Diplomarbeit

Erwerb funktionaler, räumlicher und kausaler Beziehungen von Fahrzeugteilen aus einer technischen Dokumentation

Mark Siebert

Diplomarbeit
am Fachbereich Informatik
der Universität Dortmund

29. Januar 1997

Betreuer :

Prof. Dr. Katharina Morik
Dipl.-Inform. Peter Brockhausen

Erklärung

Hiermit erkläre ich, die vorliegende Arbeit selbständig verfaßt und keine anderen als die hier angegebenen Hilfsmittel verwendet, sowie Zitate kenntlich gemacht zu haben.

Dortmund, im Januar 1997

Mark Siebert

Abstract

Wenn man aus sehr großen Datenbanken lernt, ist eine Reduzierung der Komplexität sehr wichtig. Eine Möglichkeit, um die Komplexität zu verringern, ist die Verwendung von Hintergrundwissen. Aber Hintergrundwissen liegt oft in einer unstrukturierten Form vor. Damit man es aber verwenden kann, muß es zuerst strukturiert werden.
In dieser Arbeit wird ein Ansatz vorgestellt, wie Hintergrundwissen über Teile von Fahrzeugen strukturiert werden kann. Dabei werden aus vorhanden Wissensquellen einstellige Fakten erzeugt. Dies kann zum einen automatisch erfolgen, zum anderen ist ein Interpretationsschritt notwendig. Anschließend wird mit Hilfe des Sort Taxonomie Tools STT von MOBAL [2] eine Klassenhierarchie über Fahrzeugteilen generiert.

Inhaltsverzeichnis

1 Einleitung *6*

1.1 Wissenserwerb und Wissensmodellierung bei Expertensystemen **6**

1.2 Ziel und Übersicht dieser Arbeit **9**

2 Wissensquellen und Wissenserwerb *13*

2.1 Arten von Beziehungen zwischen Bauteilen **13**

2.2 Ansatz : Modellierung funktionaler Abhängigkeiten zwischen Fahrzeugkomponenten durch Material- und Energieflüsse **15**

2.2.1 Material- und Energieflüsse 15
2.2.2 Verkettung von Flüssen 16
2.2.3 Flußbildung durch Verkettung 17

2.3 Analyse von Wissensquellen **18**

2.4 Praxiserfahrungen beim Wissenserwerb **19**

3 Darstellung von MOBAL und STT *20*

3.1 STT **21**

3.1.1 Extensionen von Sorten 21
3.1.2 Äquivalenzklassen von Sorten 23
3.1.3 Schnittsorten 26

4 Das Schadensbuch der Daimler Benz AG *28*

4.1 Gründe für den Einsatz des Schadensbuchs **28**

4.2 Aufbau der Schadensbuchs **28**

4.2.1 Struktur des Schadensbuchs 29
4.2.2 Aufbau einer Funktionsuntergruppe 31
4.2.3 Das Schadensbuch in gedruckter Form 33
4.2.4 Das Schadensbuch in maschinenlesbarer Form 35

4.3 Verschlüsselung von Schäden an Fahrzeugkomponenten **38**

5 Gruppenbildung anhand des Schadensbuches *40*

5.1 Gruppenbildung mit der Datei SSLBEZUG **41**

5.2 Gruppenbildung mit der Datei SSLTEIL durch Textinterpretation **42**

6 Generierung von MOBAL - Fakten aus dem Schadensbuch : DOCTOFAC *46*

6.1 Darstellung von DOCTOFAC **47**

6.1.1 Die Eingabedatei von DOCTOFAC 47
6.1.2 Die Konfigurationsdatei von DOCTOFAC 47
6.1.3 Die Ausgabedatei von DOCTOFAC 51
6.1.4 Die Arbeitsweise von DOCTOFAC 52

6.2 Anwendung von DOCTOFAC auf die Datei SSLBEZUG **54**

6.2.1 Einschränkung von SSLBEZUG auf SSL0-93 54
6.2.2 Erstellen der Konfigurationsdateien 55

6.3 Anwendung von DOCTOFAC auf Einzelteilgruppen **56**

6.4 Realisierung von DOCTOFAC **57**

7 Klassifikation von Schadensteilen mit STT *59*

7.1 Bilden der Äquivalenzklassen 59

7.2 Berechnung von Schnittsorten 60

7.3 Eigenschaften der generierten Klassen 61

7.4 Ergebnisse 62

8 Überführen der generierten Klassen in ein Datenbankformat 65

8.1 Auslesen der generierten Klassen 66

8.2 Umformen der genierten Klassen mit MOBTODB 67

8.2.1 Die Eingabedatei von MOBTODB 68

8.2.2 Die Konfigurationsdatei von MOBTODB 68

8.2.3 Die Ausgabedatei von MOBTODB 70

8.2.4 Die Arbeitsweise von MOBTODB 70

8.3 Realisierung von MOBTODB 71

9 Beispieldurchlauf mit dem PKW-Schadensbuch 73

9.1 Generierung von Fakten mit DOCTOFAC 73

9.2 Generierung von Klassen mit STT 74

9.3 Auslesen der generierten Klassen und Bearbeitung mit MOBTODB 75

9.4 Generierte Dateien des Beispieldurchlaufs 76

10 Literatur 84

11 Anhang 86

11.1 Programmlistings 86

11.1.1 DOCTOFAC 86

11.1.2 MOBTODB 94

11.1.3 LIB 99

11.2 Einzelteilgruppen 105

1 Einleitung

1.1 Wissenserwerb und Wissensmodellierung bei Expertensystemen

Die Wirtschaftssysteme der Industriestaaten haben sich in der Zeit nach dem zweiten Weltkrieg entscheidend gewandelt. In der Zeit unmittelbar nach dem Krieg hatte man ein produktionsorientiertes Wirtschaftssystem, das durch Massenproduktion gekennzeichnet war. Im Mittelpunkt stand die kostengünstige Produktion. Eine kleine Anzahl von Anbietern stand einer großen Anzahl von Kunden gegenüber. Im Laufe der Zeit kam es aber immer mehr zu Sättigungserscheinungen, und die Konkurrenz zwischen den Unternehmen wurde immer größer. Sie mußten sich nicht mehr hauptsächlich auf die Produktion konzentrieren, sondern der Verkauf trat in den Vordergrund (Verkäufermarkt). In unserer heutigen Überflußgesellschaft hat sich diese Situation der Unternehmen abermals verschärft. Vorhandene Märkte müssen gepflegt werden, um vorhandene Kunden nicht an Konkurrenten zu verlieren. Neue Märkte für Produktinnovationen müssen systematisch erschlossen werden. Einerseits sind die Märkte schnellebiger geworden und es kommen immer häufiger neue Produkte auf den Markt. Dadurch bedingt wird im allgemeinen der Marktzyklus eines Produktes immer kürzer. Andererseits werden die Produkte immer komplizierter und technisch aufwendiger, was wiederum zu längeren Entwicklungszeiten führt (äußerst dynamischer Verkäufermarkt). Die neuen Marktgegebenheiten verlangen von Unternehmen die Fähigkeit, sich schnell an veränderte Marktgegebenheiten anpassen zu können. Ein Schlagwort zur Lösung der Probleme beim Übergang vom Verkäufermarkt zum äußerst dynamischen Verkäufermarkt ist das Konzept des „Lean Production“. Geprägt wurde dieser Begriff von japanischen Unternehmen, die „bessere Produkte in kürzerer Zeit zu geringeren Kosten“ produzieren konnten. In der betriebswirtschaftlichen Literatur bezeichnet dies der folgende Sachverhalt:

Unter Lean Production versteht man [7] :
„Ein ganzheitliches, kundenorientiertes und resourcenminimales Zusammenwirken einzelner Bereiche des Unternehmens in einer optimal abgestimmten Prozesskette unter organisatorischer Einbindung von Zulieferern und Händlern.“

Im Rahmen des Lean Production bedient man sich auch mehr und mehr der Expertensysteme. Sie kommen in unterschiedlichen Teilkonzepten des Lean Production zum Einsatz, wie zum Beispiel in der Sicherheit eines Fertigungsprozesses.

Weitere Einsatzmöglichkeiten für Expertensysteme sind [6] :
- Interpretation (Auswertung und Analyse von Informationen)
- Diagnostik (Ursachenforschung bei Systemfehlern durch Dateninterpretation)
- Planung von Aktionsfolgen zur Erreichung eines gewünschten Zustandes
- Beratung

In der heutigen Zeit ist das technische Wissen ein entscheidender Wettbewerbsfaktor geworden. Die Maschinen und Anlagen werden immer aufwendiger, und zu ihrer Konstruktion wird Wissen von Experten benötigt. Aber Maschinen werden nicht nur aufgestellt und zur Produktion eingesetzt. Sie müssen bedient und in regelmäßigen Abständen gewartet werden, wofür oft fachlich speziell ausgebildetes Personal benötigt wird. Für alle diese Aufgaben ist man auf Experten angewiesen, da nur sie das nötige

Wissen besitzen. Ideal wäre es beispielsweise, wenn man das Wissen der Konstrukteure einer Anlage im Leitstand zur Verfügung hätte, um zukünftige Fehlfunktionen anhand von Meßergebnissen vorzeitig erkennen zu können. Da man aber einen Leitstand nicht Rund um die Uhr mit Experten besetzen kann, bieten Expertensysteme die Möglichkeit, einen Teil des Wissen des oder der Experten dem Bedienungspersonal zur Verfügung zu stellen.
Hierbei handelt es sich nur um ein kleines Beispiel für den Einsatz von Expertensystemen. Ihr Ziel ist es aber, Expertenwissen unabhängig vom Experten verfügbar zu machen, um andere Experten oder auch Nichtexperten zu unterstützen. Vorhandene Experten sollen oder können durch Expertensysteme nicht vollständig ersetzt werden.
Nun könnte man meinen, diese Aufgabe kann auch durch herkömmliche Programme erfüllt werden. Handelt es sich dabei um statisches Wissen ist das auch möglich. Allerdings ist das nur in den seltensten Fällen so, da immer wieder neues Wissen entsteht und altes Wissen revidiert werden muß. Erfolgt dabei keine Trennung zwischen Wissen und Programmcode, führt eine Änderung des Wissens auch zu einer Änderung des Programmcodes. Ein Expertensystem unterscheidet sich aber gerade durch diese Trennung von anderen Programmen. Es enthält einen Teil, der das Wissen enthält und sich leicht ändern läßt (Wissensbasis) und einen Teil, der das in der Wissensbasis vorhandene Wissen verarbeitet (Inferenzteil). Durch die Trennung von Wissen und Inferenzteil bleibt das Expertensystem unabhängig von der bearbeiteten Domäne und kann so durch einen Austausch der Wissensbasis für weitere Domänen eingesetzt werden.
Bei der Verarbeitung des Wissens durch den Inferenzteil wird aus dem vorhandenen Wissen neues Wissen hergeleitet, das in die Wissensbasis aufgenommen wird. Dieses Wissen kann wiederum zur Herleitung weiteren Wissens dienen kann. So kann man versuchen, zu einer gegebenen Hypothese eine Lösung zu finden, indem das Expertensystem versucht, die Hypothese anhand des Wissens in der Wissensbasis zu bestätigen oder zu widerlegen. Damit der Benutzer diese Lösungsschritte nachvollziehen kann, enthält das Expertensystem eine weitere Komponente, die Erklärungskomponente. Sie hilft dem Benutzer zu verstehen, auf welchem Wissen bestimmte Schlüsse beruhen, so daß er Lösungswege nachvollziehen kann.
Ein Expertensystem besteht also aus Komponenten, die Wissen darstellen, neues Wissen aus vorhandenem Wissen herleiten und dem Benutzer den Weg der Herleitung erklären, sowie der Möglichkeit, dem System neues Wissen von außen hinzuzufügen oder vorhandenes Wissen zu entfernen. Man kann also zwischen folgenden Komponenten differenzieren [6] :

- **Wissensbasis**
  Die Wissensbasis enthält das gesamte Wissen des Expertensystems. Innerhalb der Wissensbasis kann man noch zwischen bereichsbezogenem Wissen, und fallspezifischem Wissen unterscheiden. Das bereichsbezogene Wissen enthält das Wissen eines Experten, das sich während der Verwendung des Expertensystems durch einen Benutzer nicht ändert. Das fallspezifische Wissen dagegen wird vom Benutzer während der Verwendung eingegeben. Außerdem enthält die Wissensbasis noch Zwischen- und Endergebnisse, die während der Verwendung hergeleitet werden.

- **Problemlösungskomponenten (Inferenzteil)**
  Die Problemlösungskomponente verarbeitet das in der Wissensbasis gespeicherte Wissen und leitet daraus neues Wissen her. Auf diese Weise wird versucht, zu einem gegebenen Problem eine Lösung zu finden. Dabei wird der Problemlösungsprozeß durch die in die Problemlösungskomponente eingebauten Strategien gesteuert.

- **Dialogkomponente**
  Die Dialogkomponente führt den Dialog mit dem Benutzer.

- **Erklärungskomponente**
  Die Erklärungskomponente hilft dem Benutzer das Vorgehen des Expertensystems bei einer Problemlösung nachzuvollziehen. So kann er sehen, wie aus vorhandenem Wissen neues Wissen hergeleitet wurde.

- **Wissenserwerbskomponente**
  Die Wissenserwerbskomponente dient der Erhebung und Formalisierung des Wissens eines Anwendungsgebietes. Dabei ist sie gerade im Rahmen der Interaktion zwischen Experte und Wissensingenieur von Bedeutung und spielt insbesondere zum Aufbau und zur Wartung der Wissensbasis eine Rolle.

Sowohl die Bezeichnungen, aber auch die Aufgaben der einzelnen Komponenten eines Expertensystems sind in der Literatur nicht eindeutig. So werden einzelne Komponenten anders bezeichnet oder ihnen wird ein anderer Aufgabenbereich zugeordnet. Ein Beispiel dafür ist die Regelerwerbskomponente [18], deren Aufgabe ähnlich der Wissenserwerbskomponente ist, die aber einen anderen Namen trägt. Wichtig ist aber vor allem die Trennung von Wissensbasis und Problemlösungskomponente.

Nun wird die Qualität eines Expertensystems von der Qualität des Wissens in der Wissensbasis entscheidend beeinflußt. Um das Expertensystem nutzen zu können, muß die Wissensbasis aber erst aufgebaut werden. Und gerade hier existiert ein entscheidendes Problem, das als „Flaschenhals des Wissenserwerbs beim Aufbau von Expertensystemen“ bezeichnet wird [12]. Der Flaschenhals besteht darin, das Wissen von einem Experten zu extrahieren und in die Wissensbasis zu übertragen. Dabei ist zum einen das Extrahieren des Wissens von einem Experten problematisch, da der Experte nicht von selbst sein gesamtes Wissen preisgibt. Man kann zwar einiges Wissen durch Gespräche und Fragen an den Experten erhalten und vertiefen, aber trotzdem bleiben immer noch Fertigkeiten des Experten verborgen, die ihn als Experten auszeichnen. Dazu gehören beispielsweise Problemlösungsstrategien des Experten, die ihm selbst nicht bewußt sind, die er aber korrekt einsetzt. Zum anderen existiert ein Problem dabei, das extrahierte Wissen in die Wissensbasis zu übertragen, da die Wissensbasis ein bestimmte Form des Wissens fordert.
Nun sind Experten nicht die einzigen Wissensquellen. Auch Bücher, Bilder und Filme enthalten Wissen, das aber ebenfalls erst extrahiert und in eine Wissensbasis übertragen werden muß. Der Wissenserwerb mit Hilfe von Experten ist besonders schwierig, da man es hier mit Menschen zu tun hat, auf deren Eigenarten man sich einstellen muß. Andererseits bietet die Arbeit mit Experten den Vorteil, das man Wissen erwerben kann, das in Texten nicht direkt oder gar nicht enthalten ist. Es existiert eine Reihe von

Methoden, um Wissen von Experten zu erwerben. Man unterscheidet direkte Methoden, deren primäres Ziel es ist, Sprechwissen von Experten zu erwerben und indirekte Methoden, die darauf abzielen Handlungswissen und Denkstrukturen des Experten zu erkennen. Zu den direkten Methoden zählen Fragebogen, Interview, Fallreportage, Protokollprüfung und Beobachtungsanalyse [5], [13]. Sie eignen sich dazu, in der Anfangsphase des Wissenserwerbs grundlegendes Wissen zu erwerben, das dem Experten bewußt ist und das er auch erklären kann. Zu den indirekten Methoden zählen „Geordnete Bäume", „Mehrdimensionales Skalieren" , „Johnsons hierarchical clustering", „Gewichtete Netzwerke" und die „Konstuktgitteranalyse" [5], [14]. Sie eignen sich dazu, in den späteren Phasen des Wissenserwerbs die Denkstrukturen und die Repräsentation von Begriffen des Experten zu erwerben. So ist beispielsweise interessant, aufgrund welcher Kriterien er einige Begriffe als ähnlich ansieht.
Nun darf man nicht davon ausgehen, daß das erworbene Wissen von einem Experten vollständig ist oder sich problemlos in eine Wissensbasis übertragen läßt. Ein Experte besitzt Wissen, das ihm selbst nicht bewußt ist, so daß er es nicht weitergeben kann. Außerdem liegt das erworbene Wissen in einer natürlichsprachlichen Form vor. Will man dieses Wissen formalisieren, muß man es erst interpretieren und geeignet repräsentieren. Dabei handelt es sich um einen Modellierungsschritt, für den es unterschiedliche Ansätze gibt, wie „KADS"[17] und „Sloppy Modeling" [12]. Während die meisten Ansätze versuchen, die Modellierung sequentiell durchzuführen, das heißt das Wissen wird erworben und anschließend in mehreren Schritten in ein Modell überführt, geht „Sloppy Modeling" davon aus, das man niemals erwarten kann ein vollständiges, korrektes und angemessenes Modell für einen Sachbereich zu finden [18]. Statt dessen wird sich ein noch so gründlich erarbeitetes Modell als unvollständig, inkorrekt oder unangemessen erweisen. Aus diesem Grund fordert „Sloppy Modeling" ein System, das den Wissenserwerb unterstützt und dem Benutzer möglichst viel Freiheit gibt. Dadurch wird die Modellierungstätigkeit zu einem kreativen Prozeß. Dem Benutzer ist es freigestellt, ob er dem System Wissen hinzufügt oder Wissen revidiert, wann er Widersprüche auflöst oder wann er welches Lernverfahren einsetzt. Wichtig beim „Sloppy Modeling" ist, daß alle diese Schritte vom System unterstützt werden und alle Entscheidungen revidierbar sind [18]. Eine Realisierung eines Wissenserwebssystems, das die Idee des „Sloppy Modeling" unterstützt, ist das an der GMD entwickelte System MOBAL [2] ,[11].

1.2 Ziel und Übersicht dieser Arbeit

Im vorherigen Abschnitt wurde die Problematik des Wissenserwerbs und der Wissensmodellierung bei Expertensystemen dargestellt. Ziel dieser Arbeit soll es gerade sein, Wissen zu erwerben und in einer Wissensbasis darzustellen. Ausgangspunkt bilden dabei Konstruktionsdaten von Kraftfahrzeugen, insbesondere PKW und LKW der Daimler Benz AG. Diese Konstruktionsdaten werden untersucht, mit dem Ziel, Beziehungen zwischen Bauteilen aufzudecken, und in einer Wissensbasis darzustellen. Bei den Beziehungen handelt es sich um Zusammenhänge zwischen Fahrzeugteilen, die funktionaler, räumlicher, kausaler oder zeitlicher Natur sein können, wobei sich die einzelnen Begriffe nicht gegenseitig ausschließen. Kaptitel 2.1 gibt einen Überblick über diese Arten von Beziehungen.
Nach den Ausführungen im vorherigen Kapitel könnte man meinen, daß die Probleme beim Wissenserwerb erst beginnen, wenn Wissen von Experten wirklich erworben wird. Bei dieser Arbeit begannen die Probleme aber bereits damit, eine geeignete Wissensquelle zu finden, denn ein Experte stand nicht zur Verfügung, so daß die oben erwähnten direkten und indirekten Methoden des Wissenserwerbs nicht anzuwenden waren. Im

weiteren Verlauf versuchte ich Bücher über Konstruktionsdaten zu erhalten, aber auch dies schlug fehl, so daß nur ein Buch zur Verfügung stand, das Werkstätten als Richtlinie dient, um aufgetretene Schäden an Fahrzeugen einheitlich zu beschreiben. Dieses Buch bildet die Grundlage für den Erwerb von Hintergrundwissen. Da es sowohl in gedruckter Form, als auch in Form mehrerer Textdateien vorliegt, besteht die Möglichkeit, einen Teil des Wissenserwerbs zu automatisieren. Dabei werden aus einer Textdatei einstellige Fakten erzeugt, die man direkt in MOBAL verwenden kann. Die Aufgabe der Generierung von Fakten aus der vorhandenen Textdatei übernimmt dabei das Tool DOCTOFAC. Außerdem besteht die Möglichkeit, durch die Interpretation von Texten weitere einstellige Fakten zu erzeugen, und diese zusammen mit den automatisch erzeugten Fakten in MOBAL zu verwenden. Da zu diesem Schritt Textverständnis notwendig ist, muß er von einem Menschen durchgeführt werden, was im Hinblick auf die große Mengen von Schadensteilen einen großen Arbeitsaufwand erfordert. Bei den so erzeugten Fakten handelt es sich um einstellige Prädikate, deren Argumente immer Schadensteile sind. So kann das Sort Taxonomy Tool von MOBAL (STT) verwendet werden, um Schadensteile, die an der Erfüllung der gleichen Aufgabe beteiligt sind oder die aufgrund ihrer baulichen Anordnung zusammengehören, zu einer Gruppe zusammenzufassen. Anschließen werden aus diesen Gruppen Klassen gebildet, wobei Gruppen mit gleichen Elementen zu einer Klasse zusammengefaßt werden. Die so erzeugten Klassen werden dann aus MOBAL ausgelesen und unter Verwendung des Tools MOBTODB in ein spezielles Format gebracht, so daß sie als Hintergrundwissen zum Lernen aus Datenbanken verwendet werden können.
In diesem Anwendungsfall handelt es sich um ein einheitliches Qualitäts-Informations-System (QUIS) [6] der Daimler Benz AG, in dem unter anderem Fahrzeugdaten und durchgeführte Reparaturen gespeichert sind. Im Hinblick auf eine Reduzierung der Kulanz- und Gewährleistungskosten ist es nun interessant, Zusammenhänge zwischen defekten Teilen aufzudecken. Hier werden im Rahmen von KDD (Knowledge discovery in databases) verschiedene Lernverfahren wie FOIL oder RDT/DB angewendet, um aus der Datenbank Regeln zu lernen, die Zusammenhänge zwischen Schadensteilen widerspiegeln. Aufgrund der Größe der Datenbank von 2.6 Gigabyte ist es aber wichtig, die Komplexität beim Lernen so weit wie möglich zu reduzieren. Hier kommt das erworbene Hintergrundwissen in Form der erzeugten Klassen zum Einsatz. Es ist nicht mehr nötig, Regeln zwischen einzelnen Schadensteilen zu lernen, sondern man kann sich auf Klassen von Teilen beschränken. Das führt zum einen zu einer Reduzierung der Komplexität, zum anderen können bessere Lernergebnisse erzielt werden. Das Problem besteht nämlich darin, daß identische Schäden in verschiedenen Werkstätten unterschiedlich beschrieben werden. Dabei wird für den gleichen Schaden ein jeweils anderes Schadensteil als schadhaft angegeben. Beispielsweise bezeichnet die eine Werkstatt die Einspritzpumpe als schadhaft, während eine andere Werkstatt genauer vorgeht, und das Durckregelventil der Einspritzpumpe als Schadensteil angibt. Besitzt man in diesem Fall kein Hintergrundwissen, das einen Zusammenhang zwischen beiden Teilen herstellt, kann das eingesetzte Lernverfahren diese Teile nur als völlig verschiedene Teile behandeln. Setzt man aber das Hintergrundwissen ein und bildet aufgrund dieser Zusammenhänge Klassen von Teilen, so ist egal, ob eine Werkstatt die Einspritzpumpe oder das Druckregelventil der Einspritzpumpe als schadhaft angegeben hat. Da beide Teile zu einer Klasse gehören, werden sie als Einheit angesehen, und das Lernverfahren muß nicht mehr zwischen beiden differenzieren. So kann man bessere Lernergebnisse erhalten, da man die unterschiedlichen Angaben der Werkstätten als

gleich erkennen kann. Außerdem wird die Komplexität beim Lernen verringert, da die Teile einer Klasse als Einheit angesehen werden können.
Die
Abbildung 1 enthält eine Übersicht, wie Hintergrundwissen unter Verwendung von DOCTOFAC, STT und MOBTODB erworben und zur Verfügung gestellt wird. Grundlage bildet dabei das Schadenbuch, das sowohl in gedruckter Form, als auch in maschinenlesbarer Form vorliegt. Aus einer Datei des Buches in maschinenlesbarer Form werden mit Hilfe des Tools DOCTOFAC einstellige Fakten generiert, die sich direkt in MOBAL einlesen lassen. Außerdem werden Texte, die Schadensteile beschreiben, interpretiert, und zur Bildung von Gruppen ähnlicher Teile verwendet. Dieser Interpretationsschritt muß allerdings von einem Menschen durchgeführt werden, und ist im Hinblick auf eine große Menge von Schadensteilen sehr mühsam. Anschließend können diese Gruppen mit Hilfe von DOCTOFAC in einstellige Fakten überführt werden. Aufgrund der in beiden Schritten erzeugten Fakten werden dann unter Verwendung von STT Klassen von Schadensteilen gebildet. Diese Klassen werden dann aus MOBAL ausgelesen und in ein gefordertes Format überführt.

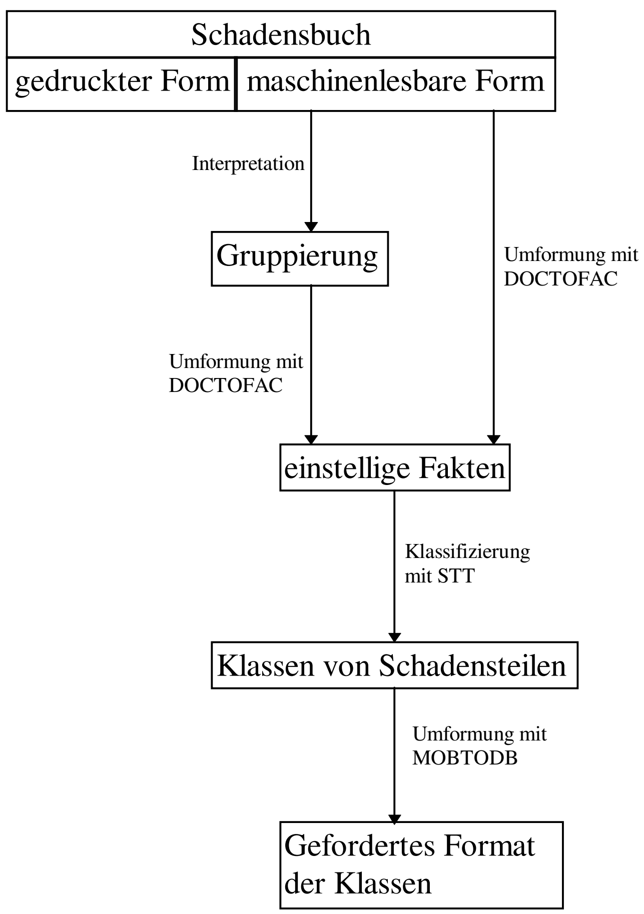


Abbildung 1 : Übersicht der Arbeit

Überblick

In Kapitel 2 werden anfangs die verschiedenen Arten von Beziehungen zwischen Bauteilen dargestellt. Danach wird ein Ansatz vorgestellt, mit dessen Hilfe man funktionale Beziehungen zwischen Bauteilen aufdecken kann, und schließlich werden vorhandene Wissensquellen analysiert. In Kapitel 3 wird die Arbeitsweise von STT vorgestellt und in Kapitel 4 wird das Schadensbuch detailliert beschrieben. In Kapitel 5 werden dann die Möglichkeiten vorgestellt, wie man aus den Dateien des Schadensbuchs in gedruckter Form Gruppen bilden kann. Kapitel 6 besteht dann aus einer Beschreibung von DOCTOFAC, während Kapitel 7 zeigt, wie STT verwendet werden kann, um aus den einstelligen Fakten Klassen von Schadensteilen zu erzeugen. Schließlich werden in Kapitel 8 die erzeugten Klassen aus MOBAL ausgelesen und mit Hilfe von MOBTODB in die gewünschte Form gebracht. Kapitel 9 besteht dann aus einem Beispiel, das die Anwendung aller vorher beschriebenen Schritte enthält.

2 Wissensquellen und Wissenserwerb

Eine wichtige Aufgabe beim Wissenserwerb besteht darin, die vorhanden Wissensquellen zu analysieren und die richtigen Wissensquellen auszuwählen. Bei der Auswahl ist es natürlich wichtig zu erkennen, welches Wissen benötigt wird. Das Ziel dieser Arbeit besteht darin, Wissen über funktionale, räumliche und kausale Beziehungen zwischen Bauteilen zu erwerben. Deshalb werden zu Beginn dieses Kapitels die Unterschiede zwischen diesen Arten von Beziehungen herausgestellt. Anschließend wird ein Ansatz dargestellt, der es ermöglicht, funktionale Beziehungen mit Hilfe von Flüssen im Fahrzeug zu beschreiben. Schließlich werden die vorhandenen Wissensquellen analysiert.

2.1 Arten von Beziehungen zwischen Bauteilen

In dieser Diplomarbeit geht es um den Erwerb von Wissen über Beziehungen zwischen Fahrzeugteilen. Von besonderem Interesse sind dabei Schäden an den Teilen und ihre Auswirkungen auf die Funktion anderer Teile. Wenn also Teil A defekt ist, kann Teil B nicht mehr korrekt arbeiten. Dieses Teil B kann dadurch ebenfalls Schaden nehmen und somit die Funktionen weiterer Teile beeinflussen. Zu Untersuchen ist nun die Art der Beziehung, die zwischen zwei Teilen besteht. Diese Beziehungen lassen sich folgendermaßen gliedern :

- **Funktionale Beziehungen**

Die Grundlage der funktionalen Beziehungen bildet die Aufgabe, die ein Teil im Fahrzeug zu erfüllen hat. Es besteht eine funktionale Beziehung zwischen zwei Teilen A und B, wenn sie an der Erfüllung derselben Aufgabe beteiligt sind. Nun kann man diese Aufgabe sehr weit fassen, wie etwa die „Fortbewegung mit einem Fahrzeug“, wobei fast alle Teile benötigt werden, oder man faßt die Aufgabe sehr eng, wie etwa „Transport des Kraftstoffes von der Benzinpumpe zum Vergaser“, wofür wesentlich weniger Teile eingesetzt werden.

- **Räumliche Beziehungen**

Die Grundlage der räumlichen Beziehungen bildet die Anordnung eines Teils im Fahrzeug. Es besteht eine räumliche Beziehung zwischen zwei Teilen A und B, wenn diese Beziehung nur aus der Anordnung von A und B im Fahrzeug abgeleitet werden kann. Auch hier hat man die Möglichkeit, die Beziehungen weit oder eng zu fassen. Beispielsweise besteht auch, wie zwischen allen Teilen, eine räumliche Beziehung zwischen der vorderen und der hinteren Stoßstange. Interessanter im Bezug auf die gegenseitige Beeinflussung zwischen den Teilen, sind aber Teile, die räumlich dicht beieinander liegen.

- **Kausale Beziehungen**

Die Grundlage kausaler Beziehungen bilden Einflüsse defekter Teile im Fahrzeug auf andere Teile. Es besteht eine kausale Beziehung zwischen zwei Teilen A und B, wenn durch einen Defekt an Teil A ein Defekt an Teil B ausgelöst wird. So kann beispielsweise eine defekte Benzinpumpe der Auslöser für einen defekten Katalysator sein, da die korrekte Funktion des Gesamtsystems nicht mehr gewährleistet ist. Ein wesentlicher Unterschied zwischen den kausalen Beziehungen und den räumlichen und funktionalen

Beziehungen besteht darin, daß kausale Beziehungen nicht in Konstruktionsplänen von Fahrzeugen enthalten sind, sondern erst durch defekte Teile hervorgerufen werden. Andererseits treten kausale Beziehungen nicht zufällig auf, sondern der Defekt eines Teils A zieht oftmals den Defekt eines bestimmten Teils B nach sich. So kann es für die Konstruktion von Fahrzeugen sinnvoll sein, diese Beziehungen zu kennen, um daraus resultierende Fehler bei zukünftigen Entwicklungen zu vermeiden.

Durch diese Trennung von funktionalen, räumlichen und kausalen Beziehungen entsteht der Eindruck, als wäre die Zuordnung zu einer dieser Beziehungsarten eindeutig. Betrachtet man aber beispielsweise die funktionalen und räumlichen Beziehungen, so passiert es häufig, daß zwei Teile, die an der Erfüllung der selben Funktion beteiligt sind, auch nah beieinander angeordnet sind. Ein Beispiel dafür wäre die Kraftstoffpumpe und die Kraftstoffleitung, die beide an der Funktion „Kraftstofförderung“ beteiligt sind, aber auch nebeneinander angeordnet sind. Würde eine defekte Kaftstoffpumpe einen Defekt an der Kraftstoffleitung nach sich ziehen, beispielsweise durch einen zu hohen Druck, hätte man zusätzlich eine kausale Beziehung. Man kann also zwei beliebige Teile nicht eine dieser 3 Arten von Beziehungen eindeutig zuordnen. Schließlich gibt es noch Beziehungen, die sich keiner dieser 3 Beziehungsarten zuordnen lassen. Ausgangspunkt ist dabei die Auswechslung eines Teils A, was zur Folge hat, daß in absehbarer Zeit ein bestimmtes Teil B defekt sein wird. Man spricht hier von zeitlichen Beziehungen. Grundlage dafür ist die Tatsache, daß sich Teile durch den ständigen Gebrauch aufeinander einspielen und so zueinander passende Abnutzungserscheinungen aufweisen. Wechselt man eines der Teile aus, paßt das neue Teil nicht mehr genau zu dem alten Teil und es kommt zu einem Defekt. Zeitliche Beziehungen sind schwer zu erfassen, da sie nicht nur von der Fahrzeugkonstruktion abhängen, sondern auch von der Art, wie das Fahrzeug benutzt wird. Die
Abbildung 2 zeigt, daß man Beziehungen zwischen Fahrzeugteilen nicht immer einer bestimmten Kategorie zuordnen kann, sondern daß es immer wieder zu Überschneidungen kommt.

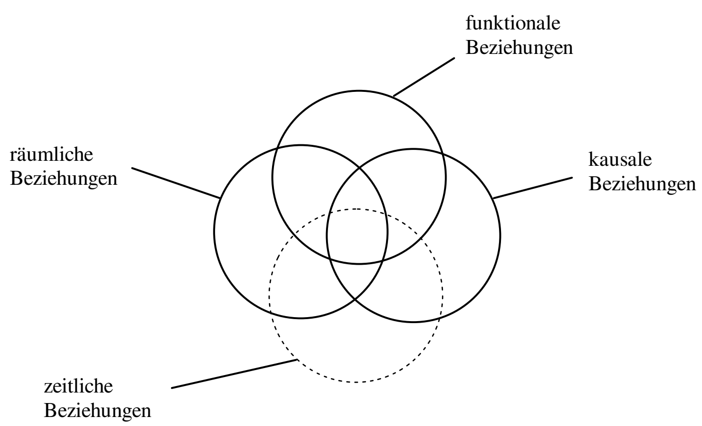


Abbildung 2 : Arten von Beziehungen zwischen Fahrzeugteilen

2.2 Ansatz : Modellierung funktionaler Abhängigkeiten zwischen Fahrzeugkomponenten durch Material- und Energieflüsse

2.2.1 Material- und Energieflüsse

Die Grundlage einer funktionalen Abhängigkeit zwischen Bauteilen bildet eine spezielle Aufgabe, an deren Erfüllung diese Bauteile beteiligt sind. Damit diese Aufgabe aber durch das Zusammenwirken dieser Teile erfüllt werden kann, müssen bestimmte Gegebenheiten vorliegen, die eine Verbindung zwischen den Teilen erzeugen. Dazu gehört selbstverständlich eine bestimmte räumliche Anordnung, denn Teile in einer beliebigen Anordnung können noch keine Aufgabe erfüllen. Aber die Anordnung der Teile allein reicht ebenfalls nicht zur Erfüllung einer Aufgabe aus. Ein weiterer wichtiger Aspekt ist die Energieeinwirkung in irgendeiner Form auf die Bauteile, die verschiedene Formen haben kann. Eine Möglichkeit ist ein Materialfluß, wie Wasser oder Benzin. Dieser Fluß wird durch bestimmte Bauteile beschrieben, die an der Erfüllung einer Aufgabe beteiligt sind, so daß zwischen ihnen eine funktionale Abhängigkeit bestehen muß. Da man aber durch Materialflüsse allein viele wichtige Funktionen im Fahrzeug nicht beschreiben kann, wird der Begriff eines Flusses um Energieflüsse erweitert. Unter Energieflüssen versteht man beispielsweise elektrischen Strom oder die Kraftübertragung vom Motor auf die Räder. Das Prinzip eines Flusses kann man sich an einem Beispiel verdeutlichen:
Ausgangspunkt ist der Kühlwasserkreislauf eines Fahrzeugs, also ein Materialfluß. Im Mittelpunkt steht das Kühlwasser, an dessen Fluß verschiedene Teile beteiligt sind. Dazu gehören beispielsweise der Kühler, die Kühlwasserpumpe, der Motor und diverse Schläuche. Damit dieser Kühlwasserkreislauf funktionieren kann, ist sicher eine bestimmte Anordnung der Teile nötig. Der gesamte Kühlwasserkreislauf erfüllt die Funktion „Kühlen der Motors“.
Konzentriert man sich nur auf Materialflüsse, klammert man einen wichtigen Fluß im Fahrzeug aus, der mit fortschreitender Technik immer mehr an Bedeutung gewinnt. Der elektrische Strom tritt heute an vielen Stellen im Fahrzeug auf, an denen man ihn gar nicht vermuten würde und ist somit an der Erfüllung der unterschiedlichsten Aufgaben beteiligt. So könnte eine Störung eines Bauteils im Stromkreislauf Störungen an anderen Teilen nach sich ziehen. Ein Extremfall wäre ein defektes Radio, daß die Funktion der Benzinpumpe beeinflußt. Das scheint auf den ersten Blick sehr unwahrscheinlich, aber gerade solche Zusammenhänge sind sehr interessant, um Defekte überhaupt diagnostizieren zu können. Dazu ein Beispiel aus meiner eigenen Erfahrung:
Ein VW Golf funktionierte, nachdem er vollgetankt wurde, einwandfrei, bis der Tank bis auf das letzte Viertel leer gefahren wurde. Dann hatte der Motor immer häufiger Zündaussetzer und das Fahrzeug begann zu rucken, bis schließlich ein Vorwärtskommen kaum mehr möglich war. Nachdem der Wagen wieder vollgetankt wurde, war kein Fehler mehr festzustellen. Verschiedene Werkstätten hatten unterschiedliche Erklärungen für dieses Phänomen. Die Benzinpumpe wurde ausgetauscht, ebenso der Vergaser, aber der Fehler wurde dadurch nicht beseitigt. In einer Werkstatt hatte ein Mechaniker langjährige Erfahrungen mit diesem Fahrzeugtyp und wechselte den Benzineinfüllstutzen aus, und beseitigte dadurch den Fehler. Der Einfüllstutzen war durchgerostet, und so konnte während der Fahrt Schmutz und Wasser in den Tank gelangen. Solange der Tank voll war, war dieser Schmutz gut im Tank verteilt und das Fahrzeug funktionierte fehlerfrei. Je leerer aber der Tank war, desto mehr Schmutz gelangte in den Motor und desto weniger Leistung brachte der Motor hervor.

