1 Gesamtsystem

1.1 Umgebung des Produktes

Bei dem zu entwickelnden System handelt es sich um ein fensterorientiertes Softwaresystem, daß sowohl unter dem Betriebssystem SUN SOLARIS als auch unter MS Windows95 lauffähig ist. Neben der Bildschirmausgabe werden auch gedruckte Listen und Pläne angeboten, außerdem werden Daten auf Festplatte gespeichert. Die Betriebssystemunabhängigkeit dieser Komponenten wird durch die Verwendung der Programmiersprache JAVA erreicht.

Ferner wird zur vollen Leistungsfähigkeit des Systems das Programm „Map&Guide“ benötigt, welches die Berechnung von Entfernungen und kürzesten Wegen zwischen beliebigen Adressen im Raum Stuttgart übernimmt. Der Datenaustausch zwischen beiden Produkten findet über die zu „Map&Guide“ gehörige Batch-Schnittstelle statt. Für den effizienten Zugriff werden die Ergebnisse von „Map&Guide“ in einer Tabelle hinterlegt, welche auch von Hand gepflegt werden kann, falls „Map&Guide“, das nur auf Windows läuft, nicht verfügbar ist. Außerdem wird durch die Anbindung mit der Tabelle die Austauschbarkeit des Verkehrstools gewährleistet.

1.2 Warum Java?

Wichtigstes Kriterium bei der Wahl der zu verwendenden Programmiersprache war die Verfügbarkeit sowohl für die UNIX-Systeme an der Universität als auch für den Windows95-PC beim DRK. Darüberhinaus durfte die Anschaffung eines Compilers sowie einer passenden Entwicklungsumgebung nur mit geringen Kosten verbunden sein.

In der engeren Wahl standen:

- C++ mit der Klassenbibliothek wxWindows (die eine Quellcodekompatible Programmierung der Benutzeroberfläche erlaubt)
- Das Smalltalk-Entwicklungssystem VisualWorks
- Java (momentan in der Version JDK 1.1)

Die Kandidaten wurden anhand verschiedener Kriterien charakterisiert und verglichen:

Portabilität C++/wxWindows läßt sich als Quellcode auf UNIX und Windows compilieren. Smalltalk und Java werden in einen hardwareunabhängigen Bytecode übersetzt.

Kosten Die Compiler für C++ und Java sind kostenlos erhältlich, für VisualWorks sind einige Lizenzen am Institut vorhanden.

Entwicklungsumgebung VisualWorks ist eine integrierte Entwicklungsumgebung, die insbesondere auch einen passenden GUI-Builder enthält. Für wxWindows gibt es einen GUI-Builder als Freeware. Für Java gibt es eine Vielzahl von Entwicklungsumgebungen, davon auch einige Freeware-Programme.

C++-Anbindung Da möglicherweise Algorithmen zum Verwalten und Durchsuchen von Graphen aus der in C++ geschriebenen Bibliothek LEDA verwendet werden sollen, wäre eine Einbindungsmöglichkeit wünschenswert.

In C++ geht dies natürlich. Sowohl Java als auch Smalltalk bieten eine Schnittstelle, um Funktionen aus C-Bibliotheken aufzurufen. Möglicherweise gibt es auch entsprechende in Java geschrieben Bibliotheken.

Bibliotheken für Standardfunktionen Damit die Projektgruppe das Rad nicht zweimal erfinden muß, sollten für Standardfunktionen (z.B. Verwaltung von Listen, vielleicht auch Zeichnen von Graphen) möglichst kostenlose Bibliotheken existieren.

VisualWorks enthält bereits eine sehr umfangreiche Klassenbibliothek. Weitergehende Bibliotheken dürften kostenlos kaum zu bekommen sein. Zu C++ existiert die Bibliothek LEDA, die einen Teil dieser Funktionen abdeckt. Die meisten frei verfügbaren Bibliotheken oder Problemlösungen gibt es vermutlich für das vor allem im Internet stark verbreitete Java.

Die Entscheidung der Projektgruppe fiel auf die Sprache Java. Dies ist eine leicht zu erlernende objekt-hybride Programmiersprache. Sie kommt im Gegensatz zu C++ ohne Zeiger aus, was eine Reihe schwierig zu entdeckender Fehler von vornherein ausschließt. Die Programmierung einer graphischen Benutzungsoberfläche ist Bestandteil der Sprachdefinition. Da seit Erfindung dieser Sprache ein regelrechter Java-Boom herrscht, existiert im Internet eine große Zahl frei verfügbarer Bibliotheken für Java, die oft benötigte Funktionen bereits fertig zur Verfügung stellen.

Das Dokumentieren wird unterstützt durch spezielle Java-Kommentare im Quellcode, aus denen automatisch ein Teil der Dokumentation erzeugt werden kann. Außerdem besitzen mehrere Mitglieder der Projektgruppe bereits Erfahrungen in der Java-Programmierung, so daß weniger Einarbeitungsaufwand erforderlich ist und Probleme bei der Programmierung bereits innerhalb der Projektgruppe behoben werden können.

1.3 Verwendete Tools

Das System wird, wie bereits in den Kapiteln 1.1 und 1.2 erwähnt, unter Verwendung des „Java Development Kit„ JDK 1.1 auf SUN SOLARIS entwickelt. Zusätzlich werden noch folgende Tools im Projektverlauf bzw. im laufenden System eingesetzt:

CVS Zur Versionsverwaltung aller Dokumente einschließlich der Quellcodes. Zuständig für CVS-Fragen ist Frank Wagner.

ObjectTeam ist ein CASE-Tool. Es wird zum Entwurf des Datenmodells und der Klassen eingesetzt. Für die Verwendung von ObjectTeam ist Jörg Fleischmann zuständig.

MS-Project wird zur Planung und Beobachtung des Projektverlaufs benutzt. Da die Aufsicht über den Projektverlauf bei Lars Hermes liegt, fällt auch MS-Project in seine Zuständigkeit.

Map&Guide ist ein Verkehrstool mit einer Batch-Schnittstelle. Wir verwenden „Map&Guide“ wie in 1.1 erwähnt im wesentlichen zur Entfernungsbestimmung zwischen Adressen. Für die Anbindung von „Map&Guide“ an das System ist Frank Wagner verantwortlich.

1.4 Funktionalität

Das System gliedert sich grob in die vier Bereiche Datenverwaltung, Planung, Optimierung und Analyse.

1.4.1 Datenverwaltung

Mit dem System können die Stammdaten für Kunden, Mitarbeiter und Fahrzeuge verwaltet werden. Dazu ist es möglich benötigte Daten wie Adressen, Bankverbindungen, Kommunikationsverbindungen, Fahrzeugausstattungen etc. einzugeben und getrennt nach Klassen zu speichern. Zur Datenspeicherung wird auf eine Datenbank verzichtet, da es im Rahmen des Projektes Transportoptimierung zu aufwendig wäre, eine Datenbankanbindung an das System zu schreiben. Die Schnittstelle an die Datenhaltung soll jedoch so offen gehalten werden, daß eine spätere Anbindung an eine Datenbank möglich ist, um die Daten persistent zu speichern, falls sich die Gegebenheiten ändern. Neben den Stammdaten beinhaltet die Datenverwaltung auch Dienstwünsche, geplante (Unter-)Touren, sowie eine Historie der erledigten Fahrten zur späteren Abrechnung, oder für Analysezwecke.

Die Datenerfassung wird in Kapitel 7 beschrieben.

1.4.2 Planung

Die Planung von Touren, Untertouren, Fahrten und Dienstplänen kann sowohl manuell als auch automatisch erfolgen, wobei die automatische Planung an jeder Stelle durch manuelle Änderungen an spezielle Bedürfnisse angepaßt werden kann.

Die automatische Planung verläuft in mehreren Schritten:

1. Aus den Dienstwünschen werden Touren zusammengestellt.
2. Touren werden zu Untertouren konkretisiert.
3. Aus Untertouren werden konkret stattfindende Fahrten abgeleitet.
4. Aus Fahrten werden Dienstpläne erstellt.

Dabei ist zu beachten, daß eine komplette Planung selten durchgeführt wird. In der Regel werden Dienstwünsche oder (Unter-)Touren in das bestehende Tourensystem eingefügt. Dies bedeutet, daß die verschiedenen Planungsschritte auch einzeln angestoßen werden können. Ferner ist es möglich ganze (Unter-)Touren oder einzelne Dienstwünsche von der weiteren Optimierung auszunehmen. Ausgenommene Teile bleiben also unbeschadet nach der Optimierungen bestehen.

Wozu dient die Unterscheidung von Tour, Untertour und Fahrt? Eine Tour faßt zunächst Dienstwünsche zusammen, die grundsätzlich miteinander bzw. nacheinander erledigt werden können, sagt jedoch noch nichts über die Reihenfolge der Dienstwünsche (und damit der Stationen) oder die Abfahrtszeit aus.

Dies ist nötig, da Kunden unabhängig von der Uhrzeit oder ob es sich um ein Hin- oder Rückfahrt handelt, immer in gleichen Gruppen befördert werden sollen. Außerdem wissen die Mitarbeiter anhand der Tournummer schon, welche Kunden sie abholen müssen.

Mit der Untertour wird dann die Dienstwunsch- bzw. Stationenreihenfolge und die Uhrzeit festgelegt, zu der eine Tourinstanz gefahren werden soll. Damit ist es also möglich, Hin- und Rückfahrten zu gestalten. Außerdem kann mit einer Untertour dargestellt werden, daß die Gruppe nicht unbedingt vollständig befördert werden muß, fehlt eine Person (regelmäßig), ändert das nur etwas an der Untertour, die Personengruppe der Tour bleibt jedoch bestehen.

Die Fahrt wird schließlich benötigt, um konkret gefahrenen Tourinstanzen darzustellen. Bei den Fahrten kann auch, da sie ein festes Datum haben, überprüft werden, ob die zugeordneten Mitarbeiter und das zugordnete Fahrzeug überhaupt verfügbar sind, und ob alle Kunden deren Dienstwünsche zur Erledigung anstehen die Ausführung wünschen. Ist beispielsweise Kunde X im Urlaub wird dessen Dienstwunsch nicht in der Fahrt berücksichtigt, ohne daß dies weitere Auswirkungen auf die bestehende (Unter-)Touren hat.

1.4.3 Optimierung

Bei dem System gibt es zwei Optimierungsansätze: die *lokale Optimierung* 5 zur Optimierung innerhalb einer Dienstart und die *inkrementelle Optimierung* 6 zum Einfügen neuer Dienstwünsche und (Unter-)Touren in ein bestehendes Tourensystem, sowie zur Optimierung einer gegebenen Menge von Touren. Auf eine *globale Optimierung* wurde verzichtet, da diese durch gezieltes Anwenden der lokalen Optimierung auf alle Dienstarten erreicht werden kann. Dem Anwender bleibt damit die Möglichkeit offen, jeweils neu festzulegen, welche Dienstart bei seiner aktuellen globalen Optimierung welche Priorität haben soll.

1.4.4 Analyse und Datenausgabe

Die Analyse und Datenausgabe dient als Grundlage zur Dienst- und Infrastrukturplanung, sowie als Überblick über den aktuellen Planungsstand.

Die Datenausgabe wird Listen zu den Stammdaten, sowie den Druck von Dienstpländen beinhalten. Zudem ermitteln die Analysefunktionen des Systems die Auslastung von Mitarbeitern und Fahrzeugen, so daß freie oder fehlende Kapazitäten aufgedeckt werden können. Eine genaue Beschreibung der Analysefunktionen können dem Kapitel 4.2 entnommen werden.

1.5 Entwicklungsphilosophie

Bei der Entwicklung des Systems wird hierarchisch vorgegangen. Dies bedeutet, daß wir einzelne Module in der Reihenfolge ihrer Wichtigkeit immer so implementieren, daß das System lauffähig ist. Dadurch können die einzelnen Module schon frühzeitig getestet werden, außerdem haben wir immer ein einsatzfähiges System, dessen Funktionalität mit der Zeit zunimmt. Wir werden zunächst die Dateneingabe (incl. manueller Tourplanung) und Datenverwaltung implementieren, anschließend Konstistenztests für die manuelle Eingabe hinzufügen und am Ende das System durch eine automatische Planung abrunden.

Durch den modularen Aufbau kann das System auch leicht an neue oder veränderte Gegebenheiten angepaßt werden. Als mögliche Erweiterungen wären ein Abrechnungsmodul zur automatischen Rechnungsstellung erledigter Dienstwünsche, sowie ein Personalplanungsmodul zur automatischen stundengenauen Abrechnung der Arbeitszeiten vorstellbar.

2 Datenmodell

Da wir zur Programmierung die objektorientierte Programmiersprache JAVA verwenden, liegt es nahe, auch zur Erstellung des Datenmodells eine objektorientierte Methode heranzuziehen. Wir verwenden daher die *objektorientierte Analyse (OOA)*, die beispielsweise in [Bal96] beschrieben wird, und benutzen dazu das CASE-Tool ObjectTeam.

Da das Datenmodell schon unsere Klassen beinhaltet, stellt dieses Kapitel nur eine Momentaufnahme des aktuellen Standes dar. Die vollständige Spezifikation findet erst in der Entwurfsphase statt, in der noch die Datentypen, Wertebereiche, etc. festgelegt werden.

2.1 Klassendiagramme

Dieses Kapitel enthält die Klassendiagramme des Datenmodells. Diese Klassen dienen als Grundlage für die anschließende Entwurfsphase, werden dort jedoch nicht unbedingt eins-zu-eins in Java-Klassen umgesetzt.

Abbildung 1 stellt eine Kurzübersicht über die OMT-Notation dar, soweit sie in den nachfolgenden Diagrammen verwendet wurde. An dieser Stelle möchten wir uns bei Stefan Krauß aus der Abteilung Software-Engineering der Universität Stuttgart bedanken, der uns für diese Referenz seine in mühevoller Kleinarbeit erstellten xfig-Bilder zur Verfügung gestellt hat.

