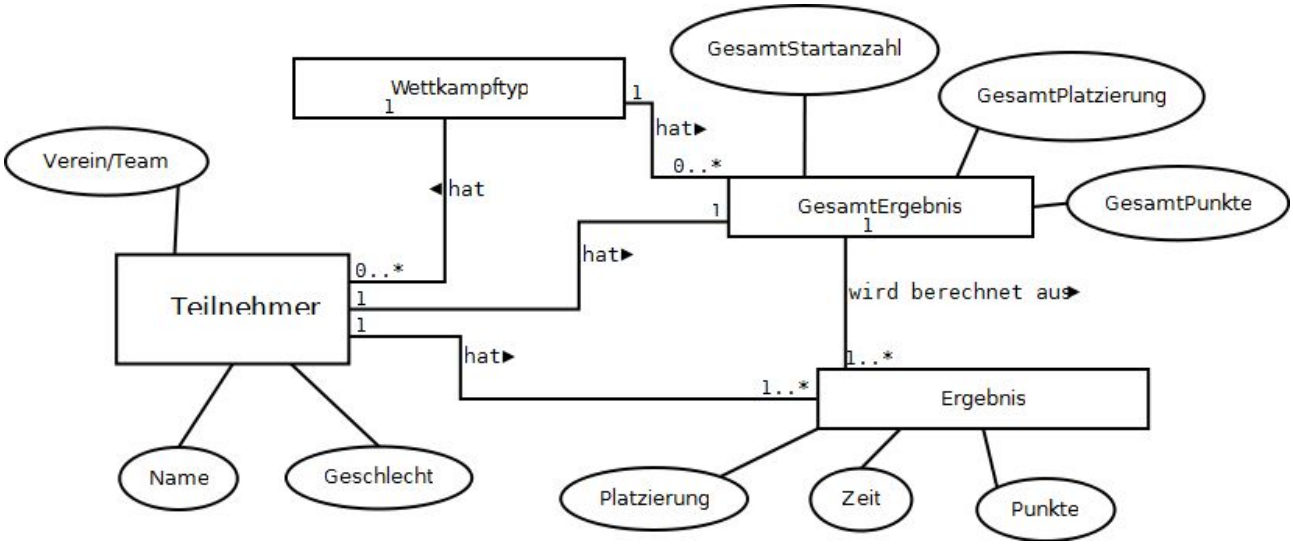


Abbildung 27: Domänendaten „Wertungsberechnung“

Systemtestplan

Testendekriterien und Grundlage werden wie in Kapitel 3.1.2 festgelegt, bei den Systemtestressourcen wird jedoch folgende Änderung vogenommen:

- Da die Genaue realisierung noch nicht fest steht, ist zu diesem Zeitpunkt noch unklar, welches Programm zur Unterstützung eingesetzt werden soll. Das Progamm soll jedoch möglichst Komplexität reduzieren und ähnlich wie Selenium automatisiert einsetzbar sein und über eine Aufnahmefunktion verfügen.

Im Schuleinsatz würden entsprechende Rahmenbedingungen für den Systemtest von der Lehrkraft vorgegeben werden.

Inspektion und Rationale

Die Inspektion erfolgt erneut auf Basis der Checkliste „Bereich Rollen, Aufgaben, NFR's, Rolleninstanzen“, die geänderte Vorgehensweise macht jedoch eine Modifikation der Checkliste notwendig, die zu den folgenden Fragen führt:

- Sind alle Anforderungen umgesetzt?
- Sind die vorhandenen Rolleninstanzen gefordert und konsistent zur Problembeschreibung?
- Sind die Texte verständlich?
- Sind alle Fachbegriffe klar
- Sind die NFR messbar?
- Wirken sich NFR aus?
- Leisten die Bestandteile einen wichtigen, neuen, einmaligen Beitrag zur Spezifikation?
- Ist jede Rolleninstanz in mindestens einen Prozess eingebunden?

Rationale ergeben sich auf dieser ersten Ebene nur zum Sollprozess Veranstaltungskonfiguration. Der Prozess ist von niedriger Komplexität und wurde nur aufgenommen um eine Vollständigkeit der Anforderungen zu gewährleisten. Auf den anderen Analyseebenen wird er daher nicht weiter detailliert. Die im Prozess geforderte Konfiguration kann einfach über eine Datei vorgenommen werden. Die Punktzahlberechnung ändert sich in der Praxis nicht, daher wird diese im Programm fest hinterlegt. Die Teilnehmenden Veranstaltungen und die Kategorien, die diese anbieten, müssen jedoch konfigurierbar sein.
Auch die Anzahl der Starts, die für eine Cupteilnahme mindestens erforderlich sind, muss festgelegt werden.
Der Prozess Datenänderung wird, vergleichbar zum ersten Prozessdurchlauf, zurückgestellt, da sich je nach gewählter Realisierungsvariante eine aufwandsarme Möglichkeit zur Änderung der einzelnen Datensätze ergibt. Aus Gründen der Vollständigkeit ist er jedoch aufgeführt.

5.1.2 Interaktionsebene

UI-Struktur

Es werden die in Abbildung 28 dargestellten Sichten entworfen und in den Arbeitsbereich Wertungsberechnung integriert.

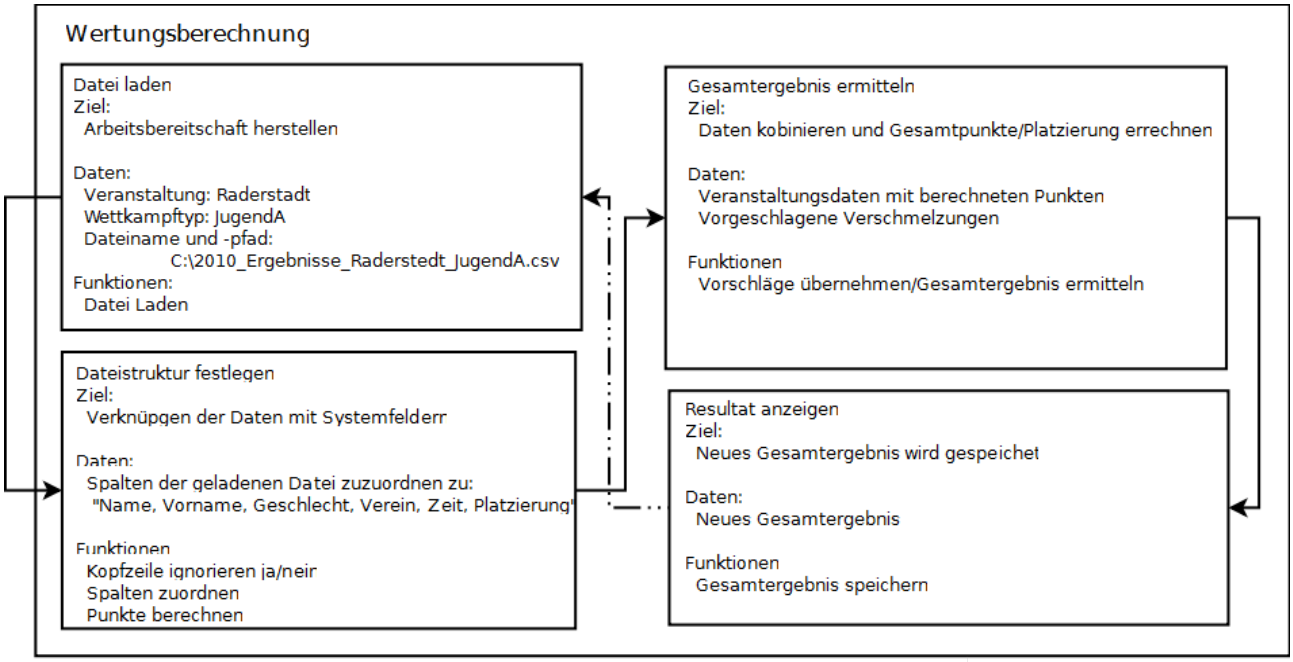


Abbildung 28: UI-Struktur

Im schulischen Einsatz erscheint eine formlose Darstellung der Sichten mittels A4 Papier an einer Wand am sinnvollsten (Papier Mock-Up). Dann können auch bereits erste Überlegungen zu konkreten Daten angestellt werden, die in den jeweiligen SIchten angezeigt werden.

Interaktionsbeschreibung

Eine unabhängige Interaktionsbeschreibung erfolgt nicht, statt dessen erfolgt eine Beschreibung im Artefakt „Szenario und Systemfunktionen“.

Verfeinertes Datenmodell

Eine Verfeinerung des Datenmodells auf diese Ebene wie bei Häfele beschrieben (vgl Häfele 2005: 59) ergibt in unserem Fall keinen erkennbaren Mehrwert und entfällt daher.

Systemfunktionen

Die Systemfunktionen werden direkt in dem Artefakt „Szenario und Systemfunktionen“ (vgl. Abbildung 29) aufgenommen, eine separate Erfassung erfolgt nur, falls mehrere komplexe Systemfunktionen notwendig sind und somit einen erkennbarer Mehrwert entsteht.

Szenarien

Das Szenario wird Schritt für Schritt entwickelt und direkt im Artefakt „Szenario und Systemfunktionen“ erfasst. Eine zusätzliche textuale Erfassung erfolgt nicht.

Nicht-funktionale Anforderungen

Auf der Interaktionsebene kommen keine neuen Kriterien hinzu, die bereits vorhan-

denen werden jedoch aufgegriffen und als Qualitätskriterien in das Artefakt „Szenario und Systemfunktionen“ übernommen. Dabei wird der betroffene Schritt durch eine Zahl und der Bereich durch einen Buchstaben (I, O, S, E) kenntlich gemacht um die Spalte Input, Output, System oder Exception zu bezeichnen.

Systemtestspezifikation

Gemäß des Systemtestplans gilt das System durch das folgende Vorgehen als getestet, wenn alle Ausnahmen ebenfalls getestet werden.

Daher wird im Artefakt „Szenario und Systemfunktionen“ zunächst ein regulärer Prozessdurchlauf erfasst (Schritt 1-4). Um den Systemtest spezifizieren zu können wird im Anschluss ein Durchlauf definiert, in dem die Ausnahmen auftreten (Schritt 5-9).

Die Ausnahmen „I/O Fehlers“, „DB Fehlers“ und „Config File nicht gefunden“ müssen separat getestet werden, da dies das Dokument zu sehr überladen würde.

Szenario und Systemfunktionen				
Name	Szenario „Schulers Wertungsberechnung“			
Akteur	Veranstaltungsleiter Max Schuler			
Ziel (Goal)	Aktualisierung der Gesamtwertung/Systemtest			
Verbindung zu Prozess	Prozess Veranstaltungskonfiguration (siehe Vorbedingungen)			
Vorbedingung (Preconditions)	-Die Datei „2010_Ergebnisse_Raderstadt_JugendA.csv“ ist im Verzeichnis „C:\“ -Die Ergebnisse der Veranstaltung Heidelberg (JugendA) stehen zur Verfügen -Die Veranstaltung wurde gemäß des Soll-Prozesses „Veranstaltungskonfiguration“ konfiguriert			
Beschreibung (Flow Of Events)				
	Input	Expected **O**utput	**S**ystem	**E**xceptions
0)	Programmstart	Sicht „Datei Laden“ wird angezeigt, die konfigurierten Veranstaltungen stehen zur Auswahl	- Lädt Konfiguration	- I/O Fehler => Test fehlgeschlagen, kein Fortfahren möglich
1)	Veranstaltung: „Raderstadt“ Wettkampftyp: „JugendA“ Datei: „C:\2010_Ergebnisse_Raderstadt_JugendA.csv“ Benutzername und Passwort für Datenbank gefüllt „Datei Laden“ geklickt	Sicht „Dateistruktur festlegen“ wird angezeigt	- Lädt Datei - Lädt Daten der Vorgängerveranstaltungen	- Pflichtfelder nicht ausgefüllt Hinweis anzeigen, Korrekturmöglichkeit - I/O Fehler => Test fehlgeschlagen, kein Fortfahren möglich - DB-Fehler => Test fehlgeschlagen, kein Fortfahren möglich
2)	Spalten „Name, Vorname, Geschlecht, Verein, Zeit, Platzierung“ zugeordnet „Kopfzeile ignorieren“ ausgewählt, „Punkte berechnen“ geklickt	Sicht „Gesamtergebnis ermitteln“ wird angezeigt	- Überführt Datenstruktur in interne Repräsentation - Berechnet Punkte	- Notwendige Felder nicht ausgewählt Hinweis anzeigen, Korrekturmöglichkeit - Berechnungsfehler => Test fehlgeschlagen - Fehler bei Felderzuordnung => Test fehlgeschlagen - Config Datei nicht gefunden => Test fehlgeschlagen
3)	1. und 3. Vorschlag ausgewählt „Vorschläge übernehmen/Gesamtergebnis ermitteln“	Sicht „Resultat anzeigen“ wird angezeigt 2. Vorschlag führt zu zwei Einträgen in	- Kombiniert Starterdaten - Macht Kombinationsvorschläge bei	-Anzahl der Einträge stimmt nicht => Test fehlgeschlagen

	geklickt	Sicht „Resultat anzeigen“. 1. und 3. Vorschlag führen zu einem Eintrag in Sicht „Resultat anzeigen“	ähnlichen Daten und kombiniert Daten falls angenommen	-Summen nicht korrekt => Test fehlgeschlagen
4)	„Gesamtergebnis speichern“ geklickt	Bestätigung „Gesamtergebnis JugendA aktualisiert“ wird angezeigt, Sicht „Datei laden“ wird angezeigt, die zuvor ausgewählten Einstellungen der Sicht „Datei Laden“ zur Veranstaltung und Datei sind vorgefüllt	- Schreibt neues Gesamtergebnis	DB Fehler => Test fehlgeschlagen
Zusatzschritte für den Systemtest				
5)	„Wettkampftyp“nicht gefüllt „Datei laden“ geklickt	Fehlermeldung: Feld muss gefüllt werden		
6)	„Veranstaltung“ nicht gefüllt „Datei laden“ geklickt	Fehlermeldung: Feld muss gefüllt werden		
7)	Veranstaltung: „Raderstadt“ Wettkampftyp: „SchülerA“ Datei: „C:\2010_Ergebnisse_Raderstadt_SchuelerA.csv“ Benutzername und Passwort für Datenbank **nicht** gefüllt. „Datei Laden“ geklickt	Fehlermeldung: Benutzername/Passwort nicht korrekt	[wie oben] - Lädt Datei	[wie oben]
7)	Veranstaltung: „Raderstadt“ Wettkampftyp: „SchülerA“ Datei: „C:\2010_Ergebnisse_Raderstadt_SchuelerA.csv“ Benutzername und Passwort für Datenbank gefüllt. „Datei Laden“ geklickt	Sicht „Dateistruktur festlegen“ wird angezeigt	[wie oben] - Lädt Datei - Lädt Daten der Vorgängerveranstaltungen	[wie oben]
8)	„Name“ nicht zugeordnet „Kopfzeile ignorieren“ nicht ausgewählt „Punkte berechnen“ geklickt [Schritt wiederholen für: Geschlecht, Verein, Zeit, Platzierung nicht zugeordnet]	Fehler „Feld nicht zugeordnet“ [Analog für Spalten „Geschlecht, Verein, Zeit, Platzierung“]	[wie oben] - Überführt Datenstruktur in interne Repräsentation - Berechnet Punkte	[wie oben]
9)	Schritt 1-4 erneut durchlaufen mit Datei „C:\2010_Ergebnisse_Neu_Raderstadt_JugendA.csv“	Benutzer sieht kein geändertes Verhalten des Systems	- System überschreibt zuerst geladenen Datensatz	

Nachbedingungen (Postconditions)	Sicht „Datei laden“ wird angezeigt, die zuvor ausgewählten Einstellungen der Sicht „Datei Laden“ sind vorgefüllt.
Testressourcen (Test Resources)	tbd
Qualitätsanforderung (Quality Constrains):	1-I) Fehlertoleranz: Felder müssen gefüllt werden 1-I) Prozesseffizienz: Datei kann im Dateisystem ausgewählt werden 2-I) Fehlertoleranz: Felder müssen gefüllt werden 2-I) Flexibiltät: Auswahl ermöglicht Handhabung von Dateien, die sich im Aufbau unterscheiden 2-O) Prozesseffizienz/Fehlertoleranz: Um lange Abfragefolgen und Einzelbestätigungen zu vermeiden werden alle Vorschläge untereinander angezeigt 3-I) Fehlertoleranz: Alle vorliegenden Datenfelder sollen angezeigt werden, damit Benutzer eine fundierte Entscheidung treffen kann 3-I) Fehlertoleranz: Vorschläge können ignoriert werden 4-O) Fehlertoleranz: Eine Vorfüllung von Wettkampftyp erfolgt nicht

Abbildung 29: Szenario und Systemfunktionen

Usabilitytestspezifikation und -plan

Benutzbarkeit spielt für die Komponente Cupwertung nur eine untergeordnete Rolle, da die Zielgruppe aus einem kleinen Kreis von Expertennutzern besteht, die eingearbeitet und unterstützt werden können. Daher erfolgt keine weitere formale Überprüfung der Benutzbarkeit.

Die entwickelte Anwendung wird jedoch als Prototyp den Leitern der Einzelveranstaltungen zur Verfügung gestellt werden und entsprechendes Feedback, insbesondere hinsichtlich der flexiblen Verwendung von Ergebnisdateien und der allgemeinen Benutzbarkeit, werden in einer späteren Version eingearbeitet werden.