An diesem Beispiel kann man sehen, wo Zusammenhänge zwischen Fahrzeugteilen bestehen können, die man nicht vermutet. Mit Hilfe der Modellierung von Flüssen sollen diese Beziehungen aufgedeckt werden. Man kann nicht nur Materialflüsse betrachten, sondern auch Energieflüsse, wie zum Beispiel den Fluß von elektrischem Strom. Zu jedem dieser Flüsse kann man aus Konstruktionsplänen die Bauteile zusammenfassen, die von diesem Fluß beeinflußt werden. Jeder Fluß besteht dann aus einem Flußmaterial und einer sortierten Liste von Bauteilen entsprechend der Flußrichtung :

Fluß A
Flußmaterial
[Teil_1,...,Teil_n)

In einem so dargestellten Fluß kann man davon ausgehen, daß alle Teile eines Flusses auf das Flußmaterial Einfluß nehmen können, so daß sie alle die eigentliche Aufgabe dieses Flußmaterials stören können. Greift man das obige Beispiel des defekten Einfüllstutzens auf, so besteht ein Fluß vom Einfüllstutzen zum Tank und von dort weiter zu allen Teilen der Kraftstoffverbrennung. Obwohl der Einfüllstutzen am Anfang dieses Flusses steht, kann er alle nachgelagerten Teile beeinflussen und somit die Verbrennung stören.

2.2.2 Verkettung von Flüssen

Nun kann man den berechtigten Einwand vorbringen, daß das Einfüllen von Benzin nicht nur zeitlich weit von der Kraftstoffverbrennung entfernt ist. Man kann also nicht von einem durchgängigen Fluß sprechen, sondern von einer Verkettung verschiedener Flüsse, bei denen bestimmte Teile, in diesem Fall der Tank, die Aufgabe der Verkettung übernehmen. Diese Teile, die eine Verkettung erzeugen, sind also in unterschiedlichen Flüssen enthalten, so daß sich ein Defekt hier in verschiedene Richtungen ausbreiten kann. Man hat also folgende Struktur :

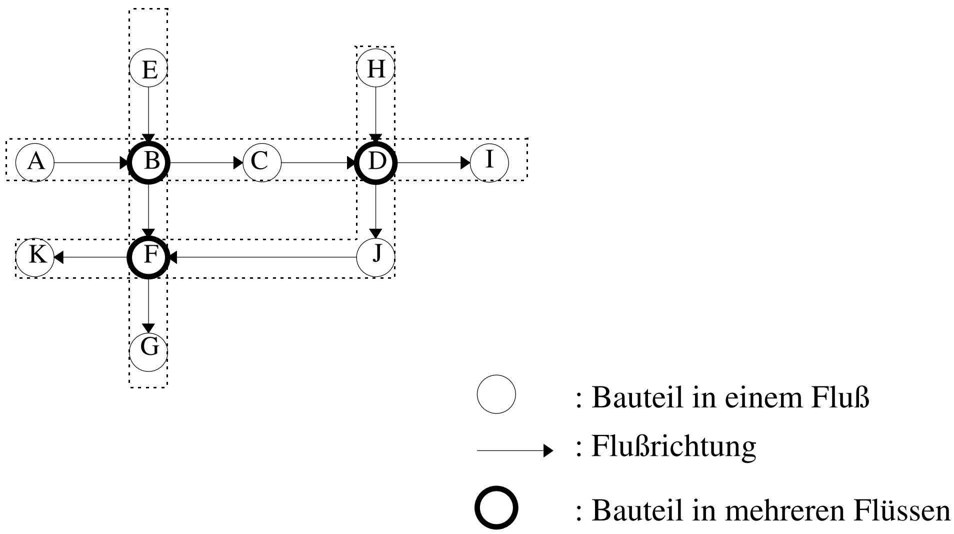


Abbildung 3 : Verkettung von Flüssen

In Abbildung 3 sind 3 Flüsse dargestellt. Diese 3 Flüsse sind durch 3 Bauteile miteinander verkettet, die in der Abbildung fett gekennzeichnet sind. Eine nicht korrekte Arbeitsweise dieser Teile kann nicht nur die Teile des Flusses beeinflussen, in dem ein Fehler aufgetreten ist, sondern kann auch Teile aus anderen Flüssen beeinträchtigen.

Die Flüsse werden aus den Konstruktionsdaten von Fahrzeugen gebildet. Dabei könnte man auch Flüsse bilden, die in anderen Flüssen vollständig enthalten sind. Dies ist aber nicht sinnvoll, da der enthaltene Fluß nur aus Teilen besteht, die sich aufgrund der Definition des übergeordneten Flusses sowieso gegenseitig beeinflussen können. Man erhält also auch durch die Verkettung dieser beiden Flüsse keinen Fortschritt. Bei der Bildung von Flüssen ist also darauf zu achten, das kein Fluß in einem anderen Fluß vollständig enthalten ist.

2.2.3 Flußbildung durch Verkettung

Läßt man die Reihenfolge der Bauteile in Flußrichtung außer acht, kann man einen Fluß als Menge von Bauteilen darstellen. Je feiner und genauer ein Fluß definiert wird, desto mehr Mengen von Bauteilen werden erzeugt und die Anzahl der Elemente einer Menge wird immer geringer. Man kann nun aus den so erzeugten Mengen von Bauteilen neue Mengen bilden, indem man zwei Mengen miteinander vereinigt, die ein gemeinsames Bauteil besitzen. Dieses gemeinsame Bauteil entspricht dann genau einem Teil, das sich in mehreren Flüssen befindet und einen Fehler auf einen anderen Fluß übertragen kann. Folgendes Beispiel soll die Mengenbildung verdeutlichen:

Gegeben seien 3 Flüsse :

Fluß_1 : $A \rightarrow B \rightarrow C \rightarrow D \rightarrow I$

Fluß_2 : $E \rightarrow B \rightarrow F \rightarrow G$
Fluß_3 : $H \rightarrow D \rightarrow J \rightarrow F \rightarrow K$

Aus diesen Flüssen kann man 3 Mengen ableiten :

$M_1 = \{A, B, C, D, I\}$
$M_2 = \{B, E, F, G\}$
$M_3 = \{D, F, H, J, K\}$

Im nächsten Schritt sollen Mengen, die ein gemeinsames Element besitzen, vereinigt werden :

$M_4 = M_1 \cup M_2 = \{A, B, C, D, E, F, G, I\}$
$M_5 = M_2 \cup M_3 = \{B, D, E, F, G, H, J, K\}$

An dieser Stelle kann man mit Hilfe der neu erzeugten Mengen M_4 und M_5 wiederum neue Mengen bilden. Eine weitere Möglichkeit wäre die Vereinigung von M_3 und M_4. Die dabei entstehenden Mengen sind allerdings identisch :

$M_6 = M_4 \cup M_5 = M_3 \cup M_4$

Für eine Fehlerdiagnose kann es sinnvoll sein, einen Fluß zu untersuchen, der in enger Beziehung zum fehlerhaften Teil steht. Aus diesem Grund sollte man sich zuerst auf eine Menge konzentrieren, die aus einem einzigen Fluß entstanden ist und somit weniger Teile enthält. Sollte dieser Weg nicht zum Ziel führen, sind als nächstes die Mengen zu betrachten, die aus zwei einzelnen Flüssen entstanden sind. Man kann also eine Funktion k definieren, die jeder Menge, die durch eine Verkettung von Flüssen entstanden ist, eine

Zahl zuordnet, die die Anzahl der enthaltenden unterschiedlichen grundlegenden Mengen widerspiegelt. Dieses k kann als Maß dafür angesehen werden, wie weit die funktionale Beziehung zwischen den Teilen einer Menge definiert ist. So erfüllen die Teile einer Menge, die aus einem einzelnen Fluß entstanden ist, eine spezielle Aufgabe, während die Teile einer Menge, die aus einer Verkettung von Flüssen entstanden ist, eine übergeordnete Aufgabe erfüllen.
Wie das Beispiel M_6 zeigt, ist die Entstehung einer Menge M durch Vereinigung von anderen Mengen, nicht immer eindeutig. Wurde beispielsweise M_6 durch Vereinigung von M_4 und M_5 gebildet, so werden die Teile von M_2 sowohl durch M_4, als auch durch M_5 hinzugefügt. Das Problem bei der Bestimmung von k sind also mehrfach auftretende grundlegende Mengen. Verwendet man aber bei der Vereinigung von zwei Mengen immer eine grundlegende Menge, so kann man zum einen alle möglichen Mengen bilden, zum anderen kann man k bei jeder Vereinigung direkt ablesen.

Bei diesem Ansatz handelt es sich im eine Idee zur Modellierung von funktionalen Abhängigkeiten zwischen Fahrzeugteilen. Um diese Modellierung durchzuführen, ist zum einen die bauliche Anordnung der Teile notwendig, zum anderen muß das Flußmaterial definiert werden können. Das Ergebnis ist eine Menge von Teilen, zwischen denen eine funktionale Abhängigkeit bestehen muß. Aus diesen Mengen lassen weitere Mengen bilden, deren Teile sich gegenseitig beeinflussen können und die an der Erfüllung einer übergeordneten Funktion beteiligt sind.

2.3 Analyse von Wissensquellen

Durch den Wissenserwerb sollen die funktionalen, räumlichen und kausalen Beziehungen zwischen Bauteilen aufgedeckt werden. Aber welche Wissensquellen sind dafür geeignet und auch gut zugänglich? Ideal wäre ein Experte von Daimler Benz, der ständig zur Verfügung stehen würde. Wirklich vorhanden waren folgende Wissensquellen.

- Durch einen Mitarbeiter der Universität Karlsruhe bestand ein indirekter Kontakt zu einem Experten bei Daimler Benz. Ich hatte aber eigentlich kaum eine Möglichkeit, spezielles Wissen über den Fahrzeugaufbau zu erhalten. Ein Problem war dabei die Entfernung von Karlsruhe bis Dortmund und die Tatsache, daß ich nicht direkt mit einem Experten sprechen konnte.

- Bei Daimler Benz gibt es ein Schadensbuch für PKW und LKW. Diese beiden Bücher lagen in gedruckter Form vor und umfassen mehrere hundert Seiten. In Kapitel 4 werden diese Bücher genauer erläutert.

- Die Schadensbücher lagen, wenn auch in anderer Form, als Dateien vor. Dabei handelte es sich um die Textdateien SSLBEZUG, SSLBILD, SSLTEIL, SSLLANG, SSLFGR, die ebenfalls im nächsten Kapitel genauer erläutert werden.

- In Bibliotheken konnte man einige Bücher über den Aufbau von Fahrzeugen und Motoren erhalten. Dabei handelte es sich um technische Literatur, die zwar den Aufbau einiger Fahrzeugkomponenten behandelte, aber darauf abzielte, allgemeine Kenntnisse der Konstruktion von Fahrzeugen zu vermitteln. Spezielle Informationen über Fahrzeuge der Daimler Benz AG waren nicht enthalten, so daß diese Literatur nicht zum Erwerb von Wissen über Beziehungen zwischen speziellen Fahrzeugkomponenten geeignet war.

- Ich hatte direkten Kontakt zu einem Mitarbeiter einer Kfz-Werkstatt, in der unter anderem auch Daimler Benz Fahrzeuge repariert werden. Allerdings stand mir dieser Mitarbeiter nur für wenige Fragen zur Verfügung, obwohl er anfangs versicherte, mich beim Wissenserwerb zu unterstützen. Außerdem wollte er mich mit speziellen Büchern versorgen, die Reparaturwerkstätten von Daimler Benz normalerweise zur Verfügung stehen. Wie sich aber nach kurzer Zeit herausstellte, hatte er kein spezielles Wissen über Fahrzeuge von Daimler Benz, so daß er mir auch als Experte nicht zur Verfügung stand. Zum anderen kümmerte er sich nicht weiter um die zugesicherten Bücher, so daß ich keinen Nutzen aus dieser Wissensquelle ziehen konnte.

- Mir erschien eine Übersicht über die räumliche Anordnung von Fahrzeugteilen als wichtig, und ich hatte die Vorstellung, daß einer Vertragswerkstatt oder einem Fachhändler von Daimler Benz eine solche Übersicht zur Verfügung stehen müßte. Also versuchte ich bei verschiedenen Werkstätten und Händler so etwas zu bekommen, allerdings immer mit dem selben negativen Ergebnis.

2.4 Praxiserfahrungen beim Wissenserwerb

Die Beziehungen, um die es bei diesem Wissenserwerb geht, sind recht nah an die Konstruktion von bestimmten Fahrzeugen angelehnt. Dabei geht es nicht um die Fahrzeugbauweise, die bei vielen Fahrzeugen übereinstimmt, sondern um ganz spezielle Fahrzeuge mit ganz speziellen Bauteilen. Da es sich dabei um Konstruktionsdaten handelt, ist die Literatur über die Fahrzeugkonstruktion sehr oberflächlich oder nicht allgemein zugänglich. Das erschwert den Wissenserwerbsprozeß natürlich sehr. Aus diesem Grund erhoffte ich mir ein Gespräch mit einem Experten, um von ihm Wissen über die bauliche Anordnung von Fahrzeugteilen zu erhalten, da die bauliche Anordnung die räumlichen Beziehungen widerspiegelt. Auch im Hinblick auf den Ansatz zur Modellierung von funktionalen Abhängigkeiten mit Flüssen ist bauliche Anordnung von Bedeutung, da es ansonsten nicht möglich, ist die grundlegenden Flüsse zu beschreiben. Anfangs bestand zwar ein Kontakt zu einem Experten, auch wenn ich nie direkt mit ihm sprechen konnte, aber bereits nach kurzer Zeit wurde auch dieser indirekte Kontakt von dem Experten abgebrochen. Nachdem sich der Mitarbeiter der Kfz-Werkstatt nicht als Experte herausstellte, blieb mir keine andere Möglichkeit, als mich auf die verfügbare Literatur zu beschränken. Das Ziel bestand weiterhin darin, Informationen über die räumliche Anordnung von Bauteilen zu erhalten, damit ich den Ansatz zur Modellierung mit Hilfe von Flüssen durchführen konnte. Aber sowohl Anfragen an Daimler Benz selbst als auch bei verschieden Vertragswerkstätten und Vertragshändlern blieben erfolglos. Die Folge war, daß mir das Schadensbuch als einzige Wissensquelle zur Verfügung stand. Da das Schadensbuch aber nur Teile nach unterschiedlichen Aspekten gruppiert, und keine bauliche Anordnung der Teile enthält, war die Bildung von Flüssen nicht durchführbar. Mir blieb also nur die Möglichkeit mich auf das vorhandene Schadensbuch zu konzentrieren.

3 Darstellung von MOBAL und STT

MOBAL ist ein System zur Entwicklung, Kontrolle und Wartung operationaler Modelle eines Anwendungsbereichs. Es enthält eine Wissensbasis, eine Wissenserwerbskomponente, eine Problemlösungskomponente, verschiedene Methoden des maschinellen Lernens sowie ein Wissensrevisionstool. Mit Hilfe der Wissenserwerbskomponente kann ein Benutzer ein Modell des Anwendungsbereichs entwickeln. Dabei bestimmt er die Reihenfolge seiner Schritte selbst und ist nicht an Vorgaben des Systems gebunden. Die Entwicklung des Modells wird dadurch zu einem kreativen Prozeß. Dargestellt wird das Modell in Form von logischen Fakten und Regeln. Das eingegebene Wissen kann dann in textueller oder graphischer Form betrachtet werden. Die Inferenzmaschine kann die eingegebenen Regeln anwenden und so resultierende Fakten erzeugen oder je nach vorhandenem Wissen auf Fragen antworten. Außerdem bildet MOBAL eine dynamische Sortentaxonomie, die Informationen über die Sorten der verwendeten Objekte enthält. Die Methoden des maschinellen Lernens können automatisch Regeln erkennen, die auf den eingegebenen Fakten basieren. Schließlich besteht noch die Möglichkeit, Widersprüche in der Wissensbasis mit Hilfe des Wissensrevisionstools zu erkennen und zu beseitigen.
Im folgenden wird eine Einführung der in dieser Arbeit benötigten Komponenten von MOBAL gegeben. Im wesentlichen handelt es sich dabei um die Definition von Fakten und um die Anwendung von STT auf automatisch generierte, einstellige Fakten. Eine ausführliche Beschreibung von MOBAL mit allen seinen Möglichkeiten gibt [2],[11].

- **Fakten**

Fakten drücken Relationen oder Eigenschaften von Objekten oder Konzepten aus. Ein Faktum besteht aus einem n-stelligen Prädikat p und aus n Termen $t_1,...,t_n$, in der Form $p(t_1,...,t_n)$. Wichtig ist, daß für die t_i nur Atome zugelassen sind, Variablen sind nicht erlaubt. Variablen werden in MOBAL mit einem großen Anfangsbuchstaben bezeichnet, während Konstanten[1] mit einem kleinen Buchstaben beginnen. Man kann positive und negierte Fakten unterscheiden. Während es sich bei der oben dargestellten Form $p(t_1,...,t_n)$ um ein positives Faktum handelt, wird die negative Form mit $not(p(t_1,...,t_n))$ bezeichnet.

- **Deklaration von Prädikaten**

Jede Argumentstelle eines Prädikats gehört einer bestimmten Sorte an, d.h. für die Argumente $t_1,...,t_n$ eines Prädikats der Form $p(t_1,...,t_n)$ gilt $t_1 \in s_1$,..., $t_n \in s_n$. Obwohl MOBAL’s Sort Taxonomy Tool (STT) die Sortenstruktur automatisch und dynamisch selbst berechnet, hat der Benutzer die Möglichkeit, symbolische Sortennamen für jede Argumentposition eines Prädikats anzugeben. Hat man beispielsweise folgende Deklaration :

p/2: $<sort_1>$, $<sort_2>$

Dann wird STT alle Objekte, die an der ersten Argumentstelle des Prädikats p stehen, der Sorte $sort_1$ zuordnen.

[1] Atome sind entweder konstante Zeichenkette oder Zahlen. Eine Konstante bezeichnet hier eine Zeichenkette, bei der es sich aber nicht um eine Variable handelt.

3.1 STT [2]

Legt man eine mehrsortige Logik zu Grunde, müssen alle Prädikate und ihre Argumentstellen definiert werden, bevor man sie verwenden kann. In der Deklaration eines Prädikats wird für jede Argumentstelle definiert, zu welcher Sorte sie gehört. In MOBAL kann man das System die Aufgabe der Deklaration übernehmen. Wenn ein Prädikat in MOBAL zum ersten Mal verwendet wird, generiert das System automatisch die Deklaration des Prädikats. Dabei wird für jede Argumentstelle jedes Prädikats eine neue Sorte eingeführt. Sammelt man dann alle Terme, die in den eingegebenen Fakten für eine bestimmte Argumentstelle benutzt werden, kann man daraus die Menge der Terme, die zur Sorte dieser Argumentstelle gehören, ableiten. Man bezeichnet diese Menge von Termen auch als Extension einer Argumentsorte. Durch diese Sammlung von Termen kann für jede Argumentsorte die zugehörige Extension berechnet werden. Diese Extensionen bilden die Grundlage zur Berechnung der Gleichheit der Argumentsorten. Danach sind zwei Argumentsorten gleich, wenn sie auf die gleiche Extension abgebildet werden. Man erhält so die feinste Sortentaxonomie, die auf der aktuellen Menge von Fakten basiert.

3.1.1 Extensionen von Sorten

Da die Prädikatsnamen eindeutig sind, kann jeder Argumentposition ein eindeutiger Sortenname zugeordnet werden, der auf dem Prädikatennamen und der Argumentposition basiert. Diese Zuordnung wird durch die Funktion arg_sort vorgenommen :

Definition : (Argumentsorte)

$arg\_sort: I, PN \rightarrow SN$

PN : die Menge der Prädikatennamen
I : der Index einer legalen Argumentposition in einem Prädikat
SN : eine Menge eindeutiger Sortennamen

Nun kann man eine Funktion definieren, die jedem Sortennamen die Menge von Termen zuordnet, die zu dieser Sorte gehören. Dabei handelt es sich um die oben erwähnte Extensionsfunktion ext :

Definition : (Extensionsfunktion ext für Argumentsorten)

$$ext: \quad SN \rightarrow pset(TERM), \text{ mit } t_i \in ext(arg\_sort(i, p)) \Leftrightarrow$$
$$p(t_1,...,t_i,...,t_n) \in FB \vee \neg\, p(t_1,...,t_i,...,t_n) \in FB$$

SN : eine Menge eindeutiger Sortennamen
TERM : Menge aller Terme
FB : Menge aller Fakten, die dem System bekannt sind (Faktenbasis)

Obwohl die Argumentsorten vom System automatisch gebildet werden können, gibt es in MOBAL außerdem die Möglichkeit, benutzerdefinierte Sorten zu spezifizieren. Dies kann sinnvoll sein ,um bedeutungsvolle Namen in die Sortentaxonomie einzuführen und so die

[2] Dieses Kapitel basiert im wesentlichen auf [2] : K. Morik, S. Wrobel, J-U. Kietz und W. Emde, *Knowledge Acquisition and Machine Learning : Theory, Methods and Applications*, Kapitel 4

Verständlichkeit zu erhöhen, oder um dem Benutzer die Möglichkeit zu geben, ihm bekannte Sorteninformationen einzugeben. Es gibt 4 Arten von Sortenspezifikationen, die der Benutzer festlegen kann :

- **Sortendeklarationen für Argumentstellen von Prädikaten**
  Die Syntax der Prädikatsdeklaration hat Ähnlichkeit mit der Definition von positiven Fakten, mit dem Unterschied, daß die Argumente keine Terme, sondern Namen für benutzerdefinierte Sorten sind. Die benutzerdefinierten Sorten werden dabei durch ‘<’ und ‘>’ eingeschlossen. Für den Fall, daß der Benutzer an einer Argumentstelle keine Argumentsorte festlegen will, kann er an dieser Stelle ‘< >’ verwenden.

  Ein zweistelliges Prädikat hat dann folgende Form :

  p/2: $<\mathrm{sort}_1>$, $<\mathrm{sort}_2>$

  Dann wird STT alle Objekte, die an der ersten Argumentstelle der Prädikats p stehen, der Extension der Sorte sort_1 zuordnen.

- **Definition von Sorten als Schnittmenge von Sorten**
  Es besteht die Möglichkeit, eine neue Sorte als Schnittmenge von zwei vorhandenen Sorten zu definieren. Dabei steht die Schnittsorte auf der linken Seite von ‘:=’, während Sorten, aus denen die Schnittsorte gebildet wird, auf der rechten Seite stehen. Man hat also folgende Form :

  Schnittsorte : = Sorte_1 , Sorte_2

- **Inklusionen für einfache Sorten und Schnittmengen von Sorten**
  Will man Inklusionen zwischen Sorten ausdrücken, so kann man die Relation :< zwischen Sorten verwenden. Dabei steht auf der linken Seite die Teilmenge, auf der rechten Seite die Obermenge. Sie hat also folgende Form.

  Untersorte <: Obersorte

  Außerdem gibt es eine syntaktische Variante, bei der die Untersorte als Schnittmenge von zwei anderen Sorten definiert wird. Dabei handelt es sich nur um eine syntaktische Kurzform für die Definition einer Schnittsorte und anschließender Definition einer Inklusion dieser Schnittsorte. Man hat also folgende Form :

  Untersorte <: $\mathrm{Obersorte}_1$, $\mathrm{Obersorte}_2$

- **Überschneidungsfreie, unabhängige Sorten**
  Schließlich besteht die Möglichkeit, die Überschneidungsfreiheit zwischen zwei Sorten zu definieren. Dabei handelt es sich um eine Variante der Schnittmengenbildung, bei der zwei Sorten als überschneidungsfrei definiert werden, wenn ihre Schnittsorte die spezielle, leere Sorte NIL ergibt.

  NIL : < Sort_1 , Sort_2

Man kann eine Funktion definieren, die einen Prädikatsnamen und eine Nummer einer legalen Argumentposition benutzt, um auf den benutzerdefinierten Sortennamen zu zugreifen

Definition : (Benutzerdefinierte Sorte)

$user\_sort : I, PN \rightarrow SN, \text{ mit } user\_sort(i,p) = s_i \Leftrightarrow p(<s_1>,...,<s_i>,...,<s_n>) \in PB$

PN : die Menge der Prädikatsnamen
PB : die Menge der benutzerdefinierten Prädikatsdeklarationen
I : eine Nummer, die eine legale Argumentposition in einem Prädikat bezeichnet
SN : eine Menge eindeutiger Sortennamen

Nachdem nun beschrieben wurde, welche Möglichkeiten MOBAL zur Definition neuer Sorten bietet, stellt sich die Frage, wie die Extensionen der neuen Sorten aussehen. Aus diesem Grund wird die Extensionsfunktion ext um benutzerdefinierte Sorten erweitert.

Definition : (Extensionsfunktion ext, erweitert für benutzerdefinierte Sorten)

Die Funktion $ext : SN \rightarrow pset(TERM)$ *mit* $t_i \in ext(s) \Leftrightarrow$

$$(s = ALL \wedge t_i \in TERM) \vee$$
$$(s = user\_sort(i,p) \wedge (p(t_1,...,t_i,...,t_n) \in FB \vee \neg\, p(t_1,...,t_i,...,t_n) \in FB)) \vee$$
$$(s = arg\_sort(i,p) \wedge (p(t_1,...,t_i,...,t_n) \in FB \vee \neg\, p(t_1,...,t_i,...,t_n) \in FB)) \vee$$
$$(s_0 :< s_1,...,s,...,s_n \in SB \wedge t_i \in ext(s_0)) \vee$$
$$(s_0 := s_1,...,s,...,s_n \in SB \wedge t_i \in ext(s_0)) \vee$$
$$(s_0 := s_1,...,s_j,...,s_n \in SB \wedge \forall j \in 1,...,n : t_i \in ext(s_j))$$

weist jedem Sortennamen eine Extension zu, die auf der aktuellen Faktenbasis und der Menge der Spezifikationen der Sorten basiert.

SN : Menge eindeutiger Sortennamen, einschließlich der speziellen Sorten ALL und NIL
TERM : Menge aller Terme
FB : Menge aller Fakten, die dem System bekannt sind (Faktenbasis)
PB : Menge der benutzerdefinierten Prädikatsdeklarationen
SB : Menge der Untersorten und der definierten Sortenspezifikationen

pset(TERM) bezeichnet die Potenzmenge von TERM

3.1.2 Äquivalenzklassen von Sorten

Im vorherigen Abschnitt wurde festgelegt, wie Sorten in MOBAL automatisch aus vorhandenen Prädikaten gebildet werden und wie diese Menge von Sorten durch benutzerspezifizierte Sorten erweitert wird. Wichtig war dabei, welche Objekte welchem Sortennamen zugeordnet wurden. Dies wurde durch die Extensionsfunktion ext realisiert, die für benutzerspezifizierte Sorten erweitert wurde.
Nun stellt sich die Frage, wie die Sorten von verschiedenen Argumentstellen von Prädikaten oder benutzerdefinierten Sorten zusammengeführt werden können, wenn die

Sorten doch eigentlich identisch sind. Das führt zu einer Äquivalenzrelation zwischen Sorten, wonach Sorten der gleichen Äquivalenzklasse angehören, wenn ihre Extensionen identisch sind. Damit kann man einer Äquivalenzklasse eine Extension zuordnen, die der Extension der enthalten Sorten entspricht. Die Äquivalenzklassen können folgendermaßen definiert werden.

Definition : (Äquivalenzrelation über Sorten)

Die Relation $\approx \subseteq SN \times SN$ ist eine Äquivalenzrelation wie '=', die folgendermaßen definiert ist :

$$\forall s_1,s_2 \in SN : s_1 \approx s_2 \Leftrightarrow ext(s_1) = ext(s_2)$$

SN/≈ : Menge der Klassen von Sorten mit der gleichen Extension

Die Äquivalenzklassen enthalten nun das, was eigentlich mit einer Sorte gemeint ist. Sie enthalten alle Sorten, die aufgrund gleicher Objekte der Argumentstellen von Prädikaten gebildet wurden, oder durch den Benutzer definiert wurden. Was noch fehlt ist ein Sortenname für eine Äquivalenzklasse und eine Funktion, die für jede Sorte die Klasse berechnet, zu der sie gehört:

Definition : (Name einer Äquivalenzklasse)

*Die bijektive Funktion cn : **SN/≈** $\rightarrow$ CLASS, ordnet jeder Äquivalenzklasse einen eindeutigen Namen zu.*

CLASS : Menge eindeutiger Klassennamen

Da jedem Klassennamen eine Klasse zugeordnet wird und jede Klasse mindestens eine Sorte enthalten muß, wird jedem Klassennamen mindestens eine Sorte zugeordnet.

Schließlich gibt es noch eine Funktion class : SN $\rightarrow$ CLASS mit

$$\forall s_1,s_2 \in SN : \mathrm{class}(s_1) = \mathrm{class}(s_2) \Leftrightarrow s_1 \approx s_2 \Leftrightarrow \mathrm{ext}(s_1) = \mathrm{ext}(s_2)$$

die für jede Sorte die Klasse berechnet, zu der sie gehört.

Durch die Definition der Äquivalenzrelation $\approx$ wurde die Menge der Sorten in Äquivalenzklassen aufgeteilt. Die Extensionsfunktion ext wurde zur Definition von $\approx$ verwendet, und aufgrund dieser Definition ist es offensichtlich, wie die Extension einer Klasse aussehen wird, da alle Sorten einer Klasse die selbe Extension haben. Dies soll durch die Definition einer Extensionsfunktion cext für Klassen dargestellt werden. Außerdem ist es so einfacher zwischen Klassen und Sorten zu unterscheiden.

Definition : (Extension von Äquivalenzklassen)

cext : CLASS $\rightarrow$ pset(TERM) ist eine Funktion, die den Klassen ihre Extensionen zuordnet:

$\forall s \in SN : cext(class(s)) = ext(s)$

Lemma
cext ist eine injektive Funktion :
$\forall c_1, c_2 \in CLASS : cext(c_1) = cext(c_2) \Rightarrow c_1 = c_2$,

$cext(class(s_1)) = cext(class(s_2))$
$\Leftrightarrow ext(s_1) = ext(s_2)$
$\Leftrightarrow s_1 \approx s_2$
$\Leftrightarrow class(s_1) = class(s_2)$

Nachdem festgelegt wurde, welche Sorten äquivalent sind, muß man festlegen welche Sorten miteinander vereinbar sind. Das wird benötigt, um die Sortenkorrektheit von Variablenbindungen in Formeln zu entscheiden. Die Vereinbarkeit zwischen Klassen führt zur Vereinbarkeit zwischen Sorten. Die Vereinbarkeit zwischen Sorten benötigt als Minimalanforderung eine partielle Ordnung. Man erhält eine partielle Ordnung, wenn man die Teilmengenbeziehungen zwischen Extensionen von Klassen betrachtet. Das bedeutet, daß eine Klasse c_1 Unterklasse einer Klasse c_2 ist, wenn die Extension von c_1 eine Teilmenge der Extension von c_2 ist.

Definition : (Die partielle Ordnung über Äquivalenzklassen)

$\leq \subseteq CLASS \times CLASS$ *mit* $c_1 \leq c_2 \Leftrightarrow cext(c_1) \subseteq cext(c_2)$

(CLASS,≤) ist eine partielle Ordnung, da (pset(TERM), ⊆) eine partielle Ordnung und cext ein Homomorphismus von CLASS nach pset(TERM) ist.

Mit Hilfe von ≤ können Funktionen und Relationen definiert werden :
subclass-relation :
$subclass \subseteq CLASS \times CLASS$
$subclass(c_1,c_2) \Leftrightarrow (c_1 \leq c_2) \wedge \neg (c_2 \leq c_1)$
predecessor :
$subs : CLASS \rightarrow pset(CLASS)$ *mit*
$subs(c_1) := \{c \in CLASS\ c \leq c_1 \wedge c \neq c_1\}$
successor :
$supers : CLASS \rightarrow pset(CLASS)$ *mit*
$supers(c_1) := \{c \in CLASS\ c_1 \leq c \wedge c \neq c_1\}$
d-subclass-Relation
$d\text{-}subclass \subseteq CLASS \times CLASS$
$d\text{-}subclass(c_1,c_2) \Leftrightarrow (c_1 \leq c_2) \wedge (\neg\, c_3 : (c_1 \leq c_3) \wedge (c_3 \leq c_2))$
direct predecessors :
$d\text{-}subs : CLASS \rightarrow pset(CLASS)$ *mit*
$d\text{-}subs(c_1) := \{c \in subs(c_1) \mid \neg\, c_2 \in subs(c_1) : c \leq c_2\}$
direct successors :
$d\text{-}supers : CLASS \rightarrow pset(CLASS)$ *mit*
$d\text{-}supers(c_1) := \{c \in supers(c_1) \mid \neg\, c_2 \in supers(c_1) : c_2 \leq c\}$

Eine partielle Ordnung kann man leicht als Ordnungsdiagramm darstellen. In Abbildung 4 ist ein Beispiel einer partiellen Ordnung dargestellt. Betrachtet man die Klasse C_5 kann man folgende Relationen verwenden.

subclass(C_6,C_5),subclass(C_8,C_5), subclass(C_7,C_5),subclass(C_5,C_2), subclass(C_5,C_1)
subs(C_5) = {C_6, C_7, C_8}
supers(C_5) = {C_1, C_2}
d-subclass(C_6,C_5), d-subclass(C_7,C_5), d-subclass(C_5,C_2)
d-subs(C_5) = {C_6,C_7}
d-supers(C_5) = {C_2}

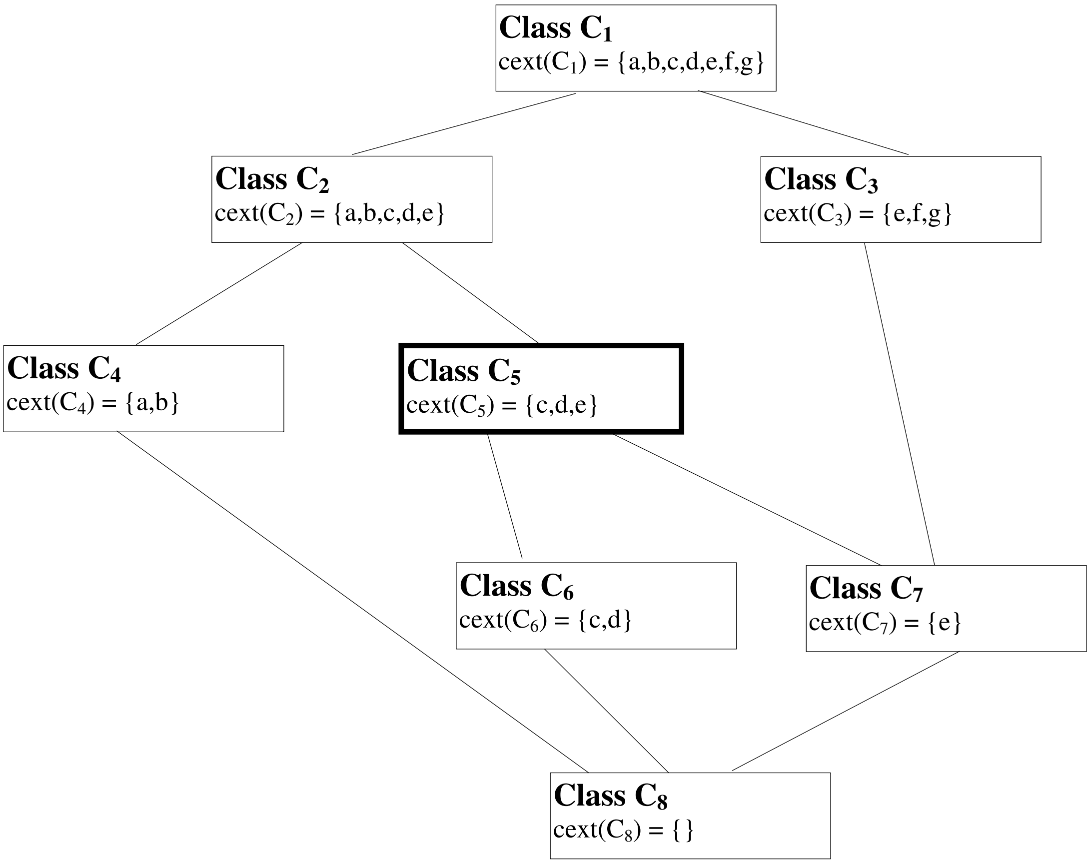


Abbildung 4 : Ordnungsdiagramm einer Klassenhierarchie

3.1.3 Schnittsorten

Oft kommt es vor, daß einige Klassen gemeinsame Elemente haben, ohne daß eine Klasse in der anderen enthalten ist. Um diese Klassen von überschneidungsfreien Klassen, die keine gemeinsamen Elemente haben, unterscheiden zu können, kann man zusätzliche Klassen erzeugen. Die Extensionen dieser neuen Klassen enthalten dann gerade die Elemente, die die beiden Ausgangsklassen gemeinsam haben, also in deren Schnittmenge liegen.

Definition : (Schnittsorten IS)

Schnittsorten IS sind alle zweielementigen Mengen von Klassen, die sich überschneiden und die keine Untersorten voneinander sind.

$IS = \{\{c_1,c_2\} \subseteq CLASS \mid \neg(c_1 \leq c_2) \wedge \neg (c_2 \leq c_1) \wedge \neg (cext(c_1) \cap cext(c_2) = \{\})\}$

Die Extensionsfunktion iext : IS $\rightarrow$ pset(TERM) ist folgendermaßen definiert :

$\forall \{c_1,c_2\} \in IS : iext(\{c_1,c_2\}) = cext(c_1) \cap cext(c_2)$

Wenn man ≈, class und cext auf ISN = SN ∪ IS erweitert, dann erzeugt diese Definition rekursiv alle Schnittsorten von sich überschneidenden Klassen. Beachtet man die Tatsache, daß man im worst case 2^n Schnitte für n Klassen zu berechnen hat, so wird klar, daß die Berechnung der Schnittsorten schon für ein relativ kleines n sehr viel Zeit in Anspruch nimmt. Aus diesem Grund kann der Benutzer von MOBAL selbst entscheiden, ob er die Schnittsorten berechnen möchte.
Man kann nun eine Funktion definieren, die für eine Klasse c alle Klassen c_i berechnet, deren Extensionen eine nicht leere Schnittmenge mit cext(c) bilden.

Definition : (Schnittklassen einer Klasse)

inters : CLASS $\rightarrow$ pset(CLASS) mit
$inters(c) = \{c_1 \in CLASS \mid \{c,c_1\} \in IS\}$

Betrachtet man wieder die Klasse C_5 in Abbildung 4, so enthält inters(C_5) alle Klassen C_k, für die weder subclass(C_5, C_k) noch subclass(C_k, C_5) gilt. Außerdem müssen beide Klassen mindestens ein gemeinsames Element haben. Die Klassen C_1, C_2, C_6, C_7, und C_8 stehen mit C_5 in einer subclass-Relation. So bleiben nur noch C_3, C_4, wobei cext(C_4)und cext(C_5) kein gemeinsames Element haben. Es bleibt also nur noch C_3, da cext(C_3) ∩ cext(C_5) = {e}. Also gilt inters(C_5) = {C_3}

4 Das Schadensbuch der Daimler Benz AG

Das Schadensbuch bildet in dieser Arbeit die Grundlage für den Erwerb von Hintergrundwissen. Es liegt zum einen in gedruckter Form als Buch vor, zum anderen in maschinenlesbarer Form als Textdateien. In diesem Kapitel wird zu Beginn die allgemeine Struktur des Schadensbuchs beschrieben, die im wesentlichen der gedruckten Form entspricht. Danach werden die einzelnen Dateien der maschinenlesbarer Form dargestellt. Abschließend wird erläutert, wie das Schadensbuch in der Praxis eingesetzt wird, um Fahrzeugschäden zu verschlüsseln.