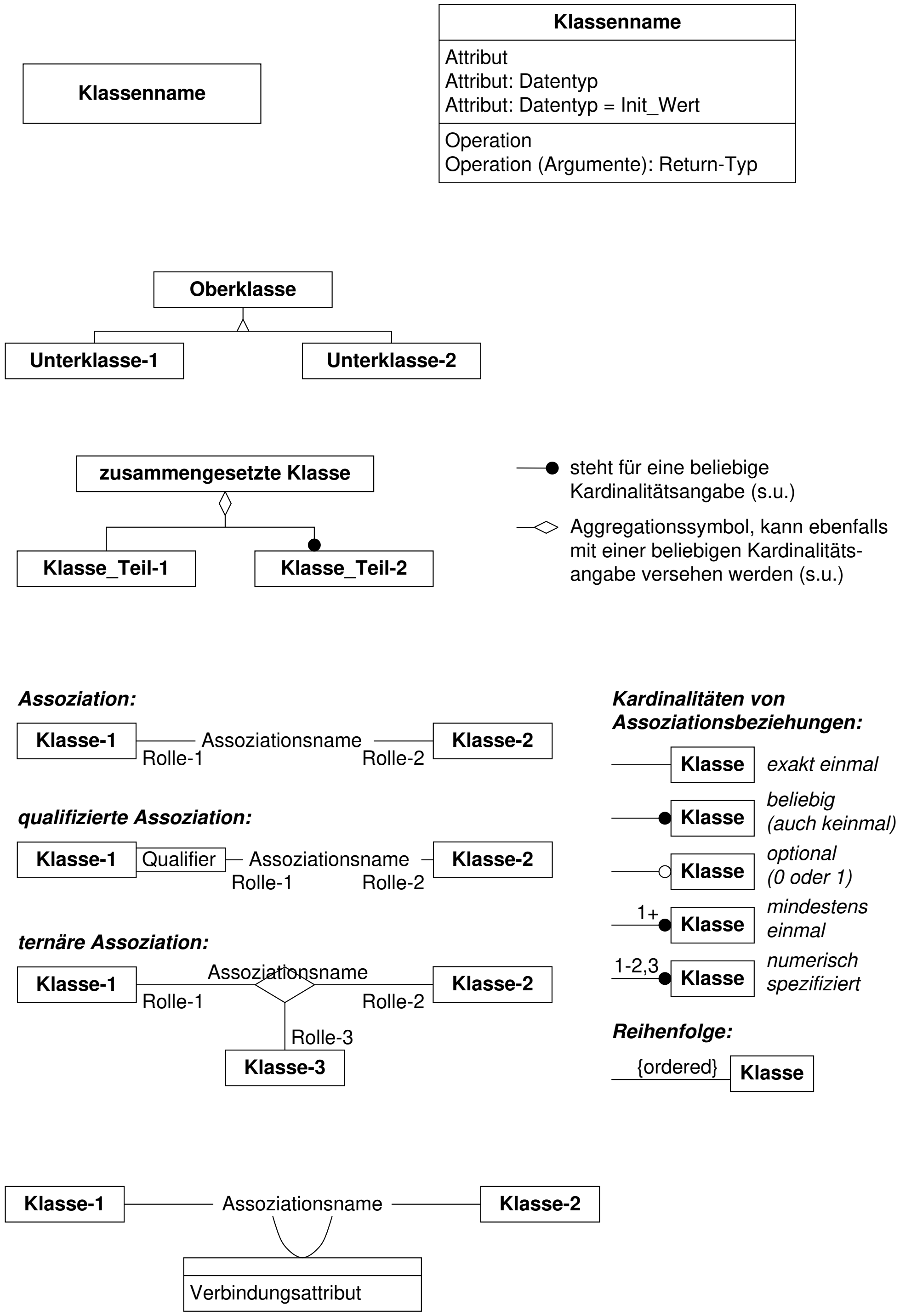

Abbildung 1: die OMT-Notation

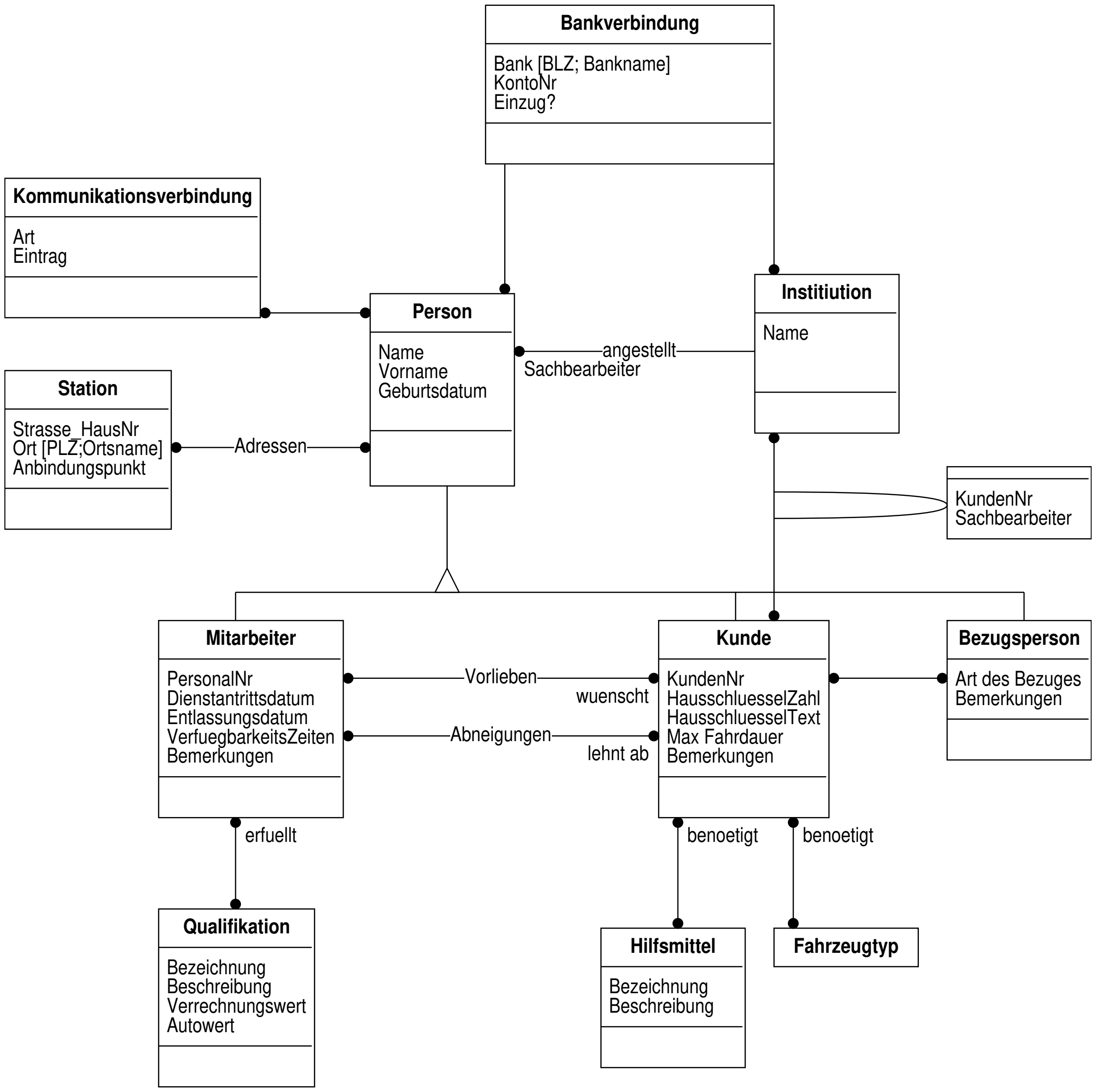

Abbildung 2: Personen

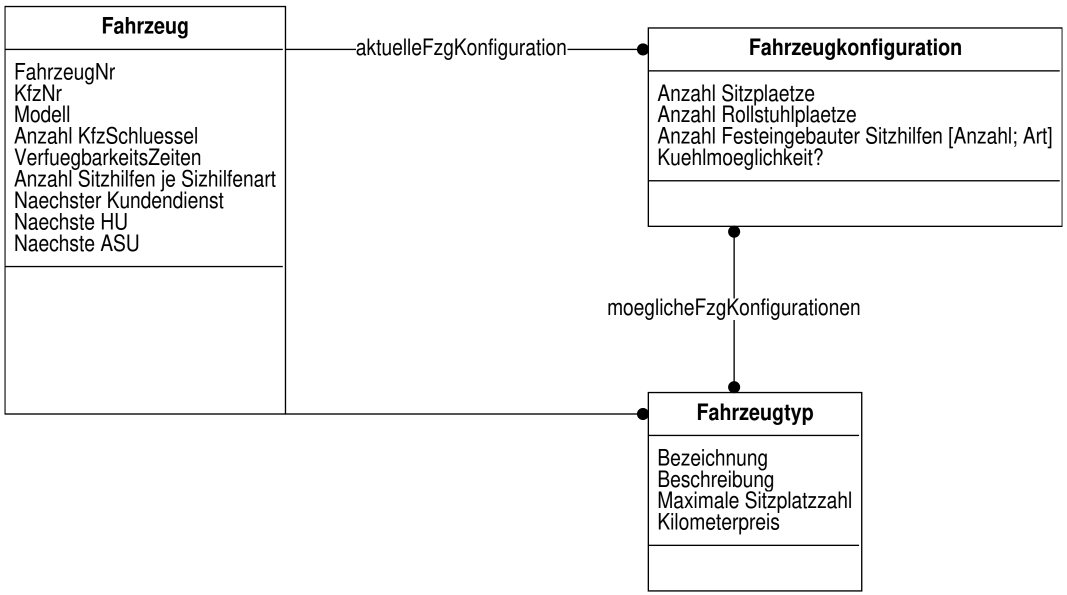

Abbildung 3: Fahrzeuge

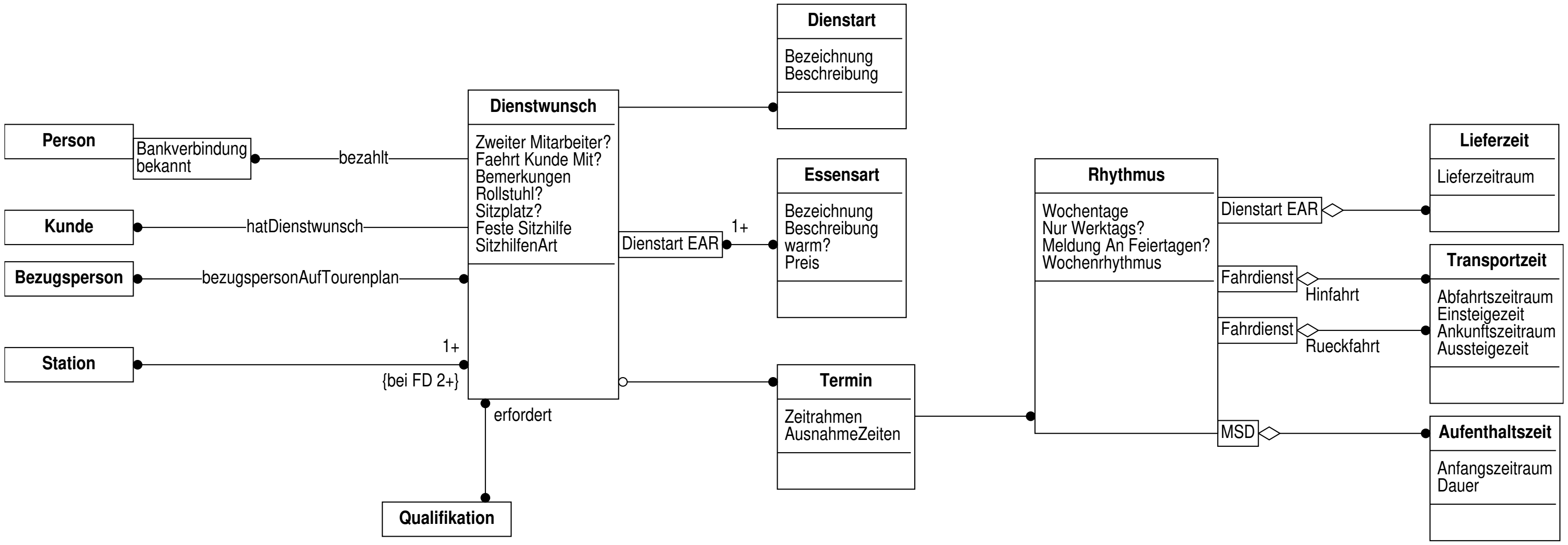

Abbildung 4: Dienstwünsche

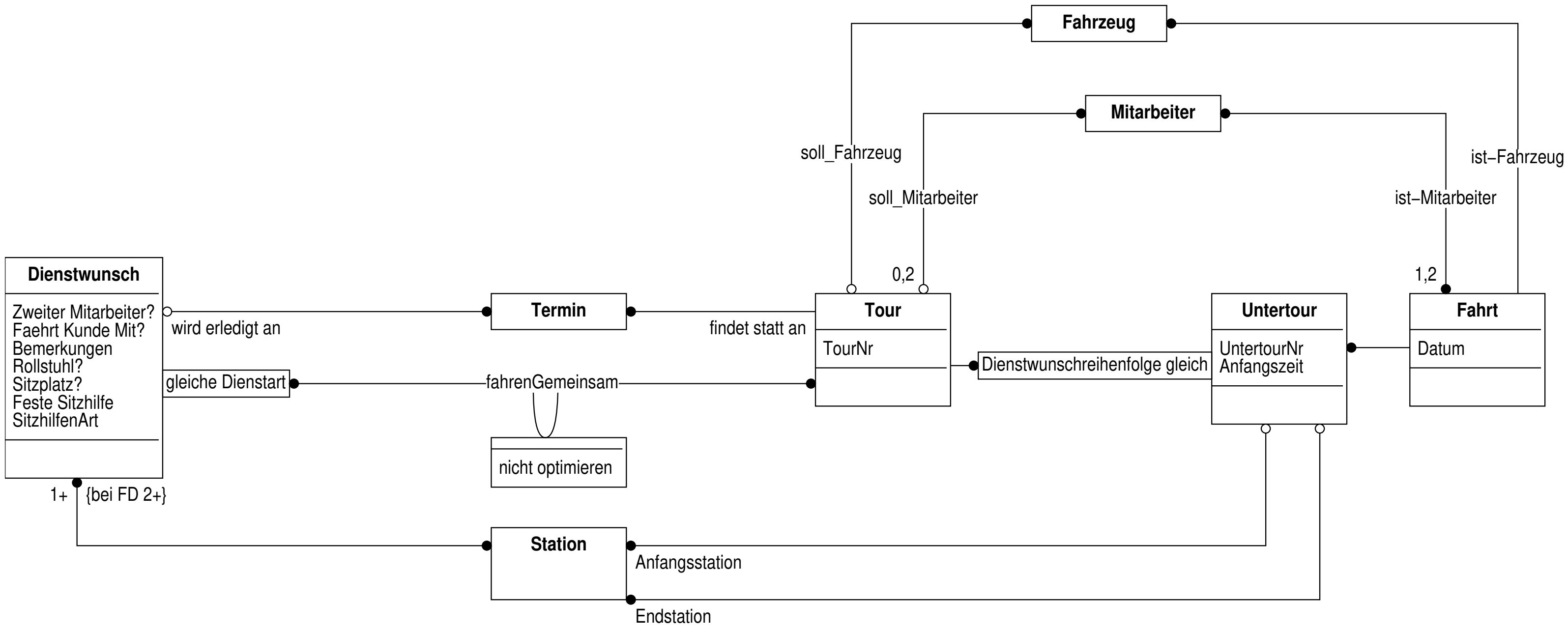

Abbildung 5: Touren

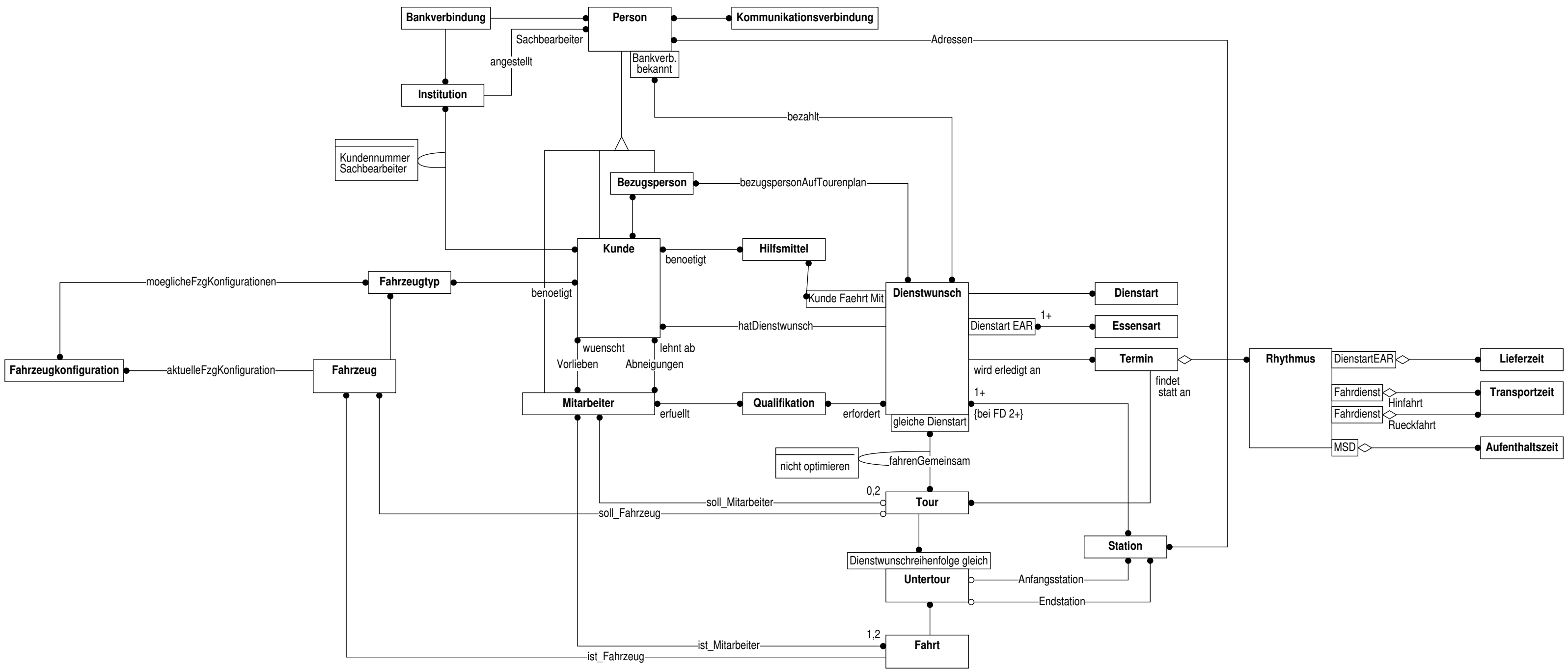

Abbildung 6: Gesamtübersicht

2.2 Klassen-Beschreibungen

Aufenthaltszeit		
Zeit bei Diensten mit Anfang und Aufenthaltsdauer		
Attribute	Spezifikation	
Anfangszeitraum	Beschreibung:	Zeitintervall innerhalb dessen der zugeordnete Dienstwunsch begonnen werden soll.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	Muß innerhalb eines Tages liegen!
	Voreinstellung:	
Dauer	Beschreibung:	Dauer des Aufenthalts, z.B. beim Kunden, d.h. die für den zugeordneten Dienstwunsch benötigte Zeit, ohne An- und Abfahrtszeit.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	

Bankverbindung		
Bankverbindungen von Personen bzw. Institutionen.		
Attribute	Spezifikation	
BLZ	Beschreibung:	Bankleitzahl
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
Bankname	Beschreibung:	Bankname zu obiger BLZ
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	

Fortsetzung der Klasse **Bankverbindung**		
Attribute	Spezifikation	
KontoNr	Beschreibung:	Kontonummer der Kontoverbindung bei obiger Bank
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
Einzug?	Beschreibung:	Liegt eine Einzugsermächtigung vor?
	Typ:	BOOLEAN
	Bereich:	TRUE, FALSE
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	

Bezugsperson		
Bezugspersonen haben einen bestimmten Bezug zu Kunden des DRK und dienen diesem als Ansprechpartner. Bezugspersonen sind z.B. Eltern, Kinder, Hausärzte.		
Attribute	Spezifikation	
Art des Bezuges	Beschreibung:	Kurzbeschreibung des Bezuges, z.B. Familienverhältnis oder Arzt
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	
Bemerkung	Beschreibung:	Freier Text zum Vermerken zusätzlicher Informationen
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	

Dienstart
Die Dienstart gibt an, um welchen, der vom DRK angebotenen, Dienste es sich handelt. Dienstarten sind z.B. MSD, EAR und Schulfahrdienst.

Attribute	Spezifikation	
Bezeichnung	Beschreibung:	Eindeutiger (Kurz-)Bezeichner einer Dienstart
	Typ:	
	Bereich:	
	Schlüssel:	ja
	Muß-Wert:	ja
	Restriktionen:	muß eindeutig sein
	Voreinstellung:	
Beschreibung	Beschreibung:	Nähere Beschreibung der Dienstart.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	

Dienstwunsch
Der Dienstwunsch repräsentiert einen Kundenauftrag.

Attribute	Spezifikation	
Zweiter Mitarbeiter?	Beschreibung:	Angabe, ob ein zusätzlicher Mitarbeiter benötigt wird.
	Typ:	BOOLEAN
	Bereich:	TRUE, FALSE
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	FALSE
Faehrt Kunde Mit?	Beschreibung:	Angabe ob der Kunde bei Ausführung des Dienstwunsches mitfahren möchte. Wird nur beim MSD benötigt um zu wissen, ob auch an das Fahrzeug Anforderungen gestellt werden.
	Typ:	BOOLEAN
	Bereich:	TRUE, FALSE
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	bei MSD und EAR: FALSE, bei Fahrdiensten: TRUE

Fortsetzung der Klasse **Dienstwunsch**		
Attribute	Spezifikation	
Bemerkungen	Beschreibung:	Beliebiger Text, der auf dem Tourplan gedruckt werden soll.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
Rollstuhl?	Beschreibung:	Ist true, wenn für den Dienstwunsch ein Rollstuhl benötigt wird.
	Typ:	BOOLEAN
	Bereich:	TRUE, FALSE
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	Kann nur true sein, wenn „Sitzplatz? 'ünd „FesteSitzhilfe?“ false sind.
	Voreinstellung:	FALSE
Sitzplatz?	Beschreibung:	Ist true, wenn ein Sitzplatz (mit oder ohne Sitzkissen) für die Ausführung des Dienstwunsches benötigt wird.
	Typ:	BOOLEAN
	Bereich:	TRUE, FALSE
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	Kann nur true sein, wenn „Rollstuhl?“ und “FesteSitzhilfe?“ false sind.
	Voreinstellung:	TRUE
FesteSitzhilfe?	Beschreibung:	Ist true, wenn eine festeingebaute Sitzhilfe benötigt wird.
	Typ:	BOOLEAN
	Bereich:	TRUE, FALSE
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	Kann nur true sein, wenn „Rollstuhl?“ und „Sitzplatz?“ false sind.
	Voreinstellung:	FALSE
Sitzhilfenart	Beschreibung:	Gibt an, welche Sitzhilfe benötigt wird (Sitzkissen, Spezialsitz).

Fortsetzung der Klasse **Dienstwunsch**		
Attribute	Spezifikation	
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nur, wenn “FesteSitzhilfe?“ true ist.
	Restriktionen:	ist “Rollstuhl?“ true, darf hier nichts stehen.
	Voreinstellung:	

Essensart
Repräsentiert die unterschiedlichen Essensarten, beispielsweise Normalkost, Diätkost, vegetarisch oder püriert.

Attribute	Spezifikation	
Bezeichnung	Beschreibung:	Eindeutige (Kurz-)Bezeichnung für die Essensart, z.B. vegetarisch, pürierte Normalkost, etc.
	Typ:	
	Bereich:	
	Schlüssel:	ja
	Muß-Wert:	ja
	Restriktionen:	muß eindeutig sein
	Voreinstellung:	
Beschreibung	Beschreibung:	Nähere Erläuterung, welches Essen mit der Bezeichnung gemeint ist.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
Warm?	Beschreibung:	Gibt an, ob es sich um ein warmes oder tiefgefrorenes Essen handelt.
	Typ:	BOOLEAN
	Bereich:	TRUE, FALSE
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	TRUE
Preis	Beschreibung:	Der Preis, der für ein Essen dieser Art bezahlt werden muß.
		↷

Fortsetzung der Klasse **Essensart**		
Attribute	Spezifikation	
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	

Fahrt		
Die Fahrt modelliert eine tatsächlich stattfindende Untertour und unterscheidet sich von dieser im wesentlichen durch ein festes Datum, an dem sie ausgeführt wird. Eine Fahrt ist also etwas einmaliges.		
Attribute	Spezifikation	
Datum	Beschreibung:	Ein festes Datum, an dem die Fahrt stattfindet.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	

Fahrzeug		
Repräsentiert ein vorhandenes Fahrzeug.		
Attribute	Spezifikation	
FahrzeugNr	Beschreibung:	DRK-interne Fahrzeugnummer
	Typ:	
	Bereich:	
	Schlüssel:	ja
	Muß-Wert:	ja
	Restriktionen:	muß eindeutig sein
	Voreinstellung:	
KfzNr	Beschreibung:	amtliche Zulassungsnummer
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	muß eindeutig sein
	Voreinstellung:	
Modell	Beschreibung:	Herstellername und Modell-Bezeichnung des Herstellers
	Typ:	

Fortsetzung der Klasse **Fahrzeug**		
Attribute	Spezifikation	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	
Anzahl Kfz-Schluessel	Beschreibung:	Anzahl der mitgelieferten bzw. vorhandenen Schlüssel für das Fahrzeug
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
Verfügbarkeitszeiten	Beschreibung:	Termine, an denen das Fahrzeug eingesetzt werden kann. In der Regel ein offener Zeitrahmen mit Ausnahmen, wenn das Fahrzeug beispielsweise in der Werkstatt ist.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
Anzahl Sitzhilfen Je Sitzhilfenart	Beschreibung:	Anzahl der im Fahrzeug vorhandenen und nicht festinstallierten Sitzhilfen je Sitzhilfenart (z.B. Kindersitze)
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	0
Naechster Kundendienst	Beschreibung:	Termin, an dem das Fahrzeug das nächste Mal zum Kundendienst muß
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein

Fortsetzung der Klasse **Fahrzeug**		
Attribute	Spezifikation	
	Restriktionen:	
	Voreinstellung:	
Naechste HU	Beschreibung:	Datum, an dem das Fahrzeug das nächste Mal zur Hauptuntersuchung muß
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
Naechste ASU	Beschreibung:	Datum, an dem das Fahrzeug das nächste Mal zur Abgassonderuntersuchung muß
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	

Fahrzeugkonfiguration		
Konfigurationsmöglichkeit eines Fahrzeugs		
Attribute	Spezifikation	
Anzahl Sitzplaetze	Beschreibung:	Anzahl der eingebauten Sitzplätze, einschließlich des Fahrersitzes und ohne Rollstuhlplätze
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	Kann nicht größer sein, als die maximale Sitzplatzzahl des zugeordneten Fahrzeugs abzüglich der Anzahl Rollstuhlplätze. Mindestens ein Sitzplatz muß exisitiern (für den Fahrer).
	Voreinstellung:	
Anzahl Rollstuhlplaetze	Beschreibung:	Anzahl der Plätze für Rollstühle, d.h. Plätze an denen kein eingebauter Sitzplatz ist.

Fortsetzung der Klasse **Fahrzeugkonfiguration**		
Attribute	Spezifikation	
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	Kann nicht größer sein, als die maximale Sitzplatzzahl des zugeordneten Fahrzeugs abzüglich der Anzahl Sitzplätze
	Voreinstellung:	
Anzahl Festeingebauter Sitzhilfen	Beschreibung:	Anzahl und Art der festeingebauten Sitzhilfen. Diese können im Gegensatz zu normalen Sitzhilfen nicht einfach vom Sitz entfernt werden
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	0
Kuehlmoeglichkeit?	Beschreibung:	Gibt an, ob das Fahrzeug eine Kühlvorrichtung für tiefgefrorenes Essen (z.B. Schienen für Kühlbox) besitzt.
	Typ:	BOOLEAN
	Bereich:	TRUE, FALSE
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	FALSE

Fahrzeugtyp		
Fahrzeuge werden in Typen eingeteilt, die für das DRK relevant sind. Beispiele: Rollstuhlbus oder hoher Bus.		
Attribute	Spezifikation	
Bezeichnung	Beschreibung:	eindeutige (Kurz-)Bezeichnung des Fahrzeugtyps
	Typ:	
	Bereich:	
	Schlüssel:	ja
	Muß-Wert:	ja
	Restriktionen:	muß eindeutig sein
	Voreinstellung:	
Beschreibung	Beschreibung:	nähere Beschreibung des Fahrzeugtyps

Fortsetzung der Klasse **Fahrzeugtyp**		
Attribute	Spezifikation	
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
Maximale Sitzplatzzahl	Beschreibung:	Maximale Anzahl der Sitzplätze incl. des Fahrersitzes (lt. KFZ-Schein).
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	
Kilometerpreis	Beschreibung:	Preis, der je Kilometer mit einem Fahrzeug diesen Typs abgerechnet wird.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	

Hilfsmittel		
Hilfsmittel die ein Kunde benötigt. Beispielsweise Rollstuhl oder Sitzkissen. Hilfsmittel werden bei der Zuordnung der Fahrzeuge beachtet, falls der Kunde in einem Fahrzeug mitfährt.		
Attribute	Spezifikation	
Bezeichnung	Beschreibung:	Eindeutiger Bezeichner für das Hilfsmittel, z.B. Krücken, Rollstuhl, Kindersitz, Sitzschale, etc.
	Typ:	
	Bereich:	
	Schlüssel:	ja
	Muß-Wert:	ja
	Restriktionen:	muß eindeutig sein
	Voreinstellung:	
Beschreibung	Beschreibung:	nähere Beschreibung für das Hilfsmittel, z.B. besonders großer Rollstuhl, Spezialsitz für Kunde X
		↷

Fortsetzung der Klasse **Hilfsmittel**		
Attribute	Spezifikation	
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	

Institution		
Private und öffentliche Institutionen wie Krankenkassen, Sozilaämter u.ä.		
Attribute	Spezifikation	
Name	Beschreibung:	Name der Institution
	Typ:	
	Bereich:	
	Schlüssel:	ja
	Muß-Wert:	ja
	Restriktionen:	muß eindeutig sein
	Voreinstellung:	

Kommunikationsverbindung		
Mögliche (technische) Kommunikationsverbindungen einer Person. Diese können frei gewählt werden, z.B. Telefon, Fax oder eMail.		
Attribute	Spezifikation	
Art	Beschreibung:	Art der Kommunikationsverbindung, z.B. Telefon, Fax, eMail, Handy
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	
Eintrag	Beschreibung:	Zur Art passender Eintrag, z.B. Telefonnummer, eMail-Adresse
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	muß zur Kommunikationsart passen
	Voreinstellung:	

Kunde		
Modelliert einen Kunden des DRK.		