Inspektion und Rationale

Auf der Interaktionsebene werden im Wesentlichen mehrere Entscheidungen getroffen. Zum Einen die Vorschlagsanzeige „im Block“ in der nicht gewünschte Vorschläge abgewählt werden können. Ob dieses Vorgehen fehlertoleranter ist als ein schnelles abnicken auf verschiedenen Bildschirmen kann nicht pauschal entschieden werden. In jedem Fall ist es effizienter (vgl. Abbildung 30). Die zweite Entscheidung betrifft die Navigation. Um eine aufwändige Funktion zur Rückgängigmachung von Schritten zu vermeiden, wurde entschieden, lediglich eine „abbrechen“ Funktion zu implementieren, die zur ersten Sicht zurückspringt und die gemachten Änderungen komplett zurücksetzt (vgl. Abbildung 31). Dennoch sollte beim Entwurf berücksichtigt werden, dass zumindest die Veranstaltungskoordinatorin die Möglichkeit haben sollte, mit vertretbarem Aufwand zu einem älteren Datenstand zurückkehren zu können.

Optionen	Kriterien		
	Fehlertoleranz	Prozesseffizienz	Summe
Vorschläge sequentiell bearbeiten	+/-	--	-2
Vorschläge in Übersicht bearbeiten	**+/-**	**++**	**2**

Abbildung 30: Rationaletabelle zur Sicht „Gesamtergebnis ermitteln“

Optionen	Kriterien		
	Kosten	Prozesseffizienz	Summe
Schrittweises Undo	-	+	0
Abbruch	**+**	**+/-**	**1**

Abbildung 31: Rationaletabelle zu Undo Funktion

Optionen	Kriterien			
	Flexibilität	Sicherheit	Prozesseffizienz	Summe
Serverseitige Eingabe/Ausgabe	+/-	+	+/-	1
Hybrid: Ausgabe Server, Eingabe Client	**+**	**+**	**+**	**3**

Abbildung 32: Rationaletabelle zur grundlegenden Struktur

Eine weitere Entscheidung ist zur Übernahme und Veröffentlichung der Daten notwendig. Die Daten könnten als CSV Datei von einem Server übernommen werden und im Anschluss wieder auf dem Server abgelegt werden. Alternativ ist der Einsatz einer Datenbank denkbar. Dabei stellt sich auch die Frage, wie die Aufgabenstellung grundsätzlich gelöst werden soll. Durch den Einsatz von serverseitiger Logik, oder auf der Clientseite oder in einem hybriden Ansatz

Beim Einsatz von Clientseitiger Logik ergibt sich das Problem, dass kein direktes Schreiben auf dem Server möglich ist, sondern der Umweg über FTP gegangen werden muss. Die Weitergabe eines FTP Zugangs ist aus Sicherheitsgründen nur bedingt sinnvoll[69]. Eine Datenbank bietet hingegen problemloses Rechtemanagement auf Tabellenebene. Auch ein Java-Applet verfügt nicht über die Möglichkeit direkt auf dem Quellserver Daten abzulegen. Daher kommt beim Einsatz von Clientlogik nur die Kombination mit einer Datenbank in Frage, wenn eine zusätzlicher Übermittlungsschritt über E-Mail oder FTP vermieden werden soll[70].

Wenn die Daten in einer Datenbank gespeichert werden, wird ein Ausgabemodul benötigt, das die Daten entsprechend aufbereiten und anzeigen kann, sinnvollerweise ist diese Komponente serverseitig, da das Verteilen eines Programms an alle Cupteilnehmer unrealistisch und unkomfortabel ist[71].

Somit besteht die Wahl entweder Eingabe und Ausgabe serverseitig zu realisieren, oder einen Client für die Veranstaltungsleiter zu entwickeln, der die Daten in einer Datenbank speichert und die Daten der Vorgängerveranstaltungen von dort liest. Das Auslesen erfolgt in diesem Fall über ein serverseitiges Skript, das HTML Seiten generieren kann.

In unserem Beispiel soll die zweite Lösung gewählt werden, da die Realisierung in ei-

69 Da keine Kontrolle der Inhalte möglich ist und das Rechtemanagement komplex ist.

70 Die Kombination eines Clients mit serverseitiger Logik, die die Daten entgegen nimmt und auf dem Server als Datei speichert, scheidet aus, da diese Lösung reichlich unelegant ist. Auch ist es recht aufwendig sicherzustellen, dass die Daten komfortabel geändert und bereits geschriebene Änderungen rückgängig gemacht werden können.

71 Der Einsatz eines Java-Applets scheidet bei der Realisierung aus, da es keine Berechtigung zum Zugriff auf andere Domains als der ausliefernden hat, was zu Problemen bei der Kombination mit einem Datenbankserver führt. Der Einsatz von Flash zur Auslieferung ist aus Benutzbarkeitsgesichtspunkten ebenfalls keine sehr elegante Lösung.

nem Client einfacher ist[72] und problemlos angepasst werden kann, während die Ausgabe von den Stärken der ersten Lösung profitiert (vgl. Abbildung 32). Dies ermöglicht auch eine einfache Verwaltung der geladenen Daten über eine Administrationsoberfläche, vergleichbar zur Komponente Teilnehmerregistrierung.

Da bei einer Clientseitigen Lösung das Speichern des Passworts im Quelltext unsicher ist, wird dem Datebankserver die Prüfung überlassen und es lediglich weitergereicht. Hierbei ist darauf zu achten, dass die Übertragung des Passworts keinesfalls im Klartext erfolgt.

Die Inspektion der Interaktionsebene erfolgt erneut mittels der Checkliste „Bereich Use Cases“. Auch hier ist eine Anpassung notwendig:

- Eindeutigkeit: Sind die beschreibenden Texte einfach, eindeutig und verständlich?
- Vollständigkeit: Ist der Soll-Prozess komplett umgesetzt?
- Korrektheit: Sind alle Schritte der Umsetzung durch den Soll Prozess gefordert?
- Korrektheit: Falls Schnittstellen zu anderen Prozessen bestehen, sind diese klar festgelegt?
- Konsistenz: Haben die Fachbegriffe im Prozess und Szenario/UI-Struktur die gleiche Bedeutung
- Eindeutigkeit: Sind alle Fachbegriffe im Glossar?
- Vollständigkeit: Gibt es zu jeder Ausnahme eine Verhaltensbeschreibung falls diese eintritt?

72 Insbesondere müssen keine Besonderheiten der Browser bei Caching und Navigation berücksichtigt werden.

5.1.3 Systemebene

Gui-Entwurf

Die bereits eingeführte „Abbrechen“ Funktion stellt eine Hilfsfunktion dar. Die Meldungen sollen als Pop-Up Fenster erscheinen. Eine weitere Ausdefinierung erfolgt auf Basis der als Prototyp fertiggestellen Komponente Cupwertung und dem eingeholten Nutzerfeedback nach Abschluss der Arbeit.

Systemkern

Im Gegensatz zum ersten Prozessdurchlauf kann hier eine weitere Verfeinerung erfolgen. Im TRAIN Prozess orientiert sich das Vorgehen an der Methode von Jacobson (vgl. Jacobson 1992).

Entitätsklassen:

1) Ergebnis
2) Gesamtergebnis
3) Teilnehmer
4) Wettkampf
5) Konfiguration

Auf eine getrennte Modellierung von „Teilnehmer“ wird verzichtet. Um Komplexität zu reduzieren werden die Attribute von Teilnehmer im Domänendatendiagramm auf Ergebnis und Gesamtergebnis übertragen. Die Entscheidung wird durch ein vorläufiges Analyseklassendiagramm erhärtet. Dort wird besonders deutlich, dass die Klasse Teilnehmer aus Köhäsionsgründen nicht sinnvoll ist.

Auch wenn es zunächst so aussieht, als könnte man Gesamtergebnis als Komposition der Ergebnisse modellieren, so ist dies nicht sinnvoll, da ein Ergebnis auch unabhängig vom Gesamtergebnis existieren kann, etwa wenn nicht genug Starts vorliegen um nach der Wertungsregel in die Gesamtwertung zu kommen. Allerdings spricht nichts gegen die Einführung einer Aggregation. Im weiteren Analyseverlauf ist jedoch zu beachten, dass bei Aggregationen die aggergierende Klasse die Operationen stellvertretend für ihre Einzelteile entgegennehmen soll[73].

Zwischen Wettkampftyp und Gesamtergebnis besteht ebenfalls eine Aggregation, da dieser eine Liste der lokalen Ergebnisse und eine Liste der geladenen Gesamtergebnisse vorhält.

Als zusätzliche Entitätsklasse kommt noch die Klasse Konfiguration hinzu, die sich aus dem Prozess Veranstaltungskonfiguration ergibt, und deren Aufgabe es ist, die aus der Konfigurationsdatei geladene Einstellungen zu speichern.

Systemfunktionen:

1) DatenSchreiben
2) DatenLaden

73 (vgl. Oestereich 2004: 59), dort finden sich auch weitere Details zu Modellierungsfragestellungen.

3) Modellgenerierung (Interne Repräsentation erstellen)

4) Punkteberechnung (Beinhaltet Vorschläge auf Basis der Ähnlichkeit)

1-3 können zur Klasse „Datenmangement“ zusammengefasst werden

Dialogklassen:

Nach Häfele soll eine Übernahme aus den Sichten erfolgen (vgl. Häfele 2005: 72). Im weiteren Verlauf übernimmt er in seinem Beispiel jedoch nur eine einzelne Sicht als Klasse. In unserem Beispiel erscheint eine Unterteilung in mehrere Dialogklassen ebenfalls nicht sinnvoll.

Daher wird lediglich die Klasse WertungsberechnungSicht festgelegt. Die im Domänendatendiagramm eingeführte Bezeichnung Wettkampftyp erscheint unklar, daher wird die Klasse in Wettkampf umbenannt.

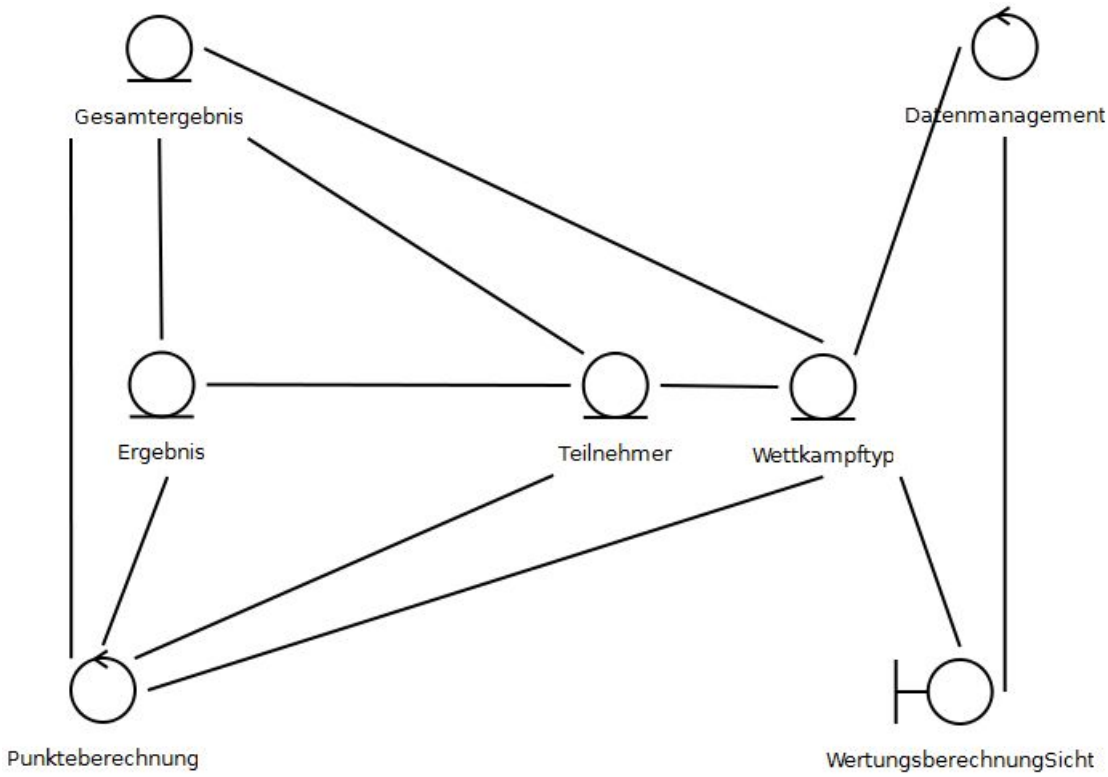


Abbildung 33: Analyseklassendiagramm, erster Entwurf

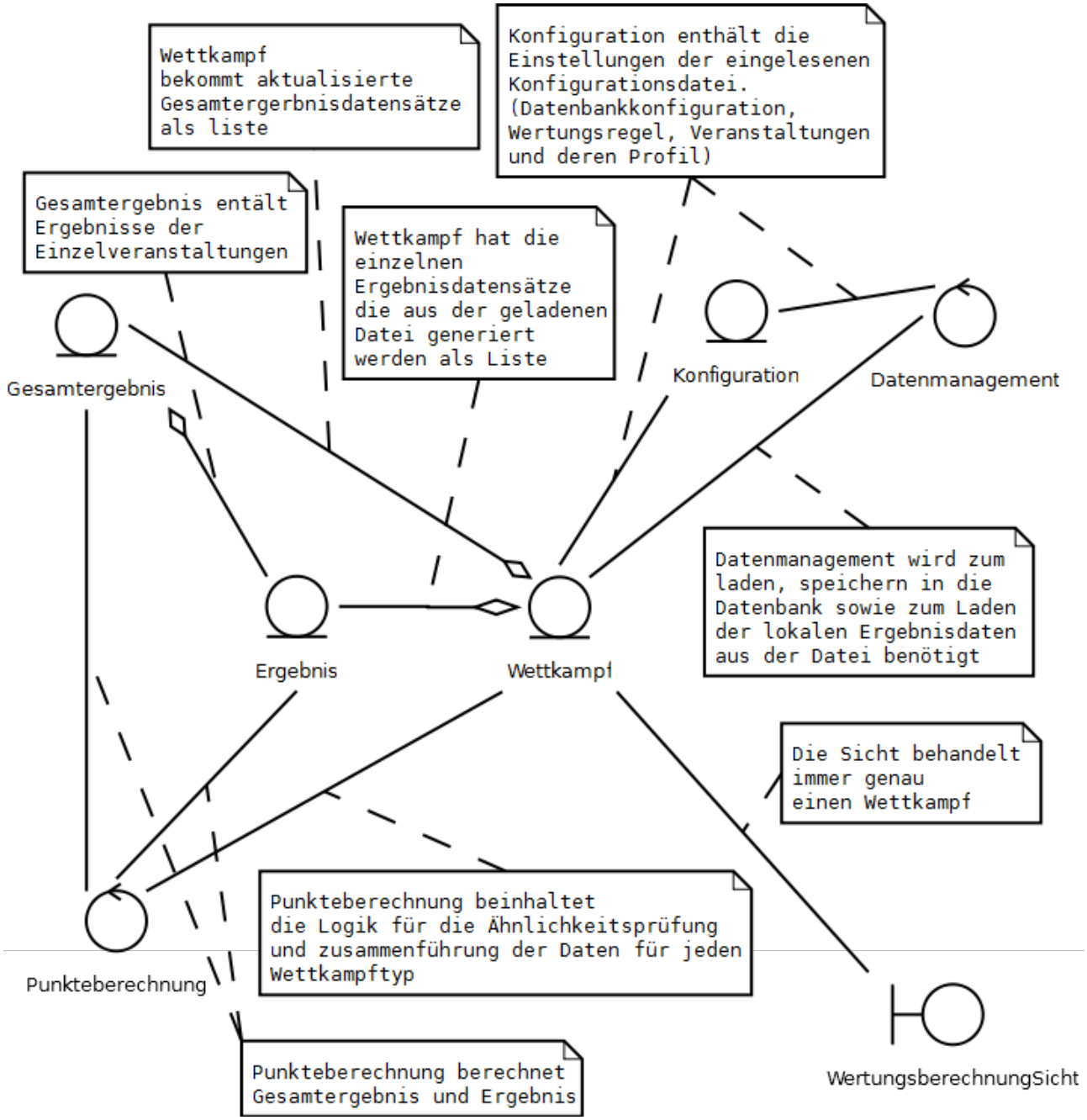


Abbildung 34: Analyseklassendiagramm

Eine Zuordnung der Steuerungsklassen zu anderen Klassen erscheint für Datenmanagement nicht sinnvoll. In dieser Klasse sollen weiterhin die entsprechenden Schnittstellenfunktionen für Ein- und Ausgabe bereitgestellt werden. Für die Punkteberechnung wird hingegen eine Zuordnung zu Ergebnis und GesamtErgebnis vorgenommen. Die Logik zur Ähnlichkeitsprüfung ist am Besten in der Klasse Wettkampf verortet, da nur diese Klasse den Überblick über alle Ergebnis- und GesamtErgebnis-Datensätze hat. Aus dem gleichen Grund wird die Aufgabe eine interne Struktur zu generieren in die Klasse Wettkampf integriert.

WertungsberechnungSicht soll möglichst keine Funktionalität, die über Bereitstellung einer Benutzungsschnittstelle hinausgeht, beinhalten. Daher ist auch keine direkte Verbindung zu Datenmanagement sinnvoll.