4.1 Gründe für den Einsatz des Schadensbuchs

Die Daimler Benz AG ist einer der führenden Kfz-Hersteller der Welt, der sich vor allem durch qualitative Merkmale von anderen Herstellern unterscheidet. Ihr Ziel ist es, diesen Wettbewerbsvorteil gegenüber anderen Unternehmen, die auf dem selben Markt tätig sind, zu verteidigen. Dieser Markt und seine Unternehmen sind durch folgende Merkmale gekennzeichnet : [10] , [7]

Ziele des Unternehmens :
- Markterfolg durch Qualität
- Einzigartigkeit der Produkte und Leistungen

Merkmale des Unternehmens :
- Einsatz moderner Technologien
- überdurchschnittliche Qualität
- überdurchschnittlicher Kundendienst

Merkmale der Produkte :
- hohe Produktqualität
- hoher Preis

Porter [10] unterscheidet zwei Möglichkeiten für Unternehmen, auf einem Markt Vorteile zu erlangen. Das ist die Differenzierung durch Kostenführerschaft und durch Qualitätsführerschaft. Aufgrund der Unternehmens- und Produktmerkmale ist die Daimler Benz AG ein Unternehmen, das sich einer Differenzierung durch Qualitätsmerkmale zuordnen läßt.
Das hier verwendete Schadensbuch ist ein Teil des Qualitätssicherungssystems, das hauptsächlich auf eine Verbesserung der Produktqualität ausgerichtet ist. Das wird durch folgenden Satz aus dem Vorwort deutlich [9]:

„Um unsere Erzeugnisse ständig weiterentwickeln zu können, müssen wir wissen, wie sie sich im täglichen Einsatz bewähren, ob und welche Schäden auftreten, und was deren Ursachen sind.“

4.2 Aufbau der Schadensbuchs

Ziel des Schadensbuchs ist eine einheitliche Erfassung aller Schäden an Fahrzeugteilen. Schäden zu erfassen ist dabei nicht so sehr problematisch, da alle Fachwerkstätten die anfallenden Schäden mehr oder weniger gut beschreiben können. Problematisch aber ist es, einen identischen Schaden an zwei Fahrzeugen in verschiedenen Werkstätten gleich zu beschreiben, da er beispielsweise durch unterschiedliche Symptome bemerkt worden sein kann. Das Schadensbuch ist dabei ein Ansatz, der die Fachwerkstätten bei der

Fehlerbeschreibung unterstützen und der Daimler Benz AG die Möglichkeit geben soll, gleiche Schäden auch als gleich zu erkennen. Sollten bestimmte Schäden nun extrem häufig auftreten, kann der Fehler von den Konstrukteuren erkannt und in zukünftigen Fahrzeuggenerationen vermieden werden. Allerdings läßt das Schadensbuch den Werkstätten noch viele Freiheiten bei der Beschreibung eines Schadens, so daß eine einheitliche Erfassung und Auswertung nur in einem eingeschränkten Rahmen möglich ist. Diese Tatsache erschwert auch das Lernen aus Datenbanken, so daß der Einsatz von Hintergrundwissen notwendig wird (vgl. Abschnitt 1.2.)

Die Daimler Benz AG baut verschiedene Fahrzeugarten, wie Personenkraftwagen, Nutzfahrzeuge und Unimog, für die jeweils ein eigenes Schadensbuch existiert. Von diesen insgesamt 3 Schadensbücher, sind aber nur die ersten beiden in gedruckter Form verfügbar. Zu beachten ist, das ein Schadensbuch nicht nur ein bestimmtes Fahrzeug beschreibt, sondern alle Ausführungen, die von Daimler Benz hergestellt werden. Beispielsweise können sich zwei Fahrzeuge durch ihre Motorausführung unterscheiden, aber ansonsten identisch sein. Ebenso unterscheiden sich ein Cabriolet und ein herkömmlicher PKW durch ihre Karosserie, sind aber ansonsten weitgehend baugleich. Wesentliche Unterschiede bestehen allerdings zwischen einem PKW und einem LKW. Sowohl die Motorbauweise, als auch die gesamten Karosserie ist unterschiedlich. Aus diesem Grund existieren zwei unterschiedliche Schadensbücher.
Wie in den folgenden Unterkapiteln dargestellt, werden Schadensteile eines Schadensbuches durch Schadensteilschlüssel beschrieben. Die Schadensteilschlüssel sind für jedes Schadensteil eindeutig, und zwar nicht nur im Bezug auf ein einzelnes Schadensbuch, sondern auch im Bezug auf alle Schadensbücher.

4.2.1 Struktur des Schadensbuchs

Das Schadensbuch läßt sich in 3 hierarchische Ebenen gliedern. Diese Ebenen sind sowohl in der gedruckten, als auch in der maschinenlesbaren Form vorhanden.
Die oberste Ebene bilden die Konstruktionsgruppen, die die Fahrzeuge nach räumlichen und funktionalen Gesichtspunkten gliedern. Es gibt Konstruktionsgruppen für alle möglichen Fahrzeugkomponenten von Daimler Benz, die über die ausschließliche Verwendung einer einzelnen Art von Fahrzeugen hinausgehen. So existieren Konstruktionsgruppen, die nur in Nutzfahrzeugen verwendet werden, wie zum Beispiel das Führerhaus eines LKW. Auf der anderen Seite gibt es auch Konstruktionsgruppen, die in beiden Fahrzeugklassen eingesetzt werden, wie beispielsweise die Motorkühlung, was allerdings nicht bedeutet, daß es sich dabei um die gleiche Konstruktion handelt. Nur die Funktion, die von der Konstruktionsgruppe erfüllt werden soll, ist gleich. Für Konstruktionsgruppen existieren zweistellige Schlüssel. Abbildung 5 enthält eine Übersicht über die Konstruktionsgruppen des Motors. Einigen Konstruktionsgruppen, z.B.: Schlüssel 04, 10, 12..., ist keine Beschreibung zugeordnet, da sie in keinem Schadensbuch vorhanden sind. Zum einen sind sie für zukünftige Entwicklungen reserviert, zum anderen existierten sie in vorherigen Ausgaben der Schadensbücher, werden aber heute nicht mehr benötigt.

Konstruktionsgruppenschlüssel	Beschreibung der Konstruktionsgruppe
01	Motorgehäuse
02	Verdichter
03	Triebwerkteile
05	Steuerung, Kettentrieb
06	Drehzahlregler u. -messer
07	Vergaser, Einspritzpumpe
08	Turbine
09	Kraftstofförder - und Ladepumpe
11	Saug- und Treibgasanlage
13	Luftpresser
14	Saug- und Auspuffkrümmer
15	Elektr. Ausrüstung am Motor
17	Luftanlaßeinrichtung
18	Ölpumpe, Schmierleitung
20	Motorkühlung
22	Motorzubehör, Aufhängung
23	Sondereinbauten

Abbildung 5 Konstruktionsgruppen des Motors

Auf der zweiten Ebene befinden sich die Funktionsgruppen. Sie sind Untergruppen der Konstruktionsgruppen und beschreiben spezielle Bauvarianten einer Konstruktionsgruppe. Betrachtet man beispielsweise die Konstruktionsgruppe 07 (Vergaser, Einspritzpumpe), so werden auf Funktionsgruppenebene die einzelnen Vergaser und Einspritzsysteme für verschiedene Motortypen beschrieben. Auch den Funktionsgruppen werden Schlüssel zugeordnet. Sie bestehen aus einer 4-stelligen Nummer, wobei die ersten beiden Ziffern der Konstruktionsgruppe entsprechen, der die jeweilige Funktionsgruppe zugeordnet ist. Die Ziffern 3 und 4 bezeichnen dann die Funktionsgruppe innerhalb ihrer Konstruktionsgruppe. Abbildung 6 zeigt eine Übersicht der Funktionsgruppen, die der Konstruktionsgruppe 07 im PKW-Schadensbuch zugeordnet sind. Auch hier sind nicht alle möglichen Funktionsgruppen benutzt worden.

Funktionsgruppenschlüssel	Beschreibung der Funktionsgruppe
0710	Diesel-Einspritzanlage allgemein
0711	Diesel-Einspritzanlage Reihenpumpe
0713	Diesel-Einspritzanlage Verteilerpumpe
0722	Vergaseranlage 2 E-E
0732	Benzin-Einspritzanlage KA/KE
0741	Benzin-Einspritzanlage LH
0750	Benzin-Einspritz- und Zündsystem HFM/PMS allgemein

Abbildung 6 : Funktionsgruppenschlüssel der Konstruktionsgruppe 07 im PKW-Schadensbuch

Auf der dritten Ebene befinden sich nun die Funktionsuntergruppen. Wie der Name schon sagt, handelt es sich dabei um Untergruppen der Funktionsgruppen. Sie beziehen sich auf

eine bestimmte Komponenten und beschreiben Details der Konstruktion. So werden die unterschiedlichen Karosserieformen eines Cabriolets und einer Limousine durch unterschiedliche Funktionsuntergruppen dargestellt. Die Funktionsuntergruppen unterscheiden sich aber von Konstruktions- und Funktionsgruppen, da ihnen Bauteile direkt zugeordnet sind, während die Konstruktionsgruppen und Funktionsgruppen die Teile untergeordneter Gruppen zusammenfassen. Diese Struktur ist einer Objekthierarchie mit Vererbung sehr ähnlich. Die unterste Ebene bilden die Funktionsuntergruppen. Sie vererben alle Teile an die über ihnen liegende Funktionsgruppe. Die Funktionsgruppen vererben ihre Teile wiederum an die über ihnen liegende Konstruktionsgruppe. Der Unterschied zu Objekthierarchien besteht aber darin, daß Funktionsgruppen und Konstruktionsgruppen nur aus den Teilen bestehen, die sie geerbt haben. Sie besitzen also keine Teile, die nicht in einer unter ihnen liegenden Gruppe bereits vorkommen.
Funktionsuntergruppen beziehen sich auf eine bestimmte Fahrzeugart, wie PKW oder LKW. Das bedeutet, sie können dem Schadensbuch einer bestimmten Fahrzeugart zugeordnet werden. Diese Zuordnung wird auch im Aufbau des Funktionsuntergruppenschlüssels deutlich, der in
Abbildung **7** dargestellt ist.

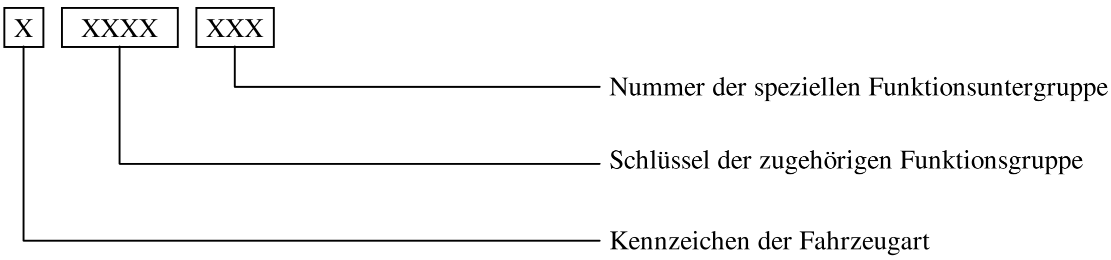


Abbildung 7 : Aufbau des Schlüssels für Funktionsuntergruppen

Der Schlüssel für der Funktionsuntergruppen besteht aus einer 7-stelligen Zeichenkette. Die erste Stelle kennzeichnet die Fahrzeugart, der diese Funktionsuntergruppe zugeordnet wird. Es stehen Schadensbücher für 3 Fahrzeugarten zur Verfügung, in denen die Fahrzeugarten jeweils durch einen anderen Buchstaben gekennzeichnet sind. Folgende Kennzeichen von Fahrzeugarten sind möglich :

P : Personenkraftwagen
L : Lastkraftwagen
U : Unimog

Die nächsten 4 Stellen geben das Kennzeichen der übergeordneten Funktionsgruppen an, woraus sich, wie oben gesehen, die übergeordnete Konstruktionsgruppe ableiten läßt.Die letzten 3 Stellen dienen nun zur Unterscheidung der Funktionsuntergruppen, die der gleichen Funktionsgruppe angehören. Der Aufbau des Schlüssels für Funktionsuntergruppen ist in Abbildung **7** dargestellt.

4.2.2 Aufbau einer Funktionsuntergruppe

Eine Funktionsuntergruppe besteht aus 5 Komponenten, die für den Wissenserwerb herangezogen werden können. Damit unterscheidet sie sich von Funktionsgruppen und Konstruktionsgruppen, die als übergeordnete Klassen nur Elemente ihrer untergeordneten

Klassen enthalten. Die 5 Komponenten lassen sich aufgrund ihres Inhalts noch weiter Zusammenfassen. Der Aufbau ist in Abbildung 8 dargestellt :

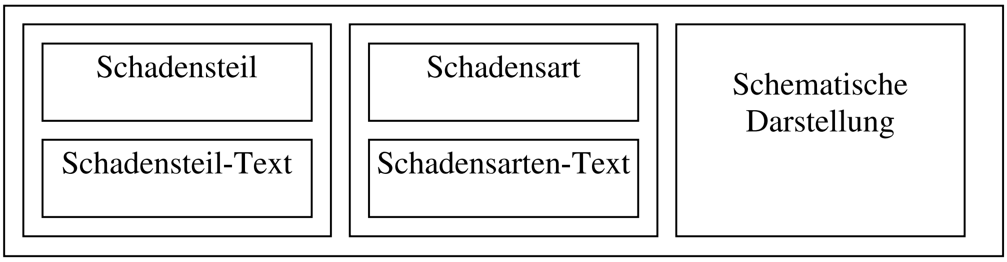


Abbildung 8 : Schematische Darstellung einer Funktionsuntergruppe

Die Komponenten einer Funktionsuntergruppe bilden die Schnittstelle zwischen den Fahrzeugteilen, über die Wissen gesammelt werden soll, und der Struktur des Schadenbuches. Wie später gezeigt wird, kann man mit Hilfe dieser Komponenten Wissen herleiten, das über die Struktur des Schadensbuches hinaus geht. Zunächst betrachten wir aber die einzelnen Komponenten :

1. **Schadensteil**
   Schadensteile bezeichnen die Fahrzeugteile, zwischen denen die funktionalen, räumlichen und kausalen Beziehungen gefunden werden sollen. Sie werden durch eine 5-stellige Nummer dargestellt, von der die ersten beiden Ziffern die Konstruktionsgruppe bezeichnen, zu der ein Teil gehört. Das bedeutet aber nicht, daß Schadensteile nur in den Funktionsuntergruppen auftreten, zu der sie aufgrund des Aufbaus ihrer Teilenummer gehören. Sie können in allen Funktionsuntergruppen auftreten, so daß sie gruppenübergreifende Verbindungen darstellen.
2. **Schadensteil-Text**
   Der Schadensteil-Text ist eine textuelle Beschreibung des Schadenteils.Um Wissen über Beziehungen zwischen Schadensteilen aus diesem Text ableiten zu können, ist eine Interpretation des Textes notwendig. Im Gegensatz zu einem Experten wird ein Laie mit diesem Beschreibungstext nicht sehr viel anfangen können. Trotzdem kann er zum Wissenserwerb durch einen Laien genutzt werden.
3. **Schadensart**
   Die Schadensarten bezeichnen mögliche Schäden, die für ein Schadensteil auftreten können. In einer Funktionsuntergruppe werden alle Schäden aufgezählt, die möglich sind. Jede Schadensart wird dabei durch eine 2-stellige Nummer dargestellt, so daß für jedes Schadensteil eine Liste möglicher Schäden entsteht.
4. **Schadensarten-Text**
   Der Schadensarten-Text ist eine textuelle Beschreibung der Schadensart. Wie beim Schadensteil-Text ist auch hier eine Interpretation des Textes notwendig, um Beziehungen zwischen Teilen ableiten zu können.

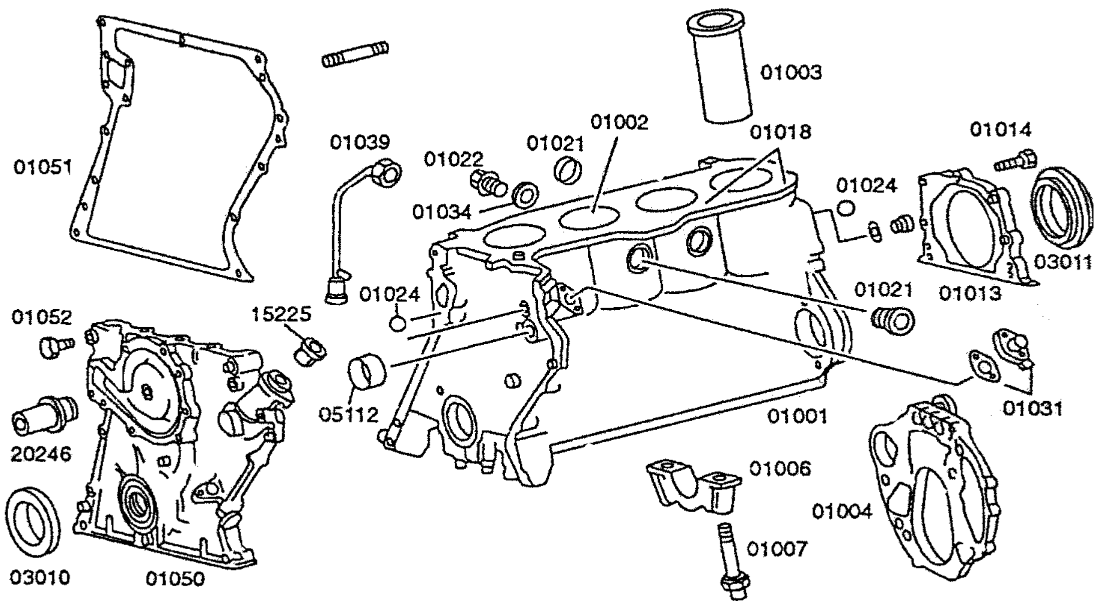
5. **Schematische Darstellung (nur gedruckte Form des Schadensbuchs)**
   Die Schematische Darstellung einer Funktionsuntergruppe enthält eine Zeichnung jedes Schadensteils, das in dieser Gruppe vorkommt, zusammen mit seiner Nummer. In einigen Zeichnungen ist zusätzlich die bauliche Anordnung der Teile andeutungsweise angegeben, was aber sehr ungenau ist und kaum wirkliche Rückschlüsse auf die Fahrzeugkonstruktion zuläßt.

4.2.3 Das Schadensbuch in gedruckter Form

Obwohl das Schadensbuch in unterschiedlichen Formen und Ausführungen vorliegt (d.h. in gedruckter und maschinenlesbarer Form oder als PKW- und LKW-Schadensbuch), spreche ich immer von dem einen Schadensbuch, da es immer die gleiche Struktur aufweist. Deutlich herausgestellt ist diese Struktur allerdings nur in den Schadensbüchern, die in gedruckter Form vorliegen. Diese Struktur wurde im vorherigen Kapitel dargestellt. Um eine genaue Vorstellung des Schadensbuchs zu bekommen, sollte man aber einen Ausschnitt gesehen haben. Abbildung 9 zeigt eine Funktionsuntergruppe, wie die im Schadensbuch in gedruckter Form vorhanden ist. Sie gehört zur Konstruktionsgruppe 01 und innerhalb dieser Konstruktionsgruppe zur Funktionsgruppe 0140. Wie man unterhalb der schematischen Darstellung erkennen kann, handelt es sich um die Funktionsuntergruppe P0140301.

01.40 Zylinderkurbelgehäuse / Steuergehäusedeckel

Teil	Art	Schadensteil-Text
01001	07 12 87	Zylinderkurbelgehäuse
01002	07 08 11 19	Zylinderbohrung Kurbelgehäuse
01003	08 11 19 38	Zylinderlaufbuchse
01004	19	Zwischenflansch Zylinderkurbelgehäuse / Getriebe
01006	07	Kurbelwellenlagerdeckel
01007	38	Schraube Kurbelwellenlagerdeckel
01013	04 12	Abschlußdeckel hinten
01014	03	Schraube Abschlußdeckel hinten
01015	04	Stiftschraube Zylinderkurbelgehäuse / Steuergehäuse
01018	07	Gewindebohrung Zylinderkurbelgehäuse
01021	03 47	Verschlußschraube / Verschlußdeckel Kernloch
01022	03	Schraube / Hahn Kühlmittelablaß
01024	03 04	Verschlußschraube / Kugel Ölkanal
01031	04	Verschlußflansch Kaftstoffpumpe
01034	04	Dichtung Schraube Kühlmittelablaß
01039	04 07	Leitung Kühlmittelablaß
01050	07 12 19	Steuergehäusedeckel
01051	82	Dichtfläche/Dichtung Steuergehäusedeckel
01052	84	Schraube Steuergehäusedeckel
03010	04	Abdichtring Kurbelwelle vorn
03011	04	Abdichtring Kurbelwelle hinten
05112	08 38	Lagerbuchse Zwischenradwelle
15225	99	Druckstück Zündverteiler
20246	08 38	Lagerzapfen Riemenspanner



P0140.3.01

Schadensarten-Text

03	lose, hierdurch undicht
04	undicht
07	gerissen/ausgerissen
08	Verschleiß/ausgeschlagen
11	riefig
12	porös
19	verzogen/verbogen/verspannt
38	lose
47	korrodiert/oxidiert
82	Ölverlust
84	Kühlmittelverlust
87	schlechter Durchgang/geknickt
99	schadhaft

Abbildung 9 : Beispiel einer Funktionsuntergruppe in gedruckter Form

4.2.4 Das Schadensbuch in maschinenlesbarer Form

Das Schadensbuch in maschinenlesbarer Form ist zwar nicht so übersichtlich und die Struktur ist nicht so leicht erkennbar, wie es bei den Schadensbüchern in gedruckter Form der Fall ist, dafür kann man es aber für einen automatischen Wissenserwerb verwenden. Es besteht aus insgesamt 5 Textdateien, die die Schadensbücher für alle 3 Fahrzeugarten enthalten. Der Aufbau dieser Textdateien richtet sich nicht mehr nach dem strukturellen Aufbau der Schadensbücher in gedruckter Form, mit Konstruktionsgruppen, Funktionsgruppen und Funktionsuntergruppen, sondern es werden nur noch die Komponenten der Funktionsuntergruppen beschrieben, worin natürlich die Struktur durch den Aufbau der Namen der Funktionsuntergruppen enthalten ist. Folgende Textdateien sind vorhanden :

1. SSLTEIL

Die Datei SSLTEIL besteht aus einer Tabelle, die eine Zuordnung von Schadensteil und Schadensteil-Text enthält. Sortiert ist diese Tabelle nach der Nummer des Schadensteils. Zu beachten ist, daß diese Datei alle Schadensteile enthält, unabhängig davon, in welchem Schadensbuchs es vorkommt. Außerdem existiert keine Unterscheidung der Schadensteilschlüssel nach Schadensbüchern, das bedeutet die Schadensteile sind eindeutig

```
...
05301  Schwinghebel/Kipphebel
05302  Buchse Schwinghebel/Kipphebel
05303  Schwing-/Kipphebelachse
05304  Schwing-/Kipphebellagerbock
05305  Schraube Schwing-/Kipphebellagerbock
05306  Federklammer/Sicherungsring Kipphebelachse
05307  Paßscheibe Kipphebelachse
05312  Kugelbolzen
05314  Einstellmutter/Hutmutter
05315  Einstellschraube Kugelbolzen
...
```

Abbildung 9 : Ausschnitt der Datei SSLTEIL

Ein Ausschnitt der Datei SSLTEIL ist in Abbildung 9 dargestellt.
Die Datei besteht insgesamt aus 8660 Schadensteilen.

2. SSLLANG

Die Datei SSLLANG besteht aus einer Tabelle, die eine Zuordnung von Schadensart und Schadensarten-Text enthält. Sortiert ist diese Tabelle nach der Nummer der Schadensart. Zusätzlich enthält sie eine Kennzeichnung, die angibt, ob es sich bei der jeweiligen Schadensart um einen Materialschaden (M) oder einen Lackschaden (L) handelt.

...
M 01 Folgearbeiten
M 02 innere Undichtheit
M 03 lose, hierdurch undicht
M 04 undicht
M 05 springt heraus
M 06 gebrochen
M 07 gerissen/ausgerissen
M 08 Verschleiß/ausgeschlagen
M 09 Oberflächenschaden
M 10 verkratzt
M 11 riefig
M 12 porös
M 13 schlecht verlegt
M 14 hakt
M 15 gefressen/sitzt fest
...

Abbildung 10 : Die ersten 15 Schadensarten der Datei SSLLANG

Ein Ausschnitt der Datei SSLLANG ist in Abbildung 10 dargestellt.
Sie enthält insgesamt 107 Schadensarten, davon 88 Materialschadensarten und 19 Lackschadensarten.

3. SSLFGR

Die Datei SSLFGR besteht aus einer Tabelle, die eine Zurodnung aller Funktionsgruppenschlüssel zu den Beschreibungen der Funktionsgruppen enthält. Zusätzlich ist noch die Fahrzeugart angegeben, zu der die Funktionsgruppe gehört. Dies ist durch die Buchstaben N, P, U gekennzeichnet. Es fällt auf, daß bei einigen Einträgen die Zugehörigkeit zu einer dieser Fahrzeugarten nicht gekennzeichnet ist. Wahrscheinlich handelt es sich um allgemeinere Funktionsgruppen, welche, wie ihre zugehörigen Funktionsuntergruppen, an keiner weiteren Stelle der Schadensbücher aufgeführt sind (weder in gedruckter noch in der maschinenlesbaren Form). Sortiert ist diese Tabelle nach dem Funktionsgruppenschlüssel.

...		
	03.00	Triebwerk allgemein
N	03.10	Triebwerk
P	03.10	Triebwerk
U	03.10	Triebwerk
	03.20	Kurbelwelle
	03.30	Schwungrad, Mitnehmerscheibe, Schwingungsdämpfer, Starterzahnkranz
N	03.30	Schwungrad/Schwungradgehäuse
	05.00	Motorsteuerung allgemein
P	05.10	Motorsteuerung
...		

Abbildung 11 : Ausschnitt aus der Datei SSLFGR

Ein Ausschnitt der Datei SSLFGR ist in Abbildung 11 dargestellt.
Sie enthält insgesamt 597 Funktionsgruppen.

4. SSLBILD
Die Datei SSLFGR besteht aus einer Tabelle, die eine Zuordnung aller Funktionsuntergruppenschlüssel zu den Beschreibungen der Funktionsuntergruppen enthält. Der Aufbau der Schlüssel der Funktionsuntergruppen ist in Abbildung 7, Seite 31 dargestellt. Die Datei enthält eine weitere Spalte, über deren genaue Bedeutung aber keine Informationen zu erhalten waren. Sortiert ist die Tabelle nach dem Schlüssel der Funktionsuntergruppen.

...		
P0110101	Motorbeanstandungen allgemein	...
P0120301	Zylinderkopfhaube	...
P0130601	Zylinderkopf	...
P0140301	Zylinderkurbelgehäuse/Steuergehäusedeckel	...
P0140302	Steuergehäusedeckel V8/V12 Verschlußdeckel Zylinderkurbelgehäuse	
...		
P0145301	Ölwanne	...
P0310401	Triebwerk	...
P0510201	Motorsteuerung	...
P0510102	Motorsteuerung Dieselmotor	...
P0520201	Nockenwellenverstellung M 104/M 111	.P.
P0520202	Nockenwellenverstellung M 119/M 120	.P.
P0530101	Ventile	
...		

Abbildung 12 : Ausschnitt aus der Datei SLLBILD

Ein Ausschnitt der Datei SSLBILD ist in Abbildung 12 dargestellt.
Sie enthält insgesamt 887 Funktionsuntergruppen.

5. SSLBEZUG
Die Datei SSLBEZUG besteht aus einer Zuordnung von Funktionsuntergruppen und Schadensarten zu Schadensteilen. Dazu werden nur die verschiedenen Schlüssel verwendet, was die Datei einerseits unübersichtlich werden läßt, andererseits aber die Zuordnungen eindeutig und unabhängig von textuellen Beschreibungen macht. Über die Verknüpfungen zwischen den jeweiligen Schlüsseln kann man die Beschreibungen aus den ersten vier Dateien erhalten.
Die Datei besteht prinzipiell aus einer Tabelle mit 3 Spalten, von denen die erste die Funktionsuntergruppe, die zweite das Schadensteil und die dritte eine Liste möglicher Schadensarten für dieses Teil angibt. Sortiert ist die Tabelle nach den Schadensteilen.

```
...
P0140301  01001  07 12 87
P0140302  01001  07 12 87
N0140301  01001  07 12 87
N0140202  01001  07 12 87
U0140101  01001  07 12 87
U0140102  01001  07 12 87
P0140301  01002  07 08 11 19
N0140202  01002  07 08 11 19
U0140101  01002  07 08 11 19
U0140102  01002  07 08 11 19
...
```

Abbildung 13 : Ausschnitt aus der Datei SSLBEZUG

Ein Ausschnitt der Datei SSLBEZUG ist in Abbildung 13 darstellt.
Sie enthält insgesamt 16353 Zuordnungen.

4.3 Verschlüsselung von Schäden an Fahrzeugkomponenten

Das Ziel des Schadensbuches besteht darin, Schäden an Fahrzeugen einheitlich zu erfassen. Aus diesem Grund wurde ein Schadensschlüssel eingeführt, der den Werkstätten vorgibt, wie Schäden zu beschreiben sind, so daß man gleiche Schäden auch gleich erfassen kann (auch wenn dieses Ziel nicht vollständig erreicht wird, vgl.4.2, Seite 28). Der Schadensschlüssel hat folgenden Aufbau :

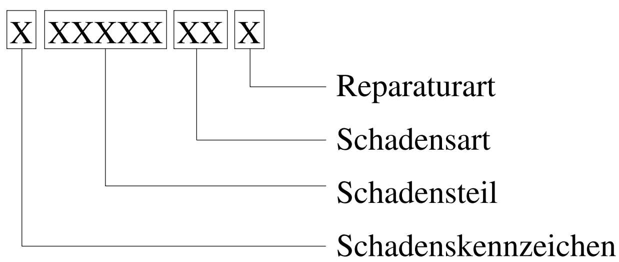


Abbildung 14 : Aufbau des Schadensschlüssels

Schadenskennzeichen
Das Schadenskennzeichen gibt an, ob es sich bei dem vorhandenen Schaden um einen Materialschaden (M) oder einen Lackschaden (L) handelt.

Schadensteil
Bezeichnet den Schlüssel eines oben beschriebenen Schadensteils

Schadensart
Bezeichnet den Schlüssel einer oben beschriebenen Schadensart. Dabei ist wichtig, daß nur eine, dem bezeichneten Schadensteil zugeordnete, Schadensart verwendet werden darf. Sollte bei möglichst weitgehender Auslegung keine der Schadensarten zutreffen, wird die Schadensart mit Sternchen '**' ausgefüllt und der Schaden im Mitteilungsfeld beschrieben.

Reparaturart
Beschreibt, wie ein Fehler repariert wurde. Je nachdem, ob es sich um einen Lackschaden oder einen Materialschaden handelte, werden unterschiedliche Schlüssel verwendet :

- Materialschäden
  1 Aggregat tauschen
  7 Instandsetzung mit Material
  8 Instandsetzung ohne Material
  9 Teile direkt von Werkstätten an Lieferanten

- Lackschäden
  7 ganze oder teilweise Nachlackierung
  8 Lackausbesserung

Der Schadensschlüssel besteht also aus einer neunstelligen Nummer, welche für jedes fehlerhafte Teil von den Werkstätten anzugeben ist. Dabei ist darauf zu achten, daß das schadensverursachende Teil angegeben wird.

5 Gruppenbildung anhand des Schadensbuches

Das Schadensbuch nimmt verschiedene Arten von Zuordnungen der Schadensteile vor. Betrachtet man die Dateien des Schadensbuches, so kann man folgende Arten von Zuordnungen erkennen:

Zuordnungen von Schlüsseln zu Schlüsseln
Diese Zuordnungen geschehen in der Datei SSLBEZUG. Die Datei besteht aus 3 Teilen, dem Schlüssel der Funktionsuntergruppe, dem Schlüssel des Schadensteil und dem Schlüssel der Schadensarten. Damit hat man folgende Möglichkeiten der Zuordnung :

Funktionsuntergruppenschlüssel ↔ Schadensteilschlüssel
Funktionsuntergruppenschlüssel ↔ Schadensartenschlüssel
Schadensartenschlüssel ↔ Schadensteilschlüssel

Das folgende Beispiel zeigt, wie man Zuordnungen vornehmen kann :

P0140301 01001 07 12

P0140301 ↔ 01001
P0140301 ↔ 07
P0140301 ↔ 12
07 ↔ 01001
12 ↔ 01001

Zuordnungen von Schlüsseln zu Texten
Diese Zuordnungen geschehen in den Dateien SSLTEIL, SSLLANG, SSLFGR und SSLBILD. Die Zeilen der Dateien bestehen aus 2 Teilen, einem Schlüssel und einem beschreibenden Text. Man hat hier also nur eine Möglichkeit der Zuordnung. Die einzelnen Dateien beschreiben also folgende Zuordnungen :

SSLTEIL	: Schadensteilschlüssel ↔ Beschreibung des Schadensteils
SSLLANG	: Schadensartenschlüssel ↔ Beschreibung der Schadensart
SSLFGR	: Funktionsgruppenschlüssel ↔ Beschreibung der Funktionsgruppe
SSLBILD	: Funktionsuntergruppenschlüssel ↔ Beschreibung einer Funktionsuntergruppe

Aus diesen Zuordnungen kann man nun Gruppen bilden. Man muß sich nur über die Elemente einer Gruppe im Klaren sein. Da es in dieser Arbeit um Beziehungen zwischen Fahrzeugteilen geht, ist es naheliegend, Fahrzeugteile in Gruppen zusammenzufassen. In einem anderen Anwendungsfall kann es vielleicht sinnvoll sein, Schadensarten zu gruppieren. Interessant sind also alle Zuordnungen, an denen Schadensteile beteiligt sind. Das ist zum einen bei den Zuordnungen der Datei SSLBEZUG der Fall, zum anderen bei der Datei SSLTEIL. Bei der Datei SSLBEZUG ist die Gruppenbildung relativ einfach, da es sich hier nur um Zuordnungen von Schlüsseln zu Schlüsseln handelt. Problematisch sind aber die Beschreibungen der Datei SSLTEIL, da hier zur Gruppenbildung eine Interpretation des Beschreibungstextes notwendig ist.

Eine weitere Möglichkeit zur Gruppenbildung wäre die Interpretation der anderen Beschreibungstexte. Da hier aber keine direkte Verbindung zu Schadensteilen besteht, wurde von dieser Möglichkeit kein Gebrauch gemacht. Beispielsweise könnte man die Beschreibung der Schadensartentexte interpretieren und ähnliche Schadenarten zu Gruppen zusammenfassen. Aus der Zuordnung von Schadensarten zu Schadensteilen könnte man dann Gruppen von Schadensteilen bilden
Im folgenden wird die Gruppenbildung mit Hilfe der SSLBEZUG und SSLTEIL beschrieben. Während die Gruppen aus der Datei SSLBEZUG automatisch gebildet werden können, ist bei der Datei SSLTEIL noch ein Interpretationsschritt notwendig. Wie die Gruppenbildung vorgenommen wird, wird in den nächsten beiden Abschnitten dargestellt.

5.1 Gruppenbildung mit der Datei SSLBEZUG

Das Schadensbuch der Daimler Benz AG gibt durch seine Struktur bestimmte Gruppen vor, denen man Schadensteile zuordnen kann. Diese Zuordnung wird durch die Zugehörigkeit zu Funktionsuntergruppen dargestellt, deren Schlüssel eine Hierarchie von Konstruktionsgruppen, Funktionsgruppen und Funktionsuntergruppen angeben. Das bedeutet, daß durch die Zugehörigkeit eines Schadensteils zu einer Funktionsuntergruppe auch die Zugehörigkeit dieses Teils zu der übergeordneten Funktionsgruppe und Konstruktionsgruppe gegeben ist. Diese Eigenschaft kann man sich bei der Gruppenbildung aus einer Zuordnung zu Nutze machen. Betrachtet man die Zuordnung aus dem obigen Beispiel, an der eine Funktionsuntergruppe beteiligt ist.

P0140301 ↔ 01001

Aus dieser einen Zuordnung kann man 3 Gruppen auf den 3 Ebenen bilden.

Konstruktionsgruppenebene :
- Konstruktionsgruppe 01 enthält Schadensteil 01001
  Kurzform : k01(t01001)

Funktionsgruppenebene
- Funktionsgruppe 0140 enthält Schadensteil 01001
  Kurzform : f0140(t01001)

Funktionsuntergruppenebene
- Funktionsuntergruppe p0140301 enthält Schadensteil 01001
  Kurzform : p0140301(t01001)

Eine weitere Möglichkeit der Gruppenbildung besteht durch die Zuordnung eines Schadensteils zu einer möglichen Schadensart. Das Beispiel enthält zwei Zuordnungen, an denen Schadensarten beteiligt sind.

07 ↔ 01001
12 ↔ 01001

Im Gegensatz zu den Schlüsseln der Funktionsuntergruppen lassen die Schlüssel der Schadensarten nicht weiter aufspalten. So bleibt für jede Zuordnung nur eine Möglichkeit der Gruppenbildung. Für das Beispiel bedeutet das folgendes :

Schadensartengruppe

- Die Schadensartengruppe 07 enthält das Schadensteil 01001
  Kurzform : s07(t01001)
- Die Schadensartengruppe 12 enthält das Schadensteil 01001
  Kurzform : s12(t01001)

Die Gruppenbildung, die am Beispiel dieser einen Zeile vorgeführt wurde, kann mit allen Zeilen der Datei SSLBEZUG durchgeführt werden.

5.2 Gruppenbildung mit der Datei SSLTEIL durch Textinterpretation

Die Datei SSLTEIL besteht aus Zuordnungen von Schadensteilen zu Schadensteilschlüsseln. Würde man hier dasselbe Prinzip zur Gruppenbildung anwenden wie bei der Datei SSLBEZUG, würde das wenig Fortschritt bringen, da man nur Gruppen erzeugen würde, die genau ein Element besitzen. Das folgende Beispiel soll die Problematik beschreiben :

05304 Schwing-/Kipphebellagerbock
05305 Schraube Schwing-/Kipphebellagerbock
05312 Kugelbolzen

Nach dem vorherigen Prinzip würde man folgende Gruppen bilden :

Schadensteilgruppe

- Die Schadensteilgruppe ‘Schwing-/Kipphebellagerbock’ enthält das Schadensteil 05304
- Die Schadensteilgruppe ‘Schraube Schwing-/Kipphebellagerbock’ enthält das Schadensteil 05305
- Die Schadensteilgruppe ‘Kugelbolzen’ enthält das Schadensteil 05312

Leider macht diese Gruppenbildung keinen Sinn, da jede so gebildete Gruppe genau ein Schadensteil enthält. Betrachtet man aber die Zeilen des Beispiels genauer, kann man anhand des Schadensteiltextes eine Verbindung zwischen den ersten beiden Schadensteilen herstellen. Selbst ein Laie kann feststellen, daß beide Schadensteile etwas mit einem „Schwinghebel-/Kipphebellagerbock“ zu tun haben. Das dritte Schadensteil kann ohne weiteres nicht eingeordnet werden. Möglicherweise gehört der „Kugelbolzen“ sogar zum „Schwinghebel-/Kipphebellagerbock“. Das kann ein Laie aber nicht entscheiden, dazu wäre ein Experte erforderlich.
Diese Fähigkeit, aufgrund der Interpretation des Schadensteiltextes auf eine Ähnlichkeit zwischen Schadensteilen zu schließen, kann man sich bei der Gruppenbildung zu Nutze machen. Man geht davon aus, daß zwischen zwei Schadensteilen eine funktionale oder räumliche Beziehung besteht, wenn aus einer Interpretation des Schadensteiltextes eine Zusammengehörigkeit der beiden Teile abgeleitet werden kann. Diese Schadensteile kann man dann zu einer Gruppe zusammenfassen. Interessant ist, daß man auf diese Art und Weise Gruppen erhält, welche die Einzelteile eines größeren, komplexeren Bauteils enthalten. In dieser größeren Gruppe sind dann wieder kleinere Gruppen enthalten, die dann die Schadensteile eines etwas kleineren Bauteils enthalten, das aber noch komplexer als ein einzelnes Schadensteil ist. Diese Gruppenbildung durch Interpretation des Schadensteiltextes soll nun an einem Beispiel dargestellt werden. (siehe Abbildung 15)

Abbildung 15 : Bilden von Einzelteilgruppen aus SSLTEIL

Aufgrund der Interpretation der Schadensteiltexte „Schwing-/Kipphebellagerbock“ und „Schraube Schwing-/Kipphebellagerbock“ der Teile 05304 und 05305 kann man vermuten, daß zwischen den Teilen ein Zusammenhang besteht, ohne zu Wissen, was ein Schwing-/Kipphebellagerbock wirklich ist. So werden die Teile zur Gruppe g00001 zusammengefaßt. Ebenso kann man die Gruppe g00002 bilden, deren Teile etwas mit der „Schwing-/Kipphebelachse“ zu tun haben. Gruppe g00003 entsteht dadurch, daß die Teile der Gruppen g00001 und g00002 zusammengefaßt werden, da alle etwas mit dem „Schwinghebel/Kipphebel“ zu tun haben. Gruppe g00004 besteht aus Teilen, die in Zusammenhang mit dem „Kugelbolzen“ stehen. Wie man an Teil 05314 sieht, kann es Teile geben, die keiner neuen Gruppe zugeordnet werden, da kein ähnliches Teil gefunden werden kann.