Attribute	Spezifikation	
KundenNr	Beschreibung:	Eindeutige Nummer, welche den Kunden innerhalb des DRK identifiziert
	Typ:	
	Bereich:	
	Schlüssel:	ja
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	
Hausschlüsselzahl	Beschreibung:	Anzahl der Kunden-Hausschlüssel, die beim DRK vorhanden sind.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	muß eindeutig sein
	Voreinstellung:	
Hausschlüsseltext	Beschreibung:	Zur freien Verwendung. Hier kann z.B. eine Schlüsselnummer, oder Türbezeichnung stehen.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
MaxFahrdauer	Beschreibung:	Gibt an, wie lange ein Kunde maximal in einem Fahrzeug zubringen darf. Global existierende Beschränkungen für die Fahrdauer (z.B. Schulfahrten 2h) müssen hier nicht eingegeben werden; diese werden automatisch berücksichtigt.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	

Fortsetzung der Klasse **Kunde**		
Attribute	Spezifikation	
Bemerkung	Beschreibung:	Hier können diverse zusätzliche Angaben zum Kunden gemacht werden. Bemerkungen werden bei der Planung nicht berücksichtigt, erscheinen jedoch auf den Tourplänen.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	

Lieferzeit		
Die Uhrzeit bei Lieferterminen (z.B. EAR)		
Attribute	Spezifikation	
Lieferzeitraum	Beschreibung:	Zeitrahmen, innerhalb dessen die Lieferung erfolgen soll.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	Muß innerhalb eines Tages liegen!
	Voreinstellung:	

Mitarbeiter		
Modelliert einen Mitarbeiter des DRK.		
Attribute	Spezifikation	
PersonalNr	Beschreibung:	Eindeutige Nummer, die den Mitarbeiter innerhalb des DRK identifiziert
	Typ:	
	Bereich:	
	Schlüssel:	ja
	Muß-Wert:	ja
	Restriktionen:	Muß eindeutig sein
	Voreinstellung:	
Dienstantrittsdatum	Beschreibung:	Datum, an dem der Mitarbeiter seinen Dienst antritt.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja

Fortsetzung der Klasse **Mitarbeiter**		
Attribute	Spezifikation	
	Restriktionen:	
	Voreinstellung:	
Entlassungsdatum	Beschreibung:	Datum, zu dem der Mitarbeiter offiziell ausscheidet, d.h. ab dem er kein Entgeld mehr bekommt. Bei festangestellten Mitarbeitern ist dieser Eintrag i.d.R. leer.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
Verfuegbarkeitszeiten	Beschreibung:	Zeitintervalle, an denen der Mitarbeiter zur Verfügung steht.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
Bemerkung	Beschreibung:	Beliebiger Text, der den Einsatzplanern in der Mitarbeitermaske zur Information angezeigt wird, der jedoch sonst nirgends auftaucht.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	

Person		
Enthält die Daten, die allen Personen innerhalb des Systems gemeinsam sind und stellt damit deren Superklasse dar.		
Attribute	Spezifikation	
Name	Beschreibung:	Nachname der Person
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	

Fortsetzung der Klasse **Person**		
Attribute	Spezifikation	
	Voreinstellung:	
Vorname	Beschreibung:	Vorname der Person
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
Geburtsdatum	Beschreibung:	Geburtsdatum der Person
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	

Qualifikation		
Qualifikationen modellieren, was ein Mitarbeiter kann, bzw. welchen Anforderungen an Ihn gestellt werden, z.B. ob er spritzen kann oder ob er Lasten heben darf.		
Attribute	Spezifikation	
Bezeichnung	Beschreibung:	Eindeutiger (Kurz-)Bezeichner für die Qualifikation
	Typ:	
	Bereich:	
	Schlüssel:	ja
	Muß-Wert:	ja
	Restriktionen:	muß eindeutig sein
	Voreinstellung:	
Beschreibung	Beschreibung:	Nähere Beschreibung der Qualifikation
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
Verrechnungswert	Beschreibung:	Kostenfaktor, der bei der Optimierung berücksichtigt wird, falls diese Qualifikation erfüllt, jedoch nicht gefordert wird. Der Verrechnungswert wird in der Regel automatisch berechnet, wenn sich Mitarbeiter oder Qualifikationen ändern. Die automatische Berechnung erfolgt jedoch nur, wenn Autowert false ist.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	
Autowert	Beschreibung:	Dieser Wert gibt an, ob der Verrechnungswert automatisch berechnet werden darf: Ist er true, darf die automatische Berrechnung erfolgen, ansonsten behält der Verrechnungswert seinen Eintrag.

Fortsetzung der Klasse **Qualifikation**		
Attribute	Spezifikation	
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	Wird ein Verrechnungswert vom Benutzer angegeben, wird der Autowert auf false gesetzt, ansonsten ist die Voreinstellung true..

Rhythmus		
der Rhythmus in dem ein Termin stattfinden soll		
Attribute	Spezifikation	
Wochentage	Beschreibung:	Gibt an, an welchen Wochentagen der zugeordnete Dienstwunsch ausgeführt werden soll. Mehrfachnennung (z.B. montags, mittwochs und freitags) ist möglich.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	
NurWerktags?	Beschreibung:	Hier wird vermerkt, ob der zugeordnete Dienstwunsch nur an Werktagen oder auch an Feiertagen ausgeführt werden soll.
	Typ:	BOOLEAN
	Bereich:	TRUE, FALSE
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	TRUE
MeldungAn-Feiertagen?	Beschreibung:	Soll ein zugeordneter Dienstwunsch an Feiertagen nicht durchgeführt werden, wird der Benutzer an Feiertagen darauf aufmerksam gemacht, daß der Dienstwunsch nicht erfüllt wird. Dies gibt ihm die Möglichkeit mit dem Kunden einen Ersatztermin zu vereinbaren.

Fortsetzung der Klasse **Rhythmus**		
Attribute	Spezifikation	
	Typ:	BOOLEAN
	Bereich:	TRUE, FALSE
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	Kann nur TRUE sein, wenn NurWerktags? TRUE ist!
	Voreinstellung:	Bei Fahrdiensten: FALSE, sonst TRUE.
Wochenrhythmus	Beschreibung:	Gibt an alle wieviel Wochen der zugeordnete Dienstwunsch ausgeführt werden soll
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	FALSE

Station
Eine Station ist im wesentlichen eine Adresse. Außerdem enthält sie für die Verarbeitung mit dem Verkehrsmodul einen Anbindungspunkt an dessen Straßensystem.

Attribute	Spezifikation	
Strasse	Beschreibung:	Straßen- oder Platzname, bzw. Postfach incl. der jeweiligen Nummer
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	
Ort	Beschreibung:	PLZ und Name des Ortes, in dem sich die Station befindet
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	
Anbindungspunkt	Beschreibung:	Für die Verarbeitung mit dem Verkehrsmodul muß eine Anbindung an dessen Straßensystem vorhanden sein. Dies ist in der Regel die Adresse der Station. Ist diese jedoch im Verkehrsmodul nicht identifizierbar, kann hier auch ein anderer Anbindungspunkt angegeben werden, der dem Verkehrsmodul bekannt ist (und möglichst nahe an der Stationsadresse liegt).
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	Die Stationsadresse im Format des Verkehrsmoduls

Termin		
Der Termin legt fest, wann zugeordnete Dienstwünsche ausgeführt werden.		
Attribute	Spezifikation	
Zeitrahmen	Beschreibung:	Das Datumsintervall, in dem der zugeordnete Dienstwunsch ausgeführt werden soll. Ist nicht bekannt, wie lange ein Dienstwunsch erfolgen soll, ist das Intervallende offen.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	Intervallanfang < Intervallende, falls Intervallende vorhanden
	Voreinstellung:	
Ausnahmezeiten	Beschreibung:	Datumsintervalle innerhalb des Zeitrahmens, in denen der zugeordnete Dienstwunsche nicht erfolgen soll, z.B. Schulferien, Kunde im Urlaub
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	

Tour		
Eine Tour faßt Dienstwünsche gleicher Dienstart zusammen, die später als „ein Block“ gefahren werden.		
Attribute	Spezifikation	
TourNr	Beschreibung:	Nummer, die die Tour eindeutig identifziert
	Typ:	
	Bereich:	
	Schlüssel:	ja
	Muß-Wert:	ja
	Restriktionen:	muß eindeutig sein
	Voreinstellung:	

Transportzeit		
Zeit bei Fahrdiensten		
Attribute	Spezifikation	
Abfahrtszeitraum	Beschreibung:	Uhrzeitintervall, in dem der Transport beginnen soll.

Fortsetzung der Klasse **Transportzeit**		
Attribute	Spezifikation	
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	Nur, wenn kein Ankunftszeitraum vorhanden ist.
	Restriktionen:	Muß innerhalb eines Tages liegen!
	Voreinstellung:	
Einstiegszeit	Beschreibung:	Aufenthaltszeit am Einstiegsort, z.B. bis ein Kind vom 3. Stock bis zum Auto getragen wird. Wird kein Wert eingetragen, wird der Pauschalwert aus den Voreinstellungen übernommen.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	
	Restriktionen:	
	Voreinstellung:	
Ankunftszeitraum	Beschreibung:	Uhrzeitintervall, in dem das Ziel erreicht werden soll.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	Nur, wenn kein Abfahrtszeitraum vorhanden ist.
	Restriktionen:	Muß innerhalb eines Tages liegen!
	Voreinstellung:	
Aussteigezeit	Beschreibung:	Aufenhaltszeit am Aussteigeort, z.B wenn ein Kind in einen Rollstuhl gesetz und in die Klasse gefahren werden muß. Wird kein Wert eingetragen, wird der Pauschalwert aus den Voreinstellungen übernommen.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	nein
	Restriktionen:	
	Voreinstellung:	

Fortsetzung der Klasse **Transportzeit**		
Attribute	Spezifikation	

Untertour		
Untertouren entstehen aus Touren durch eine festgelegte Dienstwunschreihenfolge sowie eine feste Anfangszeit.		
Attribute	Spezifikation	
UntertourNr	Beschreibung:	Setzt sich aus der Tournummer und einem alphanumerischen Teil zusammen. Die Untertournummer identifiziert eine Untertour eindeutig.
	Typ:	
	Bereich:	
	Schlüssel:	ja
	Muß-Wert:	ja
	Restriktionen:	Muß eindeutig sein.
	Voreinstellung:	
Anfangszeit	Beschreibung:	Die feste Uhrzeit, zu der die Tour beginnt.
	Typ:	
	Bereich:	
	Schlüssel:	nein
	Muß-Wert:	ja
	Restriktionen:	
	Voreinstellung:	

2.3 Tourkonzept

In diesem Kapitel soll beschrieben werden, wozu die Unterscheidung von Touren, Untertouren und Fahrten dient. Außerdem sollen die Analogien zum derzeitigen Vorgehen beim DRK aufgezeigt werden.

Die Dienstpläne ergeben sich direkt aus den Fahrten. Sie stellen also nur eine andere Sicht auf die Daten und damit keine eigenständigen Objekte dar, weshalb sie im Datenmodell auch nicht berücksichtigt wurden.

2.3.1 Tour

Eine Tour faßt zunächst Dienstwünsche zusammen, die grundsätzlich miteinander bzw. nacheinander erledigt werden können (die also insbesondere von der gleichen Dienstart sind und mit einem Fahrzeug zu bewältigen sind). Damit entspricht die Tour der DRK-Tour, wie sie durch die derzeitigen Tourpläne beim DRK repräsentiert wird.

Die Assoziation **fahrenGemeinsam** nimmt gerade die oben angegebene Partitionierung der Dienstwünsche in Touren vor. Zudem enthält sie das Attribut *nicht optimieren* als Flag mit der Semantik: die Dienstwünsche der Tour, bei denen das Flag gesetzt ist, werden bei der automatischen Touroptimierung nicht verändert (bleiben also in jedem Fall in der Tour). Dies gewährleistet die DRK-Anforderung, daß Kunden in jedem Fall in einer bestimmten Tour mitfahren sollen, z.B. damit bestimmte Kunden auch dann zusammenfahren können, wenn dies keine optimale Tour ergibt.

Die Assoziationen **sollFahrzeug** und **sollMitarbeiter** bieten die Möglichkeit, einer Tour maximal zwei Mitarbeiter und ein Fahrzeug zuzuordnen. Diese Sollwerte werden nach Möglichkeit bei der Fahrtenbildung berücksichtigt. Mit diesen Assoziationen wird also vorallem der Forderung des Mitarbeiterbezuges und der Ressourcenplanung nachgekommen. Um bei der Planung größtmögliche Flexibilität zu gewährleisten, ist die Zuweisung der Sollwerte optional. Damit werden jedoch die Analysefunktionen und Konsistenzprüfungen eingeschränkt, da nicht bekannt ist, welche Ressourcen für die Tour verplant werden.

Zu beachten ist, daß bei Touren nichts über die Reihenfolge der Dienstwünsche (und damit der Stationen) oder die Abfahrtszeit ausgesagt wird. Dies ist notwendig, da Kunden, unabhängig von den Rahmenbedingungen wie Uhrzeit oder ob es sich um eine Hin- bzw. Rückfahrt handelt, immer in gleichen Personengruppen befördert werden sollen. Außerdem wissen die Mitarbeiter anhand der eindeutigen Tournummer schon, welche Kunden sie abholen müssen.

2.3.2 Untertour

Mit der Untertour wird die Dienstwunsch- bzw. Stationenreihenfolge und der Rhythmus (Wochentag, Wochenrhythmus und Uhrzeit), zu der eine Tourinstanz gefahren werden soll, festgelegt. Damit ist es also beispielsweise möglich, Hin- und Rückfahrten zu gestalten. Außerdem kann mit einer Untertour dargestellt werden, daß die zu befördernde Personengruppe nicht unbedingt vollständig befördert werden muß (fehlt eine Person regelmäßig, ändert das nur etwas an der Untertour, die Personengruppe der Tour bleibt jedoch bestehen).

Auch die Untertouren existieren beim DRK bereits heute. Sie werden jedoch nicht explizit als solche aufgeführt, sondern finden sich als Variationen der Tour auf dem DRK-Tourplan wieder: im Kopf steht die Information zu unterschiedlichen Startzeiten, die Stationenreihenfolge bei Rückfahrten wird einfach gedanklich umgekehrt und regelmäßig ausfallende Personen werden als Bemerkungen bei den jeweiligen Stationen vermerkt.

2.3.3 Fahrt

Die Fahrten repräsentieren konkret gefahrene Untertouren und unterscheiden sich von diesen im wesentlichen durch ein festes Datum.

Da bekannt ist an welchem Datum eine Fahrt stattfindet, kann anhand der Verfügbarkeitszeiten der Mitarbeiter und Fahrzeuge überprüft werden, ob die zugeordneten Mitarbeiter und das zugeordnete Fahrzeug überhaupt verfügbar sind. Auch bei den Dienstwünschen der Untertour kann anhand ihrer Termine geprüft werden, ob sie ausgeführt werden müssen. Ist beispielsweise Kunde X im Urlaub, wird dessen Dienstwunsch nicht in der Fahrt berücksichtigt, ohne daß dies weitere Auswirkungen auf die bestehenden (Unter-)Touren hat.

Auch die TRO-Fahrten existieren schon heute beim DRK: Abweichungen von bestehenden Touren, beispielsweise wegen Mitarbeiter- oder Fahrzeugausfall, werden den betroffenen (evtl. neu eingeteilten) Mitarbeitern als ergänzende Information zum Tourenplan schriftlich oder mündlich mitgeteilt.

3 Verkehrsmodul

Das Verkehrsmodul bildet die Schnittstelle zwischen dem Transportoptimierungssystem und einem Verkehrstool (einer digitalen Straßenkarte). Es bietet dem Rest des Systems eine Möglichkeit, Fahrzeiten zwischen zwei Stationen zu ermitteln, die als Kombination aus Postleitzahl, Ortsname, Straßenname und Hausnummer gegeben sind. Das Verkehrstool wird dabei komplett eingekapselt, wodurch es auch einfach auszutauschen ist.

3.1 Das Verkehrstool: Map&Guide

Als Verkehrstool wurde Map&Guide der CAS Software GmbH in der Version 4.1 gewählt. Standardmäßig hat es nur eine interaktive Benutzerschnittstelle. Mit dem Zusatzmodul „batch-control" bietet es aber auch eine Möglichkeit, von anderen Programmen aus Anfragen zu stellen. Dazu legt man eine Auftragsdatei mit einer Liste der Stationen an, die in einer Tour angefahren werden sollen. Map&Guide sieht in regelmäßigem, einstellbaren Abstand in einem ebenfalls einstellbaren Verzeichnis nach solchen Auftragsdateien. Hat es eine gefunden, liest es sie ein und kann zunächst die Reihenfolge der Stationen optimieren, was hier aber nicht verwendet wird. Für die Tour wird dann ein kürzester Weg gesucht, wobei das Geschwindigkeitsprofil des Fahrzeugs und die Gewichtung von räumlicher Entfernung und Fahrzeit eingestelt werden kann. Der ermittelte Weg kann dann auf einem Drucker oder in Map&Guide als Liste oder in einer Karte ausgegeben werden. Für die Weiterverarbeitung gibt es die Möglichkeit, die Liste der Stationen mit Zeiten und Entfernungen, wahlweise auch mit Wegbeschreibung, in einer Ergebnisdatei zu speichern.

Im Benutzerhandbuch wird noch ein mögliches Problem angesprochen: auf dem Rechner, auf dem die batch-Aufträge abgearbeitet werden, ist darüberhinaus fast kein Arbeiten mehr möglich.

3.2 Aufbau des Verkehrsmoduls

Die Beschreibung des Verkehrstools Map&Guide macht schon deutlich, daß eine Anfrage lange dauern kann: zuerst muß Map&Guide die neue Auftragsdatei entdecken, dann muß es den Weg suchen und speichern und schließlich muß das Verkehrsmodul die Ergebnisdatei finden und auswerten. Daher sollte die Zahl der Anfragen an Map&Guide möglichst gering gehalten werden.

Der erste Ansatz, um Anfragen an das Verkehrstool zu vermeiden, ist die Speicherung bereits ermittelter Entfernungen in einer Tabelle, wo sie auch über mehrere Aufrufe des Transportoptimierungssystems hinweg erhalten bleiben. Die Größe dieser Tabelle kann der Benutzer einschränken, da sie sehr groß werden kann (bis zu $|\text{Stationen}|^2 \approx |\text{Kunden}|^2$ Einträge). Ist diese Grenze erreicht, werden alte Einträge von neuen überschrieben.

Der zweite Ansatz ist die Ankündigung großer Mengen von Anfragen. Das Verkehrsmodul kann dann alle angekündigten Entfernungen mit einer Anfrage an das Verkehrstool ermitteln und speichern. Bei den folgenden Anfragen nach einzelnen Entfernungen kann dann auf die gespeicherten Werte zurückgegriffen werden. Es sind aber trotzdem einzelne Anfragen ohne vorherige Ankündigung möglich, deren Bearbeitung dann aber lange dauern kann.

Zur Ankündigung von Anfragen wurden zwei Ansätze diskutiert: ein eher „naiver“ und ein „intelligenterer“.

Bei der „naiven“ Variante wird dem Verkehrsmodul entweder eine Liste von zu verbindenden Stationspaaren übergeben, oder eine Liste von Stationen, zwischen denen alle Entfernungen ermittelt werden sollen. Der Aufrufer, also z.B. der gerade anlaufende Optimierungsalgorithmus, muß dann zunächst eine Menge von benötigten Entfernungen bestimmen. Wenn das nicht ohne weiteres möglich ist, kann auch eine Obermenge angegeben werden.

Beim „intelligenteren“ Ansatz wird vorgesehen, daß das Verkehrsmodul auch auf die restlichen Daten des Systems zugreift. Es sind dann auch Ankündigungen der Art „alle Entfernungen zwischen Stationen von Wünschen der Dienstart Schuhlfahrdienst“ oder „alle Entfernungen zwischen Stationen von Wünschen am Montag Morgen“ möglich.

Der „naive“ Ansatz hat den Vorteil, daß das Verkehrsmodul einfacher und leichter wartbar wird, da es vollkommen unabhängig vom restlichen System ist. Der „intelligente“ Ansatz hat den Vorteil, daß die einzelnen Ankündigungen, die von mehreren Stellen des restlichen Systems kommen, einfacher werden. Jedoch werden dabei in der Regel auch viele nicht benötigte Entfernungen angekündigt.