Obwohl das entwickelte Analyseklassendiagramm einen guten Eindruck macht, muss nach Studium der Realisierungsmöglichkeiten in der Programmiersprache Java ein anderer Weg beschritten werden, aus diesem Grund macht eine unmodifizierte Übernahme des Entwurfs keinen Sinn.

Zum Anzeigen der Datentabelle wird JTable verwendet werden. JTable realisiert bereits eine Trennung in Datenmodell (TableModel) und Darstellung (JTable). Gleichzeitig werden die Gesamtergebnis Tabellen immer mit allen Spalten, so wie sie aus der Datenbank gelesen werden, ausgegeben und ein Ergebnis ist unter Auslassung von mehreren Feldern in der Tabelle darstellbar, in der auch die Gesamtergebnisse dargestellt werden. Es würde zu einem unverhältnismäßig hohen Aufwand und einer hohen Kopplung führen, immer die Objekte auszulesen, in das TableModel zu speichern, zu manipulieren und wieder das gesamte TableModel neu aufzubauen. Aus diesem Grund werden die Ergebnisse und Gesamtergebnisse nicht in Listenstrukturen, sondern direkt als Datenzeilen in ihrem jeweiligen TableModel in der Klasse Wettkampf bereitgehalten (vgl. Abbildung 35). Die Punkteberechnung erfolgt auf diesen Datenzeilen, daher wird die Funktionalität als Methode in die Klasse Wettkampf aufgenommen.

Optionen	Kriterien				
	Kosten	Kopplung	Einfachheit	Paradigma	Summe
Eigene Klassen	-	-	-	+	1
Integration in Wettkampf und Abbildung in TableModel	**+**	**+**	**+**	**-**	**3**

Abbildung 35: Rationaletabelle zur grundlegenden Struktur

Im Beispiel von Häfele folgt an dieser Stelle eine Analyse der Kommunikation zwischen den Klassen um zu Prüfen ob die getroffenen Entscheidungen bezüglich Kopplung und Kohäsion[74] zu überprüfen. Dieser Schritt erscheint mir an dieser Stelle aus mehreren Gründen nicht sinnvoll: Es ist schwierig, den sequentiellen Ablauf zu entscheiden, bevor mit einem Klassendiagramm grob Methoden und Attribute zugeordnet sind. Ein Klassendiagramm bietet hierfür einen viel besseren Überblick über den Zusammenhang.

Eine Verfeinerung des Klassendiagramms auf Basis eines anschließend erstellten Sequenzdiagramms erscheint daher intuitiver.

Um Mehrfacharbeit zu vermeiden, wurde ein Werkzeug gesucht, das möglichst bereits den Rahmen für die Codegenerierung unterstützt; das Eclipse Plugin „Green“[75].

74 Kopplung beschreibt die Abhängigkeiten zwischen Klassen, die zu viel Interaktion zwischen den Klassen führt und die möglichst reduziert werden sollte. Kohäsion (Kohäsion wird manchmal auch synonym mit Kohärenz verwendet) beschreibt den inneren Zusammenhalt einer Klasse. Die Kohäsion einer Klasse ist gering, wenn sie aus mehreren logischen Teilen besteht, die gegebenenfalls auf mehrere Klassen aufgeteilt werden können. Die Kohäsion sollte daher beim Entwurf möglichst erhöht werden.
Kopplung und Kohäsion werden jedoch in der Literatur nicht immer gleichbedeutend verwendet, wenngleich das selbe Prinzip beschrieben wird. Während im TRAIN Prozess sowie bei Oestereich und Zuser et al. (vgl. Oestereich 2004: 167 und Zuser et al. 2004: 274ff) die Begriffe für Klassen und deren Beziehungen genutzt werden, beschreiben Dutoit und Brügge damit Abhängigkeiten auf der Ebene, die logisch oberhalb der Klassen liegt. Kopplung beschreibt dort Abhängigkeiten zwischen Subsystemen/Paketen, Köhäsion Abhängigkeiten zwischen Klassen (vgl. Dutoit/Brügge 2004: 262).

75 Green (green.sourceforge.net) wurde explizit für Ausbildungszwecke und wird weiter aktiv gepflegt.

Bei diesem „Round-Trip“ Editor wirken sich Änderungen am Klassendiagramm direkt auf den Programmcode aus und umgekehrt. Dies kann den Schülern in besonderer Weise die Nützlichkeit von Modellierung für den späteren Programmiererfolg verdeutlichen.

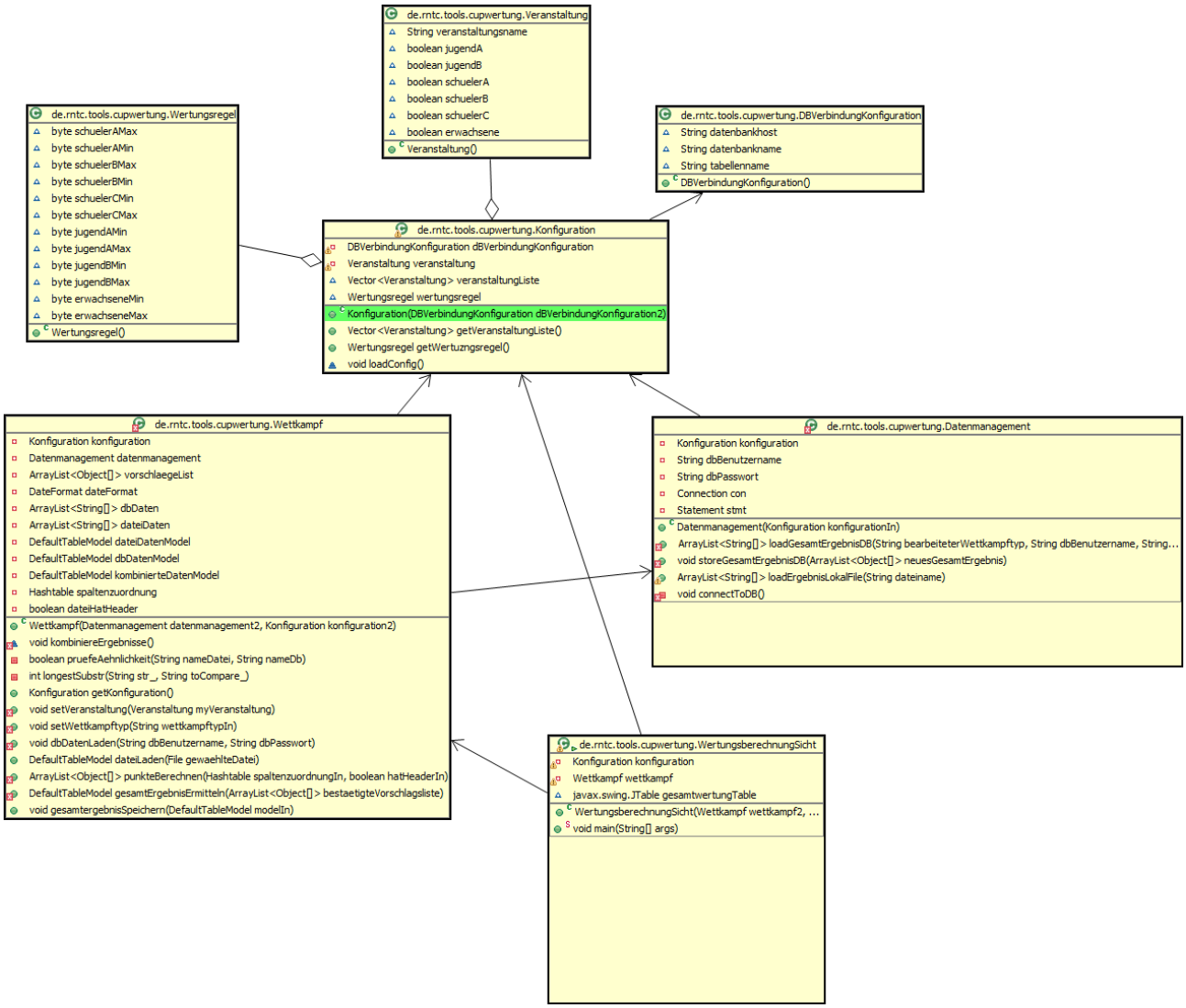


Abbildung 36: Klassendiagramm nach Verfeinerung

Abbildung 36 zeigt bereits das Ergebnis dieser Verfeinerung. Die Elemente der Konfiguration wurden dabei nochmals in einzelne Klassen unterteilt, später soll jedoch über eine Instanz von Konfiguration ein direkter Zugriff auf die in den Klassen hinterlegten Konstanten möglich sein (diese Zugriffe sind im Sequenzdigramm nicht dargestellt).

Auf Basis dieses Klassendiagramms wird nun ein Sequenzdiagramm (vgl. Abbildung 37) erstellt. Hierfür wird das Werkzeug Bouml[76] eingesetzt.

Seine Funktionalität ist zwar auf Klassendiagramme beschränkt, dafür ist es aber sehr einfach zu nutzen.

76 Bouml ist erhältlich über bouml.free.fr .

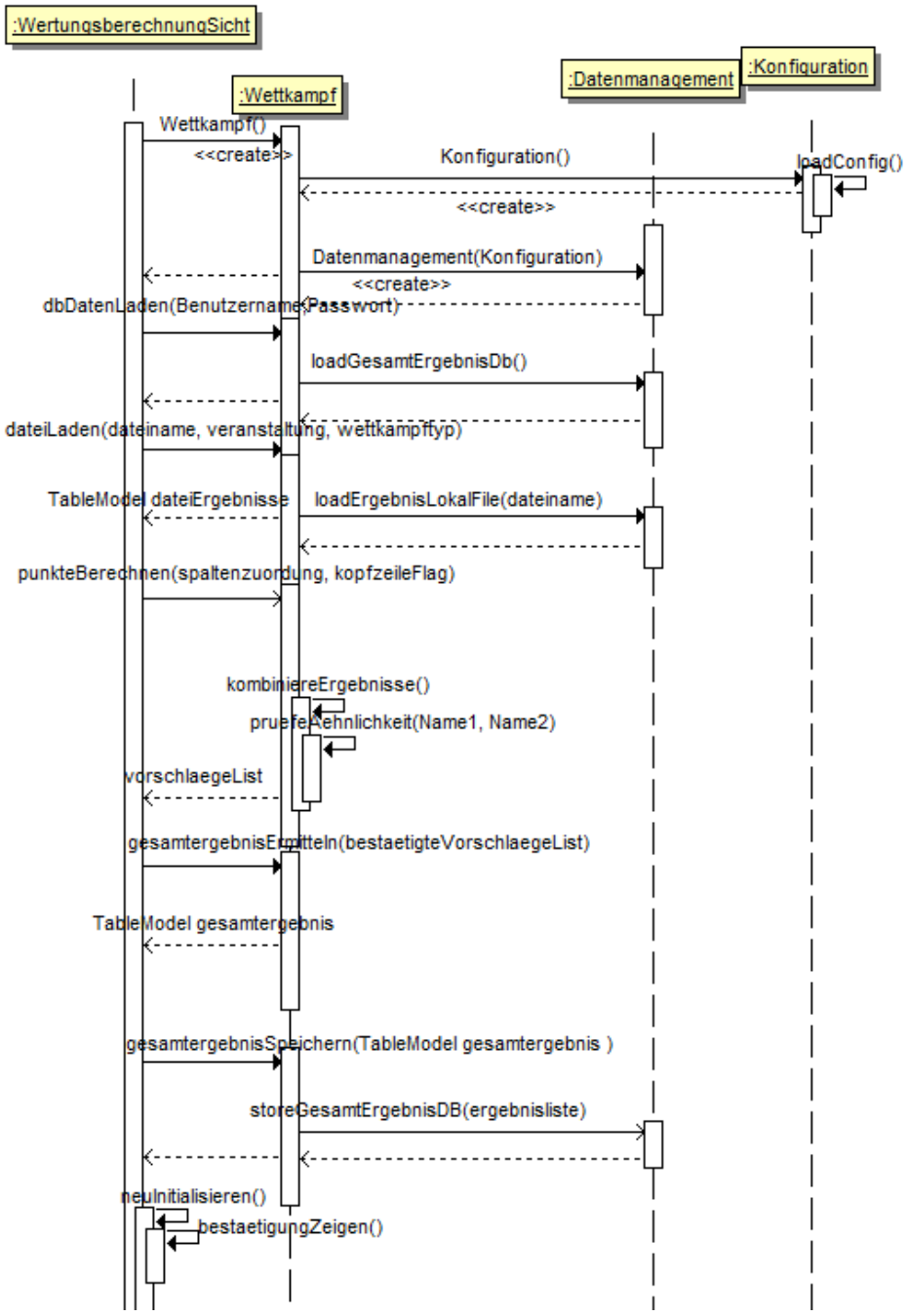


Abbildung 37: Sequenzdiagramm

Der erste Schritt erscheint sehr einleuchtend, die gefundenen Klassen und deren Unterteilung in verschiedene Typen erscheint sinnvoll, auch wenn aus technischen Gründen von der gefundenen Aufteilung Abstand genommen wird. Möglicherweise ist es an dieser Stelle nicht von „Dialogklassen“ zu sprechen, sondern besser von Dialogfunktionen, da die Schüler sonst dazu neigen könnten, diese als eigenständige Klassen beizubehalten, was dem objektorientierten Gedanken, die Funktionen möglichst an die betroffenen Objekte zu binden, widerspricht. Natürlich kann es sinnvoll sein, Funktionen in eigenen Dienstleisterklassen zusammenzufassen, aus Mangel an Erfahrung der Schüler, kann diese Abstraktionsleistung jedoch nur bedingt vorausgesetzt werden.

5.2 Architekturdefinition

Entwurfsziele

Die Entwurfsziele ergeben sich implizit aus den Nicht-Funktionalen Anforderungen, die bereits auf der Domänenebene festgelegt wurden. Dort ist jedoch nur „Sicherheit (Passwörter nicht im Klartext übertragen)“ eine Anforderung, die von der Architektur abhängt, die anderen Kriterien sind weitgehend architekturunabhängig.

Zusätzlich soll der Veranstaltungskoordinator mit vertretbarem Aufwand in der Lage sein, einen Roll-back der Daten, auf den Stand vor dem Laden einer bestimmten Veranstaltung, durchzuführen, um mit dem unbeabsichtigten Laden einer falschen Datei umgehen zu können.

Die Architektur soll sich ohne großen Aufwand und möglichst kostenneutral, in die bestehende IT Landschaft des Triathlon-Cup integrieren lassen. Dies bedeutet insbesondere, dass keine Technologien eingeführt werden sollen, die nicht bereits im Einsatz sind, sofern diese Technologien Wartungsaufand[77] erfordern.

Soll-Architektur

Als grundlegende Architektur wird eine Zwei-Schicht-Architektur gewählt. Die Funktionalität wird im Client abgebildet und die Daten in einer Datenbank abgelegt. Die Konfigurationsdateien werden zusammen mit der Anwendungskomponente bereitgestellt und jährlich aktualisiert. Als Programmiersprache für den Client wird Java gewählt. Die Entwicklungsumgebung ist eclipse[78].

Die Ausgabe der Ergebnisse erfolgt in einer Drei-Schicht-Architektur, ähnlich wie in der Komponente Cupwertung, wobei die Applikationslogik sich allerdings auf das Auslesen der Datensätze beschränkt und der Webbrowser diese nur anzeigt.

a) Komponenten und Schnittstellen

Der Java Client kommuniziert über den JDBC Treiber Connector/J mit der MySQL Datenbank. Er ist außerdem in der Lage CSV-Dateien mit den Ergebnissen einer Einzelveranstaltung aus dem lokalen Dateisystem einzulesen. Die Anwendungskonfiguration wird aus einer Konfigurationsdatei ausgelesen. Da davon auszugehen ist, dass die Nutzer mit CSV Dateien vertraut sind, wird nicht der standard Properties Mechanismus von Java verwendet, sondern zur Konfiguration ebenfalls CSV Dateien eingesetzt.

b) Ressourcen und Hardware-Software Mapping

Der Rhein-Neckar Triathlon-Cup verwendet für seine Internetdarstellung das freie CMS „Joomla!“, das auf der Datenbank MySQL und PHP aufsetzt. Realisierungsmöglichkeiten die eine andere serverseitige Logik oder eine andere Datenbank nutzen, scheiden also gemäß den Entwurfszielen aus. Die genaue Übertragung der Anwendung auf die Systeme des Triathloncups wird mit dem zuständigen IT Dienstleister festgelegt, sobald eine Entscheidung über den Produktivbetrieb getroffen ist.

Die Serverlast ist bei der Anwendung unkritisch, da es sehr wenige Nutzer gibt, de-

77 Wartungsaufwand entsteht durch Zusätzliche Hardware aber auch durch Softwaresysteme, die regelmäßig überprüft und durch Patches und Updates auf einem aktuellen Stand gehalten werden müssen. Bei neuen Systemen entsteht zusätzlich das Problem, dass Prozesse verändert und beispielsweise zusätzliche Backupmöglichkeiten geschaffen werden müssen.

78 Zu beziehen unter eclipse.org.

ren Aufgabe es ist, die Anwendung in einer definierten Reihenfolge zu unterschiedlichen Zeiten zu nutzen.

c) Betriebskonzept

Die Konfigurationsdateien werden von der Veranstaltungskoordinatoorin befüllt, und anschließend über einen FTP Zugang auf den Internetseiten des Triathloncups zur Verfügung gestellt. Betrieb und Wartung der Datenbank sowie der Internetseiten werden wie bisher vom IT Dienstleister des Triathloncups durchgeführt.