Interessant ist die entstehende Hierarchie der Gruppen g00003 mit den Untergruppen g00001 und g00002. Dabei enthält g00003 Schadensteile, eines größeren Bauteils, dem „Kipphebel“, während g00002 und g00003 diese Teile genauer gruppieren. g00001 enthält Teile des „Kipphebellagerbocks“ und g00002 Teile der „Kipphebelachse“.

Nun könnte man die Gruppen aufgrund der Zuordnung der Schadensteile benennen. Die Gruppe g00001 könnte beispielsweise den Namen „Kipphebelachse“ bekommen. Auf diese Benennung wurde allerdings bewußt verzichtet, da nicht immer sichergestellt ist, daß ein entsprechender Name von einem Laien gefunden werden kann. Außerdem würden diese für einen Laien unverständlichen Namen auch nicht unbedingt zur Übersichtlichkeit beitragen. Im Hinblick auf eine automatische Weiterverarbeitung der erzeugten Gruppen ist die Vergabe von Nummern für die einzelnen Gruppen allerdings von Vorteil.

Diese Art der Gruppenbildung erscheint in diesem Beispiel recht einfach, da der Zusammenhang zwischen den Teilen aufgrund eines gemeinsamen Wortes im Schadensteil-Text hergestellt wurde. Aber bei der großen Menge von Schadensteilen ist es problematisch, da man leicht die Übersicht verliert und so viele Zusammenhänge übersieht. Man könnte diese Gruppenbildung eventuell über einen automatischen Vergleich von Zeichenketten vornehmen. Man würde dabei aber Gruppenbildungen übersehen, die selbst ein Laie mit seinem Textverständnis finden könnte. Würde ein Experte aber diese Gruppen bilden, würde man wahrscheinlich zusätzliche Zusammenhänge entdecken. Da aber leider kein Experte zur Verfügung stand, konnte diese These nicht überprüft werden.

Nun wäre es Ideal, wenn man alle Teile der Datei SSLTEIL verwenden würde, um Gruppen zu bilden. Aufgrund der hohen Anzahl von Teilen ist dies aber nur mit sehr hohem Aufwand möglich. Außerdem wird es bei einer hohen Anzahl von Teilen und

bereits vorhandenen Gruppen immer schwieriger, ähnliche Teile zu entdecken und bereits vorhandene Gruppen zu erweitern. So habe ich mich auf eine Auswahl von Teilen beschränkt, die meiner Meinung nach im Zusammenhang mit der Funktion des Motor stehen. Dabei handelt es sich um alle Teile, deren Schlüssel mit folgenden Nummern beginnen : 00, 01, 03, 05, 07, 09, 13, 14, 15, 18, 20, 23, 24, 47, 49, 50, 51, 52, 53. Diese Nummern wurden ausgewählt, weil sie den Schlüsseln der Konstruktionsgruppen entsprechen, die mit dem Motor in Verbindung stehen. Dabei ist zu beachten, daß die Schadensteilschlüssel unabhängig von der Fahrzeugart sind, und somit kann man diese Auswahl sowohl dem PKW- als auch dem LKW-Schadensbuch zuordnen.
Nachdem die Gruppen soweit gebildet sind, wie es die Menge an Schadensteilen erlaubt, wird eine neue Datei mit Namen ETG_AUSW erzeugt. Jede Zeile dieser Datei enthält eine Zuordnung eines Schadensteils zu einer Gruppe, sowie die Beschreibung dieses Schadensteils.
Betrachtet man nur die Schadensteile des obigen Beispiels, sieht die Datei folgendermaßen aus:

```
g00001  05304  Schwing-/Kipphebellagerbock
g00001  05305  Schraube Schwing-/Kipphebellagerbock
g00002  05306  Federklammer/Sicherungsring Kipphebelachse
g00002  05307  Paßscheibe Kipphebelachse
g00002  05303  Schwing-/Kipphebelachse
g00003  05303  Schwing-/Kipphebelachse
g00003  05304  Schwing-/Kipphebellagerbock
g00003  05305  Schraube Schwing-/Kipphebellagerbock
g00003  05306  Federklammer/Sicherungsring Kipphebelachse
g00003  05307  Paßscheibe Kipphebelachse
g00003  05301  Schwinghebel/Kipphebel
g00004  05312  Kugelbolzen
g00004  05315  Einstellschraube Kugelbolzen
```

Dieses Beispiel stellt nur einen Ausschnitt der Datei ETG_AUSW dar. Insgesamt besteht sie aus 1320 Zeilen.
Betrachtet man im folgenden die Datei ETG_AUSW statt der Datei SSLTEIL, kann man an das unter Abschnitt 5.1 beschriebene Vorgehen anknüpfen. Man hat es nun mit Zuordnungen der Art Gruppenschlüssel ↔ Schadensteilschlüssel zu tun und kann die Gruppenzugehörigkeit auch ebenso aufschreiben.

Einzelteilgruppen

- Die Einzelteilgruppe 00001 enthält das Schadensteil 05304
  Kurzform : g00001(t05304)
- Die Einzelteilgruppe 00001 enthält das Schadensteil 05305
  Kurzform : g00001(t05305)
- Die Einzelteilgruppe 00002 enthält das Schadensteil 05306
  Kurzform : g00002(t05306)
- Die Einzelteilgruppe 00002 enthält das Schadensteil 05307
  Kurzform : g00002(t05307)
- Die Einzelteilgruppe 00002 enthält das Schadensteil 05303
  Kurzform : g00002(t05303)

Der Rest des Beispiels verläuft analog.

Die Kurzform der einzelnen Gruppenarten, die in diesem Kapitel gebildet wurden, entsprechen gerade den Fakten, wie sie in MOBAL verwendet werden. Dabei handelt es sich um einstellige Prädikate, deren Namen die Gruppen beschreiben und deren Argumente Schadensteilschlüssel sind. Im folgenden Kapitel wird ein Programm vorgestellt, das sowohl die Datei SSLBEZUG als auch die Datei ETG_AUSW einliest, und daraus für MOBAL lesbare Fakten generiert. Anschließend können diese Fakten mit STT weiterverarbeitet werden.

6 Generierung von MOBAL - Fakten aus dem Schadensbuch : DOCTOFAC

MOBAL bildet die Grundlage, um Beziehungen zwischen Fahrzeugteilen aufzudecken. Damit aber die Informationen des Schadensbuches mit MOBAL bearbeitet werden können, müssen sie in einer anderen Form vorliegen, als es im Schadensbuch der Fall ist. Die Aufgabe der Überführung der Fakten aus den verschiedenen Dateien des Schadensbuches in eine, für MOBAL verständliche Form, soll durch das Programm DOCTOFAC durchgeführt werden. Ausgangspunkt bilden die Dateien SSLBEZUG und SSLTEIL. Im vorherigen Kapitel wurde beschrieben, wie aus der Datei SSLTEIL durch Interpretation des Schadensteiltextes die Datei ETG_AUSW erzeugt wurde. Dabei wurden Gruppen von Schadensteilen gebildet, zwischen denen Aufgrund der Interpretation des Schadensteiltextes eine funktionale oder räumliche Beziehung abgeleitet werden konnte.
Um nur sinnvolle Fakten zu bilden, wird die Datei SSLBEZUG eingeschränkt auf die Konstruktionsgruppen 00-93, da alle weiteren Konstruktionsgruppen Lackschäden beschreiben. Die Lackschäden dürfen nicht zusammen mit anderen Schadensteilen betrachtet werden, da in der Datei SSLBEZUG keine Trennung der Schadensteilschlüssel für Materialschäden und Lackschäden vorhanden ist. Das heißt man könnte nicht unterscheiden, ob sich ein Faktum auf einen Materialschaden oder einen Lackschaden bezieht. Entfernt man alle Zeilen aus SSLBEZUG, die sich auf Lackschäden beziehen, entsteht die im folgenden verwendete Datei SSL0-93. Der Ablauf der Generierung von Fakten aus den Dateien des Schadensbuches ist in Abbildung 16 dargestellt. Zu beachten ist, das die Dateien SSLANG, SSLFGR und SSLBILD nicht verwendet werden, da sie keine Zuordnungen enthalten, an denen Schadenteile beteiligt sind (vgl. Kapitel 5)

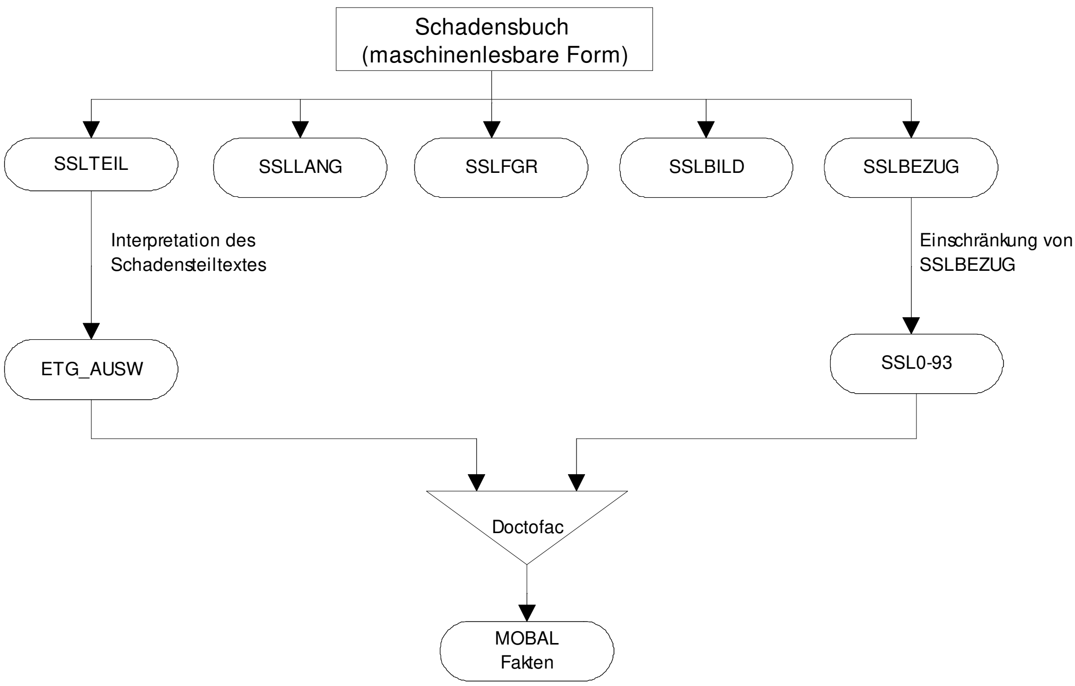


Abbildung 16 : Aufgabe von DOCTOFAC

6.1 Darstellung von DOCTOFAC

Die Aufgabe von DOCTOFAC besteht darin, aus gegebenen Dokumenten Fakten zu generieren, die von MOBAL eingelesen und zur Klassifikation von Schadensteilen weiterverarbeitet werden können. Dabei muß die Datei, die DOCTOFAC als Eingabe benötigt, ein bestimmtes Aussehen haben, das im folgenden erläutert werden soll. Die eigentliche Definition, wie die Fakten erzeugt werden sollen, erfolgt über ein Konfigurationsdatei, deren Pfad und Namen DOCTOFAC als Parameter erwartet.

6.1.1 Die Eingabedatei von DOCTOFAC

DOCTOFAC erzeugt Fakten aus einem gegebenen Dokument. Dieses Dokument ist als Textdatei gegeben und hat die Form einer Tabelle, deren Werte innerhalb einer Zeile durch Leerzeichen getrennt werden. Diese Form ist in allen Dateien des Schadensbuches gegeben, wobei aber gerade die SSLBEZUG von besonderem Interesse ist, da sie zum einen ohne Veränderung als Eingabe für DOCTOFAC dienen soll, zum anderen haben die Zeilen unterschiedliche Längen, d.h. sie besitzen eine unterschiedliche Anzahl von Werten. Dies soll nun an folgendem Ausschnitt von SSLBEZUG verdeutlicht werden.

S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
N0145101	01116	38							
N0145101	01117	06	38						
P0130601	01201	04	07	09	12	19	24	47	84
N0130101	01201	07	12	19					
U0130101	01201	07	12	19	47				
P0130601	01202	03	06	07	82	84	93		
N0130101	01202	04	06	07	93				
U0130101	01202	03	06	07	93				
P0130601	01203	03	06						

Tabelle 1 : Betrachtung der Eingabedatei von DOCTOFAC als Tabelle

Man kann diesen Ausschnitt nun zeilenweise betrachten. Jede Zeile für sich bietet eine Menge von Möglichkeiten, um Fakten zu bilden. Betrachtet man beispielsweise Zeile 3, so wäre ein mögliches Faktum P0130601(01201), was man interpretieren kann als „Das Schadensteil mit dem Schlüssel 01201 gehört zur Funktionsuntergruppe P0130601“. Betrachtet man aber die Beziehung zwischen Funktionsgruppe und Funktionsuntergruppe, so könnte man aus dieser Zeile auch das Faktum P0130(01201) ableiten. Wie die Fakten nun wirklich gebildet werden entscheidet der Benutzer, denn nur er weis, wie sinnvoll ein Faktum ist. Wichtig ist an dieser Stelle nur die Numerierung der Spalten $S_1,...,S_n$ der Eingabedatei, da dieser Bezug im Konfigurationsfile zur Definition von Fakten benötigt wird.

6.1.2 Die Konfigurationsdatei von DOCTOFAC

Die Konfigurationsdatei von DOCTOFAC enthält die nötigen Informationen, um Fakten erzeugen zu können. Die Form dieser Fakten erlaubt es, sie in MOBAL als Textdatei einzulesen. Der Inhalt der Konfigurationsdatei läßt sich in 3 Aufgabenbereiche gliedern :

- Festlegen der Eingabedatei und der Ausgabedatei

- Festlegen der Argumentposition in der Eingabedatei
- Festlegen der Prädikatsdefinition in der Eingabedatei

Die Datei ist eine reine Textdatei, in der jede Zeile einen Befehl und seine Argumente enthält. Weitere Befehle oder Argumente werden ignoriert. Der Befehl und die Argumente werden jeweils durch Leerzeichen getrennt. Die Befehle haben folgende Syntax :

INPUT <Arg$_1$>
OUTPUT <Arg$_1$>
ARGUMENT_DEFINITION <Arg$_1$> <Arg$_2$> <Arg$_3$> <Arg$_4$> <Arg$_5$>
PREDICATE_DEFINITION <Arg$_1$> <Arg$_2$> <Arg$_3$> <Arg$_4$> <Arg$_5$> <Arg$_6$> <Arg$_7$>
NO_INFORMATION
REM

Anhand dieser Befehle wird nun festgelegt, welche Datei als Eingabedatei gelesen wird, welche Spalte als Argument betrachtet wird, wie Namen von Prädikaten aus den vorhanden Spalten gebildet werden und in welche Ausgabedatei die so erzeugten Fakten geschrieben werden. Zusätzlich existiert noch eine Art von Filterfunktion, mit deren Hilfe bestimmte Zeilen der Eingabedatei bei der Erzeugung von Fakten übergangen werden können. Die Bedeutung der Befehle und ihrer Argumente wird im folgenden detailliert dargestellt.

INPUT <Arg$_1$>

Mit diesem Befehl wird die Datei angegeben, die als Eingabedatei gelesen werden soll. Arg$_1$ enthält den vollständigen Pfad zu dieser Datei einschließlich des Dateinamens. Sollte die Datei nicht geöffnet werden können, wird ein Laufzeitfehler erzeugt und das Programm abgebrochen. Der Aufbau des Pfades und des Dateinamens ist abhängig von dem System, auf dem DOCTOFAC ausgeführt wird

OUTPUT <Arg$_1$>

Mit diesem Befehl wird die Datei angegeben, in die die erzeugten Fakten geschrieben werden. Arg$_1$ enthält den vollständigen Pfad zu dieser Datei einschließlich des Dateinamens. Sollte die Datei nicht geöffnet werden können, wird ein Laufzeitfehler erzeugt, und das Programm abgebrochen. Der Aufbau des Pfades und des Dateinamens ist abhängig von dem System, auf dem DOCTOFAC ausgeführt wird.

ARGUMENT_DEFINITION <Arg$_1$> <Arg$_2$> <Arg$_3$> <Arg$_4$> <Arg$_5$>

Mit diesem Befehl wird die Spalte der Eingabedatei angeben, in der sich die Argumente der Prädikatsdefinitionen befinden. Da sich alle Prädikatsdefinitionen auf die selbe Spalte beziehen, darf dieser Befehl nur einmal verwendet werden. Sollte er mehr als einmal oder gar nicht gebraucht werden, wird das Programm mit einer Fehlermeldung abgebrochen.

Die Argumente haben folgende Bedeutung :

Arg$_1$: Typ : positive ganze Zahl

Gibt die Spalte in der Eingabedatei an, in der sich die Argumente der Prädikatsdefinitionen befinden. Die Eingabedatei wird dabei zeilenweise durchlaufen, und der jeweilige Text, der sich an der Position Arg_1 in der aktuellen Zeile befindet, wird ganz oder nur als Teil (siehe Arg_2 und Arg_3) als Argument des erzeugten Faktums eingesetzt.

Arg_2: Typ : positive ganze Zahl

Durch Arg_2 und Arg_3 wird ein Teil der Zeichenkette beschrieben, die durch Arg_1 festgelegt wurde. Arg_2 gibt die Startposition dieses Intervalls an. Dadurch besteht die Möglichkeit, daß nur ein Teil dieser Zeichenkette als Argument des Faktums eingesetzt wird.

Arg_3: Typ : positive ganze Zahl

Durch Arg_2 und Arg_3 wird ein Teil der Zeichenkette beschrieben, die durch Arg_1 festgelegt wurde. Arg_3 gibt die Endposition dieses Intervalls an.

Arg_4: Typ : String

Gibt eine Zeichenkette an, die als Kennzeichnung vor die durch Arg_1, Arg_2, und Arg_3 beschriebene Zeichenkette gehängt wird. Dies bietet die Möglichkeit, bestimmte Klassen von Argumenten zu kennzeichnen, um diese später in MOBAL besser von anderen Argumenten unterscheiden zu können.

Arg_5: Typ : String

Gibt eine Zeichenkette an, die als Muster dient, um bestimmte Zeilen bei der Faktenbildung auszulassen. Dabei wird die Zeichenkette, die durch Arg_1 beschrieben wird, mit Arg_5 zeichenweise verglichen. Bei Nichtübereinstimmung an einer Stelle der Zeichenketten, wird die aktuelle Zeile bei der Faktenbildung nicht berücksichtigt. Außerdem gibt es zwei Arten von Jokerzeichen in Arg_5 :

‘?’ Stimmt beim Vergleich mit jedem Zeichen überein. Die Zeichenketten werden danach aber weiter verglichen.
‘*’ Die Zeichenketten werden verglichen, bis Arg_5 dieses Zeichen enthält. Danach wird der Vergleich mit wahr abgebrochen.

PREDICATE_DEFINITION <Arg_1> <Arg_2> <Arg_3> <Arg_4> <Arg_5> Arg_6> <Arg_7>

Mit diesem Befehl werden die Spalten des Eingabedatei beschrieben, in der sich die Namen für die Prädikatsdefinitionen befinden. Um verschiedene Prädikate bilden zu können, kann dieser Befehl beliebig oft verwendet werden, so daß man viele verschiedene Fakten aus einer Zeile bilden kann. Der Befehl bietet auch die Möglichkeit mehrere, gleichartige Spalten gleichzeitig bei der Faktenbildung zu berücksichtigen, wodurch der Schreibaufwand verringert werden kann.

Die Argumente haben folgende Bedeutung :

$\underline{Arg_1:}$ Typ : positive ganze Zahl

Mit einer Anwendung dieses Befehls können die Namen der Prädikate aus verschiedenen gleichartigen Spalten der Eingabedatei beschrieben werden. Hierzu wird ein Intervall der Spaltennummern angegeben, die bei der Bildung der Prädikate berücksichtigt werden sollen. Arg_1 gibt dabei die untere Grenze dieses Intervalls an. Die Eingabedatei wird dabei zeilenweise durchlaufen, und in jeder Zeile werden die Spalten, die innerhalb dieses Intervalls liegen, bei der Prädikatsbildung als Namen für Prädikate berücksichtigt.

$\underline{Arg_2:}$ Typ : positive ganze Zahl oder *

Durch Arg_1 und Arg_2 wird ein Intervall der Spaltennummern der Eingabedatei beschrieben, die bei der Bildung von Prädikaten als Prädikatsnamen berücksichtigt werden. Arg_2 gibt dabei die obere Grenze dieses Intervalls an. Wird Arg_2 größer gewählt, als die tatsächlich vorhandene Anzahl der Spalten in der Eingabedatei (eine Variable Anzahl ist möglich), werden nicht vorhandene Spalten einfach nicht behandelt. Ist die maximale Anzahl von Spalten nicht bekannt, kann Anstelle einer Zahl auch ein '*' ,angegeben werden. Hierdurch werden alle weiteren Spalten der Eingabedatei bei der Faktenbildung berücksichtigt.

$\underline{Arg_3:}$ Typ : positive ganze Zahl

Durch Arg_3 und Arg_4 wird ein Teil der Zeichenketten beschrieben, die durch Arg_1 und Arg_2 festgelegt wurden. Arg_3 gibt die Startposition dieses Intervalls an. Dadurch besteht die Möglichkeit, daß nur ein Teil dieser Zeichenkette als Prädikatsname eingesetzt wird.

$\underline{Arg_4:}$ Typ : positive ganze Zahl

Durch Arg_3 und Arg_4 wird ein Teil der Zeichenketten beschrieben, die durch Arg_1 und Arg_2 festgelegt wurden. Arg_4 gibt die Endposition dieses Intervalls an.

$\underline{Arg_5:}$ Typ : positive ganze Zahl

Ebenso wie bei dem Befehl ARGUMENT_DEFINITION gibt es auch hier die Möglichkeit, bestimmte Zeilen bei der Bildung von Fakten auszulassen. Arg_5 gibt die Spalte der Eingabedatei an, auf die dieser Filter angewendet werden soll. Wird eine Spalte angeben, die in einer Zeile nicht vorhanden ist (bei unterschiedlichen Längen der Spalten), wird diese Zeile bei der Faktenbildung nicht berücksichtigt.

$\underline{Arg_6:}$ Typ : String

Gibt eine Zeichenkette an, die als Muster dient, um bestimmte Zeilen bei der Faktenbildung auszulassen. Dabei wird die Zeichenkette, die durch Arg_5 beschrieben wird, mit Arg_6 zeichenweise verglichen. Bei Nichtübereinstimmung an einer Stelle der Zeichenketten wird die aktuelle Zeile bei der Faktenbildung nicht berücksichtigt. Außerdem gibt es zwei Arten von Jokerzeichen in Arg_6 :

‘?’ Stimmt beim Vergleich mit jedem Zeichen überein. Die Zeichenketten werden danach aber weiter verglichen.
‘*’ Die Zeichenketten werden verglichen, bis Arg_6 dieses Zeichen enthält. Danach wird der Vergleich mit wahr abgebrochen.

Arg_7: Typ : String

Gibt eine Zeichenkette an, die als Kennzeichnung vor die durch Arg_1, Arg_2, und Arg_3 Arg_4 beschriebenen Zeichenketten gehängt wird. Dies bietet die Möglichkeit, bestimmte Arten von Fakten zu kennzeichnen, um später in MOBAL erkennen zu können, auf welcher Grundlage die jeweiligen Fakten entstanden sind. Beispielsweise könnte man jede Prädikatsdefinition mit einem anderen Kennzeichen versehen und so zu jeder Zeit nachvollziehen, aufgrund welcher Prädikatsdefinition ein Faktum entstanden ist.

NO_INFORMATION

Während der Ausführung von DOCTOFAC wird am Bildschirm angezeigt, wie viele Fakten bereits erzeugt wurden.
Nachdem DOCTOFAC aus der Konfigurationsdatei und der Eingabedatei Fakten gebildet hat und diese in die Ausgabedatei geschrieben hat, werden folgende Informationen am Bildschirm angezeigt :

- Anzahl der gelesenen Zeilen der Eingabedatei
- Anzahl der erzeugten Fakten für jedes verwendete Kennzeichen einer Prädikatsdefinition
- Anzahl der insgesamt erzeugten Fakten

Dieser Befehl bietet die Möglichkeit diese Anzeige zu unterdrücken.

REM
Durch diesen Befehl wird die Zeile als Kommentar angesehen

6.1.3 Die Ausgabedatei von DOCTOFAC

Die Ausgabedatei von DOCTOFAC enthält einstellige Fakten, die von MOBAL direkt eingelesen werden können. Dabei wird an jedes Faktum als Abschluß ein ‘.’ angehängt. Die einzelnen Fakten werden dann durch einen Zeilenumbruch getrennt.

Eine Zeile hat also folgenden Aufbau :

< Prädikatsname> (<Argument>).

Die Ausgabedatei könnte dann folgendermaßen aussehen :

```
k01(t00001).
f0110(t00001).
p0110101(t00001).
k18(t00001).
f1810(t00001).
```

Abbildung 17 : Beispiel für die Ausgabedatei von DOCTOFAC

6.1.4 Die Arbeitsweise von DOCTOFAC

Die Grundlage der Faktenbildung mit DOCTOFAC bilden die Eingabedatei und Definitionen von einstelligen Prädikaten und ihren Argumenten, die dann zeilenweise angewendet werden.
Wie aber die Befehle auf den Zeilen der Eingabedatei arbeiten, soll an einem Beispiel dargestellt werden. Dazu sei die folgende Eingabedatei gegeben, die bereits in Spalten aufgeteilt wurden.

	S_1	**S_2**	**S_3**	**S_4**	**S_5**	**S_6**	**S_7**	**S_8**	**S_9**	**S_{10}**
Z_1	N0145101	01116	38							
Z_2	N0145101	01117	06	38						
Z_3	P0130601	01201	04	07	09	12	19	24	47	84
Z_4	N0130101	01201	07	12	19					
Z_5	U0130101	01201	07	12	19	47				
Z_6	P0130601	01202	03	06	07	82	84	93		
Z_7	N0130101	01202	04	06	07	93				
Z_8	U0130101	01202	03	06	07	93				
Z_9	P0130601	01203	03	06						

Tabelle 2 : Beispiel für die Eingabedatei von DOCTOFAC

Außerdem sei folgende Konfigurationsdatei gegeben:

```
INPUT c:\ssl\ssl0-93
OUTPUT c:\fakten\fakten
ARGUMENT_DEFINITION 2 1 5 t *
PREDICATE_DEFINITION 1 1 2 5 1 P* f
PREDICATE_DEFINITION 3 10 1 2 1 P* s
```

Nachdem DOCTOFAC die Konfigurationsdatei gelesen und auf eventuelle Fehler überprüft hat, wird die erste Zeile Z_1 betrachtet. Begonnen wird mit der Auswertung der Argumentdefinition. Hierbei weist das erste Argument auf die Spalte hin, in der sich das Argument für alle Prädikatsdefinitionen dieser Zeile befindet. In diesem Fall ist es die Zeichenkette ‘01116’. Um nun zu bestimmen, ob die Zeile Z_1 zur Faktenbildung berücksichtigt werden soll, wird ‘01116’ mit dem 5. Argument, der Zeichenkette ‘*’, verglichen. Da es sich hier um ein Jokerzeichen handelt, wird diese Zeile weiter untersucht.

Danach wird das Argument dieser Zeile laut Argumentdefinition gebildet. Verwendet werden laut Argument 2 und Argument 3 die Zeichen 1 bis 5 der Zeichenkette, in diesem Falle also die gesamte Zeichenkette ‘01116’. Als Kennzeichnung wird ‘t’ verwendet, so daß das Ergebnis der Argumentdefinition ‘t01116’ lautet.
Im nächsten Schritt werden die Prädikatsdefinitionen betrachtet. Das Argument für alle diese Prädikate heißt ‘t01116’. Zuerst wird wieder anhand des Mustervergleichs festgestellt, ob das definierte Prädikat in dieser Zeile gebildet werden soll. Begonnen wird mit Arg_5 der ersten Prädiaktsdefinition, das einen Verweis auf die Spalte enthält, in der die Zeichenkette, die mit dem Muster, das in Arg_6 angegeben ist, verglichen werden soll. Verglichen werden soll danach die erste Spalte der aktuellen Zeile, also ’N0145101‘mit dem Muster ‘P*’. Da dieser Vergleich bereits beim ersten Zeichen fehlschlägt, wird die Prädikatsdefinition nicht weiter bearbeitet, und erzeugt somit auch kein Faktum. Da die zweite Prädikatsdefinition den selben Verweis und dasselbe Muster enthält, erzeugt auch sie kein Faktum’, so daß die gesamte Zeile kein Faktum erzeugt.

Bei Betrachtung der Zeile Z_2 wird ebenfalls kein Faktum erzeugt, da auch hier bei der Vergleich der Zeichenketten ‘N0145101’ und ‘P*’ beider Prädikatsdefinitionen fehlschlägt. Interessant wird es erst in der Zeile Z_3.

	S_1	**S_2**	**S_3**	**S_4**	**S_5**	**S_6**	**S_7**	**S_8**	**S_9**	**S_{10}**
Z_3	P0130601	01201	04	07	09	12	19	24	47	84

Die Argumentdefinition erzeugt hier die Zeichenkette ‘t01201’ als Argument aller Prädikate. Dann erfolgt der Mustervergleich der ersten Prädikatsdefinition zwischen „P0130601“ und ‘P*’, der erfolgreich abgeschlossen wird. Nun kann die eigentliche Erzeugung der Fakten beginnen. Welche Spalten der Eingabedatei dazu herangezogen werden, bestimmten Arg_2 und Arg_3 , also nur die Spalte S_1. Hier befindet sich die Zeichenkette ‘P0130601’, von der die Zeichen 2 bis 5 zur Bildung des Prädikats herangezogen werden. Als Kennzeichnung wird ‘f’ verwendet, so daß das Faktum ‘f0130(t01201).’ erzeugt wird.
Bei Anwendung der zweiten Prädikatsdefinition, besteht ein Unterschied darin, daß nicht nur eine Spalte zur Prädikatsdefinition herangezogen wird, sondern die Spalten S_3, S_4, S_5, S_6, S_7, S_8, S_9, und S_{10}. Von jeder Zeichenkette dieser Spalten werden die ersten beiden Zeichen herangezogen und durch ein ’s’ gekennzeichnet, so daß folgende Fakten entstehen: ‘s04(t01201).’, ‘07(t01201).’, ‘s09(t01201).’, ’s12(t01201).’, ‘s19(t01201).’, ‘s24(t01201).’, ‘s47(t01201).’, ‘s84(t01201).’

Für die gesamte Eingabedatei ergeben sich dann folgende Fakten :

	Fakten
Z_1	-
Z_2	-
Z_3	f0130(t01201). s04(t01201)., s07(t01201)., s09(t01201)., s12(t01201)., s19(t01201)., s24(t01201)., s47(t01201)., s84(t01201).
Z_4	-
Z_5	-
Z_5	-
Z_6	f0130(t01201) s03(t01202)., s06(t01202)., s07(t01202)., s82(t01202)., s84(t01202)., s93(t01202).
Z_7	-
Z_8	-
Z_9	f0130(t01203) s03(t01203)., s06(t01203).

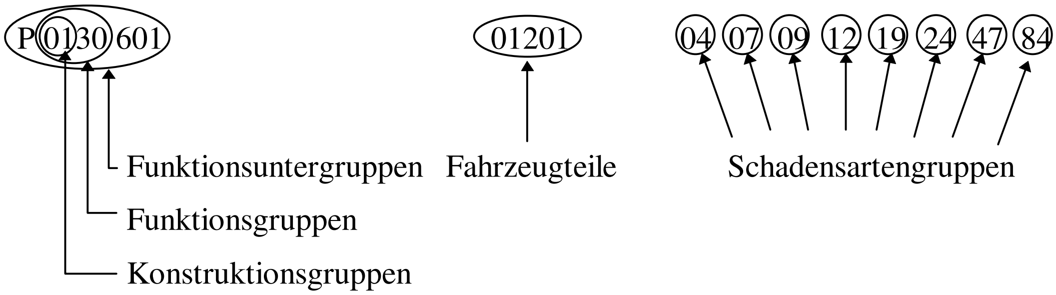
Abbildung 18 : Erzeugte Fakten des Beispiels

6.2 Anwendung von DOCTOFAC auf die Datei SSLBEZUG

Die Datei SSLBEZUG enthält eine Zuordnung von Funktionsuntergruppenschlüsseln und Schadensartenschlüsseln zu den einzelnen Bauteilen und Komponenten eine Fahrzeugs. Aus den Funktionsuntergruppenschlüsseln kann man leicht auf die übergeordneten Funktionsgruppenschlüssel und Konstruktionsgruppenschlüssel schließen Diese Zuordnungen kann man nun als für MOBAL verständliche Fakten in Form von einstelligen Prädikaten darstellen. Dabei gibt der Name des Prädikats eine Gruppe an, wie zum Beispiel eine Funktionsuntergruppe. Das Argument bezeichnet dann das Teil das dieser Gruppe angehört. Die Zugehörigkeit des Schadensteils '01201' zu der Funktionsuntergruppe 'P013601' wird dann beispielsweise beschrieben als 'P0130601(t01201).'

Bei einer genaueren Betrachtung einer Zeile aus SSLBEZUG, kann man folgende Zuordnungen von Fahrzeugteilen zu Arten von Gruppen erkennen :

Abbildung 19 : Bedeutung einer Zeile von SSLBEZUG



6.2.1 Einschränkung von SSLBEZUG auf SSL0-93

Problematisch ist, das in der Datei SSLBEZUG keine Unterscheidung zwischen Materialschäden und Lackschäden vorgenommen wird. Das bedeutet beispielsweise, daß die Schadensarten M02 und L02 in der Datei SSLBEZUG beide als Schadensart 02 dargestellt werden. Die beiden Schadensarten sind aber vollkommen verschieden. Bei M02 handelt es sich um den Schadensarten-Text „innere Undichtheit“ während L02 den Schadensarten-Text „Abklebekanten“ besitzt. Würde man nun mit der gesamten Datei SSLBEZUG Fakten bilden, könnte man anhand der Fakten nicht mehr unterscheiden, um welche Schadensart es sich wirklich handelt. Aus diesem Grund wurden die Zeilen aus der Datei SSLBEZUG entfernt, bei denen Lackschäden verwendet wurden. Das ist allerdings nur bei den Teilen der Konstruktionsgruppe 98 der Fall. Da die Konstruktionsgruppen 94-97 nicht verwendet werden, bleiben nur die Konstruktionsgruppen 00-93. Diese werden in der Datei SSL0-93 zusammengefaßt.

6.2.2 Erstellen der Konfigurationsdateien

Um die Position der Bezeichnungen für die einzelnen Gruppenarten und die Fahrzeugteile für die Bearbeitung von SSL0-93 mit DOCTOFAC festzulegen, muß eine passende Konfigurationsdatei erstellt werden. Die Konfigurationsdatei enthält den Pfad zur Eingabedatei SSL0-93 und einer Ausgabedatei, deren Name in Abhängigkeit von dem verwendeten Filter gewählt wird (vgl. Abbildung 20). So kann beispielsweise eine getrennte Betrachtung von Schadensartengruppen und Funktionsuntergruppen sinnvoll sein. Die folgenden Arten von Gruppen von Fakten sollen hier betrachtet werden.

- Konstruktionsgruppen
- Funktionsgruppen
- Funktionsuntergruppen
- Schadensartengruppen
- Konstruktionsgruppen & Funktionsgruppen & Funktionsuntergruppen
- Konstruktionsgruppen & Funktionsgruppen & Funktionsuntergruppen & Schadensartengruppen

Es bietet sich an, für jede dieser Kombinationen eine eigene Konfigurationsdatei zu erstellen, um die erzeugten Fakten getrennt mit MOBAL weiterverarbeiten zu können. Eine andere Möglichkeit wäre, alle Fakten gleichzeitig zu erstellen und in eine Datei zu schreiben. Anschließend müßten die Fakten dieser Datei aber mit einem geeigneten Programm selektiert werden. Im folgenden wird eine Konfigurationsdateien als Beispiel angegeben, durch die die gewünschten Fakten für Konstruktionsgruppen von PKW direkt aus SSL0-93 erzeugt werden. Weitere Konfigurationsdateien sowie die Anzahl der erzeugten Fakten sind in Abschnitt 7.4 aufgeführt. Da SSL0-93 die Daten von 3 verschiedenen Schadensbüchern enthält, ist es sinnvoll die Fakten nach Büchern getrennt zu erzeugen, da man so eher die Möglichkeit hat, fahrzeugspezifische Abhängigkeiten zu finden. Die Konfigurationsdateien für das PKW- Schadensbuch unterscheiden sich von denen für das LKW-Schadensbuch nur durch die Filter in der Prädikatsdefinition.

Konstruktionsgruppen (k-Fakten)
Konfigurationsdatei : pkw_kg.cfg

```
INPUT c:\ssl\ssl0-93
OUTPUT c:\Fakten\pkw_kg.fak
ARGUMENT_DEFINITION 2 1 5 t *
PREDICATE_DEFINITION 1 1 2 3 1 P* k
```

Abbildung 20 : Die Konfigurationsdatei zur Generierung von k-Fakten

6.3 Anwendung von DOCTOFAC auf Einzelteilgruppen

Analog zur Datei SSL0-93 kann man DOCTOFAC auf die Datei ETG_AUSW anwenden. Sie besteht ebenfalls aus Zuordnungen von Schadensteilen zu Gruppen, den Einzelteilgruppen. Damit man DOCTOFAC anwenden kann, muß die Datei in Spalten aufgeteilt und eine Konfigurationsdatei entstellt werden. Benötigt wird wieder ein Leerzeichen als Trennzeichen zwischen den einzelnen Spalten. Da der in der Datei enthaltene Schadensteiltext Leerzeichen enthalten kann, erhält man eine variable Anzahl Spalten. Da aber nur die ersten beiden Spalten zur Faktenbildung benötigt werden, ist der Rest der Zeile unerheblich. Er diente nur zum Bilden der Einzelteilgruppen, und kann an dieser Stelle vernachlässigt werden. Wichtig sind nur der Gruppenname und der Schadensteilschlüssel in den Spalten S_1 und S_2. Die folgende Abbildung 21 zeigt die Aufteilung einer Zeile in Spalten. Die Spalten S_3 bis S_5 resultieren aus Leerzeichen im Schadensteil-Text.

	S_1	**S_2**	**S_3**	**S_4**	**S_5**
Z_1	g00052	05084	Kettenrad	Nockenwellenverstellung	rechts

Abbildung 21 : Aufteilung einer Zeile von ETG_AUSW

Mit Hilfe von DOCTOFAC werden nun aus den Spalten S_1 und S_2 Fakten erzeugt. Dabei soll aus S_1 der Prädikatsname und aus S_2 das Argument gebildet werden. Die Konfigurationsdatei enthält die gleiche Zeile für die Argumentdefinition, wie die Konfigurationsdateien, die auf der Datei SSL0-93 arbeiten, da das Schadensteil ebenfalls in der Spalte S_2 in derselben Form steht.
Für die Prädikatsdefinition werden die Zeichen 2-6 der Zeichenkette in Spalte S_1 verwendet. Anschließend wird das ‘g’ zur Kennzeichnung verwendet.