3.3 Korrektur von Entfernungen

Eine Anforderung des Kunden ist, daß er die gelieferten Entfernungen der Realität anpassen kann. Deshalb verwaltet das Verkehrsmodul eine Liste mit Änderungen, in der immer als erstes nach einer Entfernung gesucht wird. Grundsätzlich sollte der Benutzer aber zuerst nachsehen, weshalb eine Entfernung nicht stimmt. Führt zum Beispiel der Weg zwischen zwei Kunden, den das Verkehrstool als kürzesten ermittelt, durch eine gesperrte Straße, so hat es wenig Sinn, nur die Entfernung zwischen diesen beiden Kunden anzupassen. Besser wäre es dann, die fehlerhafte Straße im Verkehrstool zu korrigieren. Dazu wird bei Map&Guide aber das Zusatzmodul „road editor plus“ benötigt, das uns jedoch nicht zur Verfügung steht.

4 Datenausgabe

In diesem Abschnitt werden mögliche Ausgaben von TRO vorgestellt. Diese werden schrittweise, je nach vorhandener Zeit entsprechend ihrer Priorität[1] implementiert.

4.1 Dienstpläne

Dienstpläne zeigen die Zuordnung von Fahrten zu Mitarbeitern bzw. zu Fahrzeugen und dienen sowohl als Übersicht für die DRK-Planer, als auch für die persönliche Zeitplanung der Zivis. Die unterschiedlichen Ansprüche werden durch drei unterschiedliche Dienstpläne berücksichtigt:

- Ein Mitarbeiter / eine Woche (A)
- Alle Mitarbeiter / eine Woche (A)

[1] es gibt die Prioritäten A,B und C, wobei A die höchste darstellt. Die jeweiligen Prioritäten werden in Klammern am Ende des jeweiligen Eintrags angegeben.

- Alle Fahrzeuge / eine Woche (A)

Die Dienstpläne könnten in textueller Form als Tabelle oder als Ganttcharts dargeboten werden. Ein Beispiel für ein Ganttchart ist Abb. 7 .

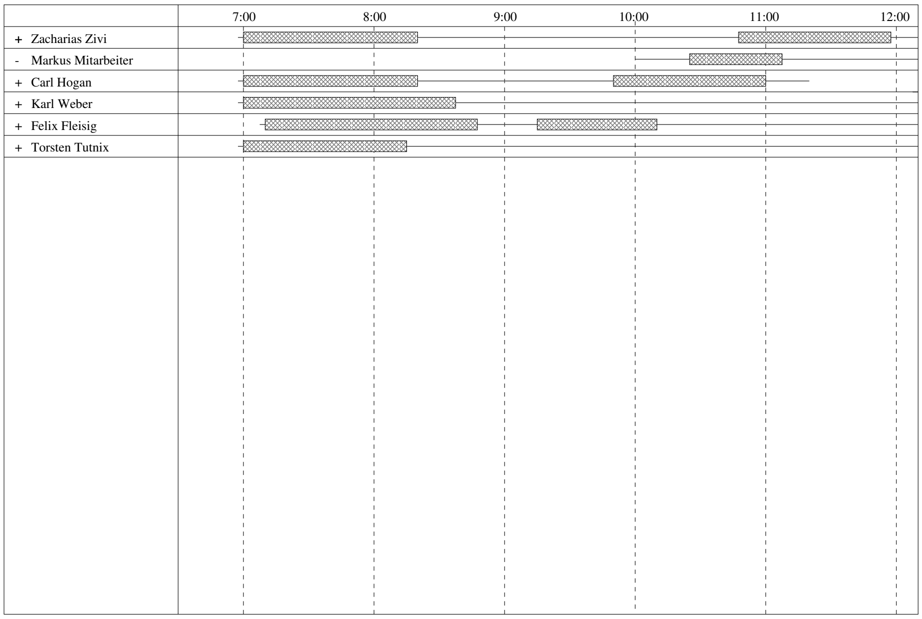

Abbildung 7: Beispiel für einen Dienstplan als Ganttchart

4.2 Analysedaten

Analysedaten dienen als Grundlage zur Infrastrukturplanung des DRK. Mit deren Hilfe sollen Fragen wie „werden mehr oder weniger Fahrzeuge benötig?“, oder „werden mehr oder weniger Mitarbeiter benötigt?“ beantwortet werden.

Folgende Analysedaten werden jeweils für Auslastung [2] und Ausfall angeboten:

- ein Mitarbeiter / Zeitraum (A)
- alle Mitarbeiter / Zeitraum (A)
- ein Fahrzeug / Zeitraum (A)
- alle Fahrzeuge / Zeitraum (A)
- alle Fahrzeuge einer Fahrzeugklasse / Zeitraum (A)

Die Auslastung bedeutet dabei: $\frac{\text{Einsatzzeit}}{\text{verfügbare Zeit}}$. Bei Fahrzeugen bzw. Fahrzeugklassen kann die Auslastung alternativ auch auf die Besetzung bezogen

[2] Damit Aussagen über eine Auslastung sinnvoll sind, muß es einen Referenz- bzw. Maximalwert geben. D.h., daß für Mitarbeiter und Fahrzeuge eine Soll-Arbeitszeit für einen Tag und evtl. auch für eine Woche (da es beim DRK die 5- und die 7-tage Woche gibt) eingegeben werden muß

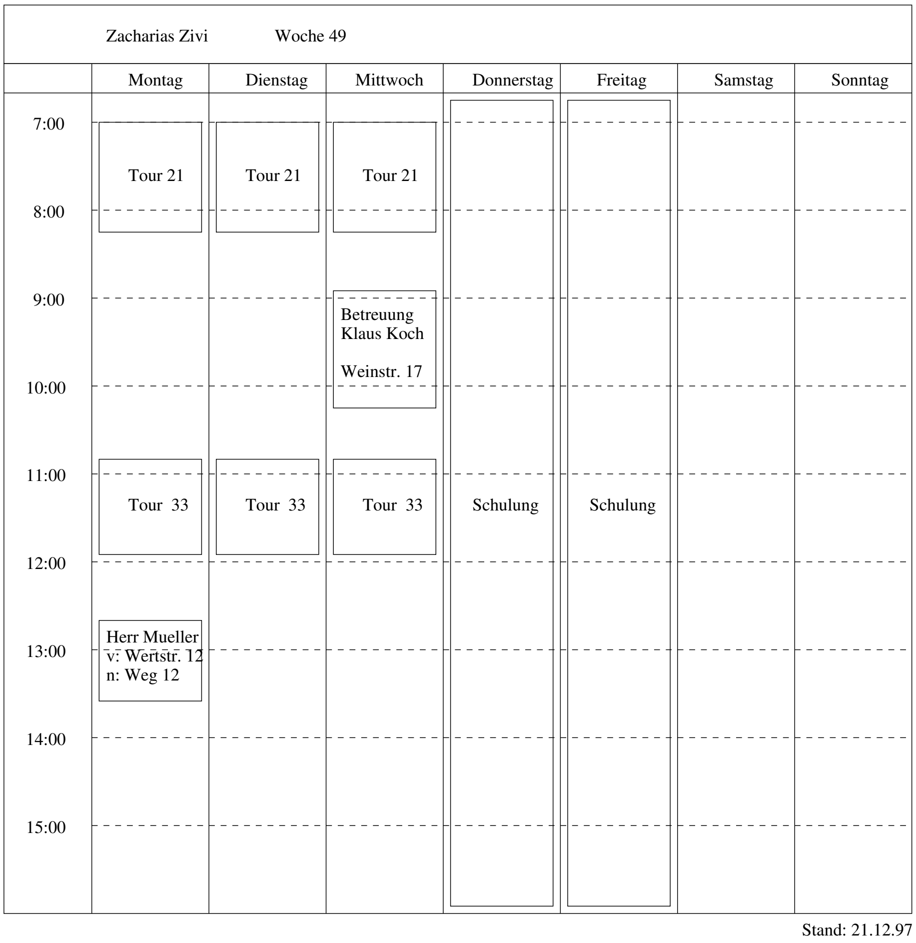

Abbildung 8: Beispiel für einen Dienstplan als Tabelle

werden: $\frac{\text{belegte Plätze}}{\text{vorhandene Plätze}}$. Der Zeitraum ist vom Anwender einzugeben und kann beispielsweise ein Wochentag, eine Woche, ein Quartal oder ein Jahr sein. Ist der Zeitraum länger als ein Tag, kann er zustäzlich angeben, ob er die Auslastung für jeden Tag in diesem Zeitraum haben möchte, oder ob ein Durchschnitt der Auslastung über alle Tage gebildet werden soll. Desweiteren kann auch nach einem bestimmten Wochentag über einen Zeitraum gefragt werden, z.B. alle Dienstage im letzten Jahr.

Zusätzlich werden folgende Analysedaten angeboten:

- Kosten / Dienst (A)
- Kosten / Dienstwünsche (A)
- Kosten / Kunden (A)
- Anzahl Dienstwünsche je Dienstart (B)
- Anzahl Dienstwünsche / Anzahl Mitarbeiter (B)

- Anzahl Dienstwünsche / Anzahl Fahrzeuge (B)
- Einnahmen / Dienst (B)
- Einnahmen / Dienstwünsche (B)
- Einnahmen / Kunden (B)
- Anzahl, Zeit und Km der Leerfahrten (C)
- Anzahl, Zeit und Km je Dienstart (C)
- Durchschnittlich benötigte Zeit für die Erledigung eines Dienstwunsches (C)

4.3 Listen

Zum Gesamtüberblick über den aktuellen Bestand werden folgende Listen angeboten:

- Kunden
  - sortiert nach
    - Nachname, Vorname (B)
    - Kundennummer (B)
  - gefiltert nach Terminen ihrer Dienstwünsche (C)
- Mitarbeiter
  - sortiert nach
    - Nachname, Vorname (B)
    - Personalnummer (B)
  - gefiltert nach Qualifikation (auch mehrfach, z.B. alle Fahrer die Lasten heben können) (B)
  - nach Terminen, an denen sie verplant sind (aus Fahrten bzw. Untertouren) (C)
- Fahrzeuge
  - sortiert nach
    - Fahrzeug-Nummer (B)
    - nach Fahrzeug-Typ (B)
    - mögliche/aktuelle Konfigurationen (C)
  - nach Terminen, an denen sie verplant sind (aus Fahrten bzw. Untertouren) (C)

Als Ausgabeform der Analysedaten sind Balken- und Tortendiagramme vorgesehen.

Desweiteren werden für die Küchen Listen mit den benötigten Essen erstellt.

4.4 Anfragen

Anfragen sollen die DRK-Planer vor allem bei der Frage unterstützen, ob und wann ein bestimmter Dienst noch angenommen werden kann, bzw. wo es noch freie Kapazitäten gibt. Folgende Anfragen sind vorstellbar:

- Kann ein bestimmter Dienstwunsch mit/ohne Tourumstellungen erfüllt werden? (A)
- Ist zum Zeitpunkt (Datum + Uhrzeit) ein Mitarbeiter und/oder ein Fahrzeug verfügbar? (A)
- Wieviele Schultouren können mit/ohne Tourumstellung noch angenommen weden? (A)
- Rechnen sich die Km-Pauschalen? (B)

4.5 Konsistenztests

Konsistenztest sollen Unstimmigkeiten bei Touren, Untertouren und Fahrten aufdecken. Vorstellbar wären etwa:

- Gibt es unerfüllte Dienstwünsche? (A)
- Werden Zeitgrenzen eingehalten? (A)
- Ist kein Fahrzeug überfüllt? (A)
- Stimmen die Mitarbeiterqualifikationen und Fahrzeugtypen der Touren/Fahrten mit den Anforderungen der Dienstwünsche überein? (A)
- Hat jede Fahrt einen Fahrer? (A)
- Sind eingeplante Mitarbeiter und Fahrzeuge einsatzfähig? (A)
- Fahren Mitarbeiter bzw. Fahrzeuge mehrere Touren gleichzeitig? (A)
- Steigen alle Kunden, die in ein Fahrzeug einsteigen, auch aus dem gleichen wieder aus (und umgekehrt)?[3] (A)

5 Lokale Optimierung

„Lokale“ Optimierung bedeutet hier, daß die Optimierung nach Dienstarten getrennt vorgeht und immer nur Dienstwünsche einer Dienstart kombiniert, um möglichst optimale Touren und Untertouren zu bilden. Dabei müssen je nach Dienstart unterschiedlich viele Randbedingungen berücksichtigt werden:

	Orte	Qualifikation der Mitarbeiter	Fahrzeug-Konfiguration	Zeit
EAR	1			(×)
MSD	1	×	(×)	×
Schulfahrten / Dialysefahrten / Tagespflegefahrten	1	×	×	(×)
Individualfahrten	2	×	×	×

[3] Wenn Stationenfolge im Dienstwunsch nicht korrekt ist

Daraus ergibt sich eine Hierarchie hinsichtlich der Anzahl der zu berücksichtigenden Parameter:

1. EAR: Wohnort des Kunden. In seltenen Fällen auch Vorgaben für den Lieferzeitpunkt.

2. MSD: Wohnort des Kunden + Anforderungen an die Qualifikation der Mitarbeiter + Vorgaben zur Uhrzeit.

3. Schulfahrten, Dialysefahrten und Fahrten zur Tagespflege: Hier werden viele Kunden zu wenigen Zielen gefahren, so daß die Kunden nach dem Ziel gruppiert werden können. Dadurch muß die Optimierung lediglich die unterschiedlichen Startpunkte berücksichtigen. Hinzu kommen Anforderung an die Qualifikation der Mitarbeiter sowie an die Ausstattung des Fahrzeugs. Außer der Ankunftszeit ist nur mit wenigen zeitlichen Anforderungen zu rechnen.

4. Individualfahrten: Hier gibt jeder Kunde individuell Start- und Zielort vor (sowie Start- bzw. Zielzeit). Anforderungen an Mitarbeiter und Fahrzeuge existieren genauso wie bei den anderen Fahrdiensten.

5.1 Ein- und Ausgabe der Optimierung

Gegeben Eine Menge von Kunden mit ihren Dienstwünschen, bestehend aus Start- und Zielort, sowie Anforderungen an Zeiten, Qualifikationen der Mitarbeiter und Konfigurationen der Fahrzeuge. Außerdem eine Menge von Mitarbeitern mit bestimmten Qualifikationen sowie eine Menge von Fahrzeugen mit verschiedenen Konfigurationen.

Gesucht Eine Menge von Untertouren, auf denen alle Kunden gemäß ihren Wünschen mit den vorhandenen Mitarbeitern und Fahrzeugen befördert werden können. Gibt es mehrere Lösungen, sollte die beste im Sinne einer noch zu definierenden Bewertung gewählt werden.

5.2 *Was* soll optimiert werden?

1. Anzahl der eingesetzten Fahrzeuge/Mitarbeiter möglichst gering

2. Summe der Einsatzzeiten möglichst klein

3. Bei Fahrten zur Tagespflege, Essen auf Rädern sowie MSD/APD, die pauschal bezahlt werden (und nicht nach gefahrenen Kilometern) sollte auch die Anzahl der gefahrenen Kilometer möglichst klein sein.

5.3 Effizienz

Ein intuitiver Lösungsansatz könnte etwa so aussehen:

Bestimme eine Qualitätsfunktion bzw. Kostenfunktion, bezüglich derer verschiedene Lösungen (= Mengen von Untertouren) verglichen werden können. Diese könnte beispielsweise als

$$K(T) = g_1 * \text{Anzahl Fahrzeuge} + g_2 * \text{Anzahl Mitarbeiter}$$

definiert werden. Die Lösung des Problems bestünde nun darin, $K(T)$ zu minimieren, d.h. unter allen möglichen Untertouraufteilungen diejenige zu finden, für die diese Funktion einen minimalen Wert annimmt.

Da es sich bei der Optimierung von Fahrdiensten um eine NP-vollständiges Problem handelt [Buc94] (und wir davon ausgehen, daß P≠NP ist), ist ein solch allgemeines Verfahren nicht effizient lösbar (die Laufzeit der Suche steigt exponentiell zur Anzahl der Dienstwünsche, die eingeplant werden müssen). Deshalb werden im folgenden Heuristiken zum Einsatz kommen, die weder immer eine optimale Lösung finden noch überhaupt garantieren können, daß eine existierende Lösung gefunden wird, von denen man sich aber nach mäßiger Laufzeit eine zumindest akzeptable Lösung verspricht.

5.4 Maße für verschiedene Parameter

Sei W eine Menge von Dienstwünschen, die in Untertouren eingeteilt werden sollen. Seien $T_1, \ldots, T_i \subseteq W$ Mengen von Dienstwünschen, die im Verlaufe der Planung bereits zu den Untertouren 1 bis $i-1$ kombiniert wurden. Untertour i ist die momentan in Planung befindliche, der noch weitere Dienstwünsche hinzugefügt werden können. Sei weiter $R = W \setminus \bigcup_{k=1}^{i} T_k$ die Menge der restlichen Dienstwünsche, die noch nicht in Untertouren eingeteilt wurden.

Der Dienstwunsch, der als nächster an Untertour T_i angefügt wird, heiße D.

Nähe von Orten Für die Auswahl eines Dienstwunsches, der örtlich am besten zur Untertour T_i paßt, sind verschiedene Metriken denkbar:

1. D ist derjenige Dienstwunsch, der zum Ort eines beliebigen Wunsches aus T_i den minimalen Abstand hat (im Vergleich zu allen anderen Dienstwünsche aus R).
2. Die Summe der Abstände von D zu allen $D_j \in T_i$ ist minimal.
3. Die Streckenlänge von $T_i \cup \{D\}$ ist minimal. Dabei kann die Streckenlänge einer Untertour aus gegebenen Stationen approximiert werden, indem der minimale spannende Baum dieser Stationen so umlaufen wird, daß jede Station nur einmal auf diesem Weg liegt.

Anforderungen an Mitarbeiter/Fahrzeuge Ein Maß für die Stärke von Anforderungen an Mitarbeiter ist die Anzahl der Mitarbeiter, die diese Anforderungen erfüllen können. Der anspruchsvollste Dienst bzw. Dienstwunsch ist also der, den am wenigsten Mitarbeiter ausführen können. Analog lassen sich Dienstwünsche nach ihren Ansprüchen an das eingesetzte Fahrzeug ordnen.

Reihenfolge von Stationen Da die eingesetzten Fahrzeuge jeweils nur eine kleine Anzahl von Kunden transportieren können (zwischen zwei und acht), läßt sich eine hinsichtlich Fahrtdauer bzw. -strecke gute Stationenfolge recht einfach und effizient bestimmen.

5.5 *Wie* könnte eine gute Lösung gefunden werden?

5.5.1 Matching-Verfahren

Für das Problem der Bildung von allgemeinen Fahrgemeinschaften ließen sich durch Maximale Matchings auf Personengraphen gute Lösungen finden, indem die Knoten (=Personen) im Graphen genau dann verbunden werden, wenn zwei Personen zusammen fahren können [Pro97]. Ein solcher Ansatz ist für die Transporte von Personen mit besonderen Bedürfnissen (Alte, Kranke, Behinderte) nicht sinnvoll:

- Die Aussage „Person X und Y können zusammen befördert bzw. betreut werden“, liefert keine wertvolle Information. Ein hochqualifizierter Mitarbeiter kann mit einem optimal ausgerüsteten Fahrzeug (z.B. Rollstuhlbus mit Hebebühne, der zudem noch eine Sitzbank mit verschiedenen Spezialsitzen eingebaut hat) beliebige Dienstwünsche befriedigen. Das heißt, je zwei beliebige Kunden „matchen“ auf diese Art, wobei dadurch möglicherweise für die restlichen Kunden nicht mehr genügend qualifizierte Mitarbeiter und Fahrzeuge zur Verfügung stehen.

  Um die Aussage „X kann mit Y“ fahren korrekt zu modellieren, wäre ein gerichteter Graph notwendig. Denn ein gehbehinderter Kunde, den man stützen muß, der aber in einem PKW fahren kann, kann gut auf der Sitzbank eines Rollstuhlbusses mitfahren. Umgekehrt ist es nur in gewissen Fällen möglich, einen Rollstuhlfahrer in einem PKW zu transportieren.

- Ein Matching auf einem Graphen, der jeder Person mögliche Fahrzeuge zuordnet, wird dem quantitativen Aspekt nicht gerecht. Ein Maximales Matching würde Paare bilden, so daß zwar viele Kunden einem passenden Fahrzeug zugeordnet würden, aber eben jedem Fahrzeug nur ein Kunde. Da gerade die Anzahl der eingesetzten Fahrzeuge durch Bildung von Untertouren minimiert werden soll, führt dieser Ansatz zu nichts.

- Verbindet man Dienstwünsche, die nacheinander ausgeführt werden könnten, in einem Graphen durch Kanten, bildet ein Matching-Verfahren zwar korrekte, aber zu kleine Untertouren (alle Dienste sollen 45 Minuten dauern, die Zwischenstrecken sind jeweils in 15 Minuten zurückzulegen):

Kunde	Dienst-beginn
A	16 Uhr
B	17 Uhr
C	17 Uhr
D	18 Uhr
E	18 Uhr
F	19 Uhr

  Ein Maximales Matching wäre z.B. {AB, CD, EF}, eine Zusammenfassung zu zwei Untertouren {ABDF, CE} wäre hier allerdings zu bevorzugen.

5.5.2 EAR: Greedy-Ansatz nach Wohnortnähe

Für den Dienst „Essen auf Rädern“ wird angenommen, daß die meisten Kunden für jeden Tag Essen bekommen möchten (zumindest für jeden Werktag). Der Tourplan wird deshalb für die ganze Woche erstellt, so daß sich für jeden Tag die gleiche Untertour ergibt, einzelne Ausnahmen werden nach Bedarf berücksichtigt, ohne jedoch die Tour an sich umzugestalten.