Hervorgehobene Aspekte

a) Datenverwaltung (Persistenz)

Aus den Überlegungen in diesem Kapitel, folgt dass MySQL als Datenbank einzusetzen ist. Damit ergibt sich die Möglichkeit zur Administration der Daten (Soll Prozess Datenänderung) wieder die Administrationsoberfläche phpMyAdmin einzusetzen.

Um eine einfache Verwaltung der Daten zu ermöglichen, wird wie bereits bei der Komponente Teilnehmerregistrierung eine einzige Tabelle angelegt. Die dargelegte Begründung ist auch hier anwendbar.

b) Zugangskontrolle und Sicherheit

Der Client benötigt Zugang zur Datenbank. Die Benutzerverwaltung wird der Datenbank überlassen. Dort wird einen Benutzer mit Schreibrechten auf Tabellenebene anlegt. Der Benutzername und das Passwort werden einfach zum Datenbankserver durchgereicht. Das eingegebene Passwort wird im Eingabefeld unkenntlich gemacht. Der Verbindungsaufbau über JDBC erfolgt standardmäßig verschlüsselt, nicht aber die spätere Kommunikation mit dem Server. Dies ist auch nicht notwendig, da ohnehin nur Daten übertragen werden, die später unverschlüsselt ohne Anmeldung im Internet abgerufen werden können.

Die Argumentation zur Sicherheit der Logik, die für die Darstellung der Ergebnisse zuständig ist, ist vergleichbar zur Komponente Teilnehmerregistrierung.

5.3 Feinentwurf

Der Feinentwurf erfolgt als textuale Beschreibung, bei der komplexeren Klasse (Wettkampf) mit Hilfe von Aktivitätsdiagrammen. Hervorzuheben ist die Methode pruefeAehnlichkeit, die gesondert beschrieben wird.

Datenbankdesign

Um die Realisierung der Ausgabe der Gesamtergebnisliste zu erleichtern, wird ein Flag „Aktuell“ eingesetzt, dass signalisiert, dass eine bestimmter Datensatz ausgegeben werden muss. Das Flag wird so definiert, dass es immer den Zustand des Datenbestandes angibt. Das Flag nimmt den Wert „1“ an nachdem die Daten der ersten Veranstaltung geladen wurden und erhöht sich nach dem Laden jeder weiteren Veranstaltung um den Wert 1. Das Flag wird für jeden Wettkampftyp separat erhöht. Die Ausgabe muss also lediglich nach Wettkampftyp selektieren und die Datensätze mit dem Höchsten Wert im Feld „Aktuell“ auswählen und ausgeben. Dabei erfolgt eine Einschränkung der Ausgabe auf die Datenfelder bis zur Anzahl „Aktuell“. Für Aktuell=2 werden also die Felder Veranstaltung1, Zeit1, Punkte1, Veranstaltung2, Zeit2 und Punkte2 ausgegeben

Um das Verhalten der Nutzer und des Systems besser nachvollziehen zu können wird zusätzlich ein Zeitstempel eingeführt, der den Zeitpunkt speichert, an dem ein Datensatz angelegt wurde. Im einzelnen enthält die Datenbank folgende Felder:

Datenbank	
ID (PK)	mediumint(8), unsigned, autoinc
Name	varchar(50)
Geschlecht	varchar(1)
Team	varchar(50)
Gesamtstartanzahl	tinyint(4)
Gesamtplatzierung	smallint(6)
Gesamtpunkte	double(7,3)
Wettkampftyp	varchar(20)
Cupjahr	smallint(4)
Aktuell	tinyint(4)
Timestamp	varchar(20)
Veranstaltung1	varchar(20)
Zeit1	varchar(8)
Punkte1	double(7,3)
Veranstaltung2	varchar(20)
Zeit2	varchar(8)
Punkte2	double(7,3)
Veranstaltung3	varchar(20)
...	...
Punkte10	double(7,3)

Abbildung 38: Formular und Datenbankfelder

1) Klasse Konfiguration

Enthält Konfigurationseinstellungen die bei der Instanziierung aus Textdateien eingelesen werden

Attribute:

- Wertungsregel, Vector<Veranstaltung>VeranstaltungsListe, dbVerbindungKonfiguration

Konstruktor

- Attribute instanziieren
- loadConfig()

private void loadConfig()

- Datei Wertungsregel.txt lesen und in Wertungsregel speichern
  Zum Laden der Datei das Programm „Java CSV“ benutzen
- Wenn Fehler: Meldung: "Die Datei Wertungsregel.txt kann nicht gelesen werden“
- Datei Veranstaltungen.txt lesen und Veranstaltungen an Veranstaltungsliste anhängen
  Zum Laden der Datei das Programm „Java CSV“ benutzen
- Wenn Fehler: Meldung: "Die Datei Veranstaltungen.txt kann nicht gelesen werden“
- Datei Anwendungseinstellungen.txt lesen und in dbVerbindungKonfiguration speichern
  Zum Laden der Datei das Programm „Java CSV“ benutzen
- Wenn Fehler: Meldung: "Die Datei dbVerbindungKonfiguration.txt kann nicht gelesen werden“

2) Klasse Veranstaltung

Veranstaltungen werden in der Klasse Konfiguration als Liste vorgehalten, wird aus der Datei Veranstaltungen.txt mit Daten befüllt. Die boolean Werte werden auf true gesetzt, wenn die jeweilige Veranstaltung den Wettkampftyp anbietet,

Attribute

- String veranstaltungsname, boolean jugendA,jugendB,schuelerA,schuelerB, schuelerC, erwachsene

3) Klasse Wertungsregel

Wertungsregel wird in der Klasse Konfiguration vorgehalten, und aus der Datei Wertungsregel.txt mit Daten befüllt. Speichert jeweils die Wertungsregel in der Form ***Min aus ***Max Veranstaltungen

Attribute:

- byte schuelerAMax,schuelerAMin,schuelerBMax,schuelerBMin,schuelerCMin, schuelerCMax,jugendAMin,jugendAMax,jugendBMin,jugendBMax,erwachseneMin,erwachsen eMax

4) Klasse dbVerbindungKonfiguration

Speichert String datenbankhost, datenbankname, tabellenname aus Anwendungseinstellungen.txt und Speichert die Position der Datenbankfelder in der Datenbank als int Konstante in der Form DB*Feldname* (beispielsweise int DBID = 0)

5) Klasse Wettkampf

Konstruktor
Instanziiert konfiguration, vorschlaegeList, dateFormat("HH:mm:ss"), Datenmanagement

public DefaultTableModel dateiLaden(File gewaehlteDatei)

- Erzeugt dateiDatenModel aus geladenenDaten

public void dbDatenLaden(String dbBenutzername, String dbPasswort)

- Erzeugt dbDatenModel aus geladenenDaten

public void gesamtergebnisSpeichern(DefaultTableModel modelIn)

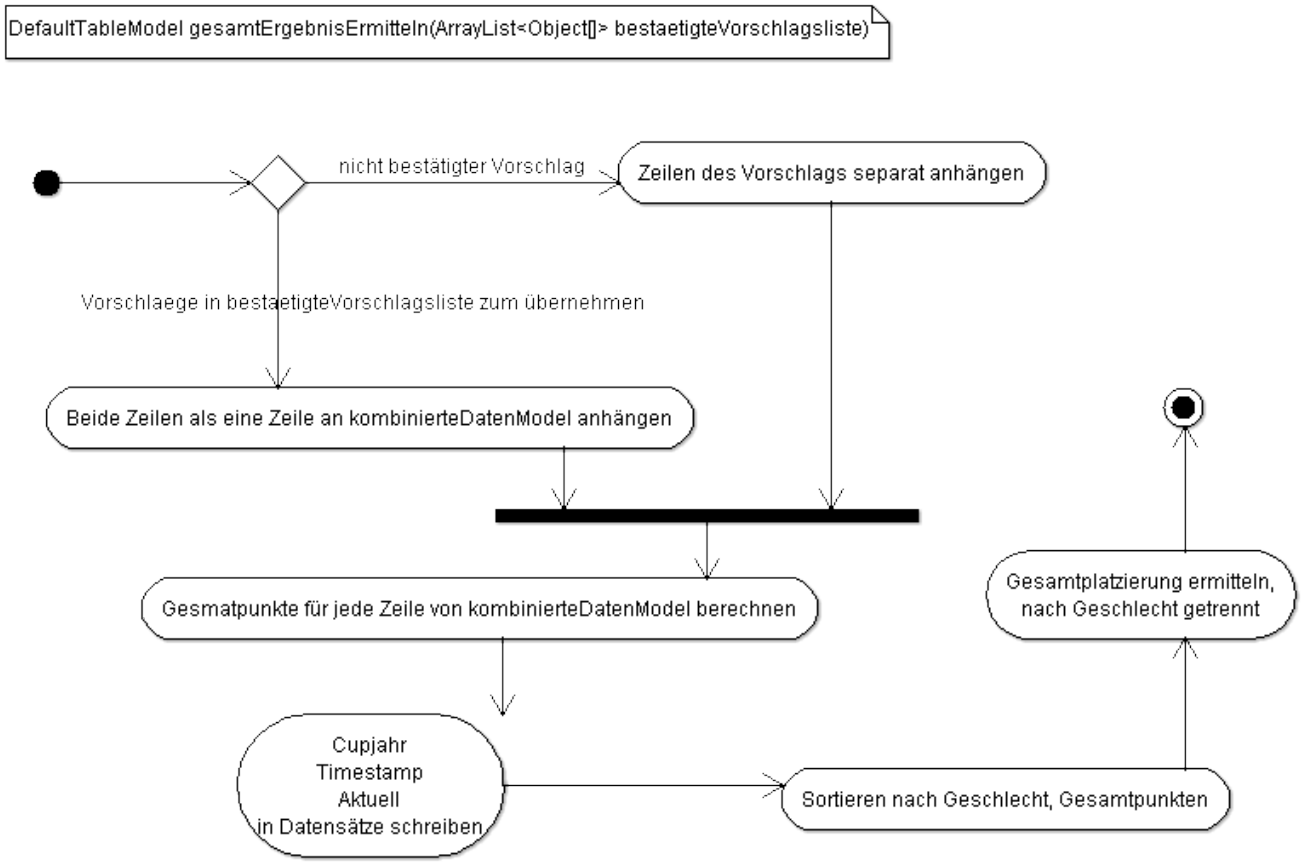
- Speichert die eingehenden Daten in der Datenbank durch Aufruf von datenmanagement.storeGesamtErgebnisDB(datenListe)

private boolean pruefeAehnlichkeit(String nameAusDatei, String nameAusDatenbank)

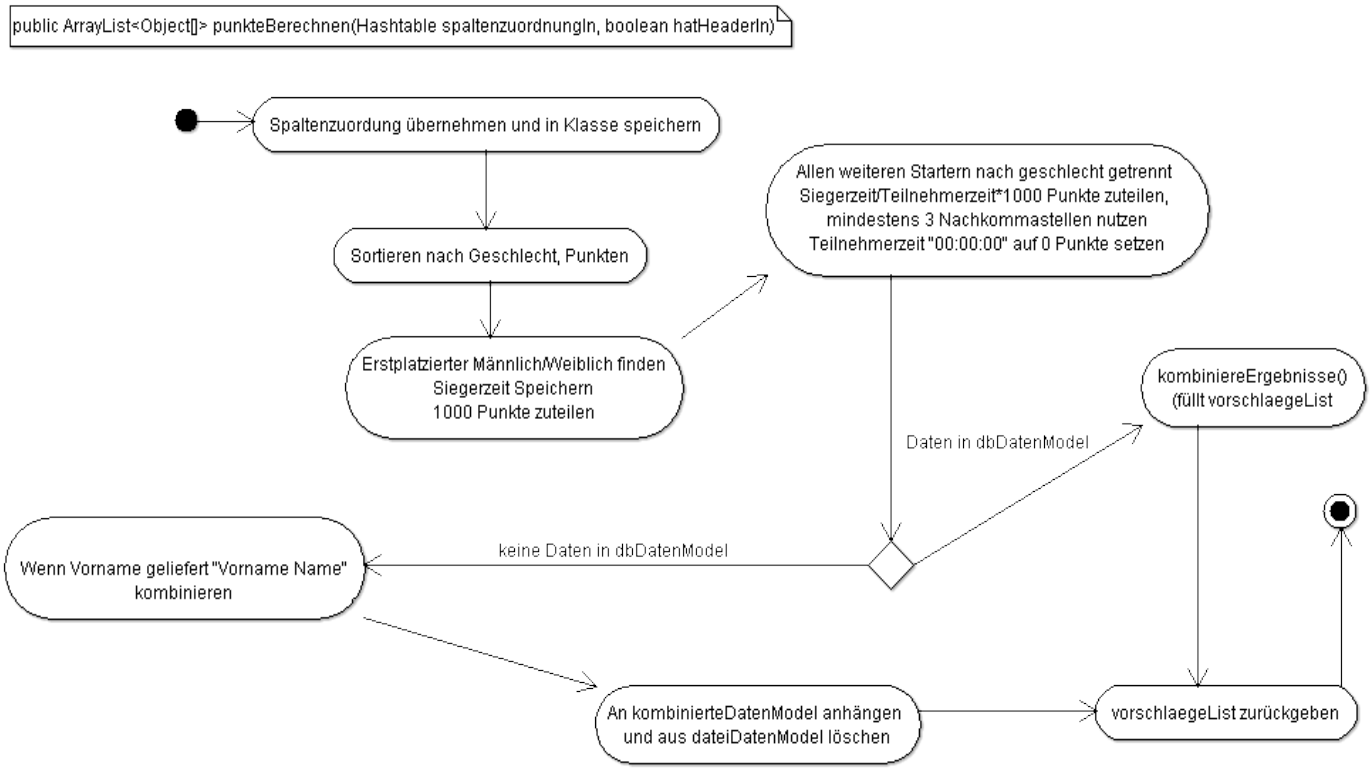
Die Ähnlichkeitsprüfung prüft, ob die kompletten Namen („Nachname Vorname“) zweier Datensätze Ähnlichkeit haben. Die Entscheidung über den eingesetzten Algorithmus wurde auf Basis von Christensen (2006: 13) getroffen, der den „Longest Common Substring“(LCS) Algorithmus empfiehlt, wenn Vorname und Nachname vertauscht sein können, was in unserem Fall verhältnismäßig häufig zu erwarten ist.
Friedman/Sideli (1992) schlagen bei der Untersuchung des Algorithmus vor, die Anwendung des Algorithmus einzuschränken. Die Mindestähnlichkeit wird definiert als die Länge des LCS geteilt durch die Länge des kürzeren der beiden verglichenen Teilstrings oder den Durchschnitt deren Länge.
Als Ähnlich werden nur Zeichenketten betrachtet, bei denen dieser wert mindestens 0,4 beträgt. Weiterhin empfehlen sie eine Mindestübereinstimmung von 3 Zeichen (vgl. Fiedman/Sideli 1992: 499). Gegebenenfalls müssen diese Parameter nach einer Testphase angepasst werden.
Daher ist die Logik wie folgt:

- LCS berechnen, hierfür wird der Algorithmus http://en.wikibooks.org/wiki/Algorithm_Implementation/Strings/Longest_common_substring verwendet
- Kürzesten Eingabestring bestimmen := Vergleichslänge
- LCS/vergleichslänge > 0.4 return true
- sonst return false

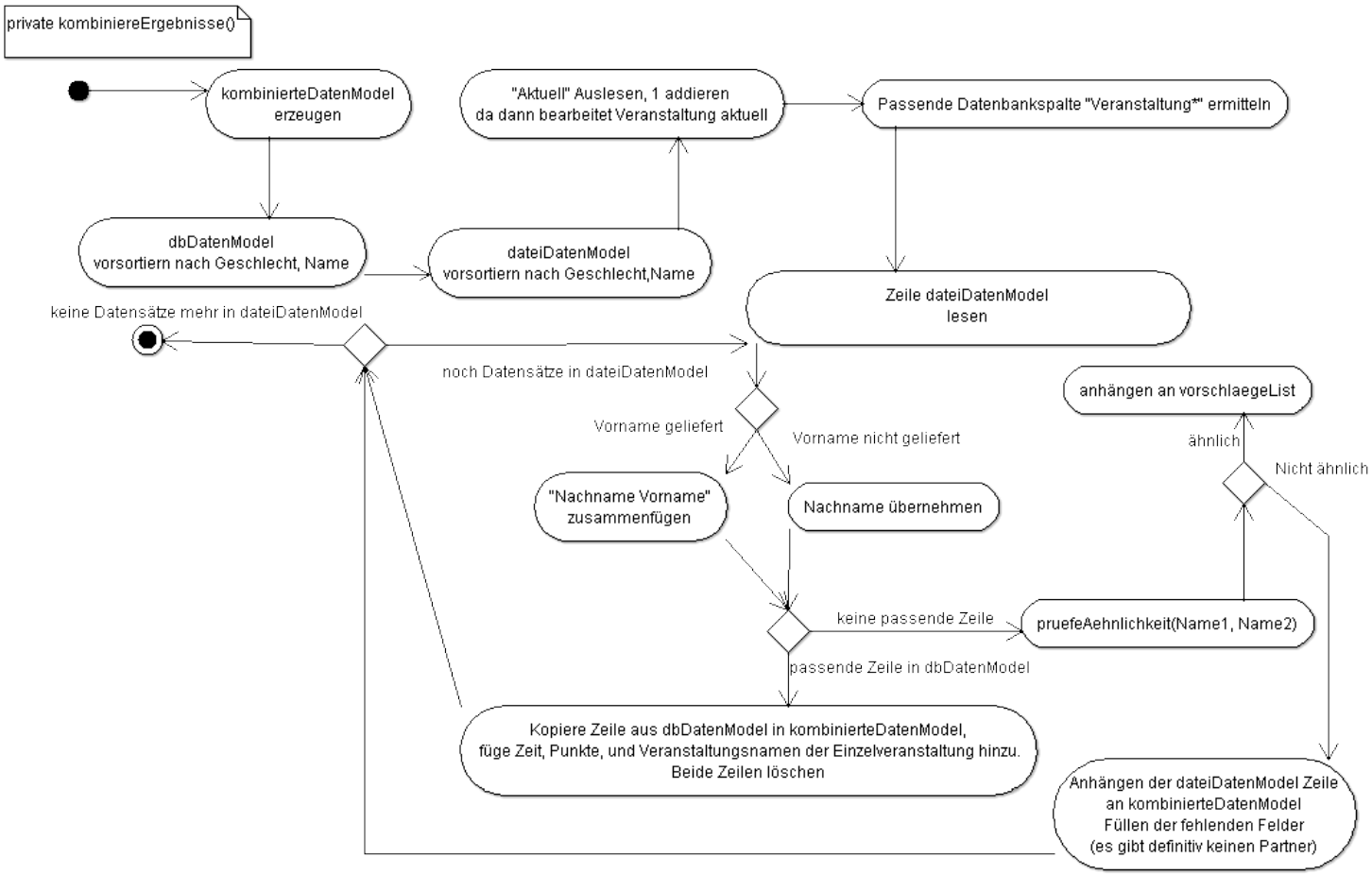
public DefaultTableModel gesamtErgebnisErmitteln(ArrayList<Object[]> bestaetigteVorschlagsliste)



public ArrayList<Object[]> punkteBerechnen(Hashtable Spaltenzuordnung, boolean hatHeader)



private kombiniereErgebnisse



6) Klasse WertungsberechnungSicht

Enthält die vier Sichten als Cards in einem CardLayout. Die Sichten werden mit einem GUI Builder erstellt. Unter Eclipse ist das Standardwerkzeug hierfür der Visual Editor.