Einzelteilgruppen (g-Fakten)
Konfigurationsdatei : etg_ausw.cfg

```
INPUT c:\ssl\etg_ausw
OUTPUT c:\Fakten\pkw_etg.fak
ARGUMENT_DEFINITION 2 1 5 t *
PREDICATE_DEFINITION 1 1 2 6 1 * g
```

Abbildung 22 : Die Konfigurationsdatei zur Generierung von g-Fakten

6.4 Realisierung von DOCTOFAC

Bei DOCTOFAC handelt es sich um ein in der Programmiersprache C++ geschriebenes Programm. Es läßt sich leicht auf verschiedene Systeme übertragen und wurde bereits mit verschiedenen Compiler übersetzt. Dazu gehören Borland C++ Version 2.0 für MS-DOS, Microsoft Visual C++ Version 4.0 für WINDOWS NT, SUN C++ Version 3.0.1 für UNIX. Der Programmablauf von DOCTOFAC läßt sich in folgende Schritte aufteilen:

- **Einlesen Konfigurationsdatei**
  Die Konfigurationsdatei wird zeilenweise gelesen und auf Fehler überprüft. Konnte kein Fehler gefunden werden, wird der Inhalt der Zeilen in einer Datenstruktur abgelegt. Diese enthält die Namen der Ein- und Ausgabedatei, die Argumentdefinition, die Anzahl der Prädikatsdefinitionen und zwei Zeiger auf eine dynamische Liste, die die Prädikatsdefinitionen enthält. Somit ist die Anzahl der Prädikatsdefinitionen nur abhängig von der Größe des verfügbaren Speichers.

- **Lesen der Eingabedatei und Generierung der Fakten**
  Die Eingabedatei wird zeilenweise gelesen und aus jeder Zeile werden entsprechend der Definition Fakten gebildet, wenn diese Zeile den Mustern der Filter für Argumentdefinition und Prädikatsdefinition entsprechen. Dabei wird in jeder Zeile zuerst das Argument des Faktums gebildet. Danach wird die Liste der Prädikatsdefinitionen durchlaufen und jeder Eintrag dieser Liste erzeugt Fakten. Die so erzeugten Fakten werden nicht sofort in die Ausgabedatei geschrieben, da es vorkommen kann, daß identische Fakten erzeugt werden. Statt dessen werden die Fakten in einer dynamischen Liste im Hauptspeicher gehalten, und für jedes neu erzeugte Faktum wird überprüft, ob es bereits vorhanden ist. Ist dies nicht der Fall, wird es an die Liste der Fakten angehängt. Während der Faktengenerierung wird die Anzahl eingetragenen Fakten und die Anzahl der gelesenen Zeilen festgehalten.

- **Schreiben der Ausgabedatei und Ausgabe der Informationen**
  Nachdem die gesamte Eingabedatei gelesen wurde, und alle Fakten generiert wurden, wird die Liste der Fakten in die Ausgabedatei geschrieben. Dabei entspricht jedes Faktum einer Zeile in der Ausgabedatei.
  Im letzten Schritt werden, wenn dies in der Konfigurationsdatei nicht unterbunden wurde, die Anzahl der gelesenen Zeilen und die Anzahl der Fakten eines bestimmten Typs ausgegeben.

Bei der Programmierung von DOCTOFAC wurde Wert auf eine dynamische Datenstruktur gelegt, damit weder die Anzahl der Prädikatsdefinitionen in der Konfigurationsdatei, noch die Anzahl der erzeugten Fakten beschränkt ist (außer durch die Größe des Hauptspeichers). Aus diesem Grunde wurden dynamische Listen als Datenstruktur verwendet. Bei der Speicherung des Inhalts der Konfigurationsdatei ist der Einsatz von Listen nicht weiter tragisch, da die Liste beim Einlesen der Konfigurationsdatei einmal erzeugt wird und danach für jede Zeile der Eingabedatei einmal vollständig durchlaufen werden muß. Weniger effizient ist die Speicherung der erzeugten Fakten in einer Liste, da bei jedem neu erzeugten Faktum ein Durchlauf der Liste erforderlich ist, um festzustellen, ob dieses Element bereits vorhanden ist. Das bedeutet im worst case eine Laufzeit von $O(n^2)$. Zur Verbesserung der Laufzeit könnte man eine Baumstruktur anstatt der Listenstruktur verwenden, was eine Laufzeit von

O(log n) zur Folge hätte. Im Hinblick auf die Laufzeit und den Speicherbedarf von STT (vgl. Abschnitt 7.4) schien es aber nicht von erforderlich, die Laufzeit von DOCTOFAC zu optimieren. Die Laufzeit von DOCTOFAC beträgt nur einen Bruchteil der Laufzeit von STT. Außerdem müssen alle Fakten nur einmal generiert werden.

Einschränkungen

- Die Anzahl von Zeichen einer Zeile der Konfigurationsdatei oder der Eingabedatei ist auf 1000 beschränkt.
- Die Anzahl von Zeichen der Namen der Eingabedatei und der Ausgabedatei sind auf 200 beschränkt.
- Die Anzahl von Zeichen der Musters bei der Argumentdefinition oder der Prädikatsdefinition ist auf 100 beschränkt. Das bedeutet
  Länge(Arg_6) in PREDICATE_DEFINITION <= 100
  Länge(Arg_5) in ARGUMENT_DEFINITION <=100.
- Die Anzahl von Zeichen zur Kennzeichnung von Gruppen oder Schadensteilen ist auf 100 beschränkt. Das bedeutet :
  Länge(Arg_7) in PREDICATE_DEFINITION <= 100
  Länge(Arg_4) in ARGUMENT_DEFINITION <=100.

Fehlermeldungen

Während DOCTOFAC die Konfigurationsdatei einliest, wird sie auf mögliche Fehler überprüft. So kann es zum Beispiel vorkommen, das die Eingabedatei oder die Ausgabedatei nicht vorhanden ist, oder das bei den Befehlen PREDICATE_DEFINITION oder ARGUMENT_DEFINITION die Intervallgrenzen vertauscht wurden. Wird ein Fehler entdeckt, bricht das Programm mit einer Fehlermeldung ab. Die möglichen Fehlermeldungen sind im folgenden aufgeführt. Sie sind im wesentlichen selbsterklärend.

```
More than one inputfile
More than one outputfile
More than one argument definition
No predicate definition
No inputfile
No outputfile
No argument definition
No configfile
No predicate definition
Incorrect configfile
Cannot open file for input: ",<filename>
Cannot open file for output: ",<filename>
Empty Pattern
Incorrect range for predicate
Unknown Syntax
Not enough memory
Line too long
Argument too long
```

7 Klassifikation von Schadensteilen mit STT

Bisher wurde immer von Gruppen von Schadensteilen gesprochen. Es gab Konstruktionsgruppen, Funktionsgruppen, Funktionsuntergruppen, Schadensartengruppen und die von Hand gebildeten Einzelteilgruppen. Mit Hilfe von DOCTOFAC wurden dann Fakten generiert, die die Gruppenzugehörigkeiten beschreiben. Diese Fakten können nun als Text in MOBAL eingelesen und mit STT weiterverarbeitet werden. Wendet man STT auf die erzeugten Fakten an, so entsprechen die Argumentsorten von SST gerade den Gruppen, die vorher bestimmt wurden. Nun könnte man sagen, daß das noch kein Fortschritt sei, den die Gruppen waren vorher auch schon vorhanden. Das ist im Prinzip richtig, die Fahrzeugteile sind einer Gruppe zugeordnet, aber man hatte die Fahrzeugteile einer Gruppe nicht vor sich liegen. Die Zugehörigkeit lag in der Datei SSLBEZUG verstreut, und man konnte die Fahrzeugteile einer Gruppe nur dadurch erhalten, daß man die gesamte Datei danach durchsuchte.

Im Blick auf die Konstruktionsgruppen, Funktionsgruppen und Funktionsuntergruppen kann man natürlich einen Blick auf Schadensbuch in gedruckter Form werfen, um diese Gruppen zu erhalten. Berücksichtigt man aber die Menge der anfallenden Daten und die Tatsache, daß man diese Daten zur elektronischen Weiterverarbeitung nutzen will, so wird der Gewinn deutlicher.

Die Schadensartengruppen sind allerdings in keiner Form der Schadensbücher direkt enthalten. Dadurch, daß STT die Argumentsorten bildet, werden die Gruppen erst aus den Schadensbüchern herausgezogen und somit für den Benutzer direkt sichtbar. Einzelteilgruppen werden von Hand gebildet und sind damit von Beginn an vorhanden.

Interessanter sind aber die Äquivalenzklassen, die STT bildet. Hier werden Gruppen, die die gleichen Fahrzeugteile beinhalten, zu einer Klasse zusammengefaßt. Die Klassen sind damit gruppenübergreifend und man erhält mit Hilfe der Teilmengenbeziehungen zwischen den Extensionen der Klassen eine partielle Ordnung auf diesen Klassen. So können beispielsweise eine Schadensartengruppe und eine Einzelteilgruppe die selben Fahrzeugteile enthalten. Da es sich bei den Argumenten der Fakten, die anhand dieser beiden Gruppen erzeugt werden, um die selben Fahrzeugteile handelt, bildet STT zwei Argumentsorten mit gleicher Extension. Diese Argumentsorten werden dann in einer Klasse zusammengefaßt.

7.1 Bilden der Äquivalenzklassen

Die von DOCTOFAC erzeugte Ausgabedatei dient MOBAL als Eingabedatei. Die Fakten werden als Textdatei eingelesen und in die Faktenbasis von MOBAL übernommen. Dazu dient der Befehl [File/Read From Text]. Nachdem die Fakten eingelesen wurden, wird STT über den Befehl [Call/Update Sorts] aktiviert und beginnt damit, die Klassen zu berechnen.[3] Um die Extensionen der Argumentsorten zu berechnen, werden alle Fakten der Faktenbasis betrachtet, und für jede Argumentstelle jedes Prädikats eine eigene Extension erstellt. Der Name der Argumentsorte wird von der Funktion arg_sort aus dem Prädikatsnamen und aktuellen Argumentstelle generiert, und ist somit eindeutig [vgl. Abschnitt 3.1]. Da DOCTOFAC aber nur einstellige Fakten generiert, existiert für jedes Prädikat genau eine Argumentsorte. Der Sortenname enthält also immer den Hinweis auf den Prädikatennamen, der dem Gruppennamen mit einer Kennzeichnung entspricht, und

[3] MOBAL bietet eine Reihe von Einstellungen. Unter anderem kann auch festgelegt werden, wann STT gestartet wird und ob Schnittsorten berechnet werden sollen.
Diese Einstellungen im Menue [View/Parameter Display] vorgenommen.

auf die erste Argumentstelle. In den Extensionen werden nun die Schadensteile einer Gruppe gesammelt. An folgendem Beispiel läßt sich die Arbeitsweise verdeutlichen :

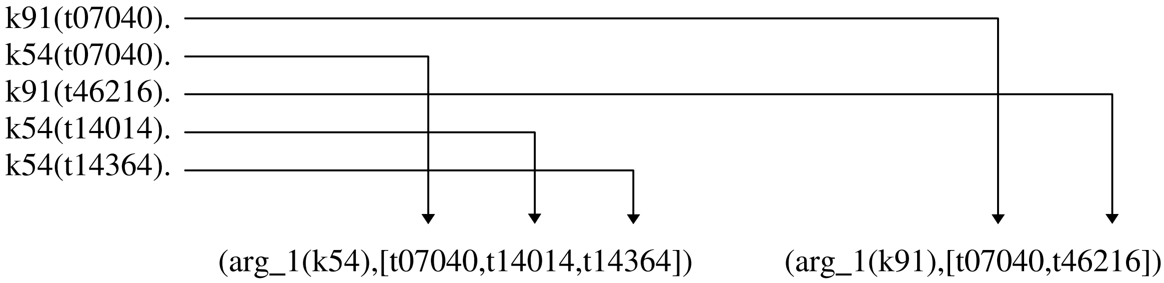


Abbildung 23 : Bilden von Extensionen mit STT

Aus diesen Extensionen werden die Äquivalenzklassen gebildet. Das bedeutet, daß Argumentsorten mit gleicher Extension zu einer Klasse zusammengefaßt werden. Dabei handelt es sich um Gruppen, die die gleichen Schadensteile beinhalten. Für Argumentsorten, deren Extension mit keiner anderen Extension übereinstimmt, wird eine eigene Klasse gebildet. Beispielsweise könnte man folgende Argumentsorten zusammenfassen.

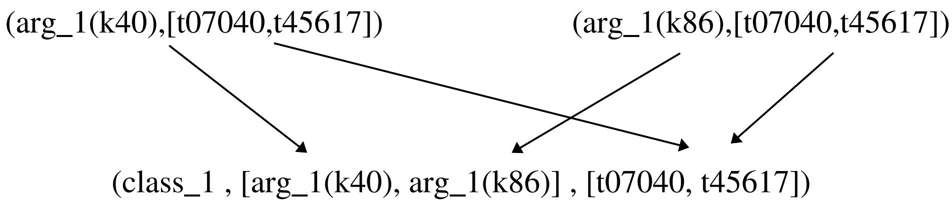


Abbildung 24 : Bilden der Äquivalenzklassen mit STT

Die Sortennamen 'arg_1(k40)’ und ‘arg_1(k86)’ werden hier in der Klasse class_1’ zusammengefaßt. Der Klassenname ‘class_1’, wird von MOBAL automatisch vergeben. Die Extensionen der beiden Argumentsorten, die laut Klassendefinition identisch sein müssen, werden in die Klasse übernommen. Werden alle Klassen berechnet, ist das Ergebnis eine Liste von Klassen in folgender Form :

(<Klassenname> , <Liste von Sortennamen> , <Liste von Argumenten>)

7.2 Berechnung von Schnittsorten

Die Möglichkeit von STT zur Berechnung von Schnittsorten könnte viele interessante Gruppen liefern, die Abhängigkeiten zwischen Bauteilen beschreiben. Allerdings ist der Aufwand zur Berechnung der Schnittsorten sehr groß. Betrachtet man die Tatsache, das im worst case 2^n Schnitte für n Klassen existieren, kann man erkennen, daß die Berechnung schon für eine relativ geringe Anzahl von Klassen sehr viel Zeit in Anspruch nimmt. Außerdem müssen eventuell neu generierte Klassen wiederum in die Berechnung weiterer Schnittsorten einbezogen werden, was wiederum viel Zeit kostet.
Aus diesem Grund ist die Berechnung der Schnittsorten nur für eine kleine Anzahl von Klassen bzw. mit wenigen Bauteile effektiv durchführbar. Will man die Berechnung mit einer geringen Anzahl von Klassen durchführen entsteht die Schwierigkeit, die richtigen

Bauteile auszuwählen, so daß aussagekräftige Klassen entstehen. Beschränkt man sich nämlich auf eine geringere Anzahl von Bauteilen, ändern sich Extensionen der Argumentsorten und es können andere Klassen erzeugt werden. Somit unterschiedet sich auch die Berechnung der Schnittsorten mit einer Auswahl von Bauteilen von der aller Bauteilen, ohne ein darin enthalten zu sein. Sinnvoll wäre nur eine gezielte Auswahl von Bauteilen, was aber weites Hintergrundwissen erfordert.

7.3 Eigenschaften der generierten Klassen

Eine Klasse besteht aus einem Klassennamen, einer Menge von zugehörigen Argumentsorten, die dem Gruppennamen entsprechen und einer Extension, die wiederum aus einer Menge Schadensteilen besteht (vgl. Abbildung 24). Sowohl die Klassennamen, als auch die Menge von Argumentsorten und die Extensionen sind voneinander verschieden. Interessant sind die Extensionen, da sie mit Hilfe der Teilmengenbeziehung eine partielle Ordnung zwischen den Klassen bilden. SST bietet die Möglichkeit, diese partielle Ordnung graphisch darzustellen.

Betrachtet man die Darstellung, in der nur Gruppierungen nach Konstruktionsgruppen, Funktionsgruppen und Funktionsuntergruppen enthalten, so kann man die implizite Struktur der Gruppenarten erkennen. Da diese Gruppen aufgrund ihrer Definition ineinander enthalten sind, bilden sie eine Baumstruktur. Abbildung 25 enthält ein Beispiel für diese Baumstruktur. Hier kann man die Ebene der Konstruktionsgruppen, Funktionsgruppen und Funktionsuntergruppen unterscheiden.

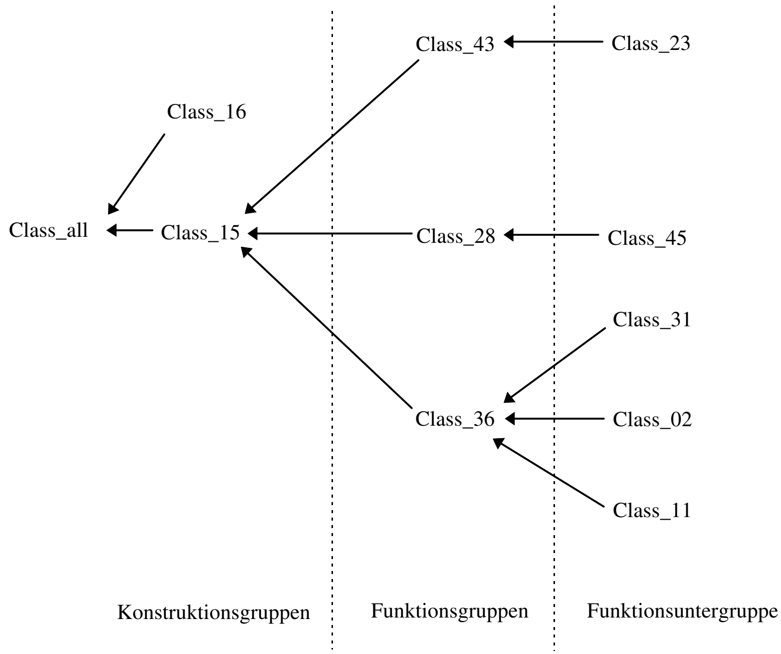


Abbildung 25 : Graphische Darstellung der Klassenhierarchie von Konstruktionsgruppen, Funktionsgruppen und Funktionsuntergruppen mit STT.

Betrachtet man zusätzlich noch die Schadensartengruppen und Einzelteilgruppen, erkennt man viele Querverbindungen. Diese Querverbindungen werden häufig durch Einzelteilgruppen erzeugt. Ihre Extensionen enthalten oft relativ wenige Schadensteile, so daß sie häufig in anderen Gruppen enthalten sind. Dadurch bleibt die Baumstruktur nicht weiter erhalten, sondern man erhält einen Graphen mit mehreren Querverbindungen. In diesem Graphen kann man auch die 3 Ebene von Konstruktionsgruppen, Funktionsuntergruppen und Funktionsuntergruppen nicht mehr erkennen. Abbildung 26 enthält ein Beispiel für diese Struktur.

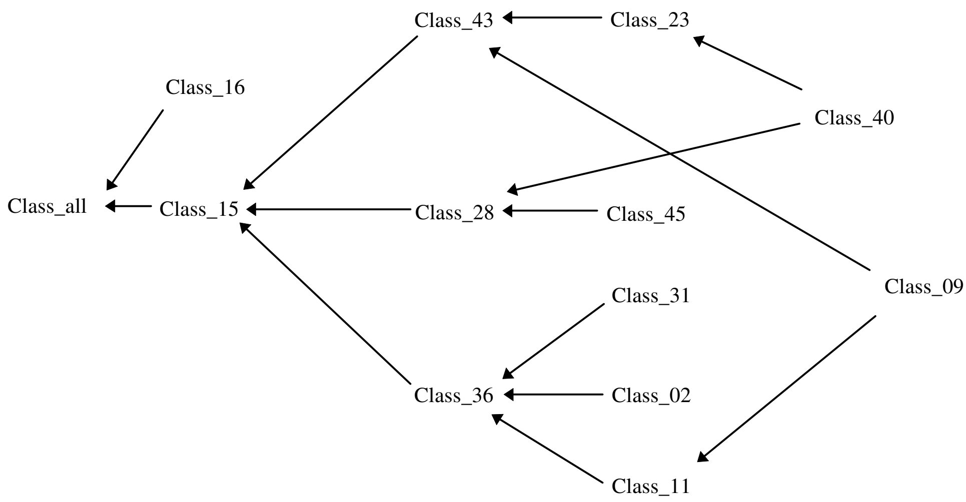


Abbildung 26 : Graphische Darstellung der Klassenhierarchie aller erzeugten Gruppen mit STT

7.4 Ergebnisse

Es ist dem Benutzers überlassen zu entscheiden, welche Fakten zur Generierung von Klassen herangezogen werden sollen. Er sollte nur eine Trennung zwischen PKW- und LKW-Fakten beibehalten. Es aber sinnvoll ist, Fakten zusammen zu betrachten und Klassen mit vielen Gruppen und Schadensteilen zu bilden, da man so eine feinere Klassenhierarchie erhalten kann. Allerdings benötigt STT bei einer größeren Anzahl von Fakten und Gruppen viel Zeit und Speicher, um daraus die Klassen zu generieren. Es wurden einige Kombinationen von Gruppen und Fakten getestet, die im folgenden beschrieben werden. Es sind nur die Konfigurationsdateien für das PKW-Schadensbuch aufgeführt. Um die Konfigurationsdateien für das LKW-Schadensbuch zu erhalten, muß nur die Filterinformation in den Prädikatsdefinitionen von 'P*' nach 'L*' geändert werden. Außerdem sollte eine andere Ausgabedatei verwendet werden.

Konstruktionsgruppen (k-Fakten)
Konfigurationsdatei : pkw_kg.cfg

INPUT c:\ssl\ssl0-93
OUTPUT c:\Fakten\pkw_kg.fak
ARGUMENT_DEFINITION 2 1 5 t *
PREDICATE_DEFINITION 1 1 2 3 1 P* k

Funktionsgruppen (f-Fakten)
Konfigurationsdatei : pkw_fg.cfg

INPUT c:\ssl\ssl0-93

OUTPUT c:\Fakten\pkw_fg.fak
ARGUMENT_DEFINITION 2 1 5 t *
PREDICATE_DEFINITION 1 1 2 5 1 P* f

Funktionsuntergruppen (p-Fakten)
Konfigurationsdatei : pkw_fug.cfg

INPUT c:\ssl\ssl0-93
OUTPUT c:\Fakten\pkw_fug.fak
ARGUMENT_DEFINITION 2 1 5 t *
PREDICATE_DEFINITION 1 1 2 8 1 P* p

Schadensartengruppen (s-Fakten)
Konfigurationsdatei : pkw_art.cfg

INPUT c:\ssl\ssl0-93
OUTPUT c:\Fakten\pkw_art.fak
ARGUMENT_DEFINITION 2 1 5 t *
PREDICATE_DEFINITION 3 100 1 2 1 P* s

Konstruktionsgruppen & Funktionsgruppen & Funktionsuntergruppen (k-, f-, p-Fakten)
Konfigurationsdatei : pkw_kfu.cfg

INPUT c:\ssl\ssl0-93
OUTPUT c:\Fakten\pkw_kfu.fak
ARGUMENT_DEFINITION 2 1 5 t *
PREDICATE_DEFINITION 1 1 2 3 1 P* k
PREDICATE_DEFINITION 1 1 2 5 1 P* f
PREDICATE_DEFINITION 1 1 2 8 1 P* p

Konstruktionsgruppen & Funktionsgruppen & Funktionsuntergruppen & Schadensartengruppen (k-, f-, p-, s- Fakten)
Konfigurationsdatei : pkw_all.cfg

INPUT c:\ssl\ssl0-93
OUTPUT c:\Fakten\pkw_all.fak
ARGUMENT_DEFINITION 2 1 5 t *
PREDICATE_DEFINITION 1 1 2 3 1 P* k
PREDICATE_DEFINITION 1 1 2 5 1 P* f
PREDICATE_DEFINITION 1 1 2 8 1 P* p
PREDICATE_DEFINITION 3 100 1 2 1 P* s

Einzelteilgruppen (etg-Fakten)
Konfigurationsdatei : etg_auw.cfg

INPUT c:\ssl\etg_ausw
OUTPUT c:\Fakten\pkw_etg.fak
ARGUMENT_DEFINITION 2 1 5 t *

PREDICATE_DEFINITION 1 1 2 6 1 * g

Im folgenden sind die Laufzeiten und Anzahl der erzeugten Fakten und Klassen aufgeführt. Die Faktenarten wurden mit den hier für das PKW-Schadensbuch aufgeführten Konfigurationsdateien erstellt. Die Konfigurationsdateien der LKW-Fakten ergeben sich durch eine Veränderung des Filters.
Die Datei SSL0-93 besteht aus insgesamt 15703, die Datei ETG_AUSW aus insgesamt 1320 Zeilen.

PKW-Schadensbuch

Faktenart	Laufzeit DOCTOFAC	Anzahl Fakten	Laufzeit STT	Anzahl Klassen
k-Fakten	0:17 min	3916	ca. 1 Stunden	47
f-Fakten	0:18 min	4327	ca. 5 Stunden	166
p-Fakten	0:20 min	5118	ca. 7 Stunden	308
s-Fakten	1:15 min	7973	ca. 3 Stunden	79
kfp-Fakten	1:49 min	13361	ca. 12 Stunden	418
kfps-Fakten	6:58 min	21334	ca. 24 Stunden	497

LKW-Schadensbuch

Faktenart	Laufzeit DOCTOFAC	Anzahl Fakten	Laufzeit STT	Anzahl Klassen
k-Fakten	0:31 min	6286	-	49
f-Fakten	0:33 min	6710	-	170
p-Fakten	0:36 min	7440	-	338
s-Fakten	2:03 min	12180	-	78
kfp-Fakten	3:39 min	20436	ca. 24 Stunden	457
kfps-Fakten	9:37 min	32616	ca. 50 Stunden	535

Einzelteilgruppen

Faktenart	Laufzeit DOCTOFAC	Anzahl Fakten	Laufzeit STT	Anzahl Klassen
etg-Fakten	0:05 min	1320	8 min	292

PKW-Fakten und Einzelteilgruppen

Faktenart	**Laufzeit DOCTOFAC**	**Anzahl Fakten**	**Laufzeit STT**	**Anzahl Klassen**
kfpsFakten und etg-Fakten	Fakten wurden getrennt generiert	22654	ca. 75 Stunden	

Diese Laufzeiten wurden auf einer Sun Sparc Station 20 mit 64 Megabyte Hauptspeicher durchgeführt. Vergleicht man die Laufzeiten von DOCTOFAC und STT wird deutlich, daß DOCTOFAC nur einen Bruchteil der Laufzeiten von STT einnimmt.
Während DOCTOFAC mit wenig Speicher ausgekommen ist, traten bei STT Speicherprobleme auf, wenn mehr als 14000 Fakten eingelesen und STT direkt aufgerufen wurde. Diese Probleme konnte man umgehen, indem die Domäne zuerst gespeichert und anschließend neu geladen wurde. Die Laufzeiten von STT sind nicht genau, da nicht automatisch festgehalten werden konnte, wann ein Lauf von STT beendet war. Aufgrund der langen Dauer eines Laufs konnte so nur eine ungefäre Laufzeit ermittelt werden.

8 Überführen der generierten Klassen in ein Datenbankformat

Wie oben beschrieben, sollen die generierten Klassen als Hintergrundwissen beim Lernen aus Datenbanken eingesetzt werden. (vgl. Abschnitt1.2). Die Klassen müssen nur noch aus MOBAL ausgelesen und ein passendes Format transferiert werden. Da MOBAL in Prolog realisiert ist, können die Klassen unter Verwendung einer Pologschnittstelle in eine Datei ausgelesen werden. Mit Hilfe von MOBTODB wird die so erzeugte Datei dann in ein passendes Format transferiert.

8.1 Auslesen der generierten Klassen

Um die generierten Klassen nun zum Lernen von Regeln, in diesem Fall mit RDT/DB, verwenden zu können, müssen sie aus MOBAL ausgelesen werden. MOBAL bietet keine direkte Möglichkeit, die generierten Klassen in eine Textdatei auszugeben, in der sowohl die Klassennamen, die enthaltenen Argumentsorten (in diesem Fall die Gruppennamen) und die Extensionen enthalten sind. Allerdings bietet MOBAL eine Schnittstelle zu Prolog, um auf interne Daten zugreifen zu können. Ein kleines Prologprogamm kann nun die interne Datenstruktur in eine Datei umleiten, so daß die generierten Klassen von externen Programmen verwendet werden können. Grundlage dieses Programms ist das interne Prädikat get_class von STT. Ein Aufruf von get_class (Classname,Argsort,Extension) liefert ein Tupel, dessen Komponenten folgende Bedeutung haben :

Classname
Classname bezeichnet den Namen der Klasse. Dieser Klassenname wird von STT automatisch vergeben.

Argsort
Argsort bezeichnet eine Liste von Sortennamen der Argmentsorten, die in der Klasse enthalten sind. Wichtig ist, daß die Liste durch eckige Klammer '[' und ']' dargestellt wird.

Extension
Bezeichnet die Extension der Klasse. Da es sich hierbei ebenfalls um eine Liste handelt, werden die Elemente wiederum in eckige Klammern '[' und ']' eingeschlossen.

Ein Aufruf von get_class liefert beispielsweise folgendes Tupel :

(class_29,[arg_1(k60)],[t32135,t63116,t65028])

Führt man nun das folgende Prologprogramm 'get_mobal_class' aus, werden alle Klassen in eine Textdatei geschrieben. Der Name der Textdatei wird durch F angegeben.

get_mobal_class :

```
write_class_to_file(F) :-
        tell(F) ,
        ((stt:get_class(X,Y,Z) , write((X,Y,Z)),nl,fail) ; true),
        close(F),
        tell(user).
```

Abbildung 27 : get_mobal_class zum Auslesen von Klassen aus STT

Das Ergebnis ist eine Textdatei, die alle Klassen einschließlich Argumentsorten und Extension enthält. Jede Zeile enthält genau ein Klasse, die wiederum als Tupel in der oben beschiebenen Form dargestellt ist. Dieses Programm kann dem System leicht über die Prologschnittstelle von MOBAL hinzugefügt und ausgeführt werden. Die Anwendung von get_mobal_class wird in Kapitel 9 anhand eines Beispieldurchlaufs beschrieben.

8.2 Umformen der genierten Klassen mit MOBTODB

Die generierten Klassen liegen nun zwar als Textdatei vor, jedoch bereitet die vorhandene prologähnliche Schreibweise Probleme, wenn man die Klassen in der Tabelle einer Datenbank ablegen will, es sei denn, man möchte die Listen innerhalb eines Tupel, in einem Feld darstellen. Das Programm MOBTODB bietet nun die Möglichkeit, die Schreibweise so umzuformen, daß die Klassen in eine Datenbank übernommen werden können. Dabei ist es im allgemeinen üblich, beim Datenimport in Datenbanken zwischen Zeichenketten und Zahlen zu unterscheiden, wobei Zeichenketten in Anführungszeichen eingeschlossen werden. Diese Unterscheidungsmöglichkeit bietet MOBTODB ebenfalls, so daß dieser Schritt nicht später in der Datenbank ausgeführt werden muß.
Es gibt nun eine Reihe von Darstellungsmöglichkeiten der Klassen in einer Datenbank, von denen man viele durch Umformungen mit Datenbankfunktionen erzeugen kann. So kann man beispielsweise die Feldreihenfolge oder das Feldformat ändern, oder man kann Felder zusammenfassen. Das erfordert aber immer einen mehr oder weniger großen Aufwand des Benutzers. Mit Hilfe von MOBTODB kann man aber nun mit relativ geringem Aufwand die prologähnliche Schreibweise in eine bestimmte neue Schreibweise überführen. Die hier verwendete Schreibweise soll an folgendem Beispiel erläutert werden.
Die Eingabedatei von MOBTODB besteht aus mehreren Zeile, die alle so aufgebaut sind wie die folgende Zeile. Aus dieser Zeile sind die nachfolgenden Zeilen entstanden, die die Zugehörigkeit der Gruppen ‘k60’ und ‘k62’ zur Klasse ‘class_29’ enthält, sowie die Zugehörigkeit der Schadensteile zu den Gruppen und damit zur Klasse.

class_29,[arg_1(k60),arg_1(k62)],[t32135,t63116,t65028]

```
class_29,k60,t32135
class_29,k60,t63116
class_29,k60,t65028
class_29,k62,t32135
class_29,k62,t63116
class_29,k62,t65028
```

Das Bilden dieser neuen Zeilen ist im wesentlichen der Schritt, den MOBTODB ausführt. Aufgrund der speziellen Anforderung zum Lernen mit RDT/DB, müssen diese Zeilen noch geringfügig abgeändert werden. Die Datenbank der Daimler-Benz AG, die zum lernen verwendet wird, enthält eine Aufspaltung des Schadensteilschlüssel, so daß diese Form auch hier verlangt wird. Diese Änderung ist kann mit Hilfe der Konfigurationsdatei von MOBTODB festgelegt werden. Die obigen Zeilen haben in der geforderten Form folgendes Aussehen.

32,135,'k60', 'class_29'
63,116,'k60', 'class_29'
65,028,'k60', 'class_29'
32,135,'k62', 'class_29'
63,116,'k62', 'class_29'
65,028,'k62' ,'class_29'

8.2.1 Die Eingabedatei von MOBTODB

Die Eingabedatei von MOBTODB entspricht gerade der Textdatei, die das Prologprogramm get_mobal_class erzeugt. Erwartet werden in jeder Zeile 3 Komponenten, die jeweils durch Komma getrennt sind. Entsprechen die Zeilen der Eingabedatei nicht den oben beschriebenen Anforderungen, wird die Ausführung von MOBTODB mit einer Fehlermeldung abgebrochen.

8.2.2 Die Konfigurationsdatei von MOBTODB

Der Aufbau der Konfigurationsdateien von MOBTODB und DOCTOFAC sind sehr ähnlich. Sie enthalten beide zwei Befehle für die Namen der Eingabe- und Ausgabedatei, sowie einen Teil, der beschreibt, wie die Ausgabedatei aus der Eingabedatei gebildet wird. Bei der Konfigurationsdatei von MOBTODB handelt es sich wiederum um eine reine Textdatei, die in jeder Zeile einen Befehl und seine Argumente enthält.
Weiter Befehle oder Argumente werden ignoriert. Der Befehl und die Argumente werden jeweils durch Leerzeichen getrennt. Die Befehle haben folgende Syntax :

INPUT <Arg_1>
OUTPUT <Arg_1>
ARGUMENT <Arg_1> <Arg_2> <Arg_3>
PREDICATE_NAME <Arg_1>
MOBAL_CLASS <Arg_1>

Anhand dieser Befehle wird nun festgelegt, welche Datei als Eingabedatei gelesen wird, in welcher Reihenfolge und in welcher Form der Klassenname, der Name eines enthaltenen Prädikats und die Extension der Klasse in die Ausgabedatei geschrieben wird. Die Bedeutung der Befehle und ihrer Argumente wird im folgenden detailliert dargestellt.

INPUT <Arg_1>

Mit diesem Befehl wird die Datei angegeben, die als Eingabedatei gelesen werden soll. Arg_1 enthält den vollständigen Pfad zu dieser Datei einschließlich des Dateinamens. Sollte die Datei nicht geöffnet werden können, wird ein Laufzeitfehler erzeugt und das Programm abgebrochen. Der Aufbau des Pfades und des Dateinamens ist abhängig von dem System, auf dem MOBTODB ausgeführt wird

OUTPUT <Arg$_1$>

Mit diesem Befehl wird die Datei angegeben, in die die erzeugte Ausgabe geschrieben werden soll. Arg$_1$ enthält den vollständigen Pfad zu dieser Datei einschließlich des Dateinamens.

ARGUMENT <Arg$_1$> <Arg$_2$> <Arg$_3$>

Dieser Befehl beschreibt, welcher Teil eines Elements einer Extension ausgeben wird. Zusätzlich kann noch angegeben werden, ob die Ausgabe als Zeichenkette oder als Zahl erfolgen soll. Die Argumente haben folgende Bedeutung :

<Arg$_1$> : Typ : ‘NUMBER’ oder ‘STRING’

Arg$_1$ gibt an, ob die Ausgabe als Zahl oder als Zeichenkette erfolgen soll. Dabei führt MOBTODB keinerlei Umformung durch, sondern setzt die Ausgabe bei Zeichenketten in Anführungszeichen, während bei Zahlen die Ausgabe ohne Anführungszeichen geschrieben wird. Diese Unterscheidung ist für Importfunktionen von Datenbanken von Bedeutung. Es wird auch nicht geprüft, ob eine Darstellung einer Zeichenkette als Zahl sinnvoll ist. Diese Aufgabe kommt der jeweiligen Importfunktion der verwendeten Datenbank zu.

<Arg$_2$> : Typ : positive ganze Zahl

Durch Arg$_2$ und Arg$_3$ wird ein Intervall über den Schadensteilschlüsseln der Extensionen beschrieben, wobei diese als Zeichenketten gesehen werden. Arg$_2$ gibt dabei die unter Grenze dieses Intervalls an.

<Arg$_3$> : Typ : positive ganze Zahl

Arg$_3$ gibt dabei die obere Grenze des Intervalls an, daß von Arg$_2$ und Arg$_3$ gebildet wird. Wird Arg$_3$ größer gewählt, als die tatsächlich vorhandene Anzahl von Zeichen, werden nur wirklich vorhandene Zeichen berücksichtigt.

PREDICATE_NAME <Arg$_1$>

Dieser Befehl beschreibt einen Sortennamen der aktuell bearbeiten Klasse. Außerdem wird festgelegt, ob die Schreibweise als Zeichenkette oder als Zahl verwendet werden soll.

<Arg$_1$> : Typ : ‘NUMBER’ oder ‘STRING’

Arg$_1$ gibt an, ob die Ausgabe als Zahl oder als Zeichenkette erfolgen soll. Dabei führt MOBTODB keinerlei Umformung durch, sondern setzt die Ausgabe bei Zeichenketten in Anführungszeichen, während bei Zahlen die Ausgabe ohne Anführungszeichen geschrieben wird.

MOBAL_CLASS <Arg$_1$>

Dieser Befehl beschreibt den Namen der aktuell bearbeiten Klasse. Außerdem wird festgelegt, ob die Schreibweise als Zeichenkette oder als Zahl verwendet werden soll.

<Arg_1> : Typ : ‘NUMBER’ oder ‘STRING’

Arg_1 gibt an, ob die Ausgabe als Zahl oder als Zeichenkette erfolgen soll.

8.2.3 Die Ausgabedatei von MOBTODB

Bei der Ausgabedatei von MOBTODB handelt es sich um eine Textdatei, die man mit Hilfe der Importfunktion von Datenbanksystemen leicht in eine Datenbank konvertieren kann. Je nachdem in welcher Reihenfolge die Befehle der Konfigurationsdatei verwendet wurden, werden die Felder in der Ausgabedatei angeordnet und somit später in die Datenbank übernommen. Alle Befehle außer ‘INPUT’ und ‘OUTPUT’ dürfen häufiger verwendet werden, und erzeugen somit mehrere Felder. Sollte keiner dieser Befehle verwendet werden, so daß kein Feld definiert wird, bricht das Programm mit einer Fehlermeldung ab.