Lieferungen von kaltem/tiefgefrorenem und warmem Essen werden getrennt geplant.

Algorithmus Beginne mit einem beliebigen Kunden. Füge solange den jeweils nächsten Kunden zur Untertour hinzu, bis entweder alle Kunden eingeplant sind oder die Ladekapazität des Fahrzeugs bzw. der späteste erlaubte Lieferzeitpunkt überschritten ist. Im letzteren Fall werden analog weitere Untertouren erzeugt.

Ausgabe Partitionierung der Kunden. Zwei Kunden sind in derselben Partition, wenn sie mit demselben Fahrzeug beliefert werden. Wird als Maß für den „nächsten Kunden“ der geometrische/zeitliche Abstand genommen, ergibt sich die Reihenfolge der Stationen aus der Reihenfolge, in der die Kunden zur Untertour hinzugefügt wurden. In anderen Fällen (z.B. Abstandssumme) muß diese Stationenfolge ggf. noch ermittelt werden.

Eigenschaften

Vorteile Die Summe der Fahrtzeiten in allen Untertouren ist recht gering.

Nachteile Vorgaben für die Lieferzeit werden nicht berücksichtigt. Dies könnte jedoch dadurch erreicht werden, daß ein Kunde, der zeitlich nicht paßt, bei der Suche nach dem nächsten Kunden übersprungen wird.

Laufzeit Um den nächsten Kunden zu ermitteln, müssen alle restlichen verglichen werden. Es entsteht also der Aufwand $\mathcal{O}(w^2)$ (w sei die Anzahl der Dienstwünsche).

5.5.3 MSD/APD: Auswahl nach Beginn der Dienste

Da Dienste dieser Art im allgemeinen eine Zeit vorgegeben haben, wann sie durchgeführt werden sollen, liegt es nahe, Untertouren nach diesem Kriterium zusammenzustellen.

Algorithmus Beginne mit dem Dienst, der am frühesten begonnen werden kann. Setze die Untertourzeit auf dessen frühestmögliche Anfangszeit plus dessen Dauer. Wähle passende Mitarbeiter und ein passendes Fahrzeug.

Wiederhole folgende Schritte, solange wie möglich:

- Bestimme alle Dienste, die zur ermittelten Untertourzeit (= nach dem letzten Dienst) möglich wären. Wähle darunter einen aus, der eine möglichst kleine zeitliche Lücke zum vorigen läßt und möglichst gleiche Anforderungen an Mitarbeiter und Fahrzeug hat. Bei mehreren gleichwertigen wird der am nächsten gelegene gewählt.
- passe gegebenenfalls Mitarbeiter und Fahrzeug so an, daß diese die Anforderungen aller Kunden auf der Untertour erfüllen
- Addiere zur Untertourzeit die Fahrzeit vom letzten Dienstort zum gewählten, sowie die Dienstdauer. (Alle Dienste, die zu dieser Untertourzeit bereits begonnen haben müßten, können hier weggelegt werden und werden bei der nächsten Untertour wieder berücksichtigt)

Damit ist eine Untertour ermittelt und mit den restlichen Dienstwünschen wird das Verfahren wiederholt, bis alle Kunden eingeplant sind.

Eine Verbesserung des Verfahrens läßt sich durch Parallelisierung der Untertourplanung erreichen: Wenn ein Dienst an keine Untertour angehängt werden kann, wird eine Paralleluntertour eröffnet, die mit anderen Mitarbeitern und mit einem anderen Fahrzeug gleichzeitig zu den bisherigen Untertouren ausgeführt werden soll. Im folgenden wird jeder Dienstwunsch an die Untertour angehängt, an die er hinsichtlich Entfernung der Stationen und Pausen (=„Löchern“) zwischen den Diensten am besten paßt.

Ausgabe Eine Menge von Untertouren.

Eigenschaften

Vorteile Die Untertouren werden zeitlich zusammenhängend. Es entstehen keine unnötig großen Löcher zwischen einzelnen Diensten einer Untertour.

Nachteile Persönliche Vorlieben und Abneigungen der Kunden werden nicht berücksichtigt. Abneigungen könnten berücksichtigt werden, indem der jeweils nächste Dienst so gewählt wird, daß er zu den momentan gewählten Mitarbeitern paßt. Da diese sich aber im Laufe des Algorithmus ändern können, wird hier u.U. unnötig eingeschränkt und dadurch fallen mögliche Lösungen weg.

Anforderungen an die Qualifikation der Mitarbeiter werden kaum berücksichtigt. Dadurch können die Kunden mit hohen Ansprüchen so stark verteilt werden, daß nicht ausreichend qualifizierte Mitarbeiter für alle Untertouren zur Verfügung stehen.

Laufzeit Da nach jedem eingeplanten Dienstwunsch alle verbleibenden bezüglich Beginn und Fahrzeit zwischen den Stationen verglichen werden müssen, entsteht hier Aufwand $\mathcal{O}(w^2)$.

Positiv-Beispiel Sechs Dienste ohne besondere Qualifikationsanforderungen sind zu erledigen (z.B. Einkaufen, Kehrwoche). Alle Dienste dauern 45 Minuten, die Zwischenstrecken sind jeweils in 15 Minuten zurückzulegen:

Kunde	Dienst-beginn
A	16 Uhr
B	17 Uhr
C	17 Uhr
D	18 Uhr
E	18 Uhr
F	19 Uhr

Es werden zwei Untertouren zusammengestellt: ABDF und CE.

Negativ-Beispiel Ein Fachpfleger und ein Zivi stehen zur Verfügung, um folgende Dienstwünsche zu erfüllen (Fahrtzeiten werden hier vernachlässigt):

Kunde	Beginn	Tätigkeit	Dauer
A	13 Uhr	Einkaufen	1 h
B	14 Uhr	Wohnung putzen	1 h
C	15 Uhr	Kehrwoche	1 h
D	15:30 Uhr	Krankenpflege	2 h
E	18 Uhr	Einkaufen	1 h
F	18:30 Uhr	Spritzen	10 min

Die Dienstwünsche A, B und C werden zunächst dem Zivi zugeordnet. Für den Dienst D wird eine zweite Untertour eröffnet, die der Pfleger übernimmt. Da Dienst E zeitlich für letzteren günstiger liegt, führt er E nach D aus. Dadurch steht aber für Dienst F, der zeitlich mit E überlappt, kein qualifizierter Mitarbeiter zur Verfügung.

5.5.4 MSD: Auswahl nach Anforderung an Mitarbeiter

Hierfür müssen alle Dienstwünsche nach Anspruch an die Qualifikation der Mitarbeiter geordnet werden (siehe 5.4).

Algorithmus Es wird der anspruchsvollste noch nicht verplante Dienstwunsch gewählt (bei mehreren Möglichkeiten wird nach örtlicher Nähe zu einem bereits verplanten Dienst vorgegangen). Dieser wird in die begonnene Untertour integriert, indem er an der zeitlich passenden Stelle eingefügt wird. Gegebenenfalls werden hierzu die anderen Dienste im Rahmen der zeitlichen Toleranzen verschoben, um Platz zu machen. Sollte ein Dienstwunsch nicht in die Lücken der bisher gewählten passen, wird er zur Seite gelegt und der nächste probiert.

Dies wird wiederholt, bis sich kein Dienstwunsch mehr in die bis dahin ermittelte Untertour einreihen läßt. Mit den zur Seite gelegten sowie den bisher noch nicht betrachteten Dienstwünschen werden weitere Untertouren auf dieselbe Art zusammengestellt.

Auf ähnliche Weise könnten persönliche Vorlieben und Abneigungen von Kunden berücksichtigt werden: Zu einem Mitarbeiter werden zunächst Kunden gewählt, die gerne von diesem Mitarbeiter betreut werden wollen. Dann solche, die möglichst viele der Qualifikationen benötigen, die dieser Mitarbeiter hat.

Ausgabe Eine Menge von Untertouren.

Eigenschaften

Vorteile Auch hohe Anforderungen an die Qualifikation der Mitarbeiter können erfüllt werden (falls dies mit dem vorhandenen Personal überhaupt möglich ist).

Nachteile Es können Untertouren entstehen, die sehr weite Wege beinhalten. Die Nähe der Kunden spielt eine sehr untergeordnete Rolle. Außerdem werden die später entstehenden Untertouren immer schlechter (größere Lücken, weniger Dienste) und möglicherweise unnötig zahlreich (und daurch evtl. mit der vorhandenen Menge an Fahrzeugen nicht mehr fahrbar).

Laufzeit Im schlimmsten Fall müssen alle unverplanten Dienstwünsche mit der Untertour abgeglichen werden, bis dann jeweils der letzte dazu paßt. Es kann also zu $\mathcal{O}(w^2)$ Einfügeversuchen kommen.

Positiv-Beispiel Das Negativ-Beispiel aus 5.5.3 würde mit diesem Verfahren korrekt gelöst, da der Fachpfleger hier zunächst die Dienste D und F zugeordnet bekäme. Dadurch entstünde für den Zivi automatisch die Untertour ABCE, was auch die einzige Möglichkeit ist (und somit trotz großer Lücken das Optimum).

Negativ-Beispiel Zwei Zivis mit pflegerischer Grundausbildung stehen zur Verfügung, um folgende Dienstwünsche zu erfüllen (Fahrtzeiten werden wieder vernachlässigt):

Kunde	Beginn	Tätigkeit	Dauer
A	17 Uhr	Grundpflege	1 h
B	17:30 Uhr	Einkaufen	1 h
C	18 Uhr	Kehrwoche	1 h
D	18:30 Uhr	Grundpflege	1 h

Zivi 1 bekommt zunächst die beiden anspruchsvollsten Dienste zugeteilt (A und D). Dienstwunsch B und C können aber weder in die Untertour von Zivi 1 integriert werden, noch kann Zivi 2 sie alleine erledigen. Die korrekte Lösung (AC und BD) muß in diesem Fall gerade Dienste mit gleichen Anforderungen trennen, da sich sonst die Zeitvorgaben nicht erfüllen lassen.

5.5.5 Fahrdienste: Greedy-Ansatz über Anforderung an Fahrzeug

Die Kunden werden hinsichtlich ihrer Ansprüche an das Fahrzeug in verschiedene Klassen eingeteilt, z.B. 1. Rollstuhlfahrer; 2. Kunden, die besondere Sitze benötigen; 3. übrige Kunden. Ein konkretes Maß für die Stärke der Ansprüche wäre die Anzahl der Fahrzeuge, die diese Ansprüche erüllen können: Je weniger Fahrzeuge für einen Kunden in Frage kommen, desto höher sind seine Ansprüche in diesem Punkt.

Algorithmus Beginne mit der Kundenklasse, die die „höchsten Ansprüche" hat. Im Beispiel wären das die Rollstuhlfahrer, da diese nur in einem Rollstuhlbus transportiert werden können, während z.B. ein Kind mit speziellem Kindersitz sowohl in einem Kleinbus als auch in einem Rollstuhlbus mit Sitzbank mitgenommen werden kann.

Diese Klassen werden nacheinander auf Fahrzeuge verteilt, wie im folgenden exemplarisch für die Rollstuhlfahrer angegeben:

1. Reserviere zunächst für die Rollstuhlfahrer eine ausreichende Menge Rollstuhlbusse

2. Verteile die Rollstuhlfahrer darauf:

   Beginnend mit einem beliebigen Kunden, wird immer derjenige Kunde als nächster dem Fahrzeug zugeordnet, der zu den Kunden, die schon im Fahrzeug sitzen, am nächsten wohnt. Ist ein Fahrzeug voll besetzt, wird mit einem neuen Fahrzeug und einem beliebigen übrigen Kunden fortgesetzt.

3. Fülle ggf. mit Kunden der nächsten Klasse auf, die am nächsten zu den bisher im Fahrzeug eingeplanten Kunden wohnen.

Ausgabe Eine Menge von „Fahrgemeinschaften". Die Reihenfolge, in der die Kunden abgeholt werden, muß noch festgelegt werden. Dies kann auch Fahrzeugübergreifend innerhalb der verschiedenen Klassen geschehen.

Eigenschaften

Vorteile Anforderungen an das Fahrzeug werden gut berücksichtigt. Besondere Fahrzeuge (z.B. Rollstuhlbusse), die nur in geringer Anzahl vorhanden sind, werden zunächst an diejenigen Kunden verteilt, die diese auf jeden Fall benötigen.

Nachteile Anforderungen an die Qualifikationen der Mitarbeiter werden nicht berücksichtigt. Sollten diese so hoch sein, daß nicht ausreichend viele qualifizierte Mitarbeiter für die berechneten Untertouren zur Verfügung stehen, muß versucht werden, durch inkrementelle Verfahren (Umsetzen von Kunden in andere Fahrzeuge) Kunden mit ähnlichen Anforderungen zusammenzubringen.

Anforderungen an Zeiten werden nicht berücksichtigt. Diese können teilweise beim Festlegen der Reihenfolge der Stationen in jeder Untertour erfüllt werden.

Laufzeit Die möglichen Fahrzeuge für einen Kunden können bereits bei dessen Aufnahme in das System berechnet werden, und bleiben dann gültig, solange an den Kundendaten oder Fahrzeugen nichts geändert wird. Die anfängliche Einteilung in Klassen ist dann eine normale Sortieroperation mit Aufwand $\mathcal{O}(w \log(w))$.

Während der Aufteilung müssen bei jedem neu zu besetzenden Fahrzeugplatz alle noch nicht eingeplanten Kunden hinsichtlich der Nähe der Startorte mit den bereits im Fahrzeug verplanten verglichen werden. Aufwand also $\mathcal{O}(w^2)$.

Positiv-Beispiel Sechs Rollstuhlfahrer und sechs Gehbehinderte sollen befördert werden. Dazu stehen folgende Fahrzeuge zur Verfügung: Zwei Rollstuhlbusse (mit entweder vier Rollstuhlplätzen oder zwei Rollstuhlplätzen und drei Sitzplätzen), sowie drei PKW mit je drei Sitzplätzen. Da keine besonders qualifizierten Mitarbeiter benötigt werden, können diese mit dem obigen Algorithmus folgendermaßen aufgeteilt werden:

Rollstuhlbus 1	4 Rollstuhlfahrer
Rollstuhlbus 2	2 Rollstuhlfahrer und 3 Gehbehinderte
PKW	3 Gehbehinderte

Eine sinnvolle Abholreihenfolge muß für jedes Fahrzeug noch ermittelt werden, was sich aber mit den in der Praxis vorkommenden Fahrzeuggrößen recht effizient machen läßt.

Negativ-Beispiel Die sechs Rollstuhlfahrer und sechs Gehbehinderten aus dem vorigen Beispiel wohnen diesmal so verteilt, daß jeweils zwei Rollstuhlfahrer und zwei Gehbehinderte in einem Ort wohnen. Es stehen drei Rollstuhlbusse zur Verfügung.

Die Rollstuhlfahrer würden auf zwei Rollstuhlbusse verteilt und die Untertouren unnötig lang, da pro Teilort ein Rollstuhlbus mit Sitzbank geschickt werden könnte. Allerdings benötigt der betrachtete Algorithmus nur zwei Rollstuhlbuse und einen PKW gegenüber drei Rollstuhlbussen für die Streckenoptimierte Lösung.

5.5.6 Fahrdienste: Adaptiver Ansatz über Wohnortnähe

Hier werden die Kunden nach örtlicher Nähe ihres Startorts zu „Fahrgemeinschaften“ zusammengefaßt. Fahrzeuge und Mitarbeiter werden passend gewählt, soweit möglich.

Algorithmus Die folgenden Schritte werden wiederholt, bis alle Kunden untergebracht sind:

1. Die Kunden, die im momentan betrachteten Fahrzeug zusammen eingeplant werden, heißen *momentane Besetzung*. Die Menge der *möglichen Kunden* ergibt sich aus den noch nicht eingeplanten Kunden durch Entfernung derjenigen Kunden, die nicht mit der momentanen Besetzung zusammen fahren können, weil für die Summe der Anforderungen kein Fahrzeug oder kein passender Mitarbeiter gefunden werden kann.

2. Wähle unter den möglichen Kunden denjenigen, der zur momentanen Besetzung am nächsten wohnt. Füge diesen zur momentanen Besetzung hinzu.

3. Passe ggf. das Fahrzeug und die Mitarbeiter an, so daß alle Anforderungen der momentanen Besetzung dadurch erfüllt sind.

   Ist das Fahrzeug durch den neu hinzugekommen Kunden voll, wähle ein neues, sowie zwei Mitarbeiter. Diese sollten jeweils möglichst wenig Anforderungen erfüllen. Dadurch ist gewährleistet, daß Fahrzeug und Mitarbeiter, die am Ende der Untertour zugeordnet werden, nicht unnötig mehr als die geforderten Eigenschaften aufweisen.

Ausgabe Eine Menge von Fahrzeugbesetzungen. Die Reihenfolge jeder Untertour muß auch hier noch festgelegt werden, da die Reihenfolge, in der sie zusammengestellt wurde, nicht optimal sein muß.

Eigenschaften

Vorteile Die entstehenden Untertouren haben sehr kurze Wege zwischen den Stationen.

Nachteile Müssen sehr viele Anforderungen an Mitarbeiter und Fahrzeuge berücksichtigt werden, wird dieses Verfahren vermutlich keine Lösung liefern, da Kunden mit ähnlichen Ansprüchen zu stark auf verschiedene Fahrzeuge verteilt sind.

Anforderungen an Zeiten werden nicht berücksichtigt. Diese können teilweise beim Festlegen der Reihenfolge der Stationen in jeder Untertour erfüllt werden.

Laufzeit Bei jedem neu zu besetzenden Fahrzeugplatz müssen alle noch nicht eingeplanten Kunden hinsichtlich der Nähe der Startorte mit der momentanen Besetzung verglichen werden. Aufwand also $\mathcal{O}(w^2)$.

Positiv-Beispiel Eine Gruppe von behinderten Kindern aus verschiedenen Orten soll in die Schule gebracht werden. Manche Kinder brauchen einen Kindersitz, andere können normal transportiert werden, benötigen aber neben dem Fahrer eine Aufsichtsperson.

Es werden Untertouren geplant, auf denen jeweils sechs nahe beieinander wohnende Kinder mit einem Kleinbus abgeholt werden. Auf jeder Untertour ist ein zweiter Mitarbeiter dabei.

Negativ-Beispiel Sechs Rollstuhlfahrer und sechs Gehbehinderte sollen transportiert werden. Die Rollstuhlfahrer wohnen allerdings ziemlich weit entfernt voneinander, die Gehbehinderten jeweils in der Nähe eines Rollstuhlfahrers.

Es werden einige Fahrgemeinschaften gebildet, in denen jeweils ein bis zwei Rollstuhlfahrer mit drei Gehbehinderten kombiniert werden. Dabei werden aber schnell alle verfügbaren Rollstuhlbusse verplant, so daß nicht alle Rollstuhlfahrer untergebracht werden.

5.5.7 Individualfahrten

Hier werden aus Abrechnungsgründen keine Dienstwünsche zu Touren zusammengefaßt. Deshalb kann hier auch keine Touroptimierung stattfinden.

5.6 Generischer Ansatz

In den obigen Optimierungsideen wurden für die Planung einzelner Dienstarten unterschiedliche Vorgehensweisen bei der Zusammenstellung von Touren bzw. Untertouren gewählt. Dabei wurden meist nur einige Parameter der Dienstwünsche berücksichtigt. Aus den all diesen Heuristiken gemeinsamen Punkten läßt sich ein allgemeiner Ansatz ableiten, der versucht, alle Parameter zu berücksichtigen.

Im Mittelpunkt steht dabei eine Gütefunktion, die bewertet, wie gut ein Dienstwunsch zu einer Tour paßt.

5.6.1 Touren

Da einige wichtige Bewertungsgrößen nur aus Untertouren ermittelt werden können (z.B. setzt die Ermittlung und Bewertung von Fahrtzeiten eine konkrete Reihenfolge der Stationen voraus), ist die Grundlage der Bewertung die Gütefunktion auf Untertouren (Abschnitt 5.6.5). Daraus kann dann eine Gütefunktion für Touren abgeleitet werden, die angibt, wie gut ein Dienstwunsch D zu einer Tour T (einer Menge von Untertouren) paßt:

$$G(T,D) = \frac{1}{|T|} \cdot \sum_{U \in T} \gamma(U,D) \quad \text{mit} \quad \gamma(U,D) = \begin{cases} 0 & D \notin U \\ G(U,D) & D \in U \end{cases}$$

5.6.2 Untertouren

Jede Dienstart wird einzeln geplant. Dabei wird wochentagweise vorgegangen. Es werden regelmäßige Untertouren erstellt, die i.a. jede Woche gleich gefahren werden. Einzelne Ausnahmen von der Regelmäßigkeit (z.B. Dienste, die an einigen speziellen Tagen nicht ausgeführt werden sollen) werden hier nicht berücksichtigt, sondern erst bei der Bildung von Fahrten.

5.6.3 Einfügen von Dienstwünschen

Um mögliche Kombinationen von Untertour und Dienstwunsch zu bewerten, muß jedesmal ein Dienstwunsch in die Untertour eingefügt werden (in der Gütefunktion mit $U \cup D$ bezeichnet). Da hier zum einen die zeitlichen Vorgaben der Dienstwünsche berücksichtigt werden müssen, zum anderen aber die Fahrtstrecke zwischen den einzelnen Stationen möglichst optimal sein soll, müssen hierfür noch Heuristiken entwickelt werden, die diese Aufgabe möglichst effizient lösen.