7) Klasse Datenmanagement

Enthält die Schnittstelle zur Datenbank und zum Einlesen der Ergebnisdatei

Konstruktor

- Nimmt Konfiguration entgegen und speichert diese in Instanz

public ArrayList<String[]> loadGesamtErgebnisDB(String bearbeiteterWettkampftyp, String dbBenutzername,String dbPasswort)

- String dbBenutzername,String dbPasswort in Instanz speichern
- Verbinden zur Datenbank über JDBC / Connector/J
- Selektieren der Daten für bearbeiteterWettkampftyp und Cupjahr, sortieren nach Aktuell um den höchsten Wert für Aktuell zu erhalten
- Wenn Daten existieren:
  - Selektieren der Daten für bearbeiteterWettkampftyp, Cupjahr, Aktuell, sortieren nach Geschlecht,
- Unabhängig davon existieren wenigstens die Spaltennamen, diese auslesen
- ArrayList aus String[] aufbauen
- Spaltennamen anhängen
- Datenzeilen anhängen wenn vorhanden
- return

public void storeGesamtErgebnisDB(ArrayList<Object[]> neuesGesamtErgebnis)

- Verbinden zur Datenbank über JDBC / Connector/J (gespeicherte Anmeldedaten nutzen)
- Daten für speichern vorbereiten
- Daten speichern
- Wenn erfolgreich: Meldung „Daten für *Wettkampftyp* erfolgreich geschrieben“
- Sonst: Meldung „Schreiben in Datenbank fehlgeschlagen“

public ArrayList<String[]> loadErgebnisLokalFile(String dateiname)

- Zum Laden der Datei das Programm „Java CSV“ benutzen[79]
- Datenzeilen Lesen
- Wenn fehlgeschlagen: Meldung "Fehler beim Verarbeiten der Datei dateiname“
- return

79 Java CSV kann über http://sourceforge.net/projects/javacsv/files/ bezogen werden.

5.4 Implementierung und Test

Implementierung und Test der Komponente Cupwertung werden aus Zeitgründen zurückgestellt werden. Daher ist leider noch keine abschließende Bewertung der vorgeschlagenen Lösung möglich. Im nächsten, abschließenden Kapitel erfolgt jedoch eine Zusammenfassung und Bewertung der im Verlauf der Arbeit gewonnenen Erkenntnisse im Bezug auf den Entwicklungsprozess.

6. Umfassendes Software Engineering im Schulunterricht?

Der gewählte „Objects First“ Ansatz, der mit Hilfe von konkreten Objekten an Stelle von Klassen arbeitet, erwies sich als prinzipiell sinnvoll, jedoch leider nicht überall konsequent durchsetzbar. Besonders bei den Systemfunktionen ist eine solche „konkrete“ Angabe kaum möglich und auch schwierig in das betreffende Artefakt zu integrieren. Zur Erarbeitung im Unterricht erscheinen an der Schnittstelle zwischen Domänendaten, Szenarien und Klassen weitere Schritte notwendig zu sein. Dort bietet es sich an, die Szenarien und das Interaktionsdiagramm auf Basis eines „Durchspielens“ zu erarbeiten. Dabei werden ähnlich wie in TRAIN zunächst Klassenkandidaten auf Basis der Substantive und Verben in der Aufgabenbeschreibung bestimmt[80]. Die Klassenkandidaten werden auf CRC Karten[81] notiert. Dieses Vorgehen deckt sich mit meiner Beobachtung beim zweiten Prozessdurchlauf, dass es zunächst einfacher sein dürfte ein Klassendiagramm grob mit Aufgaben, Zuständigkeiten und Partnerklassen zu entwerfen. Die Karten werden dann an die Schüler verteilt[82]. Die Schüler spielen dabei die Rolle dieser Klasse in der Interaktion. Nun wird ein Konkretes Szenario durchgespielt, wobei konkrete Daten verwendet werden, die zwischen den Klassen ausgetauscht werden[83]. Während dieses Prozesses wird dann das Artefakt „Szenario und Systemfunktionen“ erstellt und präzisiert. Gleichzeitig wird von einem anderen Protokollanten der Ablauf der Interaktion festgehalten, aus dem im Anschluss ein Sequenzdiagramm generiert werden kann. Sodann wird versucht weitere Szenarios zu finden, die ebenfalls durchgespielt werden. In einer verschärften Version werden die Teilnehmer so positioniert, dass sie die anderen Schüler nicht sehen können. Dies simuliert die beschränkte Sicht der einzelnen Objekte[84]. Auf Basis der beim Durchspielen gewonnenen Erfahrungen, werden schrittweise Klassenkandidaten konsolidiert oder neue hinzugefügt. Anschließend werden die Schnittstellen der Klassen (öffentliche Methoden) und deren Parameter, Rückgabewerte und Attribute formal beschrieben und entweder erneut auf einer CRC Karte oder im bereits eingeführten Werkzeug „Green“ (vgl. Kapitel 5.1.3) erfasst. Gegebenenfalls können dann mit einem Ausdruck des „Green“ Klassendiagramms als CRC Karten nochmals die Szenarien durchlaufen werden.[85] Der Einsatz des Round-Trip Editors Green in Kombination mit dem gewählten aufwandsreduzierten Testansatz dient dazu, die Gegenwartsbedeutung der Modellierungs- und Testmethoden bei den Schülern in die pädagogisch gewünschte Bedeutung zu transformieren, indem es ihnen die Nützlichkeit dieser Techniken erfahrbar macht. Für die Durchführung der Tests scheint mit dem

80 Beim regulären Prozess erfolgt dieser Schritt auf Basis der Use-Cases, Systemfunktionsbeschreibungen und des Domänendatendiagramms, im reduzierten TRAIN Prozess, erfolgt der Schritt auf Basis des detaillierteren Soll Prozesses.

81 CRC Karten wurden von Beck/Cunningham (1989) eingeführt. CRC Karten sind im Prinzip eine grobe Version einer Klassenbeschreibung im Klassendiagramm. Die CRC Karte enthält den Namen, die Zuständigkeiten und die Partnerklassen.

82 Der große Vorteil dieser Methode ist, dass man praktisch alle Schüler am Entwurfsprozess aktiv beteiligen kann.

83 Eine Beschreibung eines solchen Vorgehens am Beispiel findet sich bei Barnes/Kölling (2006: 484ff)

84 Dieses Vorgehen ist auch als „Objektspiel“ bekannt (vgl. Humbert 2005: 86).

85 Um diesen manuellen Prozess des Szenarioausbaus und der Klassenfindung zu automatisieren wurde an der Universität Paderborn das Werkzeug (www.fujaba.de) entworfen, das auf Basis der Szenarien dann auch direkt Testcode generieren kann. In einer reduzierten Version wurde es auch für Lehrzwecke optimiert (life.uni-paderborn.de). Jedoch ist die Benutzung relativ umständlich und gerade der interaktive Charakter der Szenarienerstellung, der alle Schüler einbezieht, geht dabei verloren. Obwohl es auch eine Beschreibung des Einsatzes nach vorherigem Nachspielen eines Szenarios gibt (vgl. Diethelm et al. 2002 und Diethelm 2007).

Code-Coverage Ansatz in Kombination mit Capture-Replay für Regressionstests bei leicht geänderten Anforderungen ein aufwandsreduziertes Vorgehen gefunden worden zu sein. Zukünftig ist zu überlegen ob für den Komponententest automatisierte Testwerkzeuge zum Einsatz kommen sollen, die selbständig sicherstellen, das alle Ausführungspfade in einem Test abgedeckt werden[86], für kleinere Projekte halte ich jedoch den „Manuellen Review“ Ansatz, den ich vorgestellt habe für geeigneter.

Usability Tests bieten, auch wenn sie in dieser Arbeit nur am Rande behandelt wurden, eine besondere Chance für den schulischen Einsatz, da die Schüler die Möglichkeit haben, ihr Handlungsprodukt anderen zu präsentieren und direktes Feedback zu bekommen. Dabei kann die bei Projekten oder Projektwochen ohnehin eingeplante Präsentationszeit für den Test genutzt werden. Raasch beschreibt die Einrichtung und Nutzung eines „Usability Labors“ im universitären Umfeld (vgl. Raasch 2006). Ein ähnlicher Aufbau ist möglicherweise auch ohne zusätzliche Hardware und Kosten durch Einsatz einer webcambasierten Lösung realisierbar[87]. Fragebögen können entweder auf den zu diesem Zweck entwickelten „Sumi“ Fragebogen aufbauen, oder selbst konstruiert werden. Hierfür stehen zahlreiche kostenlose Onlinedienste zur Verfügung[88]. Umfragen bieten immer die Gelegenheit einer vertiefenden Behandlung im Deutsch- oder Mathematikunterricht, wenn es um Methodenkompetenz, Formulierung oder Auswertung geht.

Die einzelnen Konzepte des objektorientierten Entwurfs wurden in der vorliegenden Arbeit nicht im Detail untersucht, jedoch hat Kicherer für den Gesamtthemenkomplex objektorientierte Modellierung bereits ein nützliches und umfangreiches Konzept vorgelegt (vgl. Kicherer 2008), das die einzelnen Elemente der Objektorientierung auf Vermittelbarkeit und Relevanz untersucht. Leider ist die von ihm gewählte Beispielanwendung weder nahe an einer realen Anwendung, wie sie in der Wirtschaft entwickelt werden würde, noch besonders schülernah. Hier bieten sich andere Rahmenanwendungen mittlerweile eher an. Einen solchen Rahmen bietet die Entwicklungsumgebung „Greenfoot“[89], mit der sich einfach Computerspiele erstellen lassen, wobei die Komplexität von Java zunächst verborgen wird, aber bei Bedarf der komplette Sprachumfang zur Verfügung steht.
Der Vorteil dieser Umgebung ist, dass direkt eine grafische Benutzeroberfläche zur Verfügung steht, in der sich auch einzelne Komponenten sofort testen lassen, damit wäre ein szenarienbasierter Test auf Komponentenebene für jedes einzelne Programmierteam möglich, was den Zeitaufwand für den Test nochmals reduziert.
Für eine zukünftige Weiterentwicklung des in dieser Arbeit eingeführten Ansatzes bietet es sich an das Konzept von Kicherer als Orientierungshilfe zu nutzen um die Inhalte des reduzierten TRAIN Prozesses an konkreten Unterrichtsgegenständen anzukoppeln und damit eine unterrichtsbegleitende Vermittlung zu ermöglichen. Auch bei dem Konzept von Kicherer muss zuvor nochmals eine Reduktion der Inhalte erfolgen, denn er geht bei seinem Konzept von den Idealbedingungen in einem technischen Gymnasium in Baden-Württemberg, mit entsprechend umfangreichem Zeitbudget aus.
Abschließend sind in der Abbildung 40 und 41 nochmals die Ergebnisse der Arbeit

86 Einen guten Eindruck machen die Projekte Randoop (people.csail.mit.edu/cpacheco/randoop/) und Java Path Finder (JPF) (babelfish.arc.nasa.gov/trac/jpf) als optionen.
87 Opengazer oder Ogama scheinen aussichtsreiche Kandidaten hierfür zu sein. Die Programme können unter inference.phy.cam.ac.uk/opengazer/ und didaktik.physik.fu-berlin.de/ogama/ bezogen werden.
88 Der Sumi Fragebogen ist erhältlich über sumi.ucc.ie/de/, als Umfragedienste bieten sich limesurvey.org, voycer.de oder grafstat.de an.
89 Greenfoot kann unter greenfoot.org bezogen werden.

zusammengefasst.

Allgemein	Projektarbeit	Unterrichtsbegleitend
Testen	- Komponententest mit Coverage/Debug - Systemtest/Regressionstest mit Capture/Replay - Usability Test	- Komponententest mit Coverage/Debug - Regressionstest bei entsprechender Aufgabenstellung mit Capture/Replay
Rationale	- Vorgaben durch Lehrkraft, wenn nötig	- Häufig unterrichtsbegleitend
Inspektionen	- Inspektionen von Arbeit der Partnergruppe auf Basis der reduzierten Checklisten	Bei Kleinaufagaben - Übereinstimmung mit Standards-Karte - Dokumentation
Prozessübergreifend	- Advance Organizer	- Prozess vorstellen

Abbildung 40: Allgemeine Bestandteile des reduzierten Prozesses

Artefakte	Projektarbeit	Unterrichtsbegleitend
Anforderungsanalyse		
Domänen/Aufgaben	Rolleninstanzen Soll Prozess Domänendaten Systemverantwortlichkeiten (NFR)	Soll Prozess Domänendaten
Interaktionsebene	„Szenario und Systemfunktionen“ UI Struktur (Papier Mock-Ups)	Eingeschränkt wie Projektarbeit
Systemebene - GUI	GUI: Guibuilder um Capture/Replay zu ermöglichen	Eingeschränkt wie Projektarbeit
Systemebene - Kern	CRC Szenarien spielen und optimieren (Sequenzdiagramm) Klassendiagramm (möglichst Roundtrip)	Eingeschränkt wie Projektarbeit
Architekturdefinition		
	Eingeschränkt, Optionen über Rationale vorgeben	Architekturthemen anhand von Beispielen aufgreifen
Feinentwurf		
	Beliebige Techniken (Struktogramm, Programmfluss, Aktivitätsdiagramm, Textual)	Nach Lehrplanvorgabe Techniken einführen
Implementierung	Code-Formatierer Konventionenkarte	Code-Formatierer Konventionenkarte

Abbildung 41: Artefakte des reduzierten TRAIN Prozesses

Die Essenz des TRAIN Prozesses ist es, Testen, Rationale und Inspektionen systematisch zu nutzen und mit einer gründlichen Anforderungsanalyse zu kombinieren. Diese Zielvorgabe lässt sich auch mit einem reduzierten Prozess erreichen. Das von mir eingeführte Artefakt „Szenario und Systemfunktionen“ leistet hierzu einen wertvollen Beitrag, da es wesentlich zur Straffung des Prozesses beiträgt, ohne zu viele Informationen zu verlieren. Die Durchführung des reduzierten Prozesses im „Selbstversuch“ erwies sich als hilfreich um die Nützlichkeit des Artefakts zu bestätigen. Auf Basis der gewonnenen Erfahrungen wurde vorgeschlagen, die Ermittlung des Klassenaufbaus mittels CRC Karten und einem Szenario/Objektspiel durchzuführen. An der Schule ermöglicht dies, auch schwächere oder ruhigere Schüler zu aktivieren und in den Analyseprozess zu integrieren.
Wenn immer möglich sollten Prozesselemente auch unterrichtsbegleitend eingebracht werden. So können etwa Rationale regelmäßig eingesetzt werden um die Eigenschaften von vorgestellten Konstrukten festzuhalten, ein einfaches Beispiel hierzu sind die Eigenschaften von Schleifentypen. Sind Prozesselemente bereits bekannt, können sie auch einfacher in einer Projektarbeit aufgegriffen und vertieft werden. Unterrichtsbegleitend können natürlich viele Elemente des Prozesses nur anhand von kleinen Beispielen behandelt werden. Dennoch haben meine Ausführungen gezeigt, dass ein umfassendes Software Engineering im Schulunterricht realisierbar ist.