8.2.4 Die Arbeitsweise von MOBTODB

Im folgenden soll Arbeitsweise von MOBTODB an einem Beispiel dargestellt werden. Für eine bestimmte Anforderung wird eine Konfigurationsdatei erstellt, und anschließend die Bearbeitung einer kleinen Eingabedatei unter Berücksichtigung der Konfigurationsdatei dargestellt. Folgende Eingabedatei soll bearbeitet werden:

	S_1	S_2	S_3
Z_1	class_31,	[arg_1(g00070),arg_1(g00073)]	[t07041,t07042]
Z_2	class_32,	[arg_1(g00069)],	[t07023,t07024,t07027]

Tabelle 3 : Beispiel für Eingabedatei von MOBTODB

Für oben beschriebene Anforderung an die Ausgabe eignet sich folgende Konfigurationsdatei. Wichtig ist die Reihenfolge der Befehle ‘ARGUMENT’, ‘PREDICATE_NAME’, ‘MOBAL_CLASS’, da die Ausgabe pro Zeile in dieser Reihenfolge erzeugt wird.

```
INPUT c:\class\class_bsp
OUTPUT c:\classdb\classdb_bsp
ARGUMENT NUMBER 2 3
ARGUMENT NUMBER 4 6
PREDICATE_NAME STRING
MOBAL_CLASS STRING
```

Der Start von MOBTODB erfolgt durch den Aufruf

mobtodb <configfile>

Bei der Ausführung wird zuerst die Konfigurationsdatei gelesen und auf eventuelle Fehler überprüft. Wurde die Eingabedatei oder die Ausgabedatei nicht definiert, oder konnte diese nicht geöffnet werden, bricht das Programm mit einer Fehlermeldung ab.

Anschließend wird die Struktur der Konfigurationsdatei in den Arbeitsspeicher geladen und dabei auf eventuelle Fehler überprüft. Danach beginnt die Bearbeitung der Eingabedatei mit der Zeile Z_1. Nachdem diese Zeile gelesen wurde, werden drei Variablenbindungen vorgenommen. Dabei enthält ‘Klasse’ den Namen der aktuell bearbeiteten Klasse, ‘Prädikat’ enthält den Namen der ersten zugeordneten Gruppe und ‘Schadensteil’ das erste Element der Extension.

Klasse : = ‘class_31’
Prädikat : = ‘g00070’
Argument : = ‘t07041’

Der erste Befehl, der verarbeitet wird, ist ‘ARGUMENT NUMBER 2 3’. Er bezieht auf die Zeichen 2 bis 3 des Arguments, also auf ‘07’. Darstellt werden soll diese ‘07’ als Zahl, und wird damit ohne Anführungszeichen in Ausgabe geschrieben. Danach wird der zweite Befehl verarbeitet. Er erzeugt die Zeichenkette ‘041’ die ebenfalls in der Datenbank als Zahl dargestellt werden soll, und somit auch nicht in Anführungszeichen eingeschlossen wird. Sie wird ebenfalls in die Ausgabe geschrieben, getrennt durch ein Komma. Die Bearbeitung der letzten beiden Befehle ist sehr einfach, da sie sich auf den gesamten Inhalt einer Variablen beziehen. Hier wurde auf die Möglichkeit verzichtet den Namen der Klasse oder der Gruppe aufzuspalten, da dies die von SST generierte Struktur auftrennen würde. Durch die Argumente beider Befehle wird bestimmt, daß die Ausgabe in Anführungszeichen eingeschlossen wird. Sie werden in der Reihenfolge ihres Definition in der Konfigurationsdatei, getrennt durch Komma, an die Ausgabe geschrieben. Die Ausgabedatei enthält dann folgende Zeile :

‘ 07,041,’g00030’,’class_31’ ‘

Da nun der letzte Befehl der Konfigurationsdatei abgearbeitet wurde, wird der Variablen ‘Argument’ das nächste Schadensteil zugewiesen und eine neue Ausgabezeile gebildet, die dann wiederum in die Ausgabedatei geschrieben wird.
Da nun alle Schadensteile abgearbeitet wurden, erfolgt die Bearbeitung der nächsten Gruppe ‘g00073’, wobei wiederum alle Schadensteile abgearbeitet werden. Sind auch alle Gruppen bearbeitet, ist die Bearbeitung der gesamten Klasse abgeschlossen. Die nächste Zeile der Eingabedatei wird eingelesen und die Ausgabezeilen werden erzeugt. Für das Beispiel werden insgesamt folgende Zeilen erzeugt:

07,041,’g00070’,class_31
07,042,’g00070’,class_31
07,041,’g00073’,class_31
07,042,’g00073’,class_31
07,023,’g00069’,class_32
07,024,’g00069’,class_32
07,027,’g00069’,class_32

8.3 Realisierung von MOBTODB

Ebenso wie bei DOCTOFAC handelt es sich auch bei MOBTODB um ein in der Programmiersprache C++ geschriebenes Programm, das sich leicht auf verschiedene Systeme übertragen läßt. Übersetzt wurde es mit auf den selben Compilern wie DOCTOFAC, also Borland C++ Version 2.0 für MS-DOS, Microsoft Visual C++ Version

4.0 für WINDOWS NT, SUN C++ Version 3.0.1 für UNIX. Der Programmablauf von MOBTODB läßt sich in folgende Schritte aufteilen:

- **Einlesen der Konfigurationsdatei**
  Die Konfigurationsdatei wird zeilenweise gelesen und auf Fehler überprüft. Konnte kein Fehler gefunden werden, wird der Inhalt der Zeile in einer Datenstruktur abgelegt. Diese enthält die Namen der Ein- und Ausgabedatei, und eine dynamische Liste der Definitionen der Konfigurationsdatei.

- **Lesen der Eingabedatei und schreiben der Ausgabedatei**
  Die Eingabedatei wird zeilenweise gelesen, und die Ausgabezeilen werden nach dem oben beschriebenen Schema gebildet. Dabei wird die Liste, die die Definitionen der Konfigurationsdatei enthält, zur Erzeugung jeder Ausgabezeile erneut durchlaufen, und die erzeugt Zeile wird direkt, das heißt ohne Zwischenspeicherung, in die Ausgabedatei übertragen. Da jede Zeile der Eingabedatei einer von STT erzeugten Klasse entspricht, werden die gelesenen Zeilen während der Umformung mitgezählt und nach Ausführung des Programm am Bildschirm angezeigt. Die einzige Ausnahme bilden die Klassen NIL und ALL die nicht mitgezählt werden, da MOBTODB aus diesen Klassen keine Ausgabe erzeugt.

Einschränkungen

- Die Anzahl von Zeichen einer Zeile der Konfigurationsdatei ist auf 1000 beschränkt.
- Die Anzahl von Zeichen einer Zeile der Eingabedatei ist auf 10000 beschränkt
- Die Anzahl von Zeichen der Namen der Eingabedatei und der Ausgabedatei sind auf 200 beschränkt.

Fehlermeldungen

Während MOBTODB die Konfigurationsdatei einliest, wird sie auf mögliche Fehler überprüft. Wird ein Fehler entdeckt, bricht das Programm mit einer Fehlermeldung ab. Die möglichen Fehlermeldungen sind im folgenden aufgeführt. Sie sind im wesentlichen selbsterklärend.

```
Unknown Syntax
No configfile
Incorrect range for argument definition
Incorrect arguments
No inputfile
No outputfile
Empty outputfile declared
Not enough memory
Cannot open file for input: ",filename
Cannot open file for output: ",filename
Line too long
Argument to long
```

9 Beispieldurchlauf mit dem PKW-Schadensbuch

In diesem Kapitel soll an einem Beispiel der Weg der Faktenerzeugung aus dem PKW-Schadensbuch, der Generierung von Fakten mit STT, des Auslesens der Klassen aus MOBAL und der Umformung mit Hilfe von MOBTODB verdeutlicht werden. Wollte man diesen Weg vollständig, d.h. mit allen erzeugten Daten des PKW-Schadensbuchs darstellen, würde dies allein den Umfang eines Buches einnehmen. Aus diesem Grund soll nur ein Ausschnitt des Schadensbuches dargestellt werden, obwohl das Ergebnis weniger aussagekräftig ist. Das Ziel dieses Beispiels liegt darin, den Weg zu beschreiben, und nicht in einem aussagekräftigen Ergebnis.
Ausgangspunkt bilden dabei die Dateien SSL0-93 und SSLTEIL, aus der Einzelteilgruppen gebildet werden können. Alle generierten Einzelteilgruppen sind im Anhang vollständig aufgeführt. Zu beachten ist, daß nicht alle Bauteile bei der Bildung der Gruppen berücksichtigt wurden. Der besondere Augenmerk lag auf den Schadensteilen, die aufgrund ihres Schadensteiltextes in Verbindung mit der Funktion des Motors gebracht werden konnten. Grundlage dafür war die Beschreibung der Konstruktionsgruppen, denen diese Schadensteile aufgrund der ersten beiden Ziffern ihrer Schlüssel zugeordnet werden können. Bei der Bildung der Einzelteilgruppen wurden nur Schadensteile folgender Konstruktionsgruppen berücksichtigt : 00, 01, 03, 05, 07, 09, 13, 14, 15, 18, 20, 23, 24, 47, 49, 50, 51, 52, 53. Die gebildeten Einzelteilgruppen liegen in der Datei ETG_AUSW in der oben beschriebenen Form vor.

9.1 Generierung von Fakten mit DOCTOFAC

Ausgangspunkt zur Generierung von Fakten bilden in diesem Beispiel die Dateien SSL0-93 und ETG_AUSW. Für jede Datei muß eine Konfigurationsdatei erstellt werden, die beschreibt, wie Fakten erstellt werden sollen. Zur Reduzierung der Faktenmenge sollen in diesem Beispiel nur Fakten mit Schadensteilen generiert werden, deren Schlüssel mit ‘07’ beginnen. Die folgende Konfigurationsdatei ‘bsp07_ssl.cfg’ liest die Datei SSL0-93 und schreibt die erzeugten Fakten in die Datei ‘bsp07_ssl.fak’.

bsp07_ssl.cfg

```
INPUT c:\ssl\ssl0-93
OUTPUT c:\Fakten\bsp07_ssl.fak
ARGUMENT_DEFINITION 2 1 5 t 07*
PREDICATE_DEFINITION 1 1 2 3 1 P* k
PREDICATE_DEFINITION 1 1 2 5 1 P* f
PREDICATE_DEFINITION 1 1 2 8 1 P* p
PREDICATE_DEFINITION 3 100 1 2 1 P* s
```

Der Aufruf erfolgt mit ‘doctofac bsp07_ssl.cfg’. Während der Ausführung wird angezeigt, wie viele Fakten bereits generiert wurden. Nach der Ausführung, die ca. 6 Sekunden dauert, werden folgende Informationen am Bildschirm angezeigt:

```
INSGESAMT GELESENE ZEILEN : 15703

ERZEUGTE FAKTEN MIT KENNZEICHEN k : 149
ERZEUGTE FAKTEN MIT KENNZEICHEN f : 201
ERZEUGTE FAKTEN MIT KENNZEICHEN p : 210
ERZEUGTE FAKTEN MIT KENNZEICHEN s : 266

INSGESAMT ERZEUGTE FAKTEN : 826
```

Beim zweiten Durchlauf von DOCTOFAC werden aus den Einzelteilgruppen der Datei ETG_AUSW Fakten generiert. Hierzu dient die Konfigurationsdatei ‘bsp07_etg.cfg’, die ebenfalls nur Fakten berücksichtigt, deren Schlüssel mit ‘07’ beginnen. Das Ergebnis wird in die Datei ‘bsp07_etg.fak’ geschrieben.

bsp07_etg.cfg

INPUT c:\ssl\etg_ausw
OUTPUT c:\Fakten\bsp07_etg.fak
ARGUMENT_DEFINITION 2 1 5 t 07*
PREDICATE_DEFINITION 1 1 2 6 1 * g

Der Aufruf erfolgt mit ‘doctofac bsp07_etg.cfg’. Das Ergebnis wird in die Datei ‘bsp07_etg.fak’ geschrieben. Nach der Ausführung, die ca. 2 Sekunde dauert, werden folgende Informationen angezeigt :

```
INSGESAMT GELESENE ZEILEN : 1321

ERZEUGTE FAKTEN MIT KENNZEICHEN g : 137

INSGESAMT ERZEUGTE FAKTEN : 137
```

Die generierten Fakten sind im Anhang vollständig aufgeführt.

9.2 Generierung von Klassen mit STT

Im nächsten Schritt werden die Klassen mit STT generiert. Dazu müssen die erzeugten Fakten in MOBAL eingelesen werden. Dies geschieht mit dem Befehl ‘Read from Text’ im Menue ‘Datei’ der MOBAL-Oberfläche. Der Vorgang des Einlesens der beiden Dateien ‘bsp07_ssl.fak’ und ‘bsp07_etg.fak’ dauert ca. 10 Minuten.
Zur Generierung der Klassen wird der Befehl ‘Update Sorts’ im Menue ‘Call’ aufgerufen, was STT dazu veranlaßt, die Klassen zu generieren. Dabei ist darauf zu achten, daß die Generierung von Schnittsorten (intersection sorts) nicht eingeschaltet ist, was aber auch standardmäßig der Fall ist. Nach ca. 7 Minuten hat STT die Klassen gebildet. Es ist nun möglich, sich den Graphen der generierten Klassen anzeigen zu lassen. Aber selbst bei der eingeschränkten Anzahl von Fakten in diesem Beispiel ist der Graph noch zu groß, um ihn an dieser Stelle vollständig darstellen zu können.

9.3 Auslesen der generierten Klassen und Bearbeitung mit MOBTODB

Zum Auslesen der generierten Klassen dient das Prologprogramm 'get_mobal_class'. Hierzu wird der 'Prolog Listener' im Menue 'Call' aufgerufen. Durch den Aufruf von '[get_mobal_class].' steht das Prädikat 'write_class_to_file' zur Verfügung.
Die erzeugten Klassen sollen nun in Datei 'class_bsp07' geschrieben werden. Dies geschieht durch einen erneuten Aufruf von 'Prolog Listener' mit 'write_class_to_file('class_bsp07')'.
Im letzten Schritt wird die Datei 'class_bsp07' mit MOBTODB bearbeitet. Die benötigten Informationen enthält die folgende Konfigurationsdatei 'bsp07_mob.cfg'. Sie arbeitet wie die in Kapitel 8 beschriebene Konfigurationsdatei, liest und schreibt aber andere Dateien.

bsp07_mob.cfg

INPUT c:\class\class_bsp07
OUTPUT c:\classdb\classdb_bsp07
ARGUMENT NUMBER 2 3
ARGUMENT NUMBER 4 7
PREDICATE_NAME STRING
MOBAL_CLASS STRING

Der Aufruf erfolgt mit 'mobtodb bsp07_mob.cfg'. Das Ergebnis wird in die Datei 'classdb_bsp07' gespeichert. Während der Ausführung der Programms wird angezeigt, welche Klasse gerade bearbeitet wird. Nach der Ausführung, die ca. 1 Sekunde dauert, wird die Anzahl der bearbeiteten Klassen angezeigt, wobei die Klassen 'class_all' und 'class_nil' nicht berücksichtigt werden. In diesem Beispiel werden folgende Informationen angezeigt :

```
Anzahl bearbeiteter Klassen : 95
```

An dieser Stelle ist der Beispieldurchlauf beendet, da die generierten Klassen, in dem geforderten Format in die Ausgabedatei 'classdb_bsp07' geschrieben wurden.
Abbildung 28 zeigt einen Ausschnitt der graphischen Darstellung der Klassenhierarchie, die mit STT erzeugt wurde.

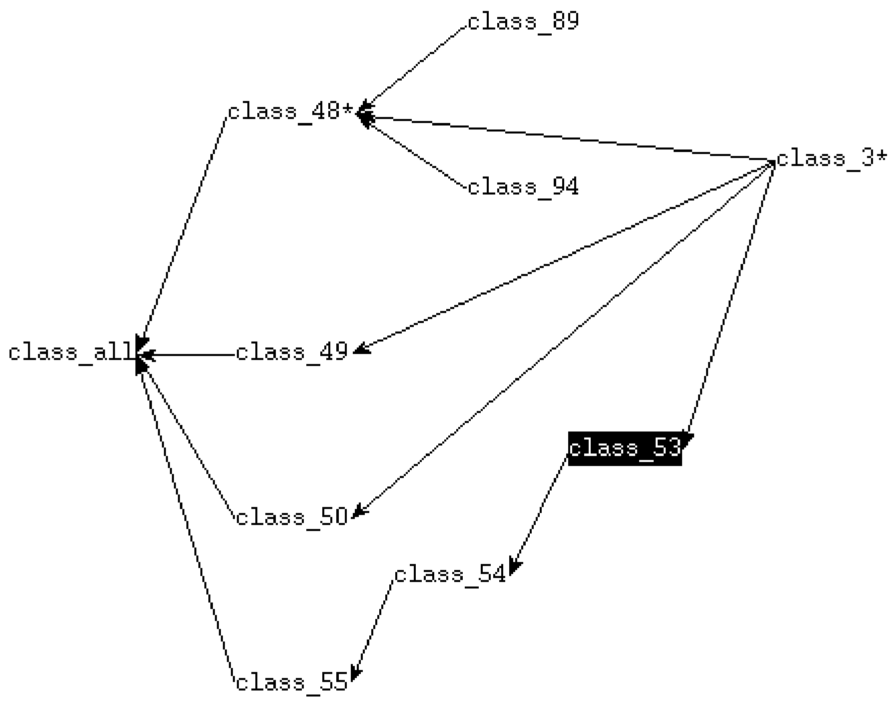


Abbildung 28 : Ausschnitt der graphischen Darstellung des Beispiels

Im folgenden sind die in diesem Beispiel generierten Dateien vollständig aufgeführt, so daß sich die Bearbeitung nachvollziehen läßt.

9.4 Generierte Dateien des Beispieldurchlaufs

bsp07_ssl.fak

```
k07(t07001).
f0711(t07001).
p0711401(t07001).
s02(t07001).
s04(t07001).
s07(t07001).
s36(t07001).
s53(t07001).
s60(t07001).
k07(t07004).
f0710(t07004).
p0710301(t07004).
s24(t07004).
k07(t07006).
f0711(t07006).
p0711401(t07006).
s04(t07006).
s36(t07006).
f0713(t07006).
p0713101(t07006).
k07(t07008).
f0711(t07008).
p0711401(t07008).
s04(t07008).
f0713(t07008).
p0713101(t07008).
k07(t07009).
f0711(t07009).
p0711401(t07009).
s06(t07009).
f0713(t07009).
p0713101(t07009).
k07(t07013).
f0711(t07013).
p0711401(t07013).
s16(t07013).
k07(t07018).
f0710(t07018).
p0710402(t07018).
s04(t07018).
s16(t07018).
k07(t07026).
f0711(t07026).
p0711401(t07026).
s04(t07026).
s52(t07026).
s53(t07026).
k07(t07029).
f0711(t07029).
p0711401(t07029).
s16(t07029).
f0713(t07029).
p0713101(t07029).
k07(t07030).
f0710(t07030).
p0710402(t07030).
s04(t07030).
s73(t07030).
k07(t07039).
f0710(t07039).
p0710402(t07039).
s04(t07039).
s73(t07039).
k07(t07040).
f0710(t07040).
p0710402(t07040).
s06(t07040).
s38(t07040).
s47(t07040).
s73(t07040).
k14(t07040).
f1420(t07040).
p1420303(t07040).
k15(t07040).
f1510(t07040).
p1510301(t07040).
k54(t07040).
f5415(t07040).
p5415501(t07040).
f5465(t07040).
p5465101(t07040).
k91(t07040).
f9140(t07040).
p9140501(t07040).
k07(t07046).
f0711(t07046).
p0711401(t07046).
s04(t07046).
k07(t07047).
f0711(t07047).
p0711401(t07047).
s04(t07047).
k07(t07048).
f0710(t07048).
p0710402(t07048).
s04(t07048).
k07(t07051).
f0732(t07051).
p0732202(t07051).
s04(t07051).
s47(t07051).
k07(t07065).
f0711(t07065).
p0711401(t07065).
s47(t07065).
s73(t07065).
f0710(t07065).
p0710402(t07065).
k14(t07065).
f1420(t07065).
p1420303(t07065).
k07(t07106).
f0711(t07106).
p0711401(t07106).
s38(t07106).
f0713(t07106).
p0713101(t07106).
k07(t07111).
f0711(t07111).
p0711401(t07111).
s06(t07111).
s08(t07111).
s16(t07111).
s36(t07111).
s38(t07111).
k07(t07113).
f0713(t07113).
p0713101(t07113).
s06(t07113).
s08(t07113).
k07(t07115).
f0711(t07115).
p0711401(t07115).
s08(t07115).
k07(t07151).
f0713(t07151).
p0713101(t07151).
s04(t07151).
s07(t07151).
s36(t07151).
s53(t07151).
s60(t07151).
k07(t07152).
f0711(t07152).
p0711401(t07152).
s04(t07152).
s16(t07152).
s73(t07152).
f0713(t07152).
p0713101(t07152).
f0710(t07152).
p0710402(t07152).
k07(t07153).
f0713(t07153).
p0713101(t07153).
s04(t07153).
s73(t07153).
k07(t07154).
f0713(t07154).
p0713101(t07154).
s04(t07154).
s73(t07154).
k07(t07155).
f0713(t07155).
p0713101(t07155).
s04(t07155).
s73(t07155).
k07(t07156).
f0713(t07156).
p0713101(t07156).
s04(t07156).
s73(t07156).
k07(t07157).
f0713(t07157).
p0713101(t07157).
s04(t07157).
s52(t07157).
k07(t07158).
f0713(t07158).
p0713101(t07158).
s04(t07158).
s73(t07158).
k07(t07159).
f0713(t07159).
p0713101(t07159).
s73(t07159).
k07(t07160).
f0713(t07160).
p0713101(t07160).
s73(t07160).
k07(t07161).
f0711(t07161).
p0711401(t07161).
s58(t07161).
s74(t07161).
s92(t07161).
f0713(t07161).
p0713101(t07161).
f0710(t07161).
p0710402(t07161).
k07(t07162).
f0711(t07162).
p0711401(t07162).
s04(t07162).
s16(t07162).
s73(t07162).
k07(t07201).
f0711(t07201).
p0711401(t07201).
s04(t07201).
s16(t07201).
s51(t07201).
s93(t07201).
f0732(t07201).
p0732202(t07201).
k07(t07202).
f0711(t07202).
p0711401(t07202).
s04(t07202).
k07(t07205).
f0732(t07205).
p0732202(t07205).
s06(t07205).
k07(t07209).
f0710(t07209).
p0710301(t07209).
s06(t07209).
s38(t07209).
k47(t07209).
f4720(t07209).
p4720201(t07209).
p4720302(t07209).
k07(t07213).
f0710(t07213).
p0710402(t07213).
s04(t07213).
s38(t07213).
s87(t07213).
f0722(t07213).
p0722201(t07213).
f0741(t07213).
p0741201(t07213).
f0750(t07213).
p0750101(t07213).
k14(t07213).
f1430(t07213).
p1430601(t07213).
k15(t07213).
f1510(t07213).
p1510302(t07213).
k07(t07214).
f0732(t07214).
p0732202(t07214).
s03(t07214).
s04(t07214).
s87(t07214).
f0741(t07214).
p0741201(t07214).
f0750(t07214).
p0750101(t07214).
k47(t07214).
f4720(t07214).
p4720201(t07214).
p4720302(t07214).
k07(t07215).
f0741(t07215).
p0741201(t07215).
s04(t07215).
s06(t07215).
s22(t07215).
s87(t07215).
f0750(t07215).
p0750101(t07215).
k47(t07215).
f4720(t07215).
p4720201(t07215).
p4720302(t07215).
k07(t07216).
f0710(t07216).
p0710301(t07216).
```

s04(t07216).
k07(t07217).
f0710(t07217).
p0710301(t07217).
s03(t07217).
s04(t07217).
f0741(t07217).
p0741201(t07217).
f0750(t07217).
p0750101(t07217).
k47(t07217).
f4720(t07217).
p4720201(t07217).
p4720302(t07217).
k07(t07219).
f0710(t07219).
p0710301(t07219).
s04(t07219).
k07(t07227).
f0710(t07227).
p0710301(t07227).
s03(t07227).
s04(t07227).
s36(t07227).
f0732(t07227).
p0732202(t07227).
f0741(t07227).
p0741201(t07227).
f0750(t07227).
p0750101(t07227).
k47(t07227).
f4720(t07227).
p4720201(t07227).
p4720302(t07227).
k47(t07228).
f4720(t07228).
p4720302(t07228).
s04(t07228).
s06(t07228).
s38(t07228).
k47(t07229).
f4720(t07229).
p4720302(t07229).
s04(t07229).
s06(t07229).
s38(t07229).
k07(t07230).
f0732(t07230).
p0732202(t07230).
s51(t07230).
k07(t07231).
f0732(t07231).
p0732503(t07231).
s47(t07231).
s51(t07231).
s73(t07231).
k47(t07232).
f4720(t07232).
p4720302(t07232).
s47(t07232).
k07(t07233).
f0732(t07233).
p0732101(t07233).
s04(t07233).
s15(t07233).
s47(t07233).
s51(t07233).
k07(t07234).
f0732(t07234).
p0732101(t07234).
s15(t07234).
s19(t07234).
s38(t07234).
s51(t07234).
k07(t07236).
f0732(t07236).
p0732101(t07236).
s04(t07236).
k07(t07237).
f0732(t07237).
p0732101(t07237).
s04(t07237).
k07(t07238).
f0732(t07238).
p0732101(t07238).
s47(t07238).
s52(t07238).
s73(t07238).
k07(t07240).
f0732(t07240).
p0732503(t07240).
s04(t07240).
k07(t07241).
f0732(t07241).
p0732503(t07241).
s15(t07241).
s73(t07241).
k47(t07245).
f4720(t07245).
p4720302(t07245).
s04(t07245).
s38(t07245).
k07(t07246).
f0711(t07246).
p0711401(t07246).
s07(t07246).
k07(t07248).
f0711(t07248).
p0711401(t07248).
s06(t07248).
f0732(t07248).
p0732202(t07248).
k07(t07249).
f0711(t07249).
p0711401(t07249).
s04(t07249).
k07(t07261).
f0732(t07261).
p0732101(t07261).
s15(t07261).
s51(t07261).
s73(t07261).
k07(t07262).
f0732(t07262).
p0732101(t07262).
s04(t07262).
s16(t07262).
s36(t07262).
f0741(t07262).
p0741201(t07262).
f0750(t07262).
p0750101(t07262).
k07(t07263).
f0710(t07263).
p0710301(t07263).
s04(t07263).
k07(t07264).
f0710(t07264).
p0710301(t07264).
s03(t07264).
s04(t07264).
k47(t07265).
f4720(t07265).
p4720201(t07265).
s04(t07265).
k07(t07266).
f0732(t07266).
p0732101(t07266).
s03(t07266).
s06(t07266).
p0732202(t07266).
k07(t07267).
f0732(t07267).
p0732202(t07267).
s04(t07267).
k07(t07268).
f0732(t07268).
p0732101(t07268).
s99(t07268).
k07(t07269).
f0732(t07269).
p0732101(t07269).
s07(t07269).
k07(t07270).
f0710(t07270).
p0710301(t07270).
s04(t07270).
k07(t07271).
f0732(t07271).
p0732101(t07271).
s47(t07271).
k07(t07272).
f0732(t07272).
p0732503(t07272).
s04(t07272).
k07(t07273).
f0732(t07273).
p0732503(t07273).
s07(t07273).
k07(t07280).
f0710(t07280).
p0710301(t07280).
s04(t07280).
k07(t07281).
f0710(t07281).
p0710301(t07281).
s04(t07281).
s16(t07281).
k47(t07281).
f4720(t07281).
p4720201(t07281).
k07(t07282).
f0710(t07282).
p0710301(t07282).
s04(t07282).
k47(t07282).
f4720(t07282).
p4720201(t07282).
k07(t07283).
f0711(t07283).
p0711401(t07283).
s03(t07283).
s06(t07283).
s36(t07283).
f0732(t07283).
p0732202(t07283).
k07(t07284).
f0711(t07284).
p0711401(t07284).
s03(t07284).
s06(t07284).
s36(t07284).
f0732(t07284).
p0732202(t07284).
k07(t07285).
f0711(t07285).
p0711401(t07285).
s03(t07285).
s06(t07285).
s36(t07285).
f0732(t07285).
p0732202(t07285).
k07(t07286).
f0711(t07286).
p0711401(t07286).
s03(t07286).
s06(t07286).
s36(t07286).
f0732(t07286).
p0732202(t07286).
k07(t07287).
f0711(t07287).
p0711401(t07287).
s03(t07287).
s06(t07287).
s36(t07287).
f0732(t07287).
p0732202(t07287).
k07(t07288).
f0711(t07288).
p0711401(t07288).
s03(t07288).
s06(t07288).
s36(t07288).
f0732(t07288).
p0732202(t07288).
k07(t07289).
f0732(t07289).
p0732202(t07289).
s03(t07289).
s06(t07289).
s36(t07289).
k07(t07290).
f0732(t07290).
p0732202(t07290).
s03(t07290).
s06(t07290).
s36(t07290).
k07(t07291).
f0741(t07291).
p0741201(t07291).
s04(t07291).
s15(t07291).
s58(t07291).
s73(t07291).
f0750(t07291).
p0750101(t07291).
k07(t07292).
f0741(t07292).
p0741201(t07292).
s06(t07292).
s38(t07292).
f0750(t07292).
p0750101(t07292).
k07(t07293).
f0741(t07293).
p0741201(t07293).
s04(t07293).
s06(t07293).
s38(t07293).
f0750(t07293).
p0750101(t07293).
k07(t07294).
f0741(t07294).
p0741201(t07294).
s87(t07294).
f0750(t07294).
p0750101(t07294).
k07(t07295).
f0741(t07295).
p0741201(t07295).
s04(t07295).
f0750(t07295).
p0750101(t07295).
k07(t07301).
f0732(t07301).
p0732101(t07301).
s16(t07301).
k07(t07302).
f0732(t07302).
p0732101(t07302).
s04(t07302).
f0741(t07302).
p0741201(t07302).
f0750(t07302).
p0750101(t07302).
k07(t07311).
f0732(t07311).
p0732101(t07311).
s47(t07311).
s73(t07311).
k15(t07311).
f1510(t07311).
p1510301(t07311).
k07(t07312).
f0732(t07312).
p0732503(t07312).
s04(t07312).
k07(t07314).
f0732(t07314).
p0732503(t07314).
s04(t07314).
k07(t07315).
f0710(t07315).
p0710402(t07315).
s04(t07315).
s52(t07315).
s73(t07315).
f0722(t07315).
p0722201(t07315).
f0732(t07315).
p0732503(t07315).
f0750(t07315).
p0750102(t07315).
k14(t07315).
f1420(t07315).
p1420303(t07315).
k15(t07315).
f1510(t07315).
p1510301(t07315).
f1520(t07315).
p1520201(t07315).
k27(t07315).
f2718(t07315).
p2718101(t07315).
k14(t07316).
f1420(t07316).
p1420301(t07316).
s52(t07316).
s73(t07316).
k07(t07320).
f0732(t07320).
p0732503(t07320).
s04(t07320).
k07(t07415).
f0722(t07415).
p0722201(t07415).
s06(t07415).
f0732(t07415).
p0732101(t07415).
k30(t07415).
f3010(t07415).
p3010203(t07415).
k14(t07437).
f1430(t07437).
p1430601(t07437).
s99(t07437).
k07(t07501).
f0711(t07501).
p0711401(t07501).
s44(t07501).
s69(t07501).
f0722(t07501).
p0722201(t07501).
f0732(t07501).
p0732101(t07501).
k30(t07502).
f3010(t07502).
p3010202(t07502).
s06(t07502).
s08(t07502).
s16(t07502).
s36(t07502).
s51(t07502).
p3010203(t07502).
k30(t07503).
f3010(t07503).
p3010202(t07503).
s15(t07503).
s16(t07503).
s36(t07503).
p3010203(t07503).
k30(t07504).
f3010(t07504).
p3010202(t07504).
s06(t07504).
s38(t07504).
s47(t07504).
p3010203(t07504).
k07(t07505).
f0711(t07505).
p0711401(t07505).
s60(t07505).
s61(t07505).
s62(t07505).
s67(t07505).
f0722(t07505).
p0722201(t07505).
f0732(t07505).
p0732101(t07505).
k07(t07506).
f0711(t07506).
p0711401(t07506).
s53(t07506).
s60(t07506).
s62(t07506).
s67(t07506).
f0722(t07506).
p0722201(t07506).
f0732(t07506).
p0732101(t07506).
k07(t07507).
f0711(t07507).
p0711401(t07507).
s04(t07507).
s16(t07507).
s53(t07507).
k30(t07507).
f3010(t07507).
p3010202(t07507).
k07(t07513).
f0711(t07513).
p0711401(t07513).
s06(t07513).
k30(t07513).
f3010(t07513).
p3010202(t07513).
k30(t07514).
f3020(t07514).
p3020201(t07514).
s06(t07514).
k30(t07515).
f3020(t07515).
p3020201(t07515).
s16(t07515).
s51(t07515).
k30(t07516).
f3010(t07516).
p3010203(t07516).
s06(t07516).
k30(t07519).
f3010(t07519).
p3010202(t07519).
s06(t07519).
s08(t07519).
s16(t07519).
s38(t07519).
k07(t07601).
f0722(t07601).
p0722201(t07601).
s47(t07601).
k07(t07602).
f0722(t07602).
p0722201(t07602).
s16(t07602).
s52(t07602).
k07(t07603).
f0722(t07603).
p0722201(t07603).
s04(t07603).
s16(t07603).
s38(t07603).
k07(t07604).
f0722(t07604).
p0722201(t07604).
s04(t07604).
s16(t07604).
k07(t07605).
f0722(t07605).
p0722201(t07605).
s24(t07605).
k07(t07606).
f0722(t07606).
p0722201(t07606).
s04(t07606).
k07(t07607).
f0722(t07607).
p0722201(t07607).
s38(t07607).
k07(t07608).
f0722(t07608).
p0722201(t07608).
s04(t07608).
s47(t07608).
s73(t07608).
k07(t07611).
f0722(t07611).
p0722201(t07611).
s16(t07611).
s47(t07611).
s73(t07611).
k07(t07612).
f0722(t07612).
p0722201(t07612).
s47(t07612).
s73(t07612).
k07(t07613).
f0722(t07613).
p0722201(t07613).
s08(t07613).
k07(t07621).
f0722(t07621).
p0722201(t07621).
s04(t07621).
s38(t07621).
k07(t07622).
f0722(t07622).
p0722201(t07622).
s04(t07622).
s73(t07622).
k07(t07623).
f0722(t07623).
p0722201(t07623).
s04(t07623).
s06(t07623).
s38(t07623).
k07(t07624).
f0722(t07624).
p0722201(t07624).
s04(t07624).
s07(t07624).
s38(t07624).
k07(t07631).
f0722(t07631).
p0722201(t07631).
s38(t07631).
s73(t07631).
k07(t07632).
f0722(t07632).
p0722201(t07632).
s47(t07632).
s73(t07632).
f0732(t07632).
p0732503(t07632).
f0750(t07632).
p0750102(t07632).