5.6.4 Definitionen

Untertour Eine Untertour ist hier ein Tupel von Dienstwünschen

MA Die Menge aller Mitarbeiter

FZ Die Menge aller Fahrzeuge

QF Die Menge aller möglichen Mitarbeiter-Qualifikationen

DW Die Menge aller Dienstwünsche

$\mathbf{M_U}$ Der oder die Mitarbeiter, die der Untertour U zugeordnet sind

$\mathbf{F_U}$ Das Fahrzeug, das der Untertour U zugeordnet ist

5.6.5 Gütefunktion

Den Kern des generischen Algorithmus' bildet eine Gütefunktion, die ein Maß dafür angibt, wie gut ein neuer Dienstwunsch D zu einer in der Planung befindlichen Untertour U paßt. Diese Funktion bekommt eine Untertour und einen neuen Dienstwunsch als Eingabe und liefert eine Zahl zwischen 0 und 1 als Ergebnis. Je besser der Dienstwunsch zur Untertour paßt, desto höher die Bewertung.

Alle Teilfunktionen der Gütefunktion haben ebenfalls den Wertebereich $[0;1]$. Liefert eine Teilfunktion den Wert 0, muß sich auch für die gesamte Gütefunktion der Wert 0 ergeben, da der Wert 0 bedeutet, daß eine wesentliche Bedingung nicht erfüllt ist.

$$G(U,D) = \begin{cases} 0 & : \exists g_X = 0 \\ \begin{array}{l} (g_t * \Delta t_F(U,D) + g_M * M(D, M_U) + \\ g_F * F(D, F_U) + g_P * \Delta P(U \cup D)) \\ * \frac{1}{\sum g_X} \end{array} & : sonst \end{cases}$$

Dabei ist

g_X Gewichtungsfaktoren für die Teilfunktionen. Diese können vom Benutzer vorgegeben und verändert werden, um verschiedenen Bewertungsaspekten mehr/weniger Bedeutung zukommen zu lassen.

$\Delta t_F(U, D)$ Bewertung der zusätzlichen Fahrtdauer, die durch Hinzufügen des Dienstwunsches D zur Untertour U entsteht.

Um dies normieren zu können, muß eine Obergrenze für die Gesamtdauer einer Untertour festgelegt werden. Diese sei $t_{\max}$.

Die gesamte Fahrtdauer der Untertour U wird dann folgendermaßen bewertet:

$$t_F(U) = 1 - \frac{1}{t_{\max}} \cdot \sum_{i=1}^{|U|-1} t_F(D_i, D_{i+1})$$

Dabei bezeichnet $t_F(D_i, D_{i+1})$ die Fahrtzeit zwischen den Stationen D_i und D_{i+1}.

Damit ergibt sich die Bewertung der durch den Dienstwunsch D verursachten zusätzlichen Fahrtdauer als

$$\Delta t_F(U, D) = t_F(U \cup D) - t_F(U)$$

Berechnungszeit: $\mathcal{O}(|U \cup D|)$

$M(D, M_U)$ Bewertung der der Untertour U zugeordneten Mitarbeiter im Verhältnis zu den Anforderungen des neuen Dienstwunsches D.

Hierzu wird definiert:

$M \models Q \quad (M \in MA, Q \in QF) \iff$ Der Mitarbeiter M hat die Qualifikation Q.

$M \models D \quad (M \in MA, D \in DW) \iff$ Der Mitarbeiter M hat alle Qualifikationen, die der Dienstwunsch D fordert.

$D \nleftrightarrow M \quad (M \in MA, D \in DW) \iff$ Der Mitarbeiter M ist bei der Person, die den Dienstwunsch D geäußert hat, nicht erwünscht.

$D \heartsuit M \quad (M \in MA, D \in DW) \iff$ Der Mitarbeiter M gehört zu den bevorzugten Mitarbeitern der Person, die den Dienstwunsch D geäußert hat.

$w_Q = 1 - \frac{|\{M \in MA | M \models Q\}|}{|MA|}$ ist der Wert einer Qualifikation $Q \in QF$. Er wird bei der Änderung von Mitarbeiter-Qualifikationen aktualisiert und gespeichert, muß also nicht bei jeder Anfrage neu berechnet werden.

$\Delta Q(D, M)$: Menge der „überzähligen“ Qualifikationen, die Mitarbeiter M hat, die aber vom Dienstwunsch D nicht gefordert sind.

Damit können zwei Teilbewertungen berechnet werden:

$M_Q(D, M) = 1 - \frac{\sum_{Q \in \Delta Q(D,M)} w_Q}{\sum_{Q \in QF} w_Q}$

$M_V(D, M) = \begin{cases} 1 & D \heartsuit M \\ 0 & \text{sonst} \end{cases}$

Die Bewertung der Mitarbeiter ist damit

$$M(D, M) = \begin{cases} 0 & D \nleftrightarrow M \vee M \not\models D \\ \alpha \cdot M_Q(D, M) + (1 - \alpha) \cdot M_V(D, M) & \text{sonst} \end{cases}$$

Durch den Faktor α kann festgelegt werden, wie stark die Vorlieben der Kunden im Verhältnis zur Eignung eines Mitarbeiters (und dem Interesse des DRK, keine hochqualifizierten Mitarbeiter für einfache Dienste zu „verschwenden") berücksichtigt werden. Ein sinnvoller Wert könnte bei 0,8 liegen, da die durch Anforderungen an das Fahrzeug gegebenen Sachzwänge in der Praxis sicher stärker berücksichtigt werden müssen als Vorlieben der Kunden.

Berechnungszeit: $\mathcal{O}(|QF|)$

$F(D, F_U)$ Bewertung des der Untertour U zugeordneten Fahrzeugs im Verhältnis zu den Anforderungen des neuen Dienstwunsches D.

$F \models D \quad (F \in FZ, D \in DW) \iff$ Das Fahrzeug F kann die Anforderungen des Dienstwunsches D erfüllen.

$F < F' \quad (F, F' \in FZ) \iff |\{D \mid F \models D\}| < |\{D \mid F' \models D\}|$ (F kann weniger Dienstwünsche erfüllen als F')

$w_F \in [0; 1] \quad (F \in FZ)$: Wert des Fahrzeugs F, dieser wird vom Benutzer für die verschiedenen Fahrzeugtypen vorgegeben.

$$F(D, F) = \begin{cases} 0 & F \not\models D \\ 1 - w_F & \text{sonst} \end{cases}$$

Berechnungszeit: $\mathcal{O}(1)$

$\Delta P(U \cup D)$ Bewertung der Pausen, die in der Untertour U zusätzlich entstehen, wenn D aufgenommen wird.

Die Pausen der Untertour U werden folgendermaßen bewertet:

$$P(U) = 1 - \frac{1}{t_{\max}} \cdot \sum_{i=1}^{|T|-1} (\Delta t(D_i, D_{i+1}) - t_F(D_i, D_{i+1}))$$

Dabei ist $\Delta t(D_i, D_{i+1})$ die Zeit zwischen Erledigung des Dienstwunsches D_i und des darauffolgenden Dienstwunsches D_{i+1}, und $t_F(D_i, D_{i+1})$ die Fahrtzeit zwischen D_i und D_{i+1}.

Damit ergibt sich die Bewertung der durch den Dienstwunsch D verursachten zusätzlichen Pausen als

$$\Delta P(U \cup D) = \frac{1 + P(U \cup D) - P(U)}{2}$$

Berechnungszeit: $\mathcal{O}(|U \cup D|)$

Zwei weitere Gütefunktionen bewerten, wie gut eine Menge von Mitarbeitern (maximal zwei) bzw. ein Fahrzeug für eine Untertour geeignet ist:

$G_M(U)$ Bewertung der Mitarbeiter. Setzt sich zusammen aus den persönlichen Vorlieben und Abneigungen der Kunden auf der Untertour, sowie deren Anforderungen an die Qualifikation der Mitarbeiter.

$M_Q(U, M) = \frac{1}{|U|} \cdot \sum_{D \in U} M_Q(D, M)$

$(M_Q(U,M) = 1 - \prod_{D\in U} \frac{|\{M'\in MA | M' < M \wedge M' \models D\}|}{|MA|}$ ist vermutlich exakter, aber aufwendiger zu berechnen)

$M_V(U,M) = \frac{1}{|U|} \cdot \sum_{D\in U} M_V(D,M)$

$$G_M(U) = \begin{cases} 0 & \exists D \in U : D \leftrightarrow M \vee M \not\models D \\ \alpha \cdot M_Q(U,M) + (1-\alpha) \cdot M_V(U,M) & \text{sonst} \end{cases}$$

α ist der Gewichtungsfaktor, der schon bei $M(D, M_U)$ definiert wurde.

Berechnungszeit: $\mathcal{O}(|U| \cdot |QF|)$

$G_F(U)$ Bewertung des Fahrzeugs. Berücksichtigt die Fahrzeugklassen, mit denen ein Kunde transportiert werden darf, sowie dessen sonstige Anforderungen an die Konfiguration.

$$G_F(U) = \begin{cases} 0 & \exists D \in U : F \not\models D \\ \frac{1}{|U|} \cdot \sum_{D\in U} F(D,F) & \text{sonst} \end{cases}$$

$(G_F(U) = \begin{cases} 0 & \exists D \in U : F \not\models D \\ 1 - \prod_{D\in U} \frac{|\{F'\in FZ | F' < F \wedge F' \models D\}|}{|FZ|} & \text{sonst} \end{cases}$ wäre hier wohl exakter, aber aufwendig zu berechnen)

Berechnungszeit: $\mathcal{O}(|U|)$

5.6.6 Lineare Vorgehensweise

Der Algorithmus arbeitet nach dem Greedy-Prinzip und stellt eine Tour nach der anderen zusammen.

Sei W die Menge der einzuplanenden Dienstwünsche sowie T die gerade entstehende Tour.

while $W \neq \emptyset$
  Wähle beliebigen Dienstwunsch $D \in W$ (oBdA den anspruchsvollsten)
  Wähle F_T und M_T optimal bezüglich $G_F(T)$ bzw. $G_M(T)$.
  Kunde-gefunden := true
  $T := \{D\}; W := W \setminus \{D\}$
  while (Fahrzeug nicht voll) und (Tour nicht zu lang) und (Kunde-gefunden)
    Sortiere W bezüglich der Gütefunktion $G(T, D_i)$ für alle $D_i \in W$
    Kunde-gefunden := false
    for D_k := (bester Dienstwunsch)
    to (schlechtester Dienstwunsch $\geq$ Minimalschranke)
      if $\exists$ Mitarbeiter $M'_T \in$ MA und Fahrzeug $F'_T \in$ FZ,
        die die Tour $T \cup D_k$ ausführen können
      then
        $T := T \cup D_k; \quad W := W \setminus \{D_k\}$
        $M_T := M'_T; \quad F_T := F'_T$
        Kunde-gefunden := true
      end if
    end for

end while
Speichere Tour T
end while

Aufwandsabschätzung Sei w die Anzahl der Dienstwünsche, t die Anzahl der entstehenden Touren. Die äußere While-Schleife wird für jede Tour einmal ausgeführt: $\mathcal{O}(t)$. Die innere While-Schleife wird höchstens so oft ausgeführt, bis das Fahrzeug voll ist (im Falle von Fahrdiensten) oder die Tour insgesamt so lang wird, daß die Mitarbeiter ihr Arbeitszeitsoll erfüllt haben. Dies ist also durch eine Konstante abschätzbar.

Innerhalb der inneren While-Schleife muß für jeden Dienstwunsch die Güte berechnet werden: $\mathcal{O}(w)$.

Die innere For-Schleife wird im schlimmsten Fall $\mathcal{O}(w)$ mal ausgeführt.

Insgesamt wird also die Gütefunktion $\mathcal{O}(t \cdot w)$ mal aufgerufen, die Suche nach passenden Mitarbeitern und Fahrzeugen muß bis zu $\mathcal{O}(t \cdot w^2)$ mal durchgeführt werden.

5.6.7 Parallele Vorgehensweise

Ein anderer Ansatz stellt nicht eine Tour nach der anderen zusammen, sondern nimmt Dienste, die nicht zu einer Tour passen, als Ausgangspunkt für eine neue Tour. Jeder Dienstwunsch wird derjenigen Tour zugeordnet, zu der er bezüglich der Qualitätsfunktion am besten paßt.

Wähle beliebigen Dienstwunsch $D \in W$ (oBdA den anspruchsvollsten)
$T_1 := \{D\}; W := W \setminus \{D\}; t := 1$
Wähle M_{T_1} und F_{T_1} optimal bezüglich $G_M(T_1)$ bzw. $G_F(T_1)$.
while $W \neq \emptyset$
  forall $D_i \in W$
    forall $j = 1 \dots t$
      berechne $G(T_j, D_i)$
    end forall
    $G_{\text{opt}}(D_i) := \min_j G(T_j, D_i)$
  end forall
  Wähle $D := \max_i D_i$ bezüglich G_{opt}.
  Prüfe Touren nach absteigendem $G(T_j, D)$
    if $T_j \cup D$ möglich (bzgl. MA und Fz) und $G(T_j, D) \geq$ Minimalschranke
    then
      $T_j := T_j \cup D$; Passe M_{T_j} und F_{T_j} an
    else
      $t := t + 1; T_t := \{D\}$
      Wähle M_{T_t} und F_{T_t} optimal bezüglich $G_M(T_t)$ bzw. $G_F(T_t)$.
    end if
  $W := W \setminus \{D\}$
end while

Aufwandsabschätzung Sei wieder w die Anzahl der Dienstwünsche, t die Anzahl der entstehenden Touren. Die äußere While-Schleife wird hier für jeden

Dienstwunsch einmal ausgeführt: $\mathcal{O}(w)$.

Die Gütefunktion muß für jede Kombination aus Dienstwunsch und Tour berechnet werden: $\mathcal{O}(t \cdot w)$.

Insgesamt finden also $\mathcal{O}(t \cdot w^2)$ Aufrufe der Gütefunktion statt, die Suche nach passenden Mitarbeitern und Fahrzeugen muß bis zu $\mathcal{O}(t \cdot w)$ mal durchgeführt werden.

5.6.8 Variante der parallelen Vorgehensweise

Eine Laufzeit von $\mathcal{O}(t \cdot w^2)$ Aufrufen der Gütefunktion kann bei realen Werten von etwa $w = 140$ und $t = 40$ bereits recht groß werden ($40 \cdot 140^2 = 784000$). Deshalb könnte die folgende Variante des obigen Algorithmus in der Praxis wertvoller sein. Dabei werden die Dienstwünsche nach Anfangszeit sortiert und dann in dieser Reihenfolge eingeplant.

Die Gewinne bei der Laufzeit werden allerdings mit vermutlich schlechteren Ergebnissen erkauft. Da die Dienstwünsche zunächst nur hinsichtlich ihrer Anfangszeit gewählt werden, werden hier möglicherweise Dienstwünsche zu Touren kombiniert, deren Bewertung nur knapp die Minimalschranke übersteigt. Im vorigen Ansatz wurde dagegen immer der bestpassende Dienstwunsch gewählt. Dies kann möglicherweise durch Anwendung eines inkrementellen Optimierungsverfahrens auf die berechneten Touren ausgeglichen werden.

Sortiere Dienstwünsche nach Anfangszeit
$t := 0$
while $W \neq \emptyset$
  $D :=$ frühester Dienstwunsch $\in W$
  Wähle $T_{\text{opt}} = \min_i T_i$ bezüglich $G(T_i, D)$
  if $T_{\text{opt}} \neq \emptyset \wedge G(T_{\text{opt}}, D) \geq$ Minimalschranke **then**
    $T_{\text{opt}} := T_{\text{opt}} \cup D$
    Passe $M_{T_{\text{opt}}}$ und $F_{T_{\text{opt}}}$ an
  else (keine passende Tour gefunden)
    $t := t + 1; T_t := \{D\}; W := W \setminus \{D\}$
    Wähle M_{T_t} und F_{T_t} optimal bezüglich $G_M(T_t)$ bzw. $G_F(T_t)$.
  end if
end while

Aufwandsabschätzung Die While-Schleife wird für jeden Dienstwunsch ausgeführt, die Gütefunktion muß hier aber für jeden Dienstwunsch höchstens t mal ausgeführt werden.

Es finden also $\mathcal{O}(t \cdot w)$ Aufrufe der Gütefunktion statt.

5.6.9 Einsatzbeispiele

Durch passende Gewichtung der Teilfunktionen bei der Bewertung der Untertouren können mit diesem generischen Algorithmus die den oben vorgeschlagenen speziellen Heuristiken zugrundeliegenden Ideen verwirklicht werden.

- Setzt man $g_W = 1$ und $g_M = g_F = g_A = g_P = 0$, wird immer derjenige Dienstwunsch zu einer Tour dazugenommen, der die gesamte Fahrtdauer (und damit auch den Weg) am wenigsten verlängert. Dies entspricht dem Ansatz aus 5.5.2

- Die starke Berücksichtigung der Anforderungen an das Fahrzeug in 5.5.5 kann durch ein entsprechend hohes Gewicht g_F erreicht werden. Somit könnte dieser Ansatz durch $g_W = g_F = 1$ und $g_M = g_A = g_P = 0$ simuliert werden.

6 Inkrementelle Optimierung

Die bisherigen Algorithmen gingen immer davon aus, daß eine Dienstart komplett automatisch optimiert werden soll, daß also noch keine Dienstwünsche in bereits vorhandenen Touren zusammengefaßt sind.

Der inkrementelle Ansatz geht hingegen von einer bestehenden Tourenmenge aus, in die weitere Dienstwünsche integriert werden sollen. Es können hiermit also neue Dienstwünsche einfach in das Tourensystem integriert werden, ohne eine komplette Neuplanung durchführen zu müssen. Eine weitere Anwendungsmöglichkeit besteht darin, durch nachträgliche Umgruppierung einzelner Dienstwünsche ein automatisch erzeugtes Tourensystem noch weiter zu verbessern.

6.1 Generischer Algorithmus zur Erweiterung von Touren

Da der in 5.6.8 vorgestellte Algorithmus letztlich nichts anderes tut, als Dienstwünsche auf Touren zu verteilen, kann er mit minimalen Änderungen so erweitert werden, daß damit eine Menge von Dienstwünschen auf eine Menge von Touren verteilt wird.
Eingabe:

- Eine Menge von Touren $\mathcal{T} = \{T_1, \ldots, T_n\}$ einer Dienstart (ist diese leer, verhält sich der Algorithmus genau wie der in 5.6.8)
- Eine Menge W von Dienstwünschen derselben Diensart wie die Touren
- Eine Minimalgüte $g_{\min}$. Ergibt die Bewertung, wie gut ein Dienstwunsch D zu einer Tour T paßt, weniger als $g_{\min}$, darf der Dienstwunsch nicht in diese Tour integriert werden. In diesem Falle wird eine neue Tour begonnen, die nur diesen Dienstwunsch enthält.

Ausgabe: Erweiterte Tourenmenge $\mathcal{T}$

Sortiere Dienstwünsche nach Anfangszeit
$t := n$
while $W \neq \emptyset$
  $D :=$ erster Dienstwunsch $\in W$
  Wähle $T_{\text{opt}} = \min_i T_i$ bezüglich $G(T_i, D)$
  if $T_{\text{opt}} \neq \emptyset \wedge G(T_{\text{opt}}, D) \geq g_{\min}$ **then**
    $T_{\text{opt}} := T_{\text{opt}} \cup D$
    Passe $M_{T_{\text{opt}}}$ und $F_{T_{\text{opt}}}$ an
  else (keine passende Tour gefunden)
    $t := t + 1; T_t := \{D\}; W := W \setminus \{D\};$
    Wähle M_{T_t} und F_{T_t} optimal bezüglich $G_M(T_t)$ bzw. $G_F(T_t)$.

$\mathcal{T} := \mathcal{T} \cup \{T_t\}$
end if
end while

6.2 Einfügen von Dienstwünschen in ein bestehendes Tourensystem

Neue Dienstwünsche können nun einfach in das bestehende Tourensystem integriert werden: Für jede Dienstart wird der obige Algorithmus aufgerufen, als Parameter werden alle bestehenden Touren dieser Dienstart sowie die neu einzufügenden Dienstwünsche übergeben.

Um noch bessere Touren zu erreichen, könnte man die Reihenfolge verändern, in der die Dienstwünsche in die Touren eingefügt werden. Anstatt nach der Anfangszeit könnten diese nach z.B. Anforderungen an Mitarbeiter sortiert werden (siehe 5.4). Denkbar wäre auch, den Algorithmus mit allen möglichen bzw. einer bestimmten Anzahl zufälliger Permutationen der Dienstwunschmenge aufzurufen. Man erhält dann mehrere Tourenmengen und kann davon die beste wählen (wie diese bestimmt werden kann, wird in 6.4 erläutert).

6.3 Inkrementelles Verbessern durch Umgruppieren von Dienstwünschen

Ein Versuch, eine Menge von Touren zu verbessern, besteht darin, diese Touren aufzulösen und die Dienstwünsche wie in 6.2 auf die restlichen Touren zu verteilen.

Sei $\mathcal{T}$ eine Menge von Touren, sowie $T_1, \dots, T_i \in \mathcal{T}$ die Touren, die verteilt werden sollen. Dies wird durch folgenden Aufruf des Algorithmus' aus 6.1 realisiert:

$$\mathcal{T}' = \text{Optimiere}(\mathcal{T} \setminus \{T_1, \dots, T_i\}, \{D \mid D \in \bigcup_{k=1}^{i} T_k\}, g_{\min})$$

Ist nun die Bewertung $G(\mathcal{T}') > G(\mathcal{T})$, wurde mit $\mathcal{T}'$ tatsächlich eine bessere Tourenmenge gefunden, die dann an Stelle von $\mathcal{T}$ benutzt werden kann.

6.4 Bewertung Tourenmengen

Um entscheiden zu können, ob sich eine Tourenmenge durch Umordnung von Dienstwünschen verbessert hat, muß eine Bewertungsfunktion vorliegen, mit der Mengen von Touren verglichen werden können.