Glossar

Anmeldeformular

=>Onlineanmeldung

BASF Triathlon-Cup Rhein-Neckar

=> Cup

Cup

Der Cup besteht aus mehreren unabhängigen „=>Veranstaltungen“. Bei jeder Veranstaltung wird auf Basis der erreichten Zeiten und mit Hilfe einer Formel eine Punktzahl berechnet. Wenn ein Starter bei mehr als einer Veranstaltung startet, werden die Punktzahlen nach einer =>Wertungsmodalität addiert.

Cupwertung

Die =>“Veranstaltungsübergreifende Komponente“ ermöglicht es, die Daten des Zeitnahmedienstleiters flexibel entgegenzunehmen, die Punktzahl der Einzelveranstaltungen zu errechnen, die Gesamtwertung entsprechend anzupassen und das Resultat auf der Internetseite des Cups zu veröffentlichen.

Ergebnisse

Ergebnis bezeichnet das Ergebnis einer =>Einzelveranstaltung

Einzelveranstaltung

=> Veranstaltung

Gesamtergebnis

=> Cupwertung

Offlineanmeldung

Eine klassische Anmeldemethode mit Stift und Papier, die Daten werden dann vom Startlistenverwalter manuell in das System nachgetragen. In diesem Fall ist es zulässig, dass die E-Mail Adresse nicht angegeben wird.

Onlineanmeldung

Die Internetbasierte Anmeldung erfolgt über die Komponente =>„Teilnehmerregistrierung“.

Staffelwertung

Für die Staffelwertung existiert keine Cupwertung.

Startliste

Die Liste der bestätigten Teilnehmer.

Teilnehmerregistrierung

Die =>„Veranstaltungsspezifischen Komponente“ „Teilnehmerregistrierung“ soll die Verwaltung einer Einzelveranstaltung erleichtern, indem einen automatisierte Anmeldung für eine Triathlonveranstaltung bereitgestellt wird. Weiterhin ermöglicht sie eine Auflistung der registrierten Teilnehmer.

Triathlontyp

Es existieren verschiedene Wettkampftypen mit unterschiedlichen Strecken, in dem betrachteten Beispiel sind dies die Typen „Erwachsene“, „Staffel“, „JugendA“, „JugendB“, „SchülerA“, „SchülerB“, „SchülerC“.

Triathlonveranstaltung

=> „Veranstaltung“

Veranstaltung

Organisatorisch komplett unabhängiger Sportwettkampf

Veranstaltungsspezifische Komponente

Dient der Verwaltung einer Einzelveranstaltung, realisiert durch die Komponente => „Teilnehmerregistrierung“.

Veranstaltungsreihe

=> Cup

Veranstaltungsübergreifende Komponente

Dient der Datenweitergabe und Publizierung der Ergebnisse der Einzelveranstaltungen, realisiert durch die Komponente => „Cupwertung“.

Warteliste

Für eine einzelne Triathlonveranstaltung werden mehrere Wartelisten geführt, da für jeden Wettkampftyp eine bestimmte Anzahl an Startplätzen zur Verfügung steht, die dann manuell aus der Warteliste periodisch wieder aufgefüllt werden können.

Wertungsmodalität

Meist werden die besten Zwei Veranstaltungen gewertet, dies bedeutet gleichzeitig, dass bei mindestens zwei Veranstaltungen gestartet werden muss, die Wertungsmodalität kann also beispielsweise lauten „2 aus 4“ oder „2 aus 5“ =>Veranstaltungen, die an dem => Cup teilnehmen. Die Wertungsmodalitäten können sich jedes Jahr ändern, da die Anzahl der Veranstaltungen pro => Altersklasse jedes Jahr variiert.

Wertungspunkte

=> Gesamtwertung

Wettkampf

=> Veranstaltung

Wettkampftyp

=> Triathlontyp

Zeitnahmedienstleister

Bei den Einzelveranstaltungen können verschiedene Zeitnahmedienstleister zum Einsatz kommen. Die Dienstleister erfassen die Teilnehmer und ordnen ihnen eine ID zu, die den Starter mit einem RFID Chip verbindet, den er beim Wettkampf bei sich trägt. Dieser Chip dient der Messung von Zwischen- und Endzeiten. Der Dienstleister stellt in der Regel ein Portal bereit, in dem sich die Teilnehmer registrieren können. Diese Funktionalität soll durch die Komponente =>“Teilnehmerregistrierung“ ersetzt werden. Nach dem Wettkampf stellt der Dienstleister die Ergebnisse bereit. Auf Basis dieser Daten erfolgt eine manuelle Aufbereitung und Berechnung der Cupwertung, was zukünftig mit Hilfe der Komponente =>„Cupwertung“ durchgeführt werden soll.

Literatur

ARD/ZDF, Onlinestudie 2007, Mai 2007, http://www.daserste.de/service/onlinestudie-2007- vorab.pdf, (aufgerufen am 07.06.2007)

Bachfeld, Daniel, 2004, Giftspritze, SQL-Injection - Angriff und Abwehr http://www.heise.de/security/artikel/Giftspritze-270382.html

Barnes, David J./Kölling, Michael, 2006, Java lernen mit BlueJ. Eine Einführung in die objektorientierte Programmierung, 3. Auflage

Beck, Kenth/Cunningham, Ward, 1989, A Laboratory For Teaching Object-Oriented Thinking, in: OOPSLA'89 Conference Proceedings October 1-6, 1989

Borner, Lars/Paech, Barbara, 2006, Teaching Software Engineering Process Emphasizing Testing, Rationale and Inspection (TRAIN), Proceedings of the 1st AIS SIGSAND European Symposium on Systems Analysis and Design

Börstler, Jürgen, 2007, Objektorientiertes Programmieren – Machen wir irgendwas falsch?, in: INFOS 2007 - Didaktik der Informatik in Theorie und Praxis, 12. GI-Fachtagung Informatik und Schule, S. 9-20

Brügge, Bernd/Dutoit, Allen H., 2004, Objektorientierte Softwaretechnik mit UML, Entwurfsmustern und Java

Bundesbeauftragter für den Datenschutz, 2007, BfDI Info 1, Bundesdatenschutzgesetz. Text und Erläuterung, 13. Aufl.

Cristensen, Henrik Baerbak, 2008, Experiences with a Focus on Testing in Teaching, in: Jens Bennedsen/ Michael E. Caspersen/Michael Kölling (Hrsg.), Reflections on the Teaching of Programming, S. 147-165

Christen, Peter, 2006, A Comparison of Personal Name Matching: Techniques and Practical Issues, TR-CS-06-02, in: Joint Computer Science Technical Report Series of The Australian National University

DeCharms, R.,1979, Motivation in der Klasse

Diethelm, Ira, 2007, Strictly models and objects first – Unterrichtskonzept für objektorientierte Modellierung, in: INFOS 2007 - Didaktik der Informatik in Theorie und Praxis, 12. GI-Fachtagung Informatik und Schule, S. 45-56

Diethelm, Ira/Geiger, Leif/Zündorf, Albert, 2002, UML im Unterricht: Systematische objektorientierte Problemlösung mit Hilfe von Szenarien am Beispiel der Türme von Hanoi, in: Workshops der GI-Fachgruppe "Didaktik der Informatik" (DDI'02),10.-11. Oktober 2002, Witten-Bommerholz. P-22, p. 33-42 (2002).

Elmasri, Ramez/Navathe, Shamkant B., 2002, Grundlagen von Datenbanksystemen, 3. Auflage

Friedman, Carol/Sideli, Robert, 1992, Tolerating Spelling Errors during Patient Validation, in: Computers and Biomedical Research 25 486-509 1992

Gesellschaft für Informatik, 2003, Empfehlungen für ein Gesamtkonzept zur informatischen Bildung an allgemein bildenden Schulen

Gesellschaft für Informatik, 2008, Grundsätze und Standards für die Informatik in der Schule. Bildungsstandards Informatik für die Sekundarstufe I

Gudjons, Herbert, 2003, Frontalunterricht – neu entdeckt Integration in offenen Unterrichtsformen

Häfele, Philipp, Softwareentwicklung mit dem TRAIN-Prozess - Bachelor Thesis http://www-swe.informatik.uni-heidelberg.de/research/publications/TRAIN.pdf, 2005

Hartmann, Werner/Näf, Michael/Reichert, Raimond, 2007, Informatikunterricht Planen und Durchführen

Hochstätter, Christoph H., 2008, E-Mail-Provider im Praxistest: Verschlüsselung oft mangelhaft, www.zdnet.de/sicherheit_in_der_praxis_e_mail_provider_im_praxistest_verschluesselung_oft_mangelhaft_story-39001543-39196494-1.htm (20.03.2009)

Hubwieser, Peter, 2004, Didaktik der Informatik, 2. Aufl.

Humbert, Ludger, 2005, Didaktik der Informatik, 2. Auflage

Jacobson, Ivar et al., 1992, Object-Oriented Software Engineering. A Use Case Driven Approach

Janck, Werner/Meyer, Hilbert, 1991, Didaktische Modelle, 5. Aufl.

Kicherer, Walter, 2008, Objektorientierter Softwareentwurf in der Sekundarstufe II. Inaugural Dissertation. Universität Heidelberg

Lauesen, S., User Interface Design. A Software Engineering Perspective, Harlow: Pearson Education, 2005

Meyer, Hilbert, 1987a, Unterrichtsmethoden I: Theorieband, 2. Aufl

Meyer, Hilbert, 1987b, Unterrichtsmethoden II: Praxisband., 2. Aufl.

Michael, Jörg, 2007, 40 000 Namen. Anredebestimmung anhand des Vornamens, in: c't 17/07, S. 182

Ministerium für Kultus, Jugend und Sport Baden-Württemberg, 2003, Lehrpläne für das berufliche Gymnasium Wirtschaftswissenschaftliche Richtung (WG). Wirtschaftsinformatik Jahrgangsstufe 1 und 2

Ministerium für Kultus, Jugend und Sport Baden-Württemberg, 2004, Bildungsplan 2004. Allgemenbildendes Gymnasium

Ministerium für Bildung, Wissenschaft, Jugend und Kultur Rheinland-Pfalz , 2008, Lehrplanentwurf Informatik. Grund- und Leistungsfach Einführungsphase und Qualifikationsphase der gymnasialen Oberstufe

Nassi, I./Shneiderman, B., Flowchart Techniques for Structured Programming, in: SIGPLAN Notices 8, 8 (August, 1973) http://www.cs.umd.edu/hcil/members/bshneiderman/nsd/1973.pdf (01.02.2010)

Oestereich, Bernd, 2004, Objektorientierte Softwareentwicklung. Analyse und Design nmit der UML 2.0, 6. Aufl.

Paech, Barbara/Borner, Lars/Rückert, Jürgen/Dutoit, Allen H./Wolf, Timo, Vom Kode zu den Anforderungen und wieder zurück: SoftwareEngineering in sechs Semesterwochenstunden, in: Software Engineering im Unterricht der Hochschulen SEUH9, Aachen 2005, S. 56-67

Raasch, Jörg, 2006, Usability von Anwendungssystemen – didaktische Aspekte, in: HDI 2006 - Hochschuldidaktik der Informatik, 2. GI- Fachtagung S. 23-36

Romeike, Ralf, 2007, Kriterien kreativen Informatikunterrichts in: INFOS 2007 - Didaktik der Informatik in Theorie und Praxis, 12. GI-Fachtagung Informatik und Schule, S. 57-68

Schubert, Sigrid/Schwill, Andreas, 2004, Didaktik der Informatik

Spillner, Andreas/Linz, Tilo, Basiswissen Softwaretest, 3. Auflage, dpunkt. Verlag Heidelberg, 2005

Spillner, Andreas, The W-Model. Strengthening the Bond Between Development and Test, 2002, http://www.stickyminds.com/s.asp?F=S3572_ART_2

Sommerville, Ian, 2004, Software Engineering, 7. Auflage

Vocke, Heike/Woigt, Ulrike, 2005, Software-Engineering in der beruflichen Ausbildung – Simulation realer Projektsituationen, in: INFOS 2005, 11. GI-Fachtagung Informatik und Schule, S.297-397

Zuser, Wolfgang/Grechenig, Thomas/Köhle, Monika, 2004, Software Engineering mit UML und dem Unified Process, 2. Auflage

Anhang A- Prozessübersicht aus Häfele 2005

Management	Phasen	Entscheidungsebenen	Dokumentation: Artefakte IN	Dokumentation: Artefakte OUT	Dokumentation: Rationale	QS-Tätigkeiten
Projektmanagement Änderungsmanagement Konfigurationsmanagement	Analyse, Wissenserwerb, Konfliktlösung					
	Requirements-engineering	Aufgabenebene	- Erhobene Defizite des bisherigen Geschäftsprozesses (z. B. auch Probleme mit externen Systemen) - Geschäftsprozessmodelle - Protokolle	- Rollenbeschreibungen - Aufgabenbeschreibungen - Glossareinträge - Qualitätskriterien (NFR) von Aufgaben	Begründung der Aufgabenwahl und der NFRs	
		QS	- Softwaredokumentation - Lesetechnik	- Problem-/Fehler-/Korrekturliste - Korrigierte SW-Dokumentation		**Inspektion**
		Domänenebene	- Rollenbeschreibungen - Aufgabenbeschreibungen - NFRs	- Domänendaten-Diagramm - Datenbeschreibung - Systemverantwortlichkeiten-übersicht (in Form eines UC Diagramms) - Aktivitätsdiagramme für IST und SOLL - Ist-, Sollbeschreibung - Glossareinträge - Qualitätskriterien (NFR) für die Domäne - Qualitätskriterien (NFR) an die Architektur - Globale Qualitätskriterien (NFR) für Funktionale Anforderungen	Begründung des SOLL und der NFRs (aber auch aller weiteren Artefakten) aus den Artefakten und v. a. den NFRs der Aufgaben-ebene	
		QS	- Softwaredokumentation - Lesetechnik - Systemverantwortlichkeiten - NFRs	- Problem-/Fehler-/Korrekturliste - Korrigierte SW-Dokumentation - QS-Plan (z. B. was, wie intensiv, Priorisierung) - Systemtestplan	Begründung des QS- und Testplans	**Inspektion** **QS-/Testplan-erstellung**

Management	Phasen		Entscheidungsebenen	Dokumentation: Artefakte IN	Dokumentation: Artefakte OUT	Dokumentation: Rationale	QS-Tätigkeiten
Projektmanagement Änderungsmanagement Konfigurationsmanagement	Analyse, Wissenserwerb, Konfliktlösung	**Requirements-engineering**	Interaktionsebene	- Systemverantwortlichkeiten-übersicht - Sollbeschreibung - Aktivitätsdiagramme (SOLL) - NFRs - Systemverantwortlichkeiten-beschreibungen (d. h. Aufgabenbeschreibungen, die zu Systemverantwortlichkeiten geworden sind) - Domänendaten	- Use Case Beschreibungen - Systemfunktionsbeschreibungen - Nutzungsdiagramm - Szenarien - Globale Qualitätskriterien (NFR) für Funktionale Anforderungen - Qualitätskriterien (NFR) für Use Cases - Qualitätskriterien (NFR) für Systemfunktionen - Qualitätskriterien (NFR) an die Architektur - UI-Struktur-Diagramm - Interaktionsdaten-Diagramm (Verfeinertes Datendiagramm)	Begründung der Artefakte aus den Artefakten und v. a. den NFRs der Domänenebene	
			QS	- Softwaredokumentation - Lesetechnik	- Problem-/Fehler-/Korrekturliste - Korrigierte SW-Dokumentation		**Inspektion**
				- Systemtestplan - Funktionale und Nicht-funktionale Anforderungen - Szenarien	- Systemtestspezifikation - Erstimplementierung (d. h. Rahmenablauf in der Testumgebung implementieren)	Begründung der Testspezifikation	**Testspezifikations-erstellung**
				- Use Case Beschreibungen - UI-Struktur-Diagramm	- Usabilitytestplan - Usabilitytestspezifikation - Prototypes	Begründung des Testplans und der Testspezifikation	**Testplan-erstellung/ -spezifikation Prototyping**

Management	Phasen		Entscheidungsebenen		Dokumentation: Artefakte IN	Dokumentation: Artefakte OUT	Dokumentation: Rationale	QS-Tätigkeiten
Projektmanagement Änderungsmanagement Konfigurationsmanagement	Analyse, Wissenserwerb, Konfliktlösung	**Requirements-engineering**	System-ebene	Kern	- ERD, UC, Systemfunktions-beschreibungen - Szenarien - NFRs	- Analyseklassendiagramm (AKD) - Sequenzdiagramme - NFRs	Begründung der Artefakte aus den Artefakten und v. a. den NFRs der Interaktionsebene	
				GUI		- Dialog Text - Dialog Zustandsdiagramm - Hilfsfunktionen		
			QS		- Softwaredokumentation - Lesetechnik	- Problem-/Fehler-/Korrekturliste - Korrigierte SW-Dokumentation		**Inspektion**

Management	Phasen		Entscheidungsebenen	Dokumentation: Artefakte IN	Dokumentation: Artefakte OUT	Dokumentation: Rationale	QS-Tätigkeiten
Projektmanagement Änderungsmanagement Konfigurationsmanagement	Analyse, Wissenserwerb, Konfliktlösung	**Architektur-definition**		- Soll-Prozess - Analyseklassendiagramm - (Architektur-) NFRs - Ressourcen - Externe Systeme - Architekturmuster	- Entwurfziele - Konfigurationsdiagramm - Komponentendiagramm/ Blockdiagramm - Implementierungsdiagramm - Verteilungsdiagramm - Einsatzdiagramm - Betriebskonzept	Begründung der Artefakte (Architekturmuster, Betriebskonzept usw.) aus den Architektur NFRs	
			QS	- Softwaredokumentation - Lesetechnik	- Problem-/Fehler-/Korrekturliste - Korrigierte SW-Dokumentation		**Inspektion**