bsp07_etg.fak

```
00065(t07001).
g00065(t07002).
g00065(t07004).
g00065(t07006).
g00065(t07007).
g00065(t07008).
g00065(t07009).
g00065(t07010).
g00065(t07011).
g00065(t07012).
g00065(t07013).
g00065(t07014).
g00065(t07015).
g00065(t07016).
g00065(t07018).
g00065(t07020).
g00065(t07023).
g00065(t07024).
g00065(t07027).
g00065(t07041).
g00065(t07042).
g00065(t07044).
g00065(t07046).
g00065(t07065).
g00065(t07161).
g00065(t07162).
g00065(t07513).
g00066(t07006).
g00066(t07007).
g00067(t07011).
g00067(t07012).
g00067(t07013).
g00067(t07014).
g00067(t07015).
g00067(t07016).
g00068(t07015).
g00068(t07016).
g00069(t07023).
g00069(t07024).
g00069(t07027).
g00070(t07041).
g00070(t07042).
g00071(t07102).
g00071(t07104).
g00072(t07111).
g00072(t07115).
g00073(t07122).
g00073(t07123).
g00074(t07151).
g00074(t07155).
g00074(t07156).
g00074(t07157).
g00074(t07158).
g00074(t07160).
g00075(t07201).
g00075(t07205).
g00076(t07202).
g00076(t07203).
g00076(t07263).
g00077(t07233).
g00077(t07236).
g00078(t07234).
g00078(t07238).
g00079(t07230).
g00079(t07269).
g00080(t07209).
g00080(t07215).
g00080(t07217).
g00080(t07228).
g00080(t07245).
g00080(t07247).
g00080(t07264).
g00080(t07266).
g00080(t07270).
g00081(t07275).
g00081(t07278).
g00081(t07279).
g00081(t07280).
g00081(t07281).
g00081(t07282).
g00082(t07291).
g00082(t07292).
g00083(t07293).
g00083(t07294).
g00083(t07295).
g00084(t07301).
g00084(t07302).
g00085(t07312).
g00085(t07320).
g00086(t07406).
g00086(t07407).
g00087(t07410).
g00087(t07411).
g00088(t07434).
g00088(t07435).
g00089(t07501).
g00089(t07505).
g00089(t07506).
g00090(t07503).
g00090(t07504).
g00091(t07511).
g00091(t07512).
g00092(t07517).
g00092(t07518).
g00093(t07602).
g00093(t07606).
g00093(t07607).
g00094(t07603).
g00094(t07604).
g00095(t07612).
g00095(t07613).
g00216(t07051).
g00216(t07056).
g00216(t07057).
g00216(t07058).
g00216(t07059).
g00216(t07060).
g00216(t07061).
g00216(t07062).
g00216(t07064).
g00218(t07113).
g00218(t07114).
g00218(t07121).
g00219(t07502).
g00219(t07519).
g00350(t07246).
g00350(t07248).
g00350(t07283).
g00350(t07284).
g00350(t07285).
g00350(t07286).
g00350(t07287).
g00350(t07288).
g00350(t07289).
g00350(t07290).
g00366(t07401).
g00366(t07624).
```

class_bsp07

class_nil,[],[]
class_all,[],[...]
class_1,[arg_1(g00366)],[t07401,t07624]
class_2,[arg_1(g00350)],[t07246,t07248,t07283,t07284,t07285,t07286,t07287,t07288,t07289,t07290]
class_3,[arg_1(g00219)],[t07502,t07519]
class_4,[arg_1(g00218)],[t07113,t07114,t07121]
class_5,[arg_1(g00216)],[t07051,t07056,t07057,t07058,t07059,t07060,t07061,t07062,t07064]
class_6,[arg_1(g00095)],[t07612,t07613]
class_7,[arg_1(g00094)],[t07603,t07604]
class_8,[arg_1(g00093)],[t07602,t07606,t07607]
class_9,[arg_1(g00092)],[t07517,t07518]
class_10,[arg_1(g00091)],[t07511,t07512]
class_11,[arg_1(g00090)],[t07503,t07504]
class_12,[arg_1(g00089)],[t07501,t07505,t07506]
class_13,[arg_1(g00088)],[t07434,t07435]
class_14,[arg_1(g00087)],[t07410,t07411]
class_15,[arg_1(g00086)],[t07406,t07407]
class_16,[arg_1(g00085)],[t07312,t07320]
class_17,[arg_1(g00084)],[t07301,t07302]
class_18,[arg_1(g00083)],[t07293,t07294,t07295]
class_19,[arg_1(g00082)],[t07291,t07292]
class_20,[arg_1(g00081)],[t07275,t07278,t07279,t07280,t07281,t07282]
class_21,[arg_1(g00080)],[t07209,t07215,t07217,t07228,t07245,t07247,t07264,t07266,t07270]
class_22,[arg_1(g00079)],[t07230,t07269]
class_23,[arg_1(g00078)],[t07234,t07238]
class_24,[arg_1(g00077)],[t07233,t07236]
class_25,[arg_1(g00076)],[t07202,t07203,t07263]
class_26,[arg_1(g00075)],[t07201,t07205]
class_27,[arg_1(g00074)],[t07151,t07155,t07156,t07157,t07158,t07160]
class_28,[arg_1(g00073)],[t07122,t07123]
class_29,[arg_1(g00072)],[t07111,t07115]
class_30,[arg_1(g00071)],[t07102,t07104]
class_31,[arg_1(g00070)],[t07041,t07042]
class_32,[arg_1(g00069)],[t07023,t07024,t07027]
class_33,[arg_1(g00068)],[t07015,t07016]
class_34,[arg_1(g00067)],[t07011,t07012,t07013,t07014,t07015,t07016]
class_35,[arg_1(g00066)],[t07006,t07007]
class_36,[arg_1(g00065)],[t07001,t07002,t07004,t07006,t07007,t07008,t07009,t07010,t07011,t07012,t07013,t07014,t07015,t07016,t07018,t07020,t07023,t07024,t07027,t07041,t07042,t07044,t07046,t07065,t07161,t07162,t07513]
class_37,[arg_1(p0750102)],[t07315,t07632]
class_38,[arg_1(f0750)],[t07213,t07214,t07215,t07217,t07227,t07262,t07291,t07292,t07293,t07294,t07295,t07302,t07315,t07632]
class_39,[arg_1(p0732503)],[t07231,t07240,t07241,t07272,t07273,t07312,t07314,t07315,t07320,t07632]
class_40,[arg_1(f0732)],[t07051,t07201,t07205,t07214,t07227,t07230,t07231,t07233,t07234,t07236,t07237,t07238,t07240,t07241,t07248,t07261,t07262,t07266,t07267,t07268,t07269,t07271,t07272,t07273,t07283,t07284,t07285,t07286,t07287,t07288,t07289,t07290,t07301,t07302,t07311,t07312,t07314,t07315,t07320,t07415,t07501,t07505,t07506,t07632]
class_41,[arg_1(s73)],[t07030,t07039,t07040,t07065,t07152,t07153,t07154,t07155,t07156,t07158,t07159,t07160,t07162,t07231,t07238,t07241,t07261,t07291,t07311,t07315,t07316,t07608,t07611,t07612,t07622,t07631,t07632]
class_42,[arg_1(s47)],[t07040,t07051,t07065,t07231,t07232,t07233,t07238,t07271,t07311,t07504,t07601,t07608,t07611,t07612,t07632]
class_43,[arg_1(f0722),arg_1(p0722201)],[t07213,t07315,t07415,t07501,t07505,t07506,t07601,t07602,t07603,t07604,t07605,t07606,t07607,t07608,t07611,t07612,t07613,t07621,t07622,t07623,t07624,t07631,t07632]
class_44,[arg_1(k07)],[t07001,t07004,t07006,t07008,t07009,t07013,t07018,t07026,t07029,t07030,t07039,t07040,t07046,t07047,t07048,t07051,t07065,t07106,t07111,t07113,t07115,t07151,t07152,t07153,t07154,t07155,t07156,t07157,t07158,t07159,t07160,t07161,t07162,t07201,t07202,t07205,t07209,t07213,t07214,t07215,t07216,t07217,t07219,t07227,t07230,t07231,t07233,t07234,t07236,t07237,t07238,t07240,t07241,t07246,t07248,t07249,t07261,t07262,t07263,t07264,t07266,t07267,t07268,t07269,t07270,t07271,t07272,t07273,t07280,t07281,t07282,t07283,t07284,t07285,t07286,t07287,t07288,t07289,t07290,t07291,t07292,t07293,t07294,t07295,t07301,t07302,t07311,t07312,t07314,t07315,t07320,t07415,t07501,t07505,t07506,t07507,t07513,t07601,t07602,t07603,t07604,t07605,t07606,t07607,t07608,t07611,t07612,t07613,t07621,t07622,t07623,t07624,t07631,t07632]
class_45,[arg_1(s38)],[t07040,t07106,t07111,t07209,t07213,t07228,t07229,t07234,t07245,t07292,t07293,t07504,t07519,t07603,t07607,t07621,t07623,t07624,t07631]
class_46,[arg_1(s07)],[t07001,t07151,t07246,t07269,t07273,t07624]
class_47,[arg_1(s04)],[t07001,t07006,t07008,t07018,t07026,t07030,t07039,t07046,t07047,t07048,t07051,t07151,t07152,t07153,t07154,t07155,t07156,t07157,t07158,t07162,t07201,t07202,t07213,t07214,t07215,t07216,t07217,t07219,t07227,t07228,t07229,t07233,t07236,t07237,t07240,t07245,t07249,t07262,t07263,t07264,t07265,t07267,t07270,t07272,t07280,t07281,t07282,t07291,t07293,t07295,t07302,t07312,t07314,t07315,t07320,t07507,t07603,t07604,t07606,t07608,t07621,t07622,t07623,t07624]
class_48,[arg_1(s06)],[t07009,t07040,t07111,t07113,t07205,t07209,t07215,t07228,t07229,t07248,t07266,t07283,t07284,t07285,t07286,t07287,t07288,t07289,t07290,t07292,t07293,t07415,t07502,t07504,t07513,t07514,t07516,t07519,t07623]
class_49,[arg_1(s08)],[t07111,t07113,t07115,t07502,t07519,t07613]
class_50,[arg_1(s16)],[t07013,t07018,t07029,t07111,t07152,t07162,t07201,t07262,t07281,t07301,t07502,t07503,t07507,t07515,t07519,t07602,t07603,t07604,t07611]
class_51,[arg_1(s24)],[t07004,t07605]
class_52,[arg_1(s52)],[t07026,t07157,t07238,t07315,t07316,t07602]
class_53,[arg_1(p3010202)],[t07502,t07503,t07504,t07507,t07513,t07519]
class_54,[arg_1(f3010)],[t07415,t07502,t07503,t07504,t07507,t07513,t07516,t07519]
class_55,[arg_1(k30)],[t07415,t07502,t07503,t07504,t07507,t07513,t07514,t07515,t07516,t07519]
class_56,[arg_1(p3010203)],[t07415,t07502,t07503,t07504,t07516]
class_57,[arg_1(s51)],[t07201,t07230,t07231,t07233,t07234,t07261,t07502,t07515]
class_58,[arg_1(f3020),arg_1(p3020201)],[t07514,t07515]
class_59,[arg_1(f0711),arg_1(p0711401)],[t07001,t07006,t07008,t07009,t07013,t07026,t07029,t07046,t07047,t07065,t07106,t07111,t07115,t07152,t07161,t07162,t07201,t07202,t07246,t07248,t07249,t07283,t07284,t07285,t07286,t07287,t07288,t07501,t07505,t07506,t07507,t07513]
class_60,[arg_1(s53)],[t07001,t07026,t07151,t07506,t07507]
class_61,[arg_1(p0732101)],[t07233,t07234,t07236,t07237,t07238,t07261,t07262,t07266,t07268,t07269,t07271,t07301,t07302,t07311,t07415,t07501,t07505,t07506]
class_62,[arg_1(s62),arg_1(s67)],[t07505,t07506]
class_63,[arg_1(s60)],[t07001,t07151,t07505,t07506]
class_64,[arg_1(s61)],[t07505]
class_65,[arg_1(s36)],[t07001,t07006,t07111,t07151,t07227,t07262,t07283,t07284,t07285,t07286,t07287,t07288,t07289,t07290,t07502,t07503]
class_66,[arg_1(s15)],[t07233,t07234,t07241,t07261,t07291,t07503]
class_67,[arg_1(s44),arg_1(s69)],[t07501]
class_68,[arg_1(s99)],[t07268,t07437]
class_69,[arg_1(f1430),arg_1(p1430601)],[t07213,t07437]
class_70,[arg_1(k14)],[t07040,t07065,t07213,t07315,t07316,t07437]
class_71,[arg_1(p1420301)],[t07316]
class_72,[arg_1(f1420)],[t07040,t07065,t07315,t07316]
class_73,[arg_1(f1520),arg_1(f2718),arg_1(k27),arg_1(p1520201),arg_1(p2718101)],[t07315]
class_74,[arg_1(p1510301)],[t07040,t07311,t07315]
class_75,[arg_1(f1510),arg_1(k15)],[t07040,t07213,t07311,t07315]
class_76,[arg_1(p1420303)],[t07040,t07065,t07315]
class_77,[arg_1(p0710402)],[t07018,t07030,t07039,t07040,t07048,t07065,t07152,t07161,t07213,t07315]

class_78,[arg_1(f0710)],[t07004,t07018,t07030,t07039,t07040,t07048,t07065,t07152,t07161,t07209,t07213,t07216,t07217,t07219,t07227,t07263,t07264,t07270,t07280,t07281,t0
7282,t07315]
class_79,[arg_1(f0741),arg_1(p0741201),arg_1(p0750101)],[t07213,t07214,t07215,t07217,t07227,t07262,t07291,t07292,t07293,t07294,t07295,t07302]
class_80,[arg_1(s87)],[t07213,t07214,t07215,t07294]
class_81,[arg_1(s58)],[t07161,t07291]
class_82,[arg_1(s03)],[t07214,t07217,t07227,t07264,t07266,t07283,t07284,t07285,t07286,t07287,t07288,t07289,t07290]
class_83,[arg_1(p0732202)],[t07051,t07201,t07205,t07214,t07227,t07230,t07248,t07266,t07267,t07283,t07284,t07285,t07286,t07287,t07288,t07289,t07290]
class_84,[arg_1(p4720201)],[t07209,t07214,t07215,t07217,t07227,t07265,t07281,t07282]
class_85,[arg_1(f4720),arg_1(k47)],[t07209,t07214,t07215,t07217,t07227,t07228,t07229,t07232,t07245,t07265,t07281,t07282]
class_86,[arg_1(p0710301)],[t07004,t07209,t07216,t07217,t07219,t07227,t07263,t07264,t07270,t07280,t07281,t07282]
class_87,[arg_1(p4720302)],[t07209,t07214,t07215,t07217,t07227,t07228,t07229,t07232,t07245]
class_88,[arg_1(s19)],[t07234]
class_89,[arg_1(s22)],[t07215]
class_90,[arg_1(p1510302)],[t07213]
class_91,[arg_1(s93)],[t07201]
class_92,[arg_1(f0713),arg_1(p0713101)],[t07006,t07008,t07009,t07029,t07106,t07113,t07151,t07152,t07153,t07154,t07155,t07156,t07157,t07158,t07159,t07160,t07161]
class_93,[arg_1(s74),arg_1(s92)],[t07161]
class_94,[arg_1(f5415),arg_1(f5465),arg_1(f9140),arg_1(k54),arg_1(k91),arg_1(p5415501),arg_1(p5465101),arg_1(p9140501)],[t07040]
class_95,[arg_1(s02)],[t07001]

classdb_bsp07

07,401,'g00366','class_1'
07,624,'g00366','class_1'
07,246,'g00350','class_2'
07,248,'g00350','class_2'
07,283,'g00350','class_2'
07,284,'g00350','class_2'
07,285,'g00350','class_2'
07,286,'g00350','class_2'
07,287,'g00350','class_2'
07,288,'g00350','class_2'
07,289,'g00350','class_2'
07,290,'g00350','class_2'
07,502,'g00219','class_3'
07,519,'g00219','class_3'
07,113,'g00218','class_4'
07,114,'g00218','class_4'
07,121,'g00218','class_4'
07,051,'g00216','class_5'
07,056,'g00216','class_5'
07,057,'g00216','class_5'
07,058,'g00216','class_5'
07,059,'g00216','class_5'
07,060,'g00216','class_5'
07,061,'g00216','class_5'
07,062,'g00216','class_5'
07,064,'g00216','class_5'
07,612,'g00095','class_6'
07,613,'g00095','class_6'
07,603,'g00094','class_7'
07,604,'g00094','class_7'
07,602,'g00093','class_8'
07,606,'g00093','class_8'
07,607,'g00093','class_8'
07,517,'g00092','class_9'
07,518,'g00092','class_9'
07,511,'g00091','class_10'
07,512,'g00091','class_10'
07,503,'g00090','class_11'
07,504,'g00090','class_11'
07,501,'g00089','class_12'
07,505,'g00089','class_12'
07,506,'g00089','class_12'
07,434,'g00088','class_13'
07,435,'g00088','class_13'
07,410,'g00087','class_14'
07,411,'g00087','class_14'
07,406,'g00086','class_15'
07,407,'g00086','class_15'
07,312,'g00085','class_16'
07,320,'g00085','class_16'
07,301,'g00084','class_17'
07,302,'g00084','class_17'
07,293,'g00083','class_18'
07,294,'g00083','class_18'
07,295,'g00083','class_18'
07,291,'g00082','class_19'
07,292,'g00082','class_19'
07,275,'g00081','class_20'
07,278,'g00081','class_20'
07,279,'g00081','class_20'
07,280,'g00081','class_20'
07,281,'g00081','class_20'
07,282,'g00081','class_20'
07,209,'g00080','class_21'
07,215,'g00080','class_21'
07,217,'g00080','class_21'
07,228,'g00080','class_21'
07,245,'g00080','class_21'
07,247,'g00080','class_21'
07,264,'g00080','class_21'
07,266,'g00080','class_21'
07,270,'g00080','class_21'
07,230,'g00079','class_22'
07,269,'g00079','class_22'
07,234,'g00078','class_23'
07,238,'g00078','class_23'
07,233,'g00077','class_24'
07,236,'g00077','class_24'
07,202,'g00076','class_25'
07,203,'g00076','class_25'
07,263,'g00076','class_25'
07,201,'g00075','class_26'
07,205,'g00075','class_26'
07,151,'g00074','class_27'
07,155,'g00074','class_27'
07,156,'g00074','class_27'
07,157,'g00074','class_27'
07,158,'g00074','class_27'
07,160,'g00074','class_27'
07,122,'g00073','class_28'
07,123,'g00073','class_28'
07,111,'g00072','class_29'
07,115,'g00072','class_29'
07,102,'g00071','class_30'
07,104,'g00071','class_30'
07,041,'g00070','class_31'
07,042,'g00070','class_31'
07,023,'g00069','class_32'
07,024,'g00069','class_32'
07,027,'g00069','class_32'
07,015,'g00068','class_33'
07,016,'g00068','class_33'
07,011,'g00067','class_34'
07,012,'g00067','class_34'
07,013,'g00067','class_34'
07,014,'g00067','class_34'
07,015,'g00067','class_34'
07,016,'g00067','class_34'
07,006,'g00066','class_35'
07,007,'g00066','class_35'
07,001,'g00065','class_36'
07,002,'g00065','class_36'
07,004,'g00065','class_36'
07,006,'g00065','class_36'
07,007,'g00065','class_36'
07,008,'g00065','class_36'
07,009,'g00065','class_36'
07,010,'g00065','class_36'
07,011,'g00065','class_36'
07,012,'g00065','class_36'
07,013,'g00065','class_36'
07,014,'g00065','class_36'
07,015,'g00065','class_36'
07,016,'g00065','class_36'
07,018,'g00065','class_36'
07,020,'g00065','class_36'
07,023,'g00065','class_36'
07,024,'g00065','class_36'
07,027,'g00065','class_36'
07,041,'g00065','class_36'
07,042,'g00065','class_36'
07,044,'g00065','class_36'
07,046,'g00065','class_36'
07,065,'g00065','class_36'
07,161,'g00065','class_36'
07,162,'g00065','class_36'
07,513,'g00065','class_36'
07,315,'p0750102','class_37'
07,632,'p0750102','class_37'
07,213,'f0750','class_38'
07,214,'f0750','class_38'
07,215,'f0750','class_38'
07,217,'f0750','class_38'
07,227,'f0750','class_38'
07,262,'f0750','class_38'
07,291,'f0750','class_38'
07,292,'f0750','class_38'
07,293,'f0750','class_38'
07,294,'f0750','class_38'
07,295,'f0750','class_38'
07,302,'f0750','class_38'
07,315,'f0750','class_38'
07,632,'f0750','class_38'
07,231,'p0732503','class_39'
07,240,'p0732503','class_39'
07,241,'p0732503','class_39'
07,272,'p0732503','class_39'
07,273,'p0732503','class_39'
07,312,'p0732503','class_39'
07,314,'p0732503','class_39'
07,315,'p0732503','class_39'
07,320,'p0732503','class_39'
07,632,'p0732503','class_39'
07,051,'f0732','class_40'
07,201,'f0732','class_40'
07,205,'f0732','class_40'
07,214,'f0732','class_40'
07,227,'f0732','class_40'
07,230,'f0732','class_40'
07,231,'f0732','class_40'
07,233,'f0732','class_40'
07,234,'f0732','class_40'
07,236,'f0732','class_40'
07,237,'f0732','class_40'
07,238,'f0732','class_40'
07,240,'f0732','class_40'
07,241,'f0732','class_40'
07,248,'f0732','class_40'
07,261,'f0732','class_40'
07,262,'f0732','class_40'
07,266,'f0732','class_40'
07,267,'f0732','class_40'
07,268,'f0732','class_40'
07,269,'f0732','class_40'
07,271,'f0732','class_40'
07,272,'f0732','class_40'
07,273,'f0732','class_40'
07,283,'f0732','class_40'
07,284,'f0732','class_40'
07,285,'f0732','class_40'
07,286,'f0732','class_40'
07,287,'f0732','class_40'
07,288,'f0732','class_40'
07,289,'f0732','class_40'
07,290,'f0732','class_40'
07,301,'f0732','class_40'
07,302,'f0732','class_40'
07,311,'f0732','class_40'
07,312,'f0732','class_40'
07,314,'f0732','class_40'
07,315,'f0732','class_40'
07,320,'f0732','class_40'
07,415,'f0732','class_40'
07,501,'f0732','class_40'
07,505,'f0732','class_40'
07,506,'f0732','class_40'
07,632,'f0732','class_40'
07,030,'s73','class_41'
07,039,'s73','class_41'
07,040,'s73','class_41'
07,065,'s73','class_41'
07,152,'s73','class_41'
07,153,'s73','class_41'
07,154,'s73','class_41'
07,155,'s73','class_41'
07,156,'s73','class_41'
07,158,'s73','class_41'
07,159,'s73','class_41'
07,160,'s73','class_41'
07,162,'s73','class_41'
07,231,'s73','class_41'
07,238,'s73','class_41'
07,241,'s73','class_41'
07,261,'s73','class_41'
07,291,'s73','class_41'
07,311,'s73','class_41'
07,315,'s73','class_41'
07,316,'s73','class_41'
07,608,'s73','class_41'
07,611,'s73','class_41'
07,612,'s73','class_41'
07,622,'s73','class_41'
07,631,'s73','class_41'
07,632,'s73','class_41'
07,040,'s47','class_42'
07,051,'s47','class_42'
07,065,'s47','class_42'
07,231,'s47','class_42'
07,232,'s47','class_42'
07,233,'s47','class_42'
07,238,'s47','class_42'
07,271,'s47','class_42'
07,311,'s47','class_42'
07,504,'s47','class_42'
07,601,'s47','class_42'
07,608,'s47','class_42'
07,611,'s47','class_42'
07,612,'s47','class_42'
07,632,'s47','class_42'
07,213,'f0722','class_43'
07,315,'f0722','class_43'
07,415,'f0722','class_43'
07,501,'f0722','class_43'
07,505,'f0722','class_43'
07,506,'f0722','class_43'
07,601,'f0722','class_43'
07,602,'f0722','class_43'
07,603,'f0722','class_43'
07,604,'f0722','class_43'
07,605,'f0722','class_43'
07,606,'f0722','class_43'
07,607,'f0722','class_43'
07,608,'f0722','class_43'
07,611,'f0722','class_43'
07,612,'f0722','class_43'
07,613,'f0722','class_43'
07,621,'f0722','class_43'
07,622,'f0722','class_43'
07,623,'f0722','class_43'
07,624,'f0722','class_43'
07,631,'f0722','class_43'
07,632,'f0722','class_43'
07,213,'p0722201','class_43'
07,315,'p0722201','class_43'
07,415,'p0722201','class_43'
07,501,'p0722201','class_43'
07,505,'p0722201','class_43'
07,506,'p0722201','class_43'
07,601,'p0722201','class_43'
07,602,'p0722201','class_43'
07,603,'p0722201','class_43'
07,604,'p0722201','class_43'
07,605,'p0722201','class_43'
07,606,'p0722201','class_43'
07,607,'p0722201','class_43'
07,608,'p0722201','class_43'
07,611,'p0722201','class_43'
07,612,'p0722201','class_43'
07,613,'p0722201','class_43'
07,621,'p0722201','class_43'
07,622,'p0722201','class_43'
07,623,'p0722201','class_43'
07,624,'p0722201','class_43'
07,631,'p0722201','class_43'
07,632,'p0722201','class_43'
07,001,'k07','class_44'
07,004,'k07','class_44'
07,006,'k07','class_44'
07,008,'k07','class_44'
07,009,'k07','class_44'
07,013,'k07','class_44'
07,018,'k07','class_44'
07,026,'k07','class_44'
07,029,'k07','class_44'
07,030,'k07','class_44'
07,039,'k07','class_44'
07,040,'k07','class_44'
07,046,'k07','class_44'
07,047,'k07','class_44'
07,048,'k07','class_44'
07,051,'k07','class_44'
07,065,'k07','class_44'
07,106,'k07','class_44'
07,111,'k07','class_44'
07,113,'k07','class_44'
07,115,'k07','class_44'
07,151,'k07','class_44'
07,152,'k07','class_44'
07,153,'k07','class_44'
07,154,'k07','class_44'
07,155,'k07','class_44'
07,156,'k07','class_44'
07,157,'k07','class_44'
07,158,'k07','class_44'
07,159,'k07','class_44'
07,160,'k07','class_44'
07,161,'k07','class_44'
07,162,'k07','class_44'
07,201,'k07','class_44'
07,202,'k07','class_44'
07,205,'k07','class_44'
07,209,'k07','class_44'
07,213,'k07','class_44'
07,214,'k07','class_44'
07,215,'k07','class_44'
07,216,'k07','class_44'
07,217,'k07','class_44'
07,219,'k07','class_44'
07,227,'k07','class_44'
07,230,'k07','class_44'
07,231,'k07','class_44'
07,233,'k07','class_44'
07,234,'k07','class_44'
07,236,'k07','class_44'
07,237,'k07','class_44'
07,238,'k07','class_44'
07,240,'k07','class_44'
07,241,'k07','class_44'
07,246,'k07','class_44'
07,248,'k07','class_44'
07,249,'k07','class_44'
07,261,'k07','class_44'
07,262,'k07','class_44'
07,263,'k07','class_44'
07,264,'k07','class_44'
07,266,'k07','class_44'
07,267,'k07','class_44'
07,268,'k07','class_44'
07,269,'k07','class_44'
07,270,'k07','class_44'
07,271,'k07','class_44'
07,272,'k07','class_44'
07,273,'k07','class_44'
07,280,'k07','class_44'
07,281,'k07','class_44'
07,282,'k07','class_44'
07,283,'k07','class_44'
07,284,'k07','class_44'
07,285,'k07','class_44'
07,286,'k07','class_44'
07,287,'k07','class_44'
07,288,'k07','class_44'
07,289,'k07','class_44'
07,290,'k07','class_44'
07,291,'k07','class_44'
07,292,'k07','class_44'
07,293,'k07','class_44'
07,294,'k07','class_44'
07,295,'k07','class_44'
07,301,'k07','class_44'
07,302,'k07','class_44'
07,311,'k07','class_44'
07,312,'k07','class_44'
07,314,'k07','class_44'
07,315,'k07','class_44'
07,320,'k07','class_44'
07,415,'k07','class_44'
07,501,'k07','class_44'
07,505,'k07','class_44'
07,506,'k07','class_44'
07,507,'k07','class_44'
07,513,'k07','class_44'
07,601,'k07','class_44'
07,602,'k07','class_44'
07,603,'k07','class_44'
07,604,'k07','class_44'
07,605,'k07','class_44'
07,606,'k07','class_44'
07,607,'k07','class_44'
07,608,'k07','class_44'
07,611,'k07','class_44'
07,612,'k07','class_44'
07,613,'k07','class_44'
07,621,'k07','class_44'
07,622,'k07','class_44'
07,623,'k07','class_44'
07,624,'k07','class_44'
07,631,'k07','class_44'
07,632,'k07','class_44'
07,040,'s38','class_45'
07,106,'s38','class_45'
07,111,'s38','class_45'
07,209,'s38','class_45'
07,213,'s38','class_45'
07,228,'s38','class_45'
07,229,'s38','class_45'
07,234,'s38','class_45'
07,245,'s38','class_45'
07,292,'s38','class_45'
07,293,'s38','class_45'
07,504,'s38','class_45'
07,519,'s38','class_45'
07,603,'s38','class_45'
07,607,'s38','class_45'
07,621,'s38','class_45'
07,623,'s38','class_45'
07,624,'s38','class_45'
07,631,'s38','class_45'
07,001,'s07','class_46'
07,151,'s07','class_46'
07,246,'s07','class_46'
07,269,'s07','class_46'
07,273,'s07','class_46'
07,624,'s07','class_46'
07,001,'s04','class_47'
07,006,'s04','class_47'
07,008,'s04','class_47'
07,018,'s04','class_47'
07,026,'s04','class_47'
07,030,'s04','class_47'
07,039,'s04','class_47'
07,046,'s04','class_47'
07,047,'s04','class_47'
07,048,'s04','class_47'
07,051,'s04','class_47'
07,151,'s04','class_47'
07,152,'s04','class_47'
07,153,'s04','class_47'
07,154,'s04','class_47'
07,155,'s04','class_47'
07,156,'s04','class_47'
07,157,'s04','class_47'
07,158,'s04','class_47'
07,162,'s04','class_47'
07,201,'s04','class_47'
07,202,'s04','class_47'
07,213,'s04','class_47'
07,214,'s04','class_47'
07,215,'s04','class_47'
07,216,'s04','class_47'
07,217,'s04','class_47'
07,219,'s04','class_47'
07,227,'s04','class_47'
07,228,'s04','class_47'
07,229,'s04','class_47'
07,233,'s04','class_47'
07,236,'s04','class_47'
07,237,'s04','class_47'
07,240,'s04','class_47'
07,245,'s04','class_47'

07,249,'s04','class_47'
07,262,'s04','class_47'
07,263,'s04','class_47'
07,264,'s04','class_47'
07,265,'s04','class_47'
07,267,'s04','class_47'
07,270,'s04','class_47'
07,272,'s04','class_47'
07,280,'s04','class_47'
07,281,'s04','class_47'
07,282,'s04','class_47'
07,291,'s04','class_47'
07,293,'s04','class_47'
07,295,'s04','class_47'
07,302,'s04','class_47'
07,312,'s04','class_47'
07,314,'s04','class_47'
07,315,'s04','class_47'
07,320,'s04','class_47'
07,507,'s04','class_47'
07,603,'s04','class_47'
07,604,'s04','class_47'
07,606,'s04','class_47'
07,608,'s04','class_47'
07,621,'s04','class_47'
07,622,'s04','class_47'
07,623,'s04','class_47'
07,624,'s04','class_47'
07,009,'s06','class_48'
07,040,'s06','class_48'
07,111,'s06','class_48'
07,113,'s06','class_48'
07,205,'s06','class_48'
07,209,'s06','class_48'
07,215,'s06','class_48'
07,228,'s06','class_48'
07,229,'s06','class_48'
07,248,'s06','class_48'
07,266,'s06','class_48'
07,283,'s06','class_48'
07,284,'s06','class_48'
07,285,'s06','class_48'
07,286,'s06','class_48'
07,287,'s06','class_48'
07,288,'s06','class_48'
07,289,'s06','class_48'
07,290,'s06','class_48'
07,292,'s06','class_48'
07,293,'s06','class_48'
07,415,'s06','class_48'
07,502,'s06','class_48'
07,504,'s06','class_48'
07,513,'s06','class_48'
07,514,'s06','class_48'
07,516,'s06','class_48'
07,519,'s06','class_48'
07,623,'s06','class_48'
07,111,'s08','class_49'
07,113,'s08','class_49'
07,115,'s08','class_49'
07,502,'s08','class_49'
07,519,'s08','class_49'
07,613,'s08','class_49'
07,013,'s16','class_50'
07,018,'s16','class_50'
07,029,'s16','class_50'
07,111,'s16','class_50'
07,152,'s16','class_50'
07,162,'s16','class_50'
07,201,'s16','class_50'
07,262,'s16','class_50'
07,281,'s16','class_50'
07,301,'s16','class_50'
07,502,'s16','class_50'
07,503,'s16','class_50'
07,507,'s16','class_50'
07,515,'s16','class_50'
07,519,'s16','class_50'
07,602,'s16','class_50'
07,603,'s16','class_50'
07,604,'s16','class_50'
07,611,'s16','class_50'
07,004,'s24','class_51'
07,605,'s24','class_51'
07,026,'s52','class_52'
07,157,'s52','class_52'
07,238,'s52','class_52'
07,315,'s52','class_52'
07,316,'s52','class_52'
07,602,'s52','class_52'
07,502,'p3010202','class_53'
07,503,'p3010202','class_53'
07,504,'p3010202','class_53'
07,507,'p3010202','class_53'
07,513,'p3010202','class_53'
07,519,'p3010202','class_53'
07,415,'f3010','class_54'
07,502,'f3010','class_54'
07,503,'f3010','class_54'
07,504,'f3010','class_54'
07,507,'f3010','class_54'
07,513,'f3010','class_54'
07,516,'f3010','class_54'
07,519,'f3010','class_54'
07,415,'k30','class_55'
07,502,'k30','class_55'
07,503,'k30','class_55'
07,504,'k30','class_55'
07,507,'k30','class_55'
07,513,'k30','class_55'
07,514,'k30','class_55'
07,515,'k30','class_55'
07,516,'k30','class_55'
07,519,'k30','class_55'
07,415,'p3010203','class_56'
07,502,'p3010203','class_56'
07,503,'p3010203','class_56'
07,504,'p3010203','class_56'
07,516,'p3010203','class_56'
07,201,'s51','class_57'
07,230,'s51','class_57'
07,231,'s51','class_57'
07,233,'s51','class_57'
07,234,'s51','class_57'
07,261,'s51','class_57'
07,502,'s51','class_57'
07,515,'s51','class_57'
07,514,'f3020','class_58'
07,515,'f3020','class_58'
07,514,'p3020201','class_58'
07,515,'p3020201','class_58'
07,001,'f0711','class_59'
07,006,'f0711','class_59'
07,008,'f0711','class_59'
07,009,'f0711','class_59'
07,013,'f0711','class_59'
07,026,'f0711','class_59'
07,029,'f0711','class_59'
07,046,'f0711','class_59'
07,047,'f0711','class_59'
07,065,'f0711','class_59'
07,106,'f0711','class_59'
07,111,'f0711','class_59'
07,115,'f0711','class_59'
07,152,'f0711','class_59'
07,161,'f0711','class_59'
07,162,'f0711','class_59'
07,201,'f0711','class_59'
07,202,'f0711','class_59'
07,246,'f0711','class_59'
07,248,'f0711','class_59'
07,249,'f0711','class_59'
07,283,'f0711','class_59'
07,284,'f0711','class_59'
07,285,'f0711','class_59'
07,286,'f0711','class_59'
07,287,'f0711','class_59'
07,288,'f0711','class_59'
07,501,'f0711','class_59'
07,505,'f0711','class_59'
07,506,'f0711','class_59'
07,507,'f0711','class_59'
07,513,'f0711','class_59'
07,001,'p0711401','class_59'
07,006,'p0711401','class_59'
07,008,'p0711401','class_59'
07,009,'p0711401','class_59'
07,013,'p0711401','class_59'
07,026,'p0711401','class_59'
07,029,'p0711401','class_59'
07,046,'p0711401','class_59'
07,047,'p0711401','class_59'
07,065,'p0711401','class_59'
07,106,'p0711401','class_59'
07,111,'p0711401','class_59'
07,115,'p0711401','class_59'
07,152,'p0711401','class_59'
07,161,'p0711401','class_59'
07,162,'p0711401','class_59'
07,201,'p0711401','class_59'
07,202,'p0711401','class_59'
07,246,'p0711401','class_59'
07,248,'p0711401','class_59'
07,249,'p0711401','class_59'
07,283,'p0711401','class_59'
07,284,'p0711401','class_59'
07,285,'p0711401','class_59'
07,286,'p0711401','class_59'
07,287,'p0711401','class_59'
07,288,'p0711401','class_59'
07,501,'p0711401','class_59'
07,505,'p0711401','class_59'
07,506,'p0711401','class_59'
07,507,'p0711401','class_59'
07,513,'p0711401','class_59'
07,001,'s53','class_60'
07,026,'s53','class_60'
07,151,'s53','class_60'
07,506,'s53','class_60'
07,507,'s53','class_60'
07,233,'p0732101','class_61'
07,234,'p0732101','class_61'
07,236,'p0732101','class_61'
07,237,'p0732101','class_61'
07,238,'p0732101','class_61'
07,261,'p0732101','class_61'
07,262,'p0732101','class_61'
07,266,'p0732101','class_61'
07,268,'p0732101','class_61'
07,269,'p0732101','class_61'
07,271,'p0732101','class_61'
07,301,'p0732101','class_61'
07,302,'p0732101','class_61'
07,311,'p0732101','class_61'
07,415,'p0732101','class_61'
07,501,'p0732101','class_61'
07,505,'p0732101','class_61'
07,506,'p0732101','class_61'
07,505,'s62','class_62'
07,506,'s62','class_62'
07,505,'s67','class_62'
07,506,'s67','class_62'
07,001,'s60','class_63'
07,151,'s60','class_63'
07,505,'s60','class_63'
07,506,'s60','class_63'
07,505,'s61','class_64'
07,001,'s36','class_65'
07,006,'s36','class_65'
07,111,'s36','class_65'
07,151,'s36','class_65'
07,227,'s36','class_65'
07,262,'s36','class_65'
07,283,'s36','class_65'
07,284,'s36','class_65'
07,285,'s36','class_65'
07,286,'s36','class_65'
07,287,'s36','class_65'
07,288,'s36','class_65'
07,289,'s36','class_65'
07,290,'s36','class_65'
07,502,'s36','class_65'
07,503,'s36','class_65'
07,233,'s15','class_66'
07,234,'s15','class_66'
07,241,'s15','class_66'
07,261,'s15','class_66'
07,291,'s15','class_66'
07,503,'s15','class_66'
07,501,'s44','class_67'
07,501,'s69','class_67'
07,268,'s99','class_68'
07,437,'s99','class_68'
07,213,'f1430','class_69'
07,437,'f1430','class_69'
07,213,'p1430601','class_69'
07,437,'p1430601','class_69'
07,040,'k14','class_70'
07,065,'k14','class_70'
07,213,'k14','class_70'
07,315,'k14','class_70'
07,316,'k14','class_70'
07,437,'k14','class_70'
07,316,'p1420301','class_71'
07,040,'f1420','class_72'
07,065,'f1420','class_72'
07,315,'f1420','class_72'
07,316,'f1420','class_72'
07,315,'f1520','class_73'
07,315,'f2718','class_73'
07,315,'k27','class_73'
07,315,'p1520201','class_73'
07,315,'p2718101','class_73'
07,040,'p1510301','class_74'
07,311,'p1510301','class_74'
07,315,'p1510301','class_74'
07,040,'f1510','class_75'
07,213,'f1510','class_75'
07,311,'f1510','class_75'
07,315,'f1510','class_75'
07,040,'k15','class_75'
07,213,'k15','class_75'
07,311,'k15','class_75'
07,315,'k15','class_75'
07,040,'p1420303','class_76'
07,065,'p1420303','class_76'
07,315,'p1420303','class_76'
07,018,'p0710402','class_77'
07,030,'p0710402','class_77'
07,039,'p0710402','class_77'
07,040,'p0710402','class_77'
07,048,'p0710402','class_77'
07,065,'p0710402','class_77'
07,152,'p0710402','class_77'
07,161,'p0710402','class_77'
07,213,'p0710402','class_77'
07,315,'p0710402','class_77'
07,004,'f0710','class_78'
07,018,'f0710','class_78'
07,030,'f0710','class_78'
07,039,'f0710','class_78'
07,040,'f0710','class_78'
07,048,'f0710','class_78'
07,065,'f0710','class_78'
07,152,'f0710','class_78'
07,161,'f0710','class_78'
07,209,'f0710','class_78'
07,213,'f0710','class_78'
07,216,'f0710','class_78'
07,217,'f0710','class_78'
07,219,'f0710','class_78'
07,227,'f0710','class_78'
07,263,'f0710','class_78'
07,264,'f0710','class_78'
07,270,'f0710','class_78'
07,280,'f0710','class_78'
07,281,'f0710','class_78'
07,282,'f0710','class_78'
07,315,'f0710','class_78'
07,213,'f0741','class_79'
07,214,'f0741','class_79'
07,215,'f0741','class_79'
07,217,'f0741','class_79'
07,227,'f0741','class_79'
07,262,'f0741','class_79'
07,291,'f0741','class_79'
07,292,'f0741','class_79'
07,293,'f0741','class_79'
07,294,'f0741','class_79'
07,295,'f0741','class_79'
07,302,'f0741','class_79'
07,213,'p0741201','class_79'
07,214,'p0741201','class_79'
07,215,'p0741201','class_79'
07,217,'p0741201','class_79'
07,227,'p0741201','class_79'
07,262,'p0741201','class_79'
07,291,'p0741201','class_79'
07,292,'p0741201','class_79'
07,293,'p0741201','class_79'
07,294,'p0741201','class_79'
07,295,'p0741201','class_79'
07,302,'p0741201','class_79'
07,213,'p0750101','class_79'
07,214,'p0750101','class_79'
07,215,'p0750101','class_79'
07,217,'p0750101','class_79'
07,227,'p0750101','class_79'
07,262,'p0750101','class_79'
07,291,'p0750101','class_79'
07,292,'p0750101','class_79'
07,293,'p0750101','class_79'
07,294,'p0750101','class_79'
07,295,'p0750101','class_79'
07,302,'p0750101','class_79'
07,213,'s87','class_80'
07,214,'s87','class_80'
07,215,'s87','class_80'
07,294,'s87','class_80'
07,161,'s58','class_81'
07,291,'s58','class_81'
07,214,'s03','class_82'
07,217,'s03','class_82'
07,227,'s03','class_82'
07,264,'s03','class_82'
07,266,'s03','class_82'
07,283,'s03','class_82'
07,284,'s03','class_82'
07,285,'s03','class_82'
07,286,'s03','class_82'
07,287,'s03','class_82'
07,288,'s03','class_82'
07,289,'s03','class_82'
07,290,'s03','class_82'
07,051,'p0732202','class_83'
07,201,'p0732202','class_83'
07,205,'p0732202','class_83'
07,214,'p0732202','class_83'
07,227,'p0732202','class_83'
07,230,'p0732202','class_83'
07,248,'p0732202','class_83'
07,266,'p0732202','class_83'
07,267,'p0732202','class_83'
07,283,'p0732202','class_83'
07,284,'p0732202','class_83'
07,285,'p0732202','class_83'
07,286,'p0732202','class_83'
07,287,'p0732202','class_83'
07,288,'p0732202','class_83'
07,289,'p0732202','class_83'
07,290,'p0732202','class_83'
07,209,'p4720201','class_84'
07,214,'p4720201','class_84'
07,215,'p4720201','class_84'
07,217,'p4720201','class_84'
07,227,'p4720201','class_84'
07,265,'p4720201','class_84'
07,281,'p4720201','class_84'
07,282,'p4720201','class_84'
07,209,'f4720','class_85'
07,214,'f4720','class_85'
07,215,'f4720','class_85'
07,217,'f4720','class_85'
07,227,'f4720','class_85'
07,228,'f4720','class_85'
07,229,'f4720','class_85'
07,232,'f4720','class_85'
07,245,'f4720','class_85'
07,265,'f4720','class_85'
07,281,'f4720','class_85'
07,282,'f4720','class_85'
07,209,'k47','class_85'
07,214,'k47','class_85'
07,215,'k47','class_85'
07,217,'k47','class_85'
07,227,'k47','class_85'
07,228,'k47','class_85'
07,229,'k47','class_85'
07,232,'k47','class_85'
07,245,'k47','class_85'
07,265,'k47','class_85'
07,281,'k47','class_85'
07,282,'k47','class_85'
07,004,'p0710301','class_86'
07,209,'p0710301','class_86'
07,216,'p0710301','class_86'
07,217,'p0710301','class_86'
07,219,'p0710301','class_86'
07,227,'p0710301','class_86'
07,263,'p0710301','class_86'
07,264,'p0710301','class_86'
07,270,'p0710301','class_86'
07,280,'p0710301','class_86'
07,281,'p0710301','class_86'
07,282,'p0710301','class_86'
07,209,'p4720302','class_87'
07,214,'p4720302','class_87'
07,215,'p4720302','class_87'
07,217,'p4720302','class_87'
07,227,'p4720302','class_87'
07,228,'p4720302','class_87'
07,229,'p4720302','class_87'
07,232,'p4720302','class_87'
07,245,'p4720302','class_87'
07,234,'s19','class_88'
07,215,'s22','class_89'
07,213,'p1510302','class_90'
07,201,'s93','class_91'
07,006,'f0713','class_92'
07,008,'f0713','class_92'
07,009,'f0713','class_92'
07,029,'f0713','class_92'
07,106,'f0713','class_92'
07,113,'f0713','class_92'
07,151,'f0713','class_92'
07,152,'f0713','class_92'
07,153,'f0713','class_92'
07,154,'f0713','class_92'
07,155,'f0713','class_92'
07,156,'f0713','class_92'
07,157,'f0713','class_92'
07,158,'f0713','class_92'
07,159,'f0713','class_92'
07,160,'f0713','class_92'
07,161,'f0713','class_92'
07,006,'p0713101','class_92'
07,008,'p0713101','class_92'
07,009,'p0713101','class_92'
07,029,'p0713101','class_92'
07,106,'p0713101','class_92'
07,113,'p0713101','class_92'
07,151,'p0713101','class_92'
07,152,'p0713101','class_92'
07,153,'p0713101','class_92'
07,154,'p0713101','class_92'
07,155,'p0713101','class_92'
07,156,'p0713101','class_92'
07,157,'p0713101','class_92'
07,158,'p0713101','class_92'
07,159,'p0713101','class_92'
07,160,'p0713101','class_92'
07,161,'p0713101','class_92'
07,161,'s74','class_93'
07,161,'s92','class_93'
07,040,'f5415','class_94'
07,040,'f5465','class_94'
07,040,'f9140','class_94'
07,040,'k54','class_94'
07,040,'k91','class_94'
07,040,'p5415501','class_94'

```
07,040,'p5465101','class_94'
07,040,'p9140501','class_94'
07,001,'s02','class_95'
```