6.4.1 Bewertung einzelner Touren

Zunächst einmal müssen einzelne Touren bewertet werden können. Dies geht ganz analog zu 5.6.1 durch Mittelwertbildung über die Untertourgüten:

$$G(T) = \frac{1}{|T|} \cdot \sum_{U \in T} \gamma(U) \quad \text{mit} \quad \gamma(U) = \begin{cases} 0 & D \notin U \\ G(U) & D \in U \end{cases}$$

6.4.2 Bewertung einzelner Untertouren

Da in 5.6.5 bereits eine Gütefunktion angegeben wurde, die angibt, wie gut ein Dienstwunsch zu einer Untertour paßt, läßt sich die Bewertung einer fertigen Untertour leicht aus deren Komponenten ableiten:

$$G(U) = (g_M * G_M(U) + g_F * G_F(U) + g_A * A(U) + g_P * P(U)) \cdot \frac{1}{\sum g_X}$$

Dabei ist

g_X Gewichtungsfaktoren für die einzelnen Teilfunktionen

$G_M(T)$ Bewertung des/der Mitarbeiter, die der Untertour U zugeordnet sind (wie in 5.6.5 definiert).

Berechnungszeit: $\mathcal{O}(|U| \cdot |QF|)$

$G_F(U)$ Bewertung des Fahrzeugs, das der Untertour U zugeordnet ist (wie in 5.6.5 definiert).

Berechnungszeit: $\mathcal{O}(|U|)$

$A(U, F_U)$ Bewertung der Auslastung des Fahrzeugs im Verhältnis zu seiner Kapazität:

$$A(U, F) = \frac{|U|}{K(F)}$$

Dabei ist K(F) die Kapazität des Fahrzeugs (=maximal mögliche Fahrgastzahl ohne Mitarbeiter).

Berechnungszeit: $\mathcal{O}(1)$

$P(U)$ Bewertung der Pausen zwischen den Stationen einer Untertour (wie in 5.6.5 definiert).

Berechnungszeit: $\mathcal{O}(|U|)$

6.4.3 Bewertung einer Tourmenge

Sei $\mathcal{T} = \{T_1 \ldots, T_n\}$ eine Menge von Touren. Diese soll hinsichtlich der Kriterien aus 5.2 sowie der Güte der einzelnen Touren bewertet werden. Der Einfluß der verschiedenen, sich teilweise widersprechenden Kriterien kann durch Gewichtungsfaktoren angepaßt werden.

$$G(\mathcal{T}) = (\alpha \cdot \frac{1}{n} + \beta \cdot \sum_{i=1}^{n} t(T_i) + \gamma \cdot \sum_{i=1}^{n} G(T_i)) \cdot \frac{1}{\alpha + \beta + \gamma}$$

Dabei ist

α Gewichtung der Touranzahl

β Gewichtung der Bewertung der Tourdauer (=Einsatzzeit von Mitarbeitern und Fahrzeugen). Diese hängt von den unterschiedlichen Dauern der Untertouren der Tour T ab: $t(T) = \frac{1}{|T|} \cdot \sum_{U \in T} t(U)$

γ Gewichtung der Einzeltourgüten

Um die Optimierungskriterien gemäß 5.2 zu berücksichtigen, könnten bei 10 Dienstwünschen Werte von etwa $\alpha = 20, \beta = 3, \gamma = 2$ benutzt werden.

7 Die Benutzungsoberfläche

Bereits unmittelbar nach Abschluß der Anforderungsanalyse wurde parallel zur Spezifikation ein Prototyp der Benutzungsoberfläche entwickelt. Dieser wurde dem Kunden am 13.2.98 vorgeführt und stellt auf anschauliche Art die Zusammenhänge im Datenmodell sowie die geplanten Funktionen dar. Insbesondere kann aus den Bildschirmmasken genau entnommen werden, welche Daten der Anwender eingeben muß und was das Programm ausgeben und berechnen soll. Der Prototyp erhebt jedoch keinen Anspruch auf Vollständigkeit, da spätere Veränderungen am Datenmodell noch nicht in den Prototyp mit aufgenommen wurden.

7.1 Konsistenztests

Konsistenztests werden durchgeführt, wenn die Eingabe einer Maske abgeschlossen wurde. Welche Konsistenztests durchgeführt werden, ist bei den einzelnen Masken selbst beschrieben. (Zu Konsistenztest siehe auch Abschnitt **??** in der Anforderungsanalyse.)

7.2 Hauptmenu

Die wesentlichen Funktionen können über ein Menu aufgerufen werden, das aus verschiedenen, thematisch geordneten Untermenus besteht:

Daten Schnittstelle zum persistenten Speicher. Hier können Kunden, Touren, Ressourcen und Einstellungen gespeichert und wieder geladen werden.

Kunden Pflege des Kundenbestands, Erfassung von neuen Kunden und deren Dienstwünschen.

Touren Ansicht und Eingabe der Touren und Untertouren, automatische Berechnung neuer Touren und Untertouren

Datenausgabe Ausgabe von Plänen und Übersichtsdiagrammen auf Bildschirm oder Drucker

Analyse Diverse Funktionen um Fahrzeug- und Ressourcenbelegung und -auslastung zu ermitteln und darzustellen.

Ressourcen Verwaltung der Mitarbeiter und Fahrzeuge

Einstellungen Definition verschiedener grundlegender Daten: Mögliche Mitarbeiterqualifikationen, Fahrzeugtypen und -konfigurationen, Arten von Sitzhilfen und angebotene Essensarten. Außerdem können hier Parameter für die Bewertung von Touren bei der automatischen Planung sowie für die Berechnung von Fahrtzeiten und -kosten aus den Streckendaten definiert werden.

7.3 Dienste anfordern

In den Masken gibt es Ein- und Ausgabefelder. Während der Benutzer Eintragungen in den Eingabefeldern vornimmt, werden in den Ausgabefeldern Informationen aus den Stammdaten angezeigt, die u.U. aus den Eingabefeldern berechnet werden.

Zur Anforderung eines Dienstes kann es erforderlich sein, daß eine oder mehrere Kontaktpersonen (siehe 7.11) angegeben werden, die später auf den Tourplänen erscheinen sollen.

7.3.1 Essen auf Rädern

Die zugehörige Eingabemaske zeigt Abbildung 9.

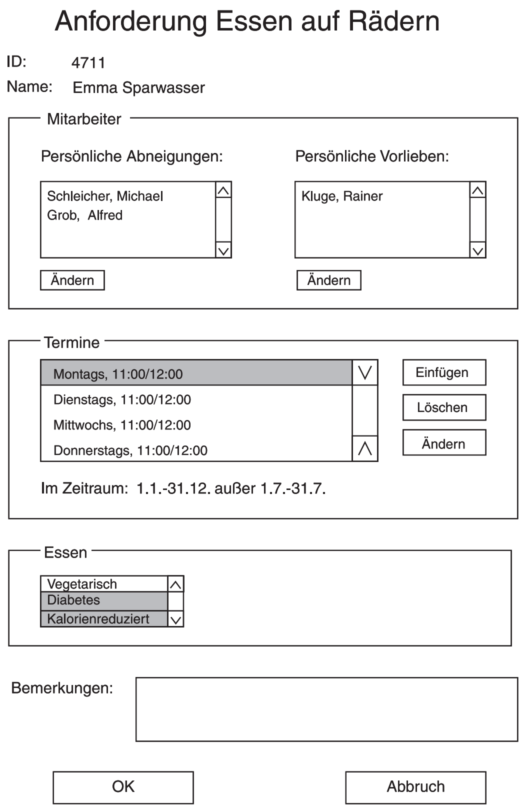

Abbildung 9: Dienstwunsch Essen auf Rädern

Eingaben des Benutzers

- Eine Menge von Terminen, an denen der Dienst stattfinden soll (siehe 7.3.5).
- Die Essensart. Diese kann aus einer Liste mit vorgegeben Essensarten gewählt werden, die z.B. verschiedene Diätessen enthält
- Bemerkungen. Hier kann ein beliebiger Text eingegeben werden, der dann im Tourplan erscheint und dem Mitarbeiter zusätzliche Informationen geben kann.

Ausgaben des Programms

- Kundennummer und -name. Für wen wird hier ein Dienst bestellt?

- Die persönlichen Zu- und Abneigungen des Kunden. (Diese Angaben sind auch änderbar)

Änderungen und Ergänzungen

- Auswahl, ob es sich um warmes oder kaltes Essen handelt.
- Es wird nicht die Art eines Essens bestimmt (durch Auswahl verschiedener kombinierbarer Eigenschaften), sondern Anzahl und Art der unterschiedlichen Essen (aus einer Auswahlliste mit lieferbaren Essensarten)

7.3.2 MSD/APD

Die zugehörige Eingabemaske zeigt Abbildung 10.

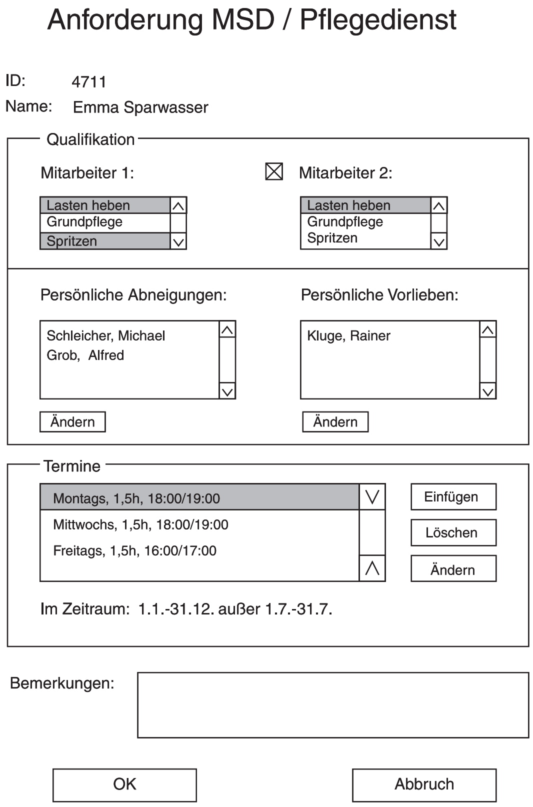

Abbildung 10: Dienstwunsch MSD/APD

Eingaben des Benutzers

- Ist ein zweiter Mitarbeiter für den Dienst nötig?
- Die erforderlichen Qualifikationen der Mitarbeiter für diesen Dienst
- Eine Menge von Terminen

- Bemerkungen

Ausgaben des Programms

- Kundennummer und -name
- Persönliche Zu- und Abneigungen des Kunden

Bemerkungen Bei der Auswahl der Qualifikationen der einzelnen Mitarbeiter kann nicht angegeben werden, daß einer der beiden Mitarbeiter die benötigte Qualifikation besitzen muß, sondern die Qualifikation hängt fest an dem Mitarbeiter, für den sie angegeben wurde. Diese Lösung schränkt zwar die Allgemeinheit ein, ist aber leichter zu handhaben.

7.3.3 Individualfahrt

Die zugehörige Eingabemaske zeigt Abbildung 11.

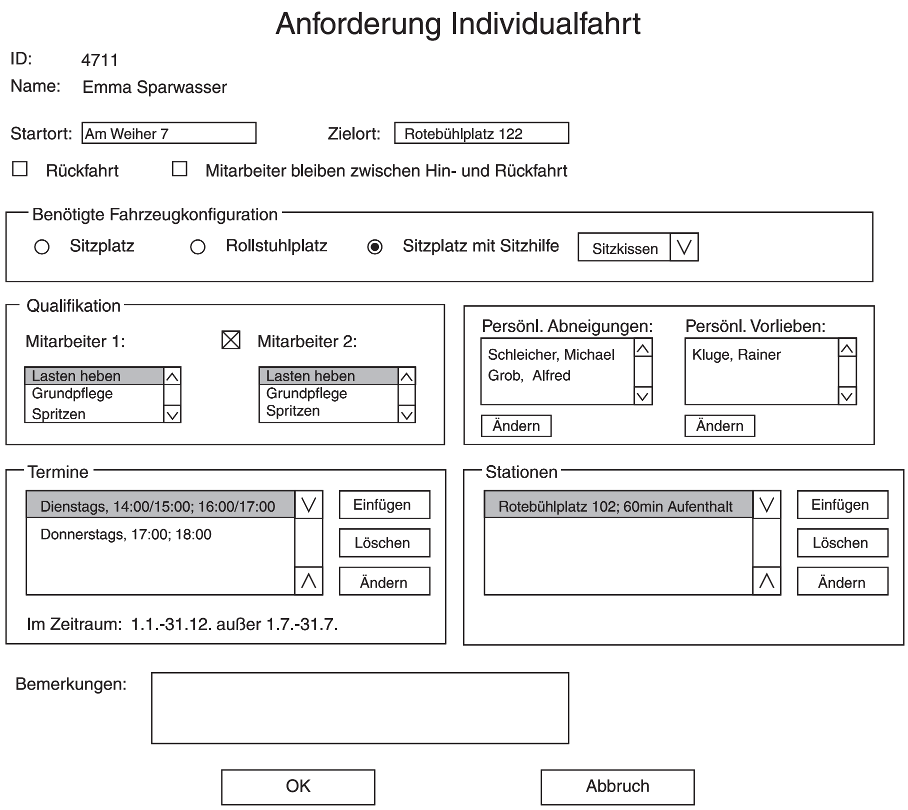

Abbildung 11: Dienstwunsch Individualfahrt

Eingaben des Benutzers

- Start- und Zielort der Fahrt
- Soll auch eine Rückfahrt vom DRK durchgeführt werden?

- Müssen die Mitarbeiter zwischen Hin- und Rückfahrt bleiben, z.B. um dem Kunden am Ziel bei seinen Tätigkeiten zu helfen?
- Ansprüche an die Konfiguration des Fahrzeugs (siehe 7.7.1), z.B. Rollstuhlplatz benötigt.
- Die erforderlichen Qualifikationen der Mitarbeiter für diesen Dienstwunsch
- Eine Menge von Terminen
- Eine Menge von Zwischenstationen, die auf der Fahrt angefahren werden sollen. Diese werden durch Adresse und Aufenthaltsdauer bestimmt.
- Bemerkungen

Ausgaben des Programms

- Kundennummer und -name
- Persönliche Zu- und Abneigungen des Kunden

Änderungen und Ergänzungen

- Die Angabe, ob eine Rückfahrt erwünscht ist, wurde herausgenommen. Diese Entscheidung stellt keine Einschränkung dar und dient zur einfacheren Handhabung des Systems. Wird trotzdem eine Rückfahrt gewünscht, so läßt sich diese durch Anhängen der Zieladresse an die Stationenliste modellieren.
- Die Angabe, ob die Mitarbeiter zwischen Hin- und Rückfahrt bleiben, wurde zur einfacheren Bedienung in die Eingabemaske für die Termine verschoben.

7.3.4 Schulfahrt, Dialysefahrt, Tagespflege

Für diese Dienste werden die selben Informationen eingegeben und angezeigt, die bereits für Individualfahrten im vorigen Abschnitt beschrieben wurden. Nur die Liste mit den Zwischenstationen kann hier entfallen.

7.3.5 Termine

Termine beschreiben, an welchen Tagen, zu welcher Zeit und in welchem Rhythmus ein Dienst ausgeführt wird. Die Eingabemaske für einen Fahrdienst zeigt Abbildung 12.

Eingaben des Benutzers

Zeitraum Eine Menge von Zeitspannen, innerhalb derer (ggf. eingeschränkt durch die Häufigkeit) der Dienst stattfindet. Der Zeitraum wird durch zwei Angaben charakterisiert:

> **Zeitrahmen** Ein umschließender Zeitraum (z.B. Schuljahr)
>
> **Ausnahmen** Zeiten innerhalb des Zeitrahmens, in denen der Dienst nicht stattfinden soll (z.B. Schulferien)

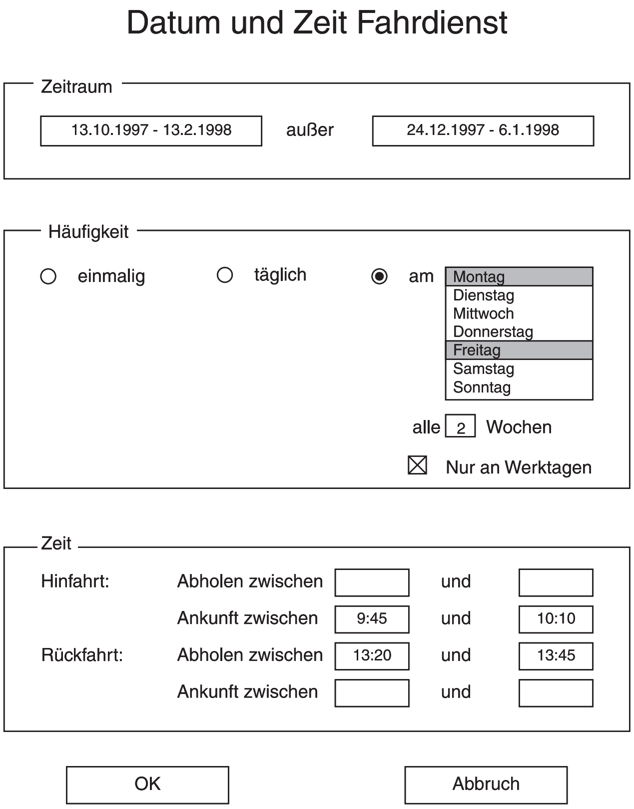

Abbildung 12: Termin für einen Fahrdienst

Beides kann sowohl ein längerer Zeitraum (wie Schuljahr und Ferien) sein, als auch eine Menge einzeln angegebener Tage (um einen Dienst an exakt diesen Tagen zu beschreiben).

Häufigkeit Für regelmäßige Dienste, die sich innerhalb des angegebenen Zeitraums periodisch wiederholen, kann hier ein Wiederholungsrhythmus angegeben werden:

einmalig Genau einmal (hier sollte der Zeitraum nur einen Tag umfassen)

täglich Jeden Tag innerhalb des angegebenen Zeitraums

regelmäßig Hier können diejenigen Wochentage gewählt werden, an denen der Dienst stattfinden soll. Zusätzlich kann angegeben werden, ob er nur an Werktagen stattfinden soll oder auch an Feiertagen, die auf einen Werktag fallen. Neben wöchentlicher Wiederholung kann ein beliebiger n-Wochen-Rhythmus angegeben werden.

Zeit Hier werden die Uhrzeiten erfaßt, an denen der Dienst an den angegebenen Tagen stattfindet. Abhängig von der Art des Dienstes müssen verschiedenen Information eingegeben werden:

Aufenthalt Hierunter fallen Dienste, bei denen Mitarbeiter zu einem Kunden fahren und dort eine gewisse Zeit verbringen, um für den

Kunden oder mit ihm verschiedene Tätigkeiten zu verrichten (z.B. MSD/APD). Nötige Angaben sind eine Zeit für den Beginn des Dienstes (optional kann auch über eine Zeitspanne der frühestmögliche/späteste Abhol- bzw. Ankunftstermin angegeben werden), sowie die Dauer.

Transport Fahrten, bei denen der Kunde an einem Ort abgeholt und zu einem anderen Ort befördert wird, brauchen folgende Angaben: Einen Zeitpunkt für den Start und für die Ankunft (optional kann auch hier eine Zeitspanne angegeben werden). Ist eine Rückfahrt erwünscht, müssen dafür dieselben Daten erfaßt werden.

Lieferung Für eine Lieferung, bei der dem Kunden eine Ware in die Wohnung gebracht wird (z.B. Essen auf Rädern), muß die Zeit(spanne) für die Ankunft beim Kunden festgehalten werden.

Änderunge und Ergänzungen

- Wie in 7.3.3 schon erläutert, wurde die Angabe, ob die Mitarbeiter zwischen Hin- und Rückfahrt bleiben, in diese Maske verschoben.
- Der Benutzer kann angeben, ob das System eine Warnmeldung ausgeben soll, wenn der Dienstwunsch auf einen Feiertag fällt (damit entschieden werden kann, ob der Dienst verlegt werden soll). Diese Warnung wird aktiviert, wenn aus Untertouren Fahrten gebildet werden (z.B. beim Erstellen der Dienstpläne). Dazu wird aber eine Eingabemöglichkeit für Feiertage benötigt.

7.4 Obertour eingeben

Die zugehörige Eingabemaske zeigt Abbildung 13.

Eingaben des Benutzers

- Die Tournummer, falls die vom System erzeugte nicht erwünscht ist
- Die Art des Dienstes
- Der oder die Mitarbeiter, die normalerweise die Fahrten fahren, die aus dieser Tour erzeugt werden.
- Das Fahrzeug, mit dem die Touren in der Regel gefahren werden
- Eine Menge von Dienstwünschen. Die Reihenfolge der Stationen spielt hier keine Rolle.

Außerdem kann der Benutzer die gesamte Tour oder einzelne Stationen für die Optimierung sperren.

Ausgaben des Programms

- Die Tournummer. Dies ist eine Vorgabe, die der Benutzer ändern kann.
- Die Qualifikation der gewählten Mitarbeiter
- Anforderungen der auf der Tour bedienten Kunden bezüglich der Mitarbeiter. Damit kann der Benutzer bereits bei der Eingabe prüfen, ob die vorgesehenen Mitarbeiter zu den Kunden passen.