Management	Phasen		Entscheidungsebenen	Dokumentation			QS-Tätigkeiten
				Artefakte		Rationale	
				IN	OUT		
Projektmanagement Änderungsmanagement Konfigurationsmanagement	Analyse, Wissenserwerb, Konfliktlösung	**Feinentwurf**	QS	- QS-Plan - Architekturdefinition - Analyseklassendiagramm	- Integrationstestplan - Komponententestplan	Begründung des Testplans	**Testplan-erstellung**
				- Analyseklassendiagramm - Architekturdefinition - Entwurfsmuster	- Entwurfklassendiagramm (EKD) - Package-, Klassen-, Operationsbeschreibungen	Begründung des Übergangs von AKD zu EKD Begründung der Implementierungs-vorschläge	
			QS	- Softwaredokumentation - Lesetechnik	- Problem-/Fehler-/Korrekturliste - Korrigierte SW-Dokumentation		**Inspektion**
				- Integrationstestplan - Sequenzdiagramme - Package-, Klassen-, Operationsbeschreibungen - Entwurfsklassen-Diagramm	- Integrationstestspezifikation - Erstimplementierung (d. h. Rahmenablauf in der Testumgebung implementieren)	Begründung der Testspezifikation	**Testspezifi-kation** **Testimple-mentierung**
				- Komponententestplan - Systemfunktionsbeschreibung, Package-, Klassen-, Operationsbeschreibungen und Operationssignatur aus Entwurfsklassen-Diagramm	- Komponententestspezifikation - Erstimplementierung (d. h. Rahmenablauf in der Testumgebung implementieren)	Begründung der Testspezifikation	**Testspezifi-kation** **Testimple-mentierung**

Management	Phasen		Entscheidungsebenen	Dokumentation			QS-Tätigkeiten
				Artefakte		Rationale	
				IN	OUT		
Projektmanagement Änderungsmanagement Konfigurationsmanagement	Analyse, Wissenserwerb, Konfliktlösung	**Implementierung**		- Entwurfsklassendiagramm - Package-, Klassen-, Operationsbeschreibungen - Implementierungsmuster	- Klassencode - Teilsystemcode - Systemcode	Begründung der Implementierung	
			QS	- Komponententestspezifikation - Erstimplementierung des Komponententests	- Komponententestcode	Begründung der Testimplementierungen	**Testimplementierung**
				- Klassencode - Komponententestcode	- Fehlerprotokoll		**Testdurchführung**
				- Fehlerprotokoll	- Fehlerursache - Geänderter Operations-/ Klassencode		**Debugging/ Ändern des Codes**
				- Integrationstestspezifikation - Erstimplementierung des Integrationstests	- Integrationstetcode		**Testimplementierung**
				- Teilsystemcode - Integrationstestcode	- Fehlerprotokoll		**Testdurchführung**
				- Fehlerprotokoll	- Fehlerursache - Geänderter Teilsystemcode		**Debugging/ Ändern des Codes**
				- Systemtestspezifikation - Erstimplementierung des Systemtests	- Systemtestcode		**Testimplementierung**
				- Systemcode - Systemtestcode	- Fehlerprotokoll		**Testdurchführung**
				- Fehlerprotokoll	- Fehlerursache - Geänderter Systemcode		**Debugging/ Ändern des Codes**

Anhang B – Code Komponente Teilnehmerregistrierung

anmeldezahlLesen.inc

```
<?php
//Bereits eingegangene Anmeldungen zählen (gesamt, staffel, jugendschueler, erwachsene)
include('config.inc');
$conn = mysqli_connect(dbhost,dbuser,dbpass) or die('Fehler beim Aufbau der Datenbankverbindung');
mysqli_select_db($conn,dbname);

//Wir wollen vorab wissen, ob noch Plätze frei sind
$sql = "SELECT COUNT(StarterID) as Gesamt FROM ".tabellenname."";
$query = mysqli_query($conn, $sql);
$datensatz = mysqli_fetch_array($query);
$gesamt = $datensatz['Gesamt'];

//Zwei Abfragen reichen für alle benötigten Werte
//#Alle Staffelstarter sollen keine Auswirkungen auf die freien Startplätze haben
$sql = "SELECT StarterID, Wettkampftyp FROM ".tabellenname." WHERE Wettkampftyp = 'Staffel'";
$query = mysqli_query($conn, $sql);
$staffel = 0;
while ($datensatz = mysqli_fetch_row($query)) {
   $staffel = $staffel + 1;
}
;
// #Alle Jugend und Schüler Anmeldungen
$sql = "SELECT StarterID, Wettkampftyp FROM ".tabellenname." WHERE Wettkampftyp != 'Erwachsene' AND Wettkampftyp != 'Staffel'";
$query = mysqli_query($conn, $sql);
$jugendschueler = 0;
while ($datensatz = mysqli_fetch_row($query)) {
   $jugendschueler = $jugendschueler + 1;
}
;
// #Erwachsenen (Von Anmeldezahl Jugend und Staffeln abziehen)
$erwachsene = $gesamt - $staffel -$jugendschueler;
mysqli_close($conn);
?>
```

auswahlhilfe.inc

```
<select name='Jahr'>
<option value selected>Bitte w&auml;hlen</option>
<option value='2008'<?php
    if ($geburtsdatum == "2008") {
      echo " selected=\"selected\"";
    }
  ?>>2008</option>
<option value='2007'<?php
    if ($geburtsdatum == "2007") {
      echo " selected=\"selected\"";
    }
 // ....................................Gekürzt.....................................................//
  ?>>1933</option>
<option value='1932'<?php
    if ($geburtsdatum == "1932") {
      echo " selected=\"selected\"";
    }
  ?>>1932</option>
<option value='1931'<?php
    if ($geburtsdatum == "1931") {
      echo " selected=\"selected\"";
    }
  ?>>1931</option>
</select>
```

bedingungen.html

```
<p>
<b>Bedingungen zum Triathlon:</b><br/>
Hier werden die Bedingungen stehen.....,
<br/>
</p>

<p>
<b>1) Streckenbeschreibung: </b> <br/>
<ul>
<li>
Schwimmen: Hallenbad Schifferstadt, auf 5 Bahnen, 5-6 Starter pro Bahn</li>
<li>Radfahren: 1 schnelle, große Runde teilweise Radwege, komplett asphaltiert, 20 km – bis auf eine Brücke absolut flach<br>
Schüler und Jugend: 1-4 flache schnelle Runden </li>
<li>Laufen: flacher Rundkurs, 3 Runden<br/>Schüler und Jugend: 1-2 flache Runden</li>
</ul>
</p>
<b>2) ... </b> <br/>
<ul>
<li>
...</li>
</ul>
</p>
```

config.inc

```
<?php
        define('dbhost', 'mysql.DLRG.de');
        define('dbuser', 'xxx');
        define('dbpass', 'xxx');
        define('dbname', 'xxx');

        // Golbale Konstanten
        define('tabellenname','Starter');
        define('startseite','http://www.dlrg.de/Gliederung/Rheinland-
Pfalz/Vorderpfalz/Schifferstadt/New folder/');
        define('jugendschuelerLimit', 999);
        define('erwachsenenLimit', 0);
        define('staffelLimit', 9999);
        define('jugendschuelerWartelistengroesse', 50);
        define('erwachsenenWartelistengroesse',1);
        define('staffelWartelistengroesse', 9999);

        define('txtSchuelerC', 'Sch&uuml;ler C (Jg. 2000/01): 0,1 - 2,5 - 0,4 km');
        define('txtSchuelerB', 'Sch&uuml;ler B (Jg. 1998/99): 0,2 - 5 - 1 km');
        define('txtSchuelerA', 'Sch&uuml;ler A (Jg. 1996/97): 0,4 - 10 - 2,5 km');
        define('txtJugendB', 'Jugend B (Jg. 1994/95):  0,4 - 10 - 2,5 km');
        define('txtJugendA','Jugend A (Jg. 1992/93):  0,4 - 20 - 5 km');
        define('txtStandard', 'Erwachsene: 0,4 - 20 - 5 km');
        define('txtStaffel', 'Mannschaft/Staffel <br />     (je 1er
Schwimmt/Fährt/Läuft):  0,4 - 20 - 5 km');
        define('txtDbFehler', 'Es liegt ein technisches Problem vor, wir bitten dies zu
entschuldigen! Bitte versuchen Sie es in ein paar Stunden noch einmal, oder
kontaktieren Sie uns!');
        define('txtWarteliste', 'Leider sind alle regulären Startplätze bereits belegt. Sie wurden
mit den aufgeführten Daten auf die Warteliste eingetragen! Erfahrungsgemäß
bestehen gute Chancen noch nachzurücken. Ihr Name ist noch nicht in der Starterliste
zu sehen, wir benachrichtigen Sie, sobald Sie nachrücken können!');
        define('txtNutzerbestaetigung', '<h1>Sind folgende Daten korrekt?</h1><br><h1>Zum
endgültigen Anmelden bitte erneut auf Absenden klicken oder vorher Daten
ändern</h1>');
?>
```

entferneSlashes.inc

```
<?php
        function entferneSlashesArray($a) {
         if (is_array($a)) {
            return array_map("entferneSlashesArray", $a);
          } else {
            return stripslashes($a);
          }
        }

         if (get_magic_quotes_gpc()) {
          $_POST = entferneSlashesArray($_POST);
          $_GET = entferneSlashesArray($_GET);
        }
?>
```

formular.inc

```
<?php
        //globale parameter
        include('config.inc');
?>
<hr>
<table>
<tr><td colspan=2><b>Anmeldeformular (Angaben mit * sind verpflichtend)</b></td></tr>
<form method='post' action='<?php echo $_SERVER['PHP_SELF']; ?>'>

<tr><td>
Vorname*</td><td><input type='text' size='40' maxlength='50' name='Vorname' value="<?php echo $vorname ?>"><br />
</td></tr><tr><td>
Nachname*</td><td><input type='text' size='40' maxlength='50' name='Nachname' value="<?php echo $nachname ?>"><br />
</td></tr><tr><td>
Geschlecht*
</td><td>
<select name='Geschlecht'>
<option value selected>Bitte w&auml;hlen</option>
 <option value="m"
 <?php
 //Vorselektierung beim erneuten Anzeigen des Formulars
  if ($geschlecht == "m") {
   echo " selected=\"selected\"";
  }
 ?>
 >Männlich</option>
 <option value="w"
 <?php
  //Vorselektierung beim erneuten Anzeigen des Formulars
  if ($geschlecht == "w") {
   echo " selected=\"selected\"";
  }
 ?>
 >Weiblich</option>
</select>
</td></tr><tr><td>
Geburtsjahr*</td><td>
```

```
<?php
// Auswahlhilfe für das Geburtsdatum
include('auswahlhilfe.inc');
?>


</td></tr>
<tr><td>
Verein/Team</td><td><input type='text' size='40' maxlength='50' name='Verein' value="<?php echo $verein ?>"><br />
</td></tr><tr><td>
PLZ*</td><td><input type='text' size='5' maxlength='5' name='PLZ' value="<?php echo $postleitzahl ?>"><br/>
</td></tr><tr><td>
Ort*</td><td><input type='text' size='40' maxlength='50' name='Ort' value="<?php echo $ort ?>"><br/>
</td></tr><tr><td>
Straße*</td><td><input type='text' size='40' maxlength='50' name='Strasse' value="<?php echo $strasse ?>"><br/>
</td></tr><tr><td>
Hausnummer*</td><td><input type='text' size='5' maxlength='9' name='Hausnummer' value="<?php echo $hausnummer ?>">
</td></tr><tr><td>
Telefon*</td><td><input type='text' size='20' maxlength='20' name='Telefon' value="<?php echo $telefon ?>">
</td></tr><tr><td>
E-Mail*</td><td><input type='text' size='40' maxlength='40' name='EMAIL' value="<?php echo $email ?>">
</td></tr>



<?php
// reduziert Auswahlmöglichkeit der Radiobuttons
//include('config.inc');
include('anmeldezahlLesen.inc');

// reduziert Auswahlmöglichkeit der Radiobuttons
                echo "<tr><td colspan='2'> </td></tr>
                <tr><td colspan='2'>
                <b>Wenn hier Auswahlmöglichkeiten fehlen, ist leider auch schon die Warteliste voll...<br/>
                Für alle Möglichkeiten, die angeboten werden, kann man sich natürlich noch anmelden</b>
                </td></tr>";
                echo "<tr><td>
```

```
Triathlontyp*
</td><td>Bitte Auswählen:</td></tr>";

//Standard noch nicht voll => einblenden
if ($erwachsene <= erwachsenenLimit + erwachsenenWartelistengroesse)
{
        echo "<tr><td></td><td>
        <input type='radio' value='Erwachsene' name='Wettkampftyp'";
                if ($wettkampftyp == "Erwachsene") {
                        echo "checked=\"checked\" ";
                }
        echo ">";
         echo txtStandard;
        echo"
        </td></tr>";
}

//Staffel noch nicht voll => einblenden
if ($staffel <= staffelLimit + staffelWartelistengroesse)
{
        echo "<tr><td></td><td>
        <input type='radio' value='Staffel' name='Wettkampftyp'";
        if ($wettkampftyp == "Staffel") {
                        echo "checked=\"checked\" ";
                }
        echo ">";
         echo txtStaffel;
        echo"
        </td></tr>";
}

//Jugend noch nicht voll => einblenden
if ($jugendschueler <= jugendschuelerLimit + jugendschuelerWartelistengroesse )
{

        echo"
        <tr><td></td><td>
        <input type='radio' value='JugendA' name='Wettkampftyp'";
                if ($wettkampftyp == "JugendA") {
                        echo "checked=\"checked\" ";
                }
```

```
echo ">";
 echo txtJugendA;

echo"
</td></tr><tr><td></td><td>
<input type='radio' value='JugendB' name='Wettkampftyp'";
            if ($wettkampftyp == "JugendB") {
                        echo "checked=\"checked\" ";
            }
echo ">";
 echo txtJugendB;

echo"
</td></tr><tr><td></td><td>
<input type='radio' value='SchuelerA' name='Wettkampftyp'";
            if ($wettkampftyp == "SchuelerA") {
                        echo "checked=\"checked\" ";
            }
echo ">";
 echo txtSchuelerA;

echo"
</td></tr><tr><td></td><td>
<input type='radio' value='SchuelerB' name='Wettkampftyp'";
            if ($wettkampftyp == "SchuelerB") {
                        echo "checked=\"checked\" ";
            }
echo ">";
 echo txtSchuelerB;


//jeweils mit Vorfüllung und Textbeschreibung aus Configvariable
echo "</td></tr><tr><td></td><td>
<input type='radio' value='SchuelerC' name='Wettkampftyp'";
            if ($wettkampftyp == "SchuelerC") {
                        echo "checked=\"checked\" ";
            }
echo ">";
 echo txtSchuelerC;

echo"
```

```
                    </td></tr>";
                }
?>

<tr><td>Anmerkungen/Kommentar</td><td><textarea rows='5' cols='40' name='Kommentar' wrap='physical'><?php echo $kommentar ?></textarea>
</td></tr>
<tr><td></td><td></td></tr>

<tr><td colspan=2>
<b>Staffelstarter(nur bei Staffelstart angeben!)</b> <br/>
Diese Angaben sind lediglich für uns, bitte unbedingt drei Namen eintragen, dann haben wir einen besseren Überblick (ihr könnt ja später trotzdem jemand auswechseln), Danke!
</td><td></td></tr>
<tr>
<td colspan=2>Staffelstarter1</td></tr>
<tr>
<td>Vorname1: <input type='text' size='15' maxlength='25' name='Vorname1' value="<?php echo $vorname1 ?>"></td><td>Nachname1: <input type='text' size='15' maxlength='24' name='Nachname1' value="<?php echo $nachname1 ?>">
</td></tr>

<tr><td colspan=2>Staffelstarter2</td></tr><tr><td>Vorname2: <input type='text' size='15' maxlength='25' name='Vorname2' value="<?php echo $vorname2 ?>"></td><td>Nachname2: <input type='text' size='15' maxlength='24' name='Nachname2' value="<?php echo $nachname2 ?>">
</td></tr><tr><td colspan=2>Staffelstarter3</td></tr><tr><td>Vorname3: <input type='text' size='15' maxlength='25' name='Vorname3' value="<?php echo $vorname3 ?>"></td><td>Nachname3: <input type='text' size='15' maxlength='24' name='Nachname3' value="<?php echo $nachname3 ?>">
</td></tr>
<tr><td><b>Anmeldung abschicken:</b> <input type='submit' value='Abschicken' id='submit' name='Submit'><td></td></tr>
<input type='hidden' name='NutzerOk' value="<?php echo $nutzerok ?>"
</form>
</table>
```

index.inc

```
<?php
// Konstanten einbinden
include ('config.inc');

// Magic-quotes beandeln, falls nötig
include ('entferneSlashes.inc');

//prüfen welche Felder noch fehlen und markieren. Wenn ein Feld Fehlt => $ok auf False zurücksetzen, dann greift der Else-Block Fehlerfelder unten
  // Inhalte der Felder aus POST holen
  $vorname = (isset($_POST["Vorname"])) ? $_POST["Vorname"] : "";
  $nachname = (isset($_POST["Nachname"])) ? $_POST["Nachname"] : "";
  $geburtsdatum = (isset($_POST["Jahr"])) ? $_POST["Jahr"] : "";
  $geschlecht = (isset($_POST["Geschlecht"])) ? $_POST["Geschlecht"] : "";
  $verein = (isset($_POST["Verein"])) ? $_POST["Verein"] : "";
  $postleitzahl = (isset($_POST["PLZ"])) ? $_POST["PLZ"] : "";
  $ort = (isset($_POST["Ort"])) ? $_POST["Ort"] : "";
  $strasse = (isset($_POST["Strasse"])) ? $_POST["Strasse"] : "";
  $hausnummer = (isset($_POST["Hausnummer"])) ? $_POST["Hausnummer"] : "";
  $telefon = (isset($_POST["Telefon"])) ? $_POST["Telefon"] : "";
  $email = (isset($_POST["EMAIL"])) ? $_POST["EMAIL"] : "";
  $wettkampftyp = (isset($_POST["Wettkampftyp"])) ? $_POST["Wettkampftyp"] : "";
  $kommentar = (isset($_POST["Kommentar"])) ? $_POST["Kommentar"] : "";
  $vorname1 = (isset($_POST["Vorname1"])) ? $_POST["Vorname1"] : "";
  $nachname1 = (isset($_POST["Nachname1"])) ? $_POST["Nachname1"] : "";
  $vorname2 = (isset($_POST["Vorname2"])) ? $_POST["Vorname2"] : "";
  $nachname2= (isset($_POST["Nachname2"])) ? $_POST["Nachname2"] : "";
  $vorname3 = (isset($_POST["Vorname3"])) ? $_POST["Vorname3"] : "";
  $nachname3 = (isset($_POST["Nachname3"])) ? $_POST["Nachname3"] : "";
  $starter1 =  ucfirst($nachname1).' '.ucfirst($vorname1);
  $starter2 =  ucfirst($nachname2).' '.ucfirst($vorname2);
  $starter3 =  ucfirst($nachname3).' '.ucfirst($vorname3);

  $nutzerok = (isset($_POST["NutzerOk"])) ? $_POST["NutzerOk"] : "";

  $ok = false;

 //Nicht gefüllte Pflichtfelder auslesen
  $fehlerfelder = array();
  if (isset($_POST["Submit"])) {
```

```
$ok = true;
if (!isset($_POST["Vorname"]) ||
    trim($_POST["Vorname"]) == "") {
  $ok = false;
  $fehlerfelder[] = "Vorname";
}

if (!isset($_POST["Nachname"]) ||
    trim($_POST["Nachname"]) == "") {
  $ok = false;
  $fehlerfelder[] = "Nachname";
}


if (!isset($_POST["Jahr"]) ||  $_POST["Jahr"] == ""){
  $ok = false;
  $fehlerfelder[] = "Geburtsjahr";
}

    if (!isset($_POST["Geschlecht"]) || $_POST["Geschlecht"] == "") {
  $ok = false;
  $fehlerfelder[] = "Geschlecht";
}

if (!isset($_POST["PLZ"]) ||
    trim($_POST["PLZ"]) == ""){
  $ok = false;
  $fehlerfelder[] = "Postleitzahl";
}


if (!isset($_POST["Ort"]) ||
trim($_POST["Ort"]) == ""){
  $ok = false;
  $fehlerfelder[] = "Ort";
}
if (!isset($_POST["Strasse"]) ||
trim($_POST["Strasse"]) == ""){
  $ok = false;
  $fehlerfelder[] = "Straße";
}
```

```
  if (!isset($_POST["Hausnummer"]) ||
      trim($_POST["Hausnummer"]) == "") {
    $ok = false;
    $fehlerfelder[] = "Hausnummer";
  }


  //TODO: einfache regex-prüfung des Mailformats
  if (!isset($_POST["EMAIL"]) ||
      trim($_POST["EMAIL"]) == "") {
    $ok = false;
    $fehlerfelder[] = "E-Mail";
  }

      if (!isset($_POST["Telefon"]) ||
      trim($_POST["Telefon"]) == "") {
    $ok = false;
    $fehlerfelder[] = "Telefon";
  }

          if (!isset($_POST["Wettkampftyp"]) ||
      trim($_POST["Wettkampftyp"]) == "") {
    $ok = false;
    $fehlerfelder[] = "Wettkampftyp";
  }


// Aus sicherheitsgründen Eingabe vorprozessieren, muss hier geschehen, da sonst ggf beim
Anzeigen der Daten
// ein Problem entstehen kann


  $vorname = htmlspecialchars($vorname);
  $nachname = htmlspecialchars($nachname);
  $geburtsdatum = htmlspecialchars($geburtsdatum);
  $geschlecht = htmlspecialchars($geschlecht);
  $verein = htmlspecialchars($verein);
  $postleitzahl = htmlspecialchars($postleitzahl);
  $ort = htmlspecialchars($ort);
  $strasse = htmlspecialchars($strasse);
  $hausnummer = htmlspecialchars($hausnummer);
  $telefon = htmlspecialchars($telefon);
```

```
  $email = htmlspecialchars($email);
  $wettkampftyp = htmlspecialchars($wettkampftyp);
  $kommentar = htmlspecialchars($kommentar);
  $starter1 = htmlspecialchars($starter1);
  $starter2 = htmlspecialchars($starter2);
  $starter3 = htmlspecialchars($starter3);



if ($ok && (!$nutzerok)) {

echo txtNutzerbestaetigung;

//formulardaten
  echo "<b>Name:</b> $vorname  ";
  echo "$nachname<br>";
  echo "<b>Geburtsjahr:</b> $geburtsdatum   ";
  echo "<b>Geschlecht:</b> $geschlecht<br>";
  echo "<b>Verein/Team:</b> $verein<br>";
  echo "<b>Adresse:</b><br>";
  echo "$postleitzahl   $ort<br>";
  echo "$strasse  ";
  echo "$hausnummer<br>";
  echo "<b>Telefon:</b> $telefon<br>";
  echo "<b>E-mail:</b> $email<br>";
  echo "<b>Wettkampftyp: </b> $wettkampftyp<br>";
  echo "<b>Starter 1:</b> $starter1 - ";
  echo "<b>Starter 2:</b> $starter2 - ";
  echo "<b>Starter 3:</b> $starter3<br>";

$nutzerok = true;

//formular direkt zum ändern anzeigen
 include ('formular.inc');


   }
elseif ($ok && $nutzerok){

                    //Bereits eingegangene Anmeldungen nochmals zählen (gesamt, staffel,
jugendschueler, erwachsene)
                    //include('anmeldezahlLesen.inc');
```

```
                // Alles Daten drin, DB-Verbindung aufbauen
                include('config.inc');
                $conn = mysqli_connect(dbhost, dbuser, dbpass) or die ('Error
connecting to mysql');
                mysqli_select_db($conn, dbname);

                //Standardanmeldungszahl überschritten => Wartelistenplatz
                if (($erwachsene >= erwachsenenLimit) && ($wettkampftyp == 'Erwachsene') &&
($erwachsene <= (erwachsenenLimit + erwachsenenWartelistengroesse)))
                {
                        include('wartelisteneintrag.inc');
                }

                //Jugendschueler Anmeldungszahl überschritten =>Wartelistenplatz (Jugend und
Schueler zusammen)
                elseif ($jugendschueler >= jugendSchuelerLimit && $wettkampftyp != 'Erwachsene'
&& $wettkampftyp != 'Staffel' && ($jugendschueler <= (jugendschuelerLimit +
jugendschuelerWartelistengroesse)))
                {
                        include('wartelisteneintrag.inc');
                }
                else
                {
                include('standardeintrag.inc');
                }

                //Starterliste immer anzeigen
                include('starterliste.inc');

mysqli_close($conn);



}
        // Else-Block Fehlerfelder
        else{
    echo "<b>Sie haben die folgenden Pflichtfelder nicht ausgefüllt: </b>";
    echo "<ul><li>";
    echo implode("</li><li>", $fehlerfelder);
    echo "</li></ul>";
        echo "<b>Bitte korrigieren sie dies und senden sie das Formular erneut ab! Danke!  </b>";
        include('formular.inc');
```

```
   }

//Bis zu dieser Klammer wird nur nach vorherigem senden ausgeführt
 }


//Seite wurde noch nicht gesendet => Startseite
 else {
                    echo "<table><tr><td>";
                    include('bedingungen.html');
                    echo"</tr></td></table>";
                    include('formular.inc');
          }
?>

      <!-- Ende Seiteninhalt -->
```

index.php

```
<html>
<head>
</head>
<body>
<?php echo" Triathlon 'Goldener Hut' am Samstag, 28.08.2010 in Schifferstadt</br>";


?>
<?php echo" Alles was sie hier sehen ist nur ein Systemtest, alle Anmeldungen werden Gelöscht! Die Anmeldefrist beginnt erst mitte bis ende Mai!";
?>


<p style='text-align: left'>
<img src='./bild/LogoTriathlon.png' width='400'/>
</p>


<?php
include('index.inc');
?>


</body>
</html>
```

standardeintrag.inc

```
<?php
	//Weiter wie gehabt
		// Zeitstempel erstellen
		$timestamp = time();
		$datum = date("Y.m.d",$timestamp);
		$uhrzeit = date("H:i",$timestamp);
		$mytime = $datum.$uhrzeit;

	//Wir speichern fürs erste ein Datum in die Datenbank um zukünftig flexibler zu sein
	$geburtsdatum = $geburtsdatum."-01-01";

		$sql = "INSERT INTO ".tabellenname." (
		Nachname,
		Vorname,
		Geburtsdatum,
		Geschlecht,
		Postleitzahl,
		Ort,
		Strasse,
		Hausnummer,
		Email,
		Telefon,
		Wettkampftyp,
		Verein,
		Kommentar,
		Starter1,
		Starter2,
		Starter3,
		Flex)
		VALUES (
		'".ucfirst(mysqli_real_escape_string($conn, $nachname )).'',
		'".ucfirst(mysqli_real_escape_string($conn, $vorname )).'',
		'".mysqli_real_escape_string($conn, $geburtsdatum ).'',
		'".mysqli_real_escape_string($conn, $geschlecht ).'',
		'".mysqli_real_escape_string($conn, $postleitzahl ).'',
		'".ucfirst(mysqli_real_escape_string($conn, $ort )).'',
		'".ucfirst(mysqli_real_escape_string($conn, $strasse )).'',
		'".mysqli_real_escape_string($conn, $hausnummer ).'',
		'".mysqli_real_escape_string($conn, $email ).'',
		'".mysqli_real_escape_string($conn, $telefon ).'',
```

```
'".mysqli_real_escape_string($conn, $wettkampftyp )."',
'".ucfirst(mysqli_real_escape_string($conn, $verein ))."',
'".mysqli_real_escape_string($conn, $kommentar )."',
'".ucfirst(mysqli_real_escape_string($conn, $starter1 ))."',
'".ucfirst(mysqli_real_escape_string($conn, $starter2 ))."',
'".ucfirst(mysqli_real_escape_string($conn, $starter3 ))."',
'".mysqli_real_escape_string($conn, $mytime )."'
)";

mysqli_query($conn, $sql);

// Pruefen ob der neue Datensatz tatsaechlich eingefuegt wurde
if (mysqli_affected_rows($conn) == 1)
 {
        echo "<b>Name:</b> $vorname  ";
        echo "$nachname<br>";
        echo "<b>Geburtsjahr:</b> $geburtsdatum   ";
        echo "<b>Geschlecht:</b> $geschlecht<br>";
        echo "<b>Verein/Team:</b> $verein<br>";
        echo "<b>Adresse:</b><br>";
        echo "$postleitzahl   $ort<br>";
        echo "<b>Wettkampftyp: </b> $wettkampftyp<br>";
        echo "<b>Starter 1:</b> $starter1 - ";
        echo "<b>Starter 2:</b> $starter2 - ";
        echo "<b>Starter 3:</b> $starter3<br>";
        echo"<p style='text-align: left'><a href='".$startseite."'>Zur&uuml;ck zur
Startseite</a></p>";
}
else
{

                echo "<b>Fehler beim Schreiben in die Datenbank, bitte
benachrichtigen sie webmaster@dlrg-schifferstadt.de</b>".mysqli_error()."<br>\n";


}
?>
```

standardeintrag.inc

```
<?php
		include('config.inc');
		mysqli_select_db($conn,$dbname);


		$query = mysqli_query($conn, "SELECT UPPER(SUBSTRING(Vorname
FROM 1 FOR 1)),Nachname, Verein,Ort, Wettkampftyp, Cupwertung FROM Starter ORDER BY
Wettkampftyp, Ort");


		//$datensatz = mysqli_fetch_array($query);


			echo "<h1>Automatisch generierte, unbestätigte Starterliste</h1>";


		echo "<table>";
	echo "<tr><td colspan='5'><b>Sortiert nach Wettkampftyp und Wohnort<br>
					Wenn ihr euch nicht findet, einfach mal im Browser auf
'aktualisieren' klicken! </b><br/>Wartelisteneinträge werden nicht angezeigt! Wir sind aber
zuversichtlich, dass wir einige Nachrücker unterbringen</td></tr>";
	echo "<tr><td
colspan='2'>Name</td><td>Verein/Team</td><td>Wohnort</td><td>Wettkampftyp</td></tr>";




	echo "<tr><td colspan='5'><hr/></td></tr>";


			while($datensatz = mysqli_fetch_row($query))
			{
				// Nur bestätigte Starter anzeigen
				if ($datensatz[5] != 'Warteliste')
				{
				 echo "<tr>";


				 echo "<td>".$datensatz[0].".";
				 echo "</td><td>".$datensatz[1];
				 echo "</td><td>".$datensatz[2];
				 echo "</td><td>".$datensatz[3];
				 echo "</td><td>".$datensatz[4]."</td></tr>";
				};
			};
			echo "</table>";
		mysqli_close($conn);
	?>
```

wartelisteneintrag.inc

```
<?php
                // Zeitstempel erstellen
                $timestamp = time();
                $datum = date("Y.m.d",$timestamp);
                $uhrzeit = date("H:i",$timestamp);
                $mytime = $datum.$uhrzeit;
                $cupwertung = 'Warteliste';

                //Wir speichern fürs erste ein Datum in die Datenbank um zukünftig flexibler zu sein
                $geburtsdatum = $geburtsdatum."-01-01";

                $sql = "INSERT INTO ".tabellenname."(
                Nachname,
                Vorname,
                Geburtsdatum,
                Geschlecht,
                Postleitzahl,
                Ort,
                Strasse,
                Hausnummer,
                Email,
                Telefon,
                Wettkampftyp,
                Cupwertung,
                Verein,
                Kommentar,
                Starter1,
                Starter2,
                Starter3,
                Flex)
                VALUES (
                '".ucfirst(mysqli_real_escape_string($conn, $nachname )).'',
                '".ucfirst(mysqli_real_escape_string($conn, $vorname )).'',
                '".mysqli_real_escape_string($conn, $geburtsdatum ).'',
                '".mysqli_real_escape_string($conn, $geschlecht ).'',
                '".mysqli_real_escape_string($conn, $postleitzahl ).'',
                '".ucfirst(mysqli_real_escape_string($conn, $ort )).'',
                '".ucfirst(mysqli_real_escape_string($conn, $strasse )).'',
                '".mysqli_real_escape_string($conn, $hausnummer ).'',
```

```
'".mysqli_real_escape_string($conn, $email )."',
'".mysqli_real_escape_string($conn, $telefon )."',
'".mysqli_real_escape_string($conn, $wettkampftyp )."',
'$cupwertung',
'".ucfirst(mysqli_real_escape_string($conn, $verein ))."',
'".mysqli_real_escape_string($conn, $kommentar )."',
'".ucfirst(mysqli_real_escape_string($conn, $starter1 ))."',
'".ucfirst(mysqli_real_escape_string($conn, $starter2 ))."',
'".ucfirst(mysqli_real_escape_string($conn, $starter3 ))."',
'".mysqli_real_escape_string($conn, $mytime )."'
)";

mysqli_query($conn, $sql);

// Pruefen ob der neue Datensatz tatsaechlich eingefuegt wurde
if (mysqli_affected_rows($conn) == 1)
 {
        echo "<b>Alle Startplätze sind bereits Vergeben - Sie wurden auf die
Warteliste gesetzt! Sorry!</b><br>Erfolgreich in die Warteliste eingetragen als: <br/>";
        echo "<b>Name:</b> $vorname  ";
        echo "$nachname<br>";
        echo "<b>Geburtsjahr:</b> $geburtsdatum   ";
        echo "<b>Geschlecht:</b> $geschlecht<br>";
        echo "<b>Verein/Team:</b> $verein<br>";
        echo "<b>Adresse:</b><br>";
        echo "$postleitzahl   $ort<br>";
        echo "<b>Wettkampftyp: </b> $wettkampftyp<br>";
        echo "<b>Starter 1:</b> $starter1 - ";
        echo "<b>Starter 2:</b> $starter2 - ";
        echo "<b>Starter 3:</b> $starter3<br>";
        echo"<p style='text-align: left'><a href='".startseite."'>Zurück zur
Startseite</a></p>";          }
else
{
            echo "<b>Fehler beim Schreiben in die Datenbank, bitte
benachrichtigen sie webmaster@dlrg-schifferstadt.de</b>".mysqli_error()."<br>\n";

}
?>
```