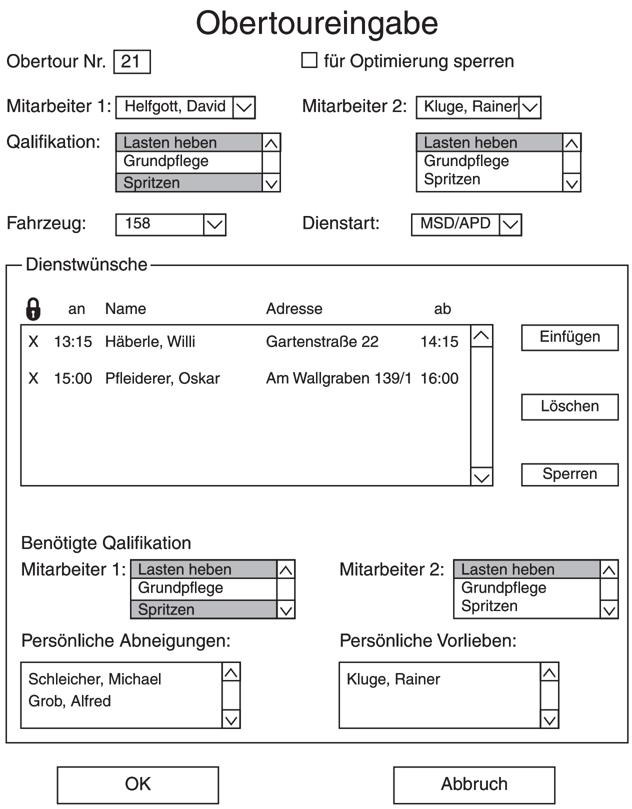

Abbildung 13: Eingabe einer Obertour

Konsistenztest Es wird geprüft, ob die Qualifikationen der eingegebenen Mitarbeiter mit den von den Kunden geforderten Qualifikationen übereinstimmen. Ebenso wird das Fahrzeug mit den Anforderungen der Dienstwünsche abgeglichen.

7.5 Untertour eingeben

Die zugehörige Eingabemaske zeigt Abbildung 14.

Eingaben des Benutzers

- Die Untertournummer, falls die vom System erzeugte nicht erwünscht ist
- Die Wochentage, an denen die Tour gefahren wird
- Die Abfahrtszeit und den Abfahrtsort
- Die Reihenfolge der Stationen der Tour
- Ob das Fahrzeug vor der nächsten Tour wieder zur Dienststelle zurück muß

Außerdem kann der Benutzer die gesamte Untertour oder einzelne Dienstwünsche für die Optimierung sperren.

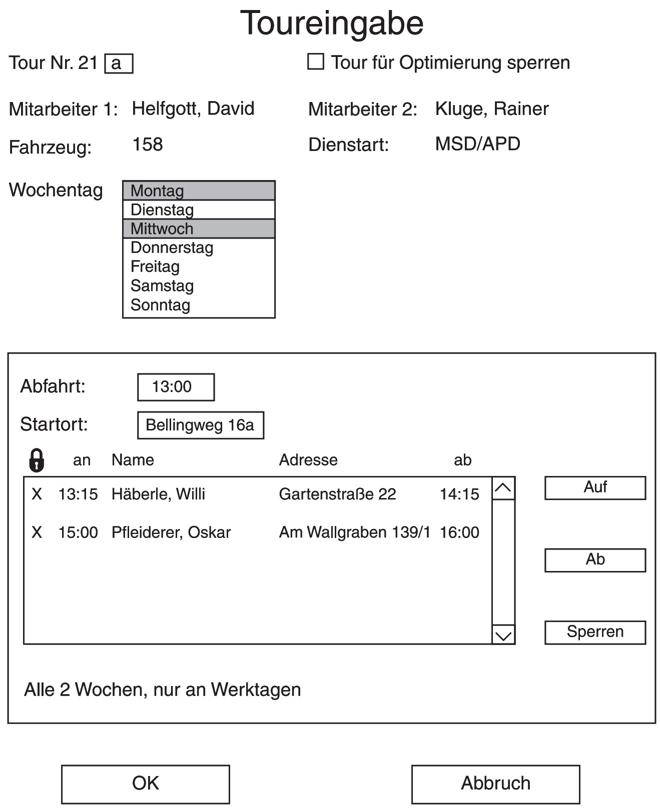

Abbildung 14: Eingabe einer Untertour

Ausgaben des Programms

- Die Tournummer. Dies ist eine Vorgabe, die der Benutzer ändern kann.
- Die Art des Dienstes
- Der oder die Mitarbeiter, die die Tour fahren sollen
- Das Fahrzeug, mit dem die Tour gefahren werden soll
- Die Dienstwüsche, die in der Tour definiert wurden.
- Der aus den Dienstwünschen ermittelte Wochenrhythmus und ggf. die Einschränkung auf Werktage.

7.6 Mitarbeiter eingeben

Die zugehörige Eingabemaske zeigt Abbildung 15.

Eingaben des Benutzers

- Persönliche Daten des Mitarbeiters: Name, Adresse, Geburtsdatum, Telephonnummer
- Eintrittsdatum

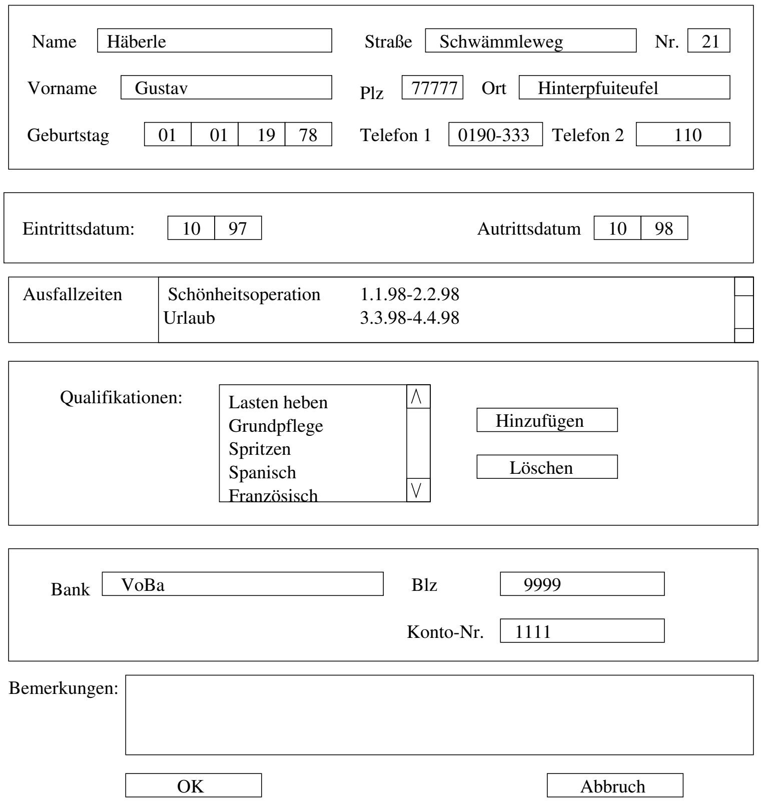

Abbildung 15: Mitarbeiter eingeben

- Austrittsdatum
- Ausfallzeiten: Eine Liste mit Zeiträumen, an denen der Mitarbeiter nicht zu Diensten eingeteilt werden kann.
- Qualifikationen: Aus einer Liste mit Qualifikationen, die für die Dienstplanung eine Rolle spielen, kann hier gewählt werden, welche Qualifikationen ein Mitarbeiter hat
- Die Bankverbindung des Mitarbeiters
- Bemerkung

7.7 Fahrzeug eingeben

Die zugehörige Eingabemaske zeigt Abbildung 16.

Eingaben des Benutzers

- Modell
- amtliches Kennzeichen
- interne Fahrzeugnummer
- Anzahl der Kfz-Schlüssel, die das DRK besitzt

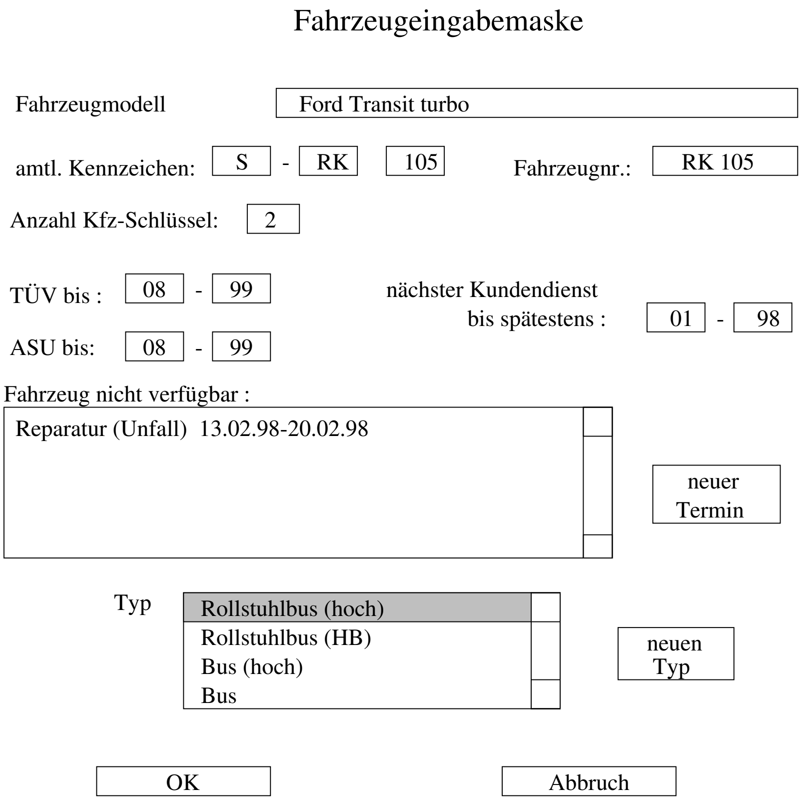

Abbildung 16: Fahrzeug eingeben

- Termine, wann das Fahrzeug in die Werkstatt muß (und damit nicht für die mobilen Dienste zur Verfügung steht): HU, ASU, Kundendienst
- Fahrzeugtyp. Aus einer Liste, die eine für die Planung wichtige Einteilung der Fahrzeuge in verschiedene Typen liefert, muß ein Typ gewählt werden. Diese Typenliste kann ggf. auch geändert bzw. erweitert werden. (Siehe 7.8)
- Aktuelle Konfiguration. Eine der für den Fahrzeugtyp möglichen Konfigurationen (siehe 7.7.1). Darüberhinaus die Anzahl der verschiedenen Sitzhilfen, die sich momentan im Fahrzeug befinden.

7.7.1 Fahrzeugkonfiguration

Eine Konfiguration beschreibt eine von mehreren Umbaumöglichkeiten eines Fahrzeugs. So können z.B. durch Ausbau einer Sitzbank zusätzliche Rollstuhlplätze geschaffen werden.

Eingaben des Benutzers

- Anzahl der Sitzplätze

- Anzahl der Sitzplätze mit fester Sitzhilfe (z.B. eingebauter Kindersitz)
- Anzahl der Rollstuhlplätze
- Sind Schienen für einen Kühlcontainer vorhanden (in dem tiefgekühltes Essen ausgeliefert wird)?

7.8 Fahrzeugtyp eingeben

Der Fahrzeugtyp beschreibt die Zusammenfassung mehrerer Fahrzeugmodelle und deren Konfigurartion unter einem einheitlichen Namen. Die Konfigurationen einzelner Modelle unter der gleichen Typbezeichnung müssen die gleichen sein.

Eingaben des Benutzers

- Typbezeichnung: Eindeutiger Bezeichner, der den Fahrzeugtyp von allen anderen unterscheidet.
- Beschreibung: Beliebiger Text, der die Typbezeichnung ergänzen kann.
- Kilometerpreis: Der Preis mit einem Fahrzeug dieses Typs gefahrenen Kilometers.

7.9 Kunden eingeben

Die zugehörige Maske zeigt Abbildung 17

Eingaben des Benutzers

- Persönliche Daten: Name, Vorname, Geburtsdatum, Adresse, Telefon
- Schlüssel: Anzahl der Hausschlüssel, zusätzlich kann optional ein Bezeichner für den Hausschlüssel vergeben werden (z.B. die Kundennummer).
- Rechnungsempfänger : siehe 7.10
- Rollstuhl: Angabe, ob der Kunde einen Rollstuhl besitzt.
- Hilfsmittel: ist aus einer Liste auszuwählen. Es kann auch festgelegt werden, ob dieses benötigte Hilfsmittel vom DRK gestellt werden muß, oder ob es der Kunde selbst mitbringt. Diese Liste kann auch erweitert und geändert werden.
- Fahrtdauer: Maximale Zeit, die der Kunde im Fahrzeug verbringen soll. Unabhängig davon existiert eine allgemeine Obergrenze (beim DRK momentan zwei Stunden), die für alle Kunden gilt und auf keinen Fall überschritten werden darf.
- Passender Fahrzeugtyp: Auswahl und Freigabe aus der Liste der definierten Fahrzeugtypen, mit denen der Kunde transportiert werden kann. (Siehe 7.8)
- Kontaktpersonen: siehe 7.11
- Bemerkungen
- persönliche Vorlieben und Abneigungen: neu, ändern und löschen
- Dienstwünsche: neu, ändern und löschen

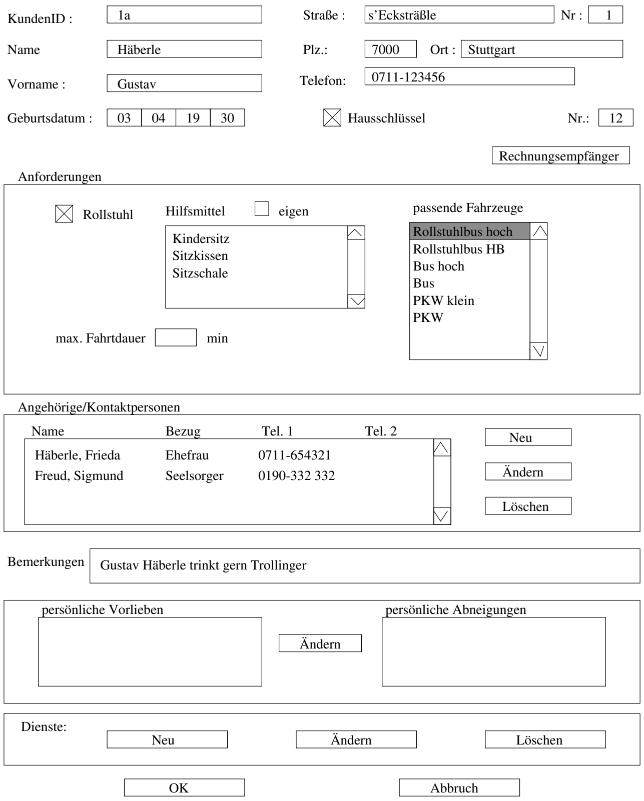

Abbildung 17: Kunden eingeben

Ausgaben des Programmes

- Kundennummer: Wird vom Programm vorgegeben, kann aber vom Benutzer durch eine andere Kundennummmer ersetzt werden.
- Listen der schon bekannten Rechnungsempfänger, Kontaktpersonen und Dienstwünsche.

7.10 Rechnungsempfänger eingeben

Unter Rechnungsempfängeren versteht man diejenigen Personen oder Institutionen, die für die Bezahlung eines oder mehrerer Dienstwünsche aufkommen. Der Kunde selbst kann dabei ebenfalls als Rechnungsempfänger auftreten. Ebenso kann der Dienstwunsch eines Kunden mehrere Rechnungsempfänger haben.

Eingaben des Benutzers

- Name des Rechnungsempfängers
- Adresse und Telefon
- Ansprechpartner: z.B. bei Institutionen der Name des zuständigen Sachbearbeiters

- Bankverbindung: Name der Bank, BLZ und Kontonummer
- Dienstwunsch: Auswahl aus einer Liste, die die vom Kunden geäußerten Dienstwünsche umfaßt
- Bemerkung

7.11 Kontaktperson eingeben

Bezugspersonen sind Personen, die bei unvorhergesehenen Ereignissen benachrichtigt werden sollen, z.B. Eltern, Hausarzt, ...

Eingaben des Benutzers

- Name
- Adresse und Telefon
- Art des Bezugs: Bezeichnung, in welchem Verhältnis die Bezugsperson zum Kunden steht (z.B. Mutter oder Hausarzt)
- Bemerkung

Literatur

[Aig93] Aigner, Martin: *Diskrete Mathematik.* Friedr. Vieweg & Sohn, Postfach 58 29, D-65048 Wiesbaden, Deutschland, 1993.

[AL95] Appelrath, Hans-Jürgen und Jochen Ludewig: *Skriptum Informatik.* B. G. Teubner, Stuttgart, Dritte Auflage, 1995.

[AMO93] Ahuja, R. K., Magnanti und J. B. Orlin: *Network flows: theory, algorithms, and applications.* Prentice-Hall, Englewood Cliffs, NJ 07632, USA, 1993.

[Bal96] Balzert, Helmut: *Lehrbuch der Softwaretechnik,* Band 1: Software-Entwicklung. Spektrum-Verlag, 1996.

[BFP+72] Blum, M., R.W. Floyd, V.R. Pratt, R.L. Rivest und R.E. Tarjan: *Time bounds for selection.* J. Computer and System Sciences, 7:488–461, 1972.

[Boo94] Booch, G.: *Objektorientierte Analyse und Design.* Addison-Wesley, 1994.

[Buc94] Buchholz, Friedhelm: *Komplexität des Fahrgemeinschaften-Problems, Studienarbeit Nr. 1327.* Institut für Informatik der Universität Stuttgart, 1994.

[Buc95] Buchholz, Friedhelm: *Entwurf eines Systems zur Vermittlung von Fahrgemeinschaften, Diplomarbeit Nr. 1226.* Institut für Informatik der Universität Stuttgart, 1995.

[FB74] Finkel, R.A. und J. L. Bentley: *Quad Trees - A Data Structure for Retrieval on Composite Keys.* Acta Informatica, 4:1–9, 1974.

[FH95] FREUDER, EUGENE C. und PAUL D. HUBBE: *A Disjunctive Decomposition Schema for Constraint Satisfaction*, Kapitel 17. In: [SvH95], 1995.

[Fla83] FLAJOLET, P.: *On the performance evaluation of extendible hashing and trie searching.* Acta Informatica, 20:345–369, 1983.

[FNPS79] FAGIN, R., J. NIEVERGELT, N. PIPPENGER und H.R. STRONG: *Extendible hashing – a fast access method for dynamic files.* ACM Trans. Database Systems, 4(3):315–344, 1979.

[FP86] FLAJOLET, P. und C. PUECH: *Partial match retrieval of multidimensional data.* Journal of the Association for Computing Machinery, 33(2):371–407, 1986.

[Gir87] GIRGENSOHN, ANDREAS: *ObjCon – Constraints in einer objektorientierten Sprache, Diplomarbeit Nr. 509.* Institut für Informatik der Universität Stuttgart, 1987.

[GJ78] GAREY, MICHAEL R. und DAVID S. JOHNSON: *Computers and Intractibility: A guide to the theory of NP-Completeness.* W. H. Freeman, San Francisko, 1978.

[Gle95] GLEICHER, MICHAEL: *Practical Issues in Graphical Constraints*, Kapitel 21. In: [SvH95], 1995.

[Grü90] GRÜN, BRIGIT: *Ein formularorientiertes Constraint-System, Studienarbeit Nr. 960.* Institut für Informatik der Universität Stuttgart, 1990.

[GT88] GOLDBERG, ANDREW V. und ROBERT E. TARJAN: *A New Approach to the Maximum-Flow Problem.* Journal of the Association for Computing Machinery, 35(4):921–940, Oktober 1988.

[Jon79] JONES, A.: *The Object Model: A Conceptual Tool for Structuring Software.* Springer Verlag, 1979.

[KW85] KRISHNARMUTHY, R. und K.-Y. WHANG: *Multilevel Grid Files – IVM Research Report*, 1985.

[Meh84] MEHLHORN, KURT: *Data Structures and Algorithms.* Springer-Verlag, 1984.

[NHS84] NIEVERGELT, J., H. HINTERBERGER und K. C. SEVICK: *The Grid File: An Adaptable, Symmetric Multikey File Structure.* ACM Transactions on Database Systems, 9(1):38–69, März 1984.

[OW93] OTTMANN, T. und P. WIDMAYER: *Algorithmen und Datenstrukturen.* BI Wissenschaftsverlag, 1993.

[Pro96] PROJEKTGRUPPE EVOLUTIONÄRE ALGORITHMEN: *Zwischenbericht – Bericht 1996/10.* Institut für Informatik der Universität Stuttgart, 1996.

[Pro97] PROJEKTGRUPPE FAHRGEMEINSCHAFTEN: *Zwischenbericht – Bericht 1997/10.* Institut für Informatik der Universität Stuttgart, 1997.

[PS95] PREPARATA, F.P. und M.I. SHAMOS: *Computational Geometry.* Springer-Verlag, 1995.

[Reg85] REGNIER, M.: *Analysis of grid file algorithms.* BIT, 25(2):335–357, 1985.

[Riv76] RIVEST, R.L.: *Partial-match retrivel algorithms.* SIAM Journal of Computing, 5(1):19–50, 1976.

[Sah74] SAHNI, SARTAJ: *Computationally related problems.* SIAM Journal of Computing, 3(4):262–279, dec 1974.

[San95] SANELLA, MICHAEL: *The SkyBlue Constraint Solver and Its Applications*, Kapitel 20. In: [SvH95], 1995.

[SB84] STEFIK, M. und D. BOBROW: *Object-Orientied Programming : Themes and Variations.* AI Magazine, 6(4):40–62, 1984.

[Sch97] SCHÄFFER, THOMAS: *Constraint Programming.* [Pro97].

[Sed91] SEDGEWICK, R.: *Algorithmen in C.* Addison-Wesley, 1991.

[SvH95] SARASWAT, VIJAY und PASCAL VAN HENTENRYCK: *Principles and Practice of Constraint Programming.* MIT Press, 1995.

[Tar91] TARJAN, ROBERT ENDRE: *Data Structures and Network Algorithms.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1991.

[Tri96] TRICK, MICHAEL A.: *Network Optimization.* http://mat.gsia.cmu.edu/OR/networks/networks.html, 1996.

[Wal92] WALTER, WOLFGANG: *Analysis 1.* Springer-Verlag, Zweite Auflage, 1992.