10 Literatur

(1) Peter Brockhausen und Katharina Morik
Direct access of an ILP Algorithm to a Database Management System
In Bernhard Pfaringer und Johannes Fürnkranz (Hrsg.) *Data Mining with Inductive Logic Programming (ILP for KDD)*, MLnet sponsored Familiarization Workshop, Bari, Italien, Juli 1996

(2) K. Morik, S. Wrobel, J-U. Kietz und W. Emde
Knowledge Acquisition and Machine Learning : Theory, Methods and Applications. Academic Press, London, 1993

(3) Katharina Morik und Peter Brockhausen
A Multistrategy Approach to Relational Knowledge Discovery in Databases,
Proceedings of the Third International Workshop on Multistrategy Learning (MSL-96), AAAI Press, 1996

(4) J-U. Kietz und K. Morik
A Polynomial Approach to the Constructive Induction of Structural Knowledge *Machine Learning Journal*, 14(2):193-217, 1994

(5) Dr. Ing. Madjid Fathi
Skriptum zur Vorlesung Wissensakquisition und Wissensmodellierung bei Expertensystemanwendungen
Universität Dortmund, Sommersemester 1992

(6) Dr. Ing. Madjid Fathi
Material zur Vorlesung Basiskonzepte für Expertensystemanwendungen,
Universität Dortmund, Wintersemester 1991 / 1992

(7) Prof. Dr. Egon Jehle
Unterlagen zur Lehrveranstaltung Industriebetriebslehre I, Strategisches und operative Produktionsmanagement,
Universität Dortmund, Wintersemester 1993 / 1994

(8) Pflichtenheft QUIS Daimler Benz AG

(9) PKW-Schadensbuch Daimler Benz AG

(10) Michael E. Porter
Wettbewerbsvorteile (Competitive Advantage)
Frankfurt/M. New York 1986

(11) Edgar Sommer, Werner Emde, Jörg-Uwe Kietz, Stefan Wrobel
MOBAL 3.X User Guide, German National Research Center of Computer Science, AI Research Division, Schloss Birlinghoven,
D-53754 St. Augustin,
1. Dezember 1994

(12) Katharina Morik
Sloppy Modeling Knowledge Representation and Organisation,
in *Machine Learning*, Springer Verlag Berlin, New York 1989
Page 107 - 134

(13) K. Milzner
Konzepte der Wissensakquisition - Direkte Methoden
Forschungsbericht Nr. 287 - Abteilung Informatik, Universität Dortmund

(14) A. Nitche
Indirekte Methoden der Wissensakquisition - Indirekte Methoden
Forschungsbericht Nr. 316 - Abteilung Informatik, Universität Dortmund
(15) T. Bylander, B. Chandrasekaran
The 'Right' level of Abstaction for Knowledge Acquisition
AAAI Workshop on Knowledge Acquisition for Knowledge Based Systems
1986, Page 7 - 13
(16) Guus Schreiber, Bob Wielinga, Joost Breuker
Introduction and Overview, Page 1 -18
KADS- A Principle Approach to Knowledge-Based System Development
Academic Press 1993
(17) C. Wolfs, C. Herold, F. Szurmann
Wissensmodellierung und Operationalisierung im Knowledge Engineering
Diplomarbeit Universität Dortmund, Lehrstuhl I, März 1992
(18) Katharina Morik
Skriptum Einführung in die künstliche Intelligenz
Universität Dortmund, Fachbereich Informatik, Lehrstuhl VIII
4. Auflage, Sommersemester 1995
(19) Peter Brockhausen
Work Report
University of Dortmund, Novermber 7, 1996

11 Anhang

11.1 Programmlistings

11.1.1 DOCTOFAC

```
/* lib.cpp entält Funktionen, die von doctofac und mobtodb gemeinsam verwendet werden */

#include "lib.cpp"

/* pred_def ist ein Elemente der Liste, die die Prädikatsdefinitionen enthält
   Betrachtet man die 7 Argumente von PREDICATE DEFINITION der Konfigurationsdatei
   so entsprechen diese Argumente genau den Feldern von pred_def */

struct pred_def
{
  int pred_position_begin;      /* Arg1 PREDICATE_DEFINITION */
  int pred_position_end;        /* Arg2 PREDICATE_DEFINITION */
  int pred_start_position;      /* Arg3 PREDICATE_DEFINITION */
  int pred_end_position;        /* Arg4 PREDICATE_DEFINITION */
  int pred_pattern_position;    /* Arg5 PREDICATE_DEFINITION */
  string pred_pattern;          /* Arg6 PREDICATE_DEFINITION */
  string pred_add;              /* Arg7 PREDICATE_DEFINITION */
  pred_def *next;               /* Verweis auf das nächste Elemente der Liste */
};

/* pred_def_list ist die Liste, die die Prädikatsdefinitionen enthält. Außerdem
   sind noch die Argumentdefinition sowie die Namen der Eingabe und Ausgabe definitionen
   enthalten */

struct pred_def_list
{
  int count_def;                /* Anzahl der gelesenen Prädikatsdefinionen */
  int pred_arg_position;        /* Arg1 ARGUMENT_DEFINITION */
  int pred_arg_start_position;  /* Arg2 ARGUMENT_DEFINITION */
  int pred_arg_end_position;    /* Arg3 ARGUMENT_DEFINITION */
  string pred_arg_add;          /* Arg5 ARGUMENT_DEFINITION */
  string pred_arg_pattern;      /* Arg4 ARGUMENT_DEFINITION */
  string input;                 /* Eingabedatei */
  string output;                /* Ausgabedatei */
  pred_def *first;              /* Verweis auf das erste Element der Liste */
  pred_def *last;               /* Verweis auf das letzte Element der Liste */
  int information;              /* Angabe von NO_INFORMATION */
};

/* fact_elem enthält ein bereits erzeugtes Faktum und einen Verweis auf das nächste
   Faktum */

struct fact_elem
{
  char *str;                    /* Bereits erzeugtes Faktum */
  fact_elem *next;              /* Verweis auf das nächste Element der Liste */
};

/* factset ist eine Liste, die die bereits erzeugten Fakten enthält */
```

```
struct factset
{
  long count_lines;          /* Anzahl geslesener Zeilen der Eingabedatei */
  long count_facts;          /* Anzahl erzeugter Fakten */
  fact_elem *first;          /* Verweis auf das erste Element der Liste */
  fact_elem *last;           /* Verweis auf das letzte Element der Liste */
};

/* init_pred_def_list setzt die Liste der Pädikatsdefinitionen in die
   Startkonfiguration */

pred_def_list * init_pred_def_list(pred_def_list *definition)
{
  definition->information = 1;
  definition->count_def = 0;
  definition->first = NULL;
  definition->last = NULL;
  definition->pred_arg_start_position = 0;
  definition->pred_arg_end_position = 0;
  definition->pred_arg_position = 0;
  emptystr(definition->pred_arg_add);
  emptystr(definition->pred_arg_add);
  emptystr(definition->input);
  emptystr(definition->output);
  return definition;
}

/* new_factset setzt die Liste der Fakten in die Startkonfiguration */

factset* new_factset()
{
  factset *p;
  p = (factset *) new_mem(sizeof(factset),1);
  p->first = NULL;
  p->last = NULL;
  p->count_facts = 0;
  p->count_lines = 0;
  return p;
}

/* add_fact fügt der Liste der Fakten ein neues Faktum hinzu */

factset* add_fact(factset *set, char* fact)
{
 fact_elem * h;
 h = (fact_elem *) new_mem(sizeof(fact_elem),1);
 h->str = (char *) new_mem(strlen(fact)+1,1);
 strcpy(h->str,fact);
 h->next = NULL;
 if (set->last != NULL)
   set->last->next = h;
 if(set->first == NULL)
   set->first = h;
 set->last = h;
 set->count_facts++;
 return set;
}

/* Prüft, ob ein Faktum bereits in der Faktenliste enthalten ist. Ist die nicht
```

```
  Fall, wird add_fact aufgerufen, und das Element hinzugefügt */

factset *add_fact_to_set(factset * set, char * str)
{
 fact_elem *h;
 h = set->first;
 while (h != NULL)
 {
  if ((strcmp(h->str,str)) == 0)
  {
   return set;
  }
  h = h->next;
 }
 add_fact(set,str);
 return set;
}

/* make_pred_def erzeugt aus einer Zeile der Konfigurationsdatei die Einträge
  für pred_def_list. Dabei wird auf eventuelle Fehler geprüft, wie
  Mehrfachdefinition von Eingabedatei, Ausgabedatei oder Argumentdefinition. */

void make_pred_def (pred_def_list *definition, char * buffer)
{
 string type_of_line;
 string h;
 get_arg_n(type_of_line,buffer,' ',1,sizeof(string));
 if (strcmp(type_of_line,"REM") == 0)
  return;
 if (strcmp(type_of_line,"INPUT") == 0)
 {
  if (strcmp(definition->input,"") == 0)
      {
       get_arg_n (h,buffer,' ',2,sizeof(string));
   delchar(definition->input,h,' ');
      }
      else
       exit_on_error("More than one inputfile");
  return;
 }
 if (strcmp(type_of_line,"OUTPUT") == 0)
 {
  if (strcmp(definition->output,"") == 0)
      {
       get_arg_n (h,buffer,' ',2,sizeof(string));
   delchar(definition->output,h,' ');
      }
      else
       exit_on_error("More than one outputfile");
  return;
 }
 if (strcmp(type_of_line,"ARGUMENT_DEFINITION") == 0)
 {
      if (definition->pred_arg_position == 0)
      {
       get_arg_n(h,buffer,' ',2,sizeof(string));
   definition->pred_arg_position = string_to_number(h);
   get_arg_n(h,buffer,' ',3,sizeof(string));
   definition->pred_arg_start_position = string_to_number(h);
```

```
      get_arg_n(h,buffer,' ',4,sizeof(string));
      definition->pred_arg_end_position = string_to_number(h);
      get_arg_n(definition->pred_arg_add,buffer,' ',5,sizeof(string));
      get_arg_n(definition->pred_arg_pattern,buffer,' ',6,sizeof(string));
         }
         else
           exit_on_error("More than one argument definition");
    return;
  }
  if (strcmp(type_of_line,"PREDICATE_DEFINITION") == 0)
  {
    pred_def *p;
    string h;
    p = (pred_def *) new_mem(sizeof(pred_def),1);
    definition->count_def++;
    if (definition->first == NULL)
      definition->first = p;
    p->next = NULL;
    if (definition->last != NULL)
      definition->last->next = p;
    definition->last = p;
    get_arg_n(h,buffer,' ',2,sizeof(string));
    p->pred_position_begin = string_to_number(h);
    get_arg_n(h,buffer,' ',3,sizeof(string));
    trim(h,h,' ');
    if (strcmp(h,"*") == 0)
       strcpy(h,"1000");
    p->pred_position_end = string_to_number(h);
    get_arg_n(h,buffer,' ',4,sizeof(string));
    p->pred_start_position = string_to_number(h);
    get_arg_n(h,buffer,' ',5,sizeof(string));
    p->pred_end_position = string_to_number(h);
    get_arg_n(h,buffer,' ',6,sizeof(string));
    p->pred_pattern_position = string_to_number(h);
    get_arg_n(p->pred_pattern,buffer,' ',7,sizeof(string));
    get_arg_n(p->pred_add,buffer,' ',8,sizeof(string));
  return;
  }
  if (strcmp(type_of_line,"NO_INFORMATION") == 0)
  {
    definition->information = 0;
    return;
  }
  exit_on_error(strcat("Unknown Syntax : ",type_of_line));
}

/* Nachdem pred_def_list vollständig erzeugt wurde, wird sie von check_pred_def
   auf Fehler überprüft, die nur nach der Verarbeitung der gesamten Konfigurationsdatei
   festgestellt werden können */

bool check_pred_def (pred_def_list *definition)
{
  bool check_ok = TRUE;
  int count = 1;
  pred_def *akt_def;
  if(definition->count_def == 0)
    exit_on_error("No predicate definition");
  if(strcmp(definition->input,"") == 0)
    exit_on_error("No inputfile");
```

```
if(strcmp(definition->output,"") == 0)
  exit_on_error("No outputfile");
if(definition->pred_arg_position == 0)
  exit_on_error("No argument definition");
if(definition->pred_arg_start_position > definition->pred_arg_end_position)
{
  printf("Argument Definition : Error in pred_arg_start/end_position\n");
  check_ok = FALSE;
}
if((!(is_add(definition->pred_arg_add)) || (strcmp(definition->pred_arg_add,"") == 0)))
{
  printf("Argument Definition : Error in pred_arg_add\n");
  check_ok = FALSE;
}
if(strcmp(definition->pred_arg_pattern,"") == 0)
{
  printf("Argument Definition : Error in pred_arg_pattern\n");
  check_ok = FALSE;
}
if((definition->pred_arg_position == 0) || \
   (definition->pred_arg_start_position == 0) || \
   (definition->pred_arg_end_position == 0))
{
  printf("Argument Definition : Impossible value\n");
  check_ok = FALSE;
}
akt_def = definition->first;
while (akt_def !=NULL)
{
  if(akt_def->pred_position_begin > akt_def->pred_position_end)
  {
    printf("Predicate Definition : %d  --  Error in pred_position_begin/end\n",count);
    check_ok = FALSE;
  }
  if(akt_def->pred_start_position > akt_def->pred_end_position)
  {
    printf("Predicate Definition : %d  --  Error in pred_start/end_position\n",count);
    check_ok = FALSE;
  }
  if((!(is_add(akt_def->pred_add)) || (strcmp(akt_def->pred_add,"") == 0)))
  {
    printf("Predicate Definition : %d  --  Error in pred_add\n",count);
    check_ok = FALSE;
  }
  if(strcmp(akt_def->pred_pattern,"") == 0)
  {
    printf("Predicate Definition : %d  --  Error in pred_pattern\n",count);
    check_ok = FALSE;
  }
  if((akt_def->pred_position_begin == 0) || \
     (akt_def->pred_position_end == 0) || \
     (akt_def->pred_start_position == 0) || \
     (akt_def->pred_end_position == 0) || \
     (akt_def->pred_pattern_position == 0))
  {
    printf("Predicate Definition : %d  --  Impossible value\n",count);
    check_ok = FALSE;
  }
```

```
    count++;
    akt_def = akt_def->next;
  }
  return check_ok;
}

/* Liest alle Zeilen der Konfigurationsdatei und ruft für jede nicht leere Zeile
   make_pred_def auf */

void read_pred_def(char *config_file, pred_def_list *definition)
{
  FILE * configfile;
  char *buffer;
  buffer = (char *) new_mem (1000,1);
  configfile = open_text_read(config_file,1);
  while (!(feof(configfile)))
  {
    read_line(configfile,buffer,1000);
    if (buffer != NULL)
    {
      trim(buffer,buffer,' ');
      if (strlen(buffer) != 0)
            make_pred_def(definition, buffer);
    }
  }
  fclose(configfile);
  if (check_pred_def(definition) == 0)
    exit_on_error("Incorrect Configfile");
}

/* Überprüft, ob eine Zeichenkette h dem Muster pattern entspricht. Wird bei der Filterfunktion
   von PREDICATE_DEFINITION und ARGUMENT_DEFINITION verwendet */

bool check_pattern (char * h, char* pattern)
{
  int i = 0;
  if (strlen(pattern) == 0)
    exit_on_error ("Empty Pattern");
  if (strlen(h) == 0)
    return FALSE;
  while (i<strlen(h))
  {
    if (pattern[i] == '*')
      return TRUE;
    if ((h[i] == pattern[i]) || (pattern[i] =='?'))
      i++;
    else
      return FALSE;
  };
  return TRUE;
}

/* make_fact nimmt als Eingabe eine Zeile der Eingabedatei und durchläuft alle Einträge der Liste
   der Prädikatsdefintionen, um Fakten zu erzeugen. Wenn diese Zeile der Filterfunktion der
aktuellen
   Prädikatsdefinition entspricht, wird nach der Definition ein Faktum erzeugt, und in die
   Liste der Fakten eingetragen. Dabei wird add_fact_to_set aufgerufen, um eventuell doppelte
   Fakten zu vermeiden. */
```

```
void make_fact (factset *fact_list , pred_def_list *definition,pred_def *p, char *buffer)
{
 int pred_position_count;
 string h;
 string m;
 string f;
 string fact;
 pred_position_count = p->pred_position_begin;
 get_arg_n(m,buffer,' ',definition->pred_arg_position,sizeof(string));
 if (!(check_pattern(m,definition->pred_arg_pattern)))
   return;
 get_arg_n(f,buffer,' ',p->pred_pattern_position,sizeof(string));
 if (!(check_pattern(f,p->pred_pattern)))
   return;
 while (pred_position_count <= p->pred_position_end)
 {
   get_arg_n(h,buffer,' ',pred_position_count,sizeof(string));
   if (strlen(h) > 0)
   {
     strcpy(fact,p->pred_add);
     if (strlen(h) < p->pred_end_position)
         exit_on_error ("Incorrect range for predicate");
     midstr(h,h,p->pred_start_position,p->pred_end_position);
     strcat(fact,h);
     strcat(fact,"(");
     strcat(fact,definition->pred_arg_add);
     midstr(m,m,definition->pred_arg_start_position,definition->pred_arg_end_position);
     strcat(fact,m);
     strcat(fact,").");
     add_fact_to_set(fact_list,fact);
     if (definition->information == 1)
         printf("Erzeugte Fakten : %d\r",fact_list->count_facts);
   }
     pred_position_count++;
 }
}

/* Liest alle Zeilen der Eingabedatei, und ruft für jede nicht leere Zeile
   make_fact auf */

void build_facts(factset *fact_list,pred_def_list *definition)
{
 FILE * inputfile;
 char *buffer;
 int i;
 pred_def * akt_def;
 buffer = (char *) new_mem (1000,1);
 inputfile = open_text_read(definition->input,1);
 while (!(feof(inputfile)))
 {
   read_line(inputfile,buffer,1000);
   if (buffer != NULL)
   {
     trim(buffer,buffer,' ');
     if (strcmp(buffer,"") != 0)
     {
               akt_def = definition->first;
               fact_list->count_lines++;
               for (i=1; i<=definition->count_def; i++)
```

```
                    {
                              make_fact (fact_list,definition,akt_def,buffer);
                              akt_def = akt_def->next;
                    }
    }
   }
 }
 fclose(inputfile);
}

/* Nachdem alle Fakten erzeugt wurden, wird write_facts aufgerufen, um alle Fakten
   in die Ausgabedatei zu schreiben */

void write_facts(factset *fact_list,pred_def_list *definition)
{
 FILE *outputfile;
 fact_elem *h;
 if (fact_list->count_facts < 1)
   return;
 outputfile = open_text_write(definition->output,1);
 h = fact_list->first;
 do
 {
   write_line(outputfile,h->str);
   h = h->next;
 }
 while (h != NULL);
 fclose(outputfile);
}

/* Gibt am Ende der Ausführung von doctofac die Informationen darüber aus,
   wieviele Fakten welchen Typs erzeugt wurden. Die Ausführung von print_information
   kann durch NO_INFORMATION in der Konfigurationsdatei vermieden werden */

void print_information(factset *fact_list,pred_def_list *definition)
{
 if (definition->information == 1)
 {
   int count;
   pred_def* akt_def;
   fact_elem * akt_fact;
   printf("INSGESAMT GELESENE ZEILEN : %d\n\n",fact_list->count_lines);

          akt_def = definition->first;
   while(akt_def != NULL)
   {
     count = 0;
     akt_fact = fact_list->first;
     while (akt_fact != NULL)
     {
       if (instr(1,akt_fact->str,akt_def->pred_add) == 1)
              count ++;
              akt_fact = akt_fact->next;
     }
     printf("ERZEUGTE FAKTEN MIT KENNZEICHEN %s : %d\n",akt_def->pred_add,count);
     akt_def = akt_def->next;
   }
   printf("\nINSGESAMT ERZEUGTE FAKTEN : %d\n",fact_list->count_facts);
 }
```

```
}


int main(int arg_num, char* arg_list[])
{
  printf("doctofac -- 1997 -- Mark Siebert\n\n\n");
  pred_def_list *definition;
  factset* fact_list;          /* Zeiger auf die Liste aller Fakten */
  definition = (pred_def_list *) new_mem(sizeof(pred_def_list),1);  /*Zeiger auf
                         die Lister alle Definitionen der Konfigurationsdatei */
  init_pred_def_list(definition);
  fact_list = new_factset();
  if (arg_num < 2)               /* Test, ob eine Kofigurationsdatei vorhanden ist */
    exit_on_error("No configfile");
  read_pred_def(arg_list[1],definition);   /* Einlesen der Konfigurationsdatei */
  build_facts(fact_list,definition);       /* Erzeugen von Fakten */
  write_facts(fact_list,definition);       /* Schreiben der Fakten in Ausgabedatei */
  print_information(fact_list,definition); /* Ausgabe von Informationen am Bildschirm */
  return 0;
}
```

11.1.2 MOBTODB

```
/* lib.cpp entält Funktionen, die von doctofac und mobtodb gemeinsam verwendet werden */

#include "lib.cpp"

/* def entält die Definition eines Feldes der Ausgabedatei. Die Definitionen werden
   in einer Liste angeordnet. Für die Definition existieren die Möglichkeit
   MOBAL_CLASS, PREDICATE_NAME, ARGUMENT, die in der Variablen type angelegt werden.
   var_type enthält die Information 'STRING' oder 'NUMBER'. Für den Fall, das type gleich
   ARGUMENT ist, enthalten start_position und end_position die Definition, aus welchem
   Intervall das Argument gebildet werden soll. */

struct def
{
  string type;         /* enthält entweder MOBAL_CLASS, PREDICATE_NAME oder ARGUMENT */
  string var_type;      /* enthält entweder STRING oder NUMBER */
  int start_position;   /* Arg2 von ARGUMENT */
  int end_position;     /* Arg2 von ARGUMENT */
  def *next;           /* Verweis auf den nächsten Listeneintrag */
};

/* def_list enthält Liste aller Definitionen von Feldern, die in der Konfigurationsdatei
   vorgenommen wurden, sowie die Namen der Eingabedatei und der Ausgabedatei */

struct def_list
{
  string input;
  string output;
  def *first;
  def *last;
};

/* Ininitalisierung einer leeren def_list */

def_list * new_def_list()
```

```
{
  def_list * p;
  p = (def_list *) new_mem(sizeof(def_list),1);
  p->first = NULL;
  p->last = NULL;
  p->input[0] ='\0';
  p->output[0] ='\0';
  return p;
}

/* make_def nimmt eine Zeile der Konfigurationsdatei als Eingabe, und bildet
   daraus eine neuen Eintrag der def_list. Dabei wird der Eintrag auf eventuelle
   Fehler überprüft */

void make_def(char * buffer, def_list *definition)
{
  def* p;
  string h;
  string type_of_line;
  get_arg_n(type_of_line,buffer,' ',1,sizeof(string));
  if (strcmp(type_of_line,"INPUT") == 0)
  {
    if (strlen(definition->input) != 0)
                      exit_on_error("More than one inputfile");
           get_arg_n(definition->input,buffer,' ',2,sizeof(string));
    return;
  }
  if (strcmp(type_of_line,"OUTPUT") == 0)
  {
    if (strlen(definition->output) != 0)
                      exit_on_error("More than one outputfile");
           get_arg_n(definition->output,buffer,' ',2,sizeof(string));
    return;
  }
  if ((strcmp(type_of_line,"MOBAL_CLASS") == 0) || \
      (strcmp(type_of_line,"PREDICATE_NAME") == 0) || \
      (strcmp(type_of_line,"ARGUMENT") == 0))
  {
    p = (def *) new_mem(sizeof(def),1);
    p->next = NULL;
    if (definition->last != NULL)
      definition->last->next = p;
    definition->last = p;
    if (definition->first == NULL)
      definition->first = p;
    strcpy(p->type,type_of_line);
    get_arg_n(h,buffer,' ',2,sizeof(string));
    strcpy(p->var_type,h);
    get_arg_n(h,buffer,' ',3,sizeof(string));
    p->start_position = string_to_number(h);
    get_arg_n(h,buffer,' ',4,sizeof(string));
    p->end_position = string_to_number(h);
           if (strcmp(type_of_line,"ARGUMENT") == 0)
    {
       if ((p->start_position <= 0) || \
                         (p->start_position > p->end_position))
                                exit_on_error("Incorrect range for argument definition");
           }
    if((strcmp(p->var_type,"NUMBER") != 0) && \
```

```
      (strcmp(p->var_type,"STRING") != 0))
       exit_on_error("Incorrect arguments");
    return;
  }
  exit_on_error(strcat("Unknown Syntax : ",type_of_line));
}

/* Liest die Zeilen der Konfigurationsdatei, ruft für jede nichtleere Zeile
   make_def auf, um einen neuen Eintrag in die Konfigurationsdatei zu erzeugen */

void read_config_file(char *filename, def_list *definition)
{
  FILE* configfile;
  char* buffer;
  configfile = open_text_read (filename,1);
  buffer = (char *) new_mem(1000,1);
  while(!(feof(configfile)))
  {
    read_line(configfile,buffer,1000);
    if (buffer != NULL)
    {
      trim(buffer,buffer,' ');
      if (strcmp(buffer,"") != 0)
                    make_def(buffer,definition);
    }
  }
  if (strlen(definition->input) == 0)
      exit_on_error("No inputfile");
  if (strlen(definition->output) == 0)
      exit_on_error("No outputfile");
  if (definition->first == NULL)
      exit_on_error("Empty outputfile declared");
  fclose(configfile);
  free(buffer);
}



void format_pred_list(char * pred_list)
{
  unsigned int i = 0;
  int k = 0;
  int level = 0;
  while (i < strlen(pred_list))
  {
    if(pred_list[i] == '(')
      level++;
    if(pred_list[i] == ')')
      level--;
    if (((level == 1) && (pred_list[i] != '(') && (pred_list[i] != ')')) || \
            pred_list[i] == ',')
    {
      pred_list[k] = pred_list[i];
      k++;
    }
    i++;
  }
  pred_list[k] ='\0';
}
```

```
/* split_input_line erhält als Eingabe eine Zeile der Eingabedatei, und unterteilt sie in die Teil
   mobal_class, Liste von Prädikatennamen und Liste von Argumenten. Dabei werden
   werden 3 Zeiger jeweils auf den Anfang der 3 Teile gesetzt.*/

void split_input_line(char *buffer, char **mobal_class,char **pred_list, char **arg_list)
{
  long i = 0;
  long level = 0;
  long elem_count = 1;
  long bufferlen;
  bufferlen = strlen(buffer);
  *mobal_class = buffer;
  *pred_list = NULL;
  while (i<bufferlen)
  {
    if (buffer[i] == '[')
      level++;
    if (buffer[i] == ']')
      level--;
    if ((level == 0) && (buffer[i]) == ',')
    {
      elem_count++;
      if (elem_count == 2)
      {
             buffer[i] = '\0';
           *pred_list = &buffer[i+2];
      }
      if (elem_count == 3)
      {
           buffer[i-1] = '\0';
           *arg_list = &buffer[i+2];
           buffer[bufferlen-1] = '\0';
           break;
      }
    }
    i++;
  }
  format_pred_list(*pred_list);
}

/* make_output erhält als Eingabe Zeile der Eingabedatei, und bildet daraus
   mit Hilfe der Liste von Definitionen der Konfigurationsdatei die Zeilen der
   Ausgabedatei. Dabei wird jede Zeile direkt in die Ausgabedatei geschrieben */


bool make_output (FILE *outputfile, char* buffer, def_list* definition)
{
  char *mobal_class, *pred_list, *arg_list;
  long pred_pos, arg_pos;
  def *p;
  string akt_pred, akt_arg, dbline, h;
  split_input_line(buffer,&mobal_class, &pred_list, &arg_list);
  if ((strcmp(pred_list,"") == 0) || (strcmp(arg_list,"") == 0))
    return FALSE;
  pred_pos = get_next_elem(akt_pred,pred_list,0,',');
  printf("Bearbeite Klasse : %s%s\r",mobal_class,"                    ");
  while(!(strcmp(akt_pred,"") == 0))
  {
```

```
    arg_pos = get_next_elem(akt_arg,arg_list,0,',');
    while(!(strcmp(akt_arg,"") == 0))
    {
      p = definition->first;
      dbline[0] = '\0';
      while(p != 0)
      {
            if(strcmp(p->var_type,"STRING") == 0)
              strcat(dbline,"'");
            if (strcmp(p->type,"MOBAL_CLASS") == 0)
              strcat(dbline,mobal_class);
            if (strcmp(p->type,"PREDICATE_NAME") == 0)
              strcat(dbline,akt_pred);
            if (strcmp(p->type,"ARGUMENT") == 0)
            {
              midstr(h,akt_arg,p->start_position,p->end_position);
              strcat(dbline,h);
            }
            if(strcmp(p->var_type,"STRING") == 0)
              strcat(dbline,"'");
            p=p->next;
            if (p != 0)
              strcat(dbline,",");
      }
      write_line(outputfile,dbline);
      arg_pos = get_next_elem(akt_arg,arg_list,arg_pos,',');
    }
    pred_pos = get_next_elem(akt_pred,pred_list,pred_pos,',');
  }
  return TRUE;
}

/* build_lines liest alle Zeilen der Eingabedatei, und ruft für jede nicht
   leere Zeile make_output auf. */

void build_lines(def_list* definition)
{
  FILE *inputfile;
  FILE *outputfile;
  char * buffer;
  long count = 0;
  buffer = (char *) new_mem(10000,1);
  inputfile = open_text_read(definition->input,1);
  outputfile = open_text_write(definition->output,1);
  while(!(feof(inputfile)))
  {
    buffer = read_line(inputfile,buffer,10000);
    if (buffer != NULL)
    {
      buffer = trim(buffer,buffer,' ');
      if (strcmp(buffer,"") != 0)
      {
        if (make_output(outputfile, buffer, definition) == 1)
            count ++;
      }
    }
  }
  fclose(inputfile);
  fclose(outputfile);
```

```
  printf("Anzahl bearbeiteter Klassen : %d\n",count);
}

int main(int arg_num, char* arg_list[])
{
  printf ("mobtodb -- 1997 -- Mark Siebert\n\n\n");
  def_list *definition;        /* Liste der Einträge der Konfigurationsdatei */
  definition = new_def_list();
  if (arg_num < 2)                       /* Test, ob eine Kofigurationsdatei vorhanden ist */
    exit_on_error("No configfile");
  read_config_file(arg_list[1],definition); /* Einlesen der Konfighurationsdatei */
  build_lines(definition);                /* Einlesen der Eingabedatei und bilden
                                       und bilden der Ausgabezeilen */
  return 0;
}
```

11.1.3 LIB

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

typedef unsigned char bool;
#define FALSE 0;
#define TRUE 1;

typedef char string[200];

char *emptystr(char * s)
{
  s[0] = '\0';
  return s;
}

/* Ausgabe einer Fehlermeldung und beenden des Programms */

void exit_on_error (char *error_text)
{
  printf("ERROR - %s\n",error_text);
  exit(1);
}

/* Anforderung eines Speicherbereichs der Größe size */

void *new_mem(long size, bool status)
{ void *h;
  h = malloc(size);
  if ((h == NULL) && (status == 1))
    exit_on_error("Not enough memory");
  return h;
}

/* open_text_read öffnet eine Textdatei zum Lesen. Abbruch mit einer Fehlermeldung
   wenn das Öffnen nicht möglich ist */

FILE* open_text_read (char* filename, bool status)
{ FILE* datei;
  datei = fopen(filename,"rt");
  if ((status) && (datei == NULL))
```

```
    exit_on_error(strcat("Cannot open file for input: ",filename));
  return datei;
}

/* read_line liest eine Zeile einer Textdatei in buffer, mit der maximalen Länge
   maxlen. Ausgabe einer Fehlermeldung und beenden des Programms, wenn
   die Zeile nicht in den Puffer passt */

char* read_line (FILE* datei, char* buffer, int maxlen)
{
  buffer = fgets(buffer,maxlen,datei);
  if (buffer == NULL)
    return NULL;
  if (sizeof(buffer) == maxlen-1)
    exit_on_error("Line too long");
  if (buffer[strlen(buffer)-1] == '\n')
    buffer[strlen(buffer)-1] = '\0';
  return buffer;
}

/* open_text_write öffnet eine Textdatei zum Schreiben. Abbruch mit einer Fehlermeldung
   wenn das Öffnen nicht möglich ist */

FILE* open_text_write (char* filename, bool status)
{ FILE* datei;
  datei = fopen(filename,"wt");
  if ((status) && (datei == NULL))
    exit_on_error(strcat("Cannot open file for output: ",filename));
  return datei;
}

/*write_line schreibt eine Zeile in eine Textdatei */

int write_line (FILE* datei, char* buffer)
{
  return fputs(strcat(buffer,"\n"),datei);
}

/* get_arg_n lieft das k-te Wort der Zeichenkette source, wobei die einzelnen Worte
   durch c voneinander getrennt werden. Ist das so erhaltene Wort größer als maxlen
   bricht das Programm mit einer Fehlermeldung ab */

char *get_arg_n(char * dest,char * source, char c, int k, int maxlen)
{
  int i = 0;
  int l = 0;
  int m = 0;
  if (source[0] != c)
    l=1;
  while (i<strlen(source))
  {
    if ((source[i] == c) && (source[i+1] != c))
    {
      l++;
      i++;
      continue;
    }
    if (l==k)
    {
```

```
      if (source[i] == c)
            break;
      dest[m] = source[i];
      m++;
      if (m == maxlen)
            exit_on_error("Argument to long");
    }
    i++;
  }
  dest[m] = '\0';
  return dest;
}

/*delchar löscht alle vorkommen des Zeichens c aus source */

char *delchar (char *dest, char *source, char c)
{
  int i = 0;
  int m = 0;
  while (i<strlen(source))
  {
    if (source[i] != c)
    {
      dest[m] = source[i];
      m++;
    }
    i++;
  }
  dest[m] = '\0';
  return dest;
}

/* string_to_number wandelt die Zeichenkette s in einer entsprechen Zahl um */
int string_to_number(char* s)
{
  int i = 0;
  int k = 0;
  while ((i< strlen(s)) && isdigit(s[i]))
    {
      k = k * 10;
      k = k + (s[i]-48);
      i++;
    }
  return k;
}

/* ltrim entfernt alle vorkommen von c auf der linken Seite von source */

char *ltrim(char *dest, char *source, char c)
{
  int i = 0;
  int m = 0;
  while(i < strlen(source))
  {
    if ( ((m == 0) && (source[i] != c)) || (m != 0))
    {
      dest[m] = source[i];
      m++;
    }
```

```
    i++;
  }
  dest[m] = '\0';
  return dest;
}
```

/* rteim entfernt alle vorkommen von c auf der rechten Seite von source */

```
char *rtrim(char *dest, char *source, char c)
{
  int i;
  int m = 0;
  i = strlen(source)-1;
  while (i>0)
  {
    if (((m == 0) && (source[i] != c)) || (m != 0))
    {
      dest[i] = source[i];
      if (m == 0)
             m = i;
    }
      i--;
  }
  dest[m+1] = '\0';
  return dest;
}
```

/* trim entfernt alle vorkommen von c auf der linken und rechten Seite
   von source */

```
char *trim (char * dest, char *source, char c)
{
  ltrim(dest,source,c);
  rtrim(dest,dest,c);
  return dest;
}
```

/* is_add test, ob die Zeichenkette nur aus Buchstaben besteht */

```
bool is_add(char *s)
{
  int i = 0;
  if (strlen(s) > 100)
      exit_on_error("String to long");
  while (i<strlen(s))
  {
    if (((s[i] < 97) || (s[i] > 122)) && (s[i] !='_') &&\
            ((s[i] < 48) || (s[i] > 59)))
      return FALSE;
    if  (((s[i] < 97) || (s[i] > 122)) && (i==0))
      return FALSE;
    i++;
  }
  return TRUE;
}
```

/* midstr liefert eine Teilzeichenkette von source, die mit dem Zeichen
   startpos beginnt, und mit dem Zeichen endpos endet */

```
char *midstr(char* dest,char*source,int startpos, int endpos)
{
  int i;
  startpos--;
  endpos --;
  if (startpos >= strlen(source)-1)
  {
    endpos = -1;
  }
  i = startpos;
  while ((i<=endpos) && (endpos != -1))
  {
    dest[i-startpos] = source[i];
    i++;
  }
  dest[i-startpos] = '\0';
  return dest;
}

/* get_next_elem liefert das nächste Wort, das sich nach der Position
   start in der Zeichenkette source ergibt. Dabei gilt c als Trennzeichen
   zwischen Wörten. Dest enthält dann dieses Wort, während die Funktion
   die Position des letzten Zeichens des Wortes in der Gesamtzeichenkette
   zurückliefert. */

long get_next_elem(char * dest, char* source, long start, char c)
{
  int i;
  int k = 0;
  i = start;
  while (i<strlen(source))
  {
    if(source[i] != c)
    {
      dest[k] = source[i];
      k++;
    }
    else
      break;
    i++;
  }
  dest[k]='\0';
  if (source[i] == '\0')
    return i;
  else
    return i+1;
}

/* instr suchen nach dem ersten Vorkommen des Zeichenkette search in s
   und liefert die Position der ersten gemeinsamen Zeichens zurück */

int instr(int begin, char* s, char* search)
{
  int i;
  int k = 0;
  i = begin-1;
  while (i<strlen(s))
  {
    if (s[i+k] == search[k])
```

```
    {
      if (k==strlen(search)-1)
             return i+1;
      k++;
    }
    else
    {
      k=0;
      i++;
    }
  }
  return 0;
}
```

11.2 Einzelteilgruppen

g00001 00001 Motor
g00001 00201 Kundendienst Motor
g00002 01001 Zylinderkurbelgehaeuse
g00002 01002 Zylinderbohrung Kurbelgehaeuse
g00002 01015 Stiftschraube Zylinderkurbelgehaeuse/Steuergehaeuse
g00002 01018 Gewindebohrung Zylinderkurbelgehaeuse
g00002 01048 Dichtung Steuergehaeuse/Kurbelgehaeuse
g00002 18108 Dichtung Oelfilter/Kurbelgehaeuse
g00002 18214 Dichtung Oelkuehler/Kurbelgehaeuse
g00002 18215 Schraube Oelkuehler Kurbelgehaeuse
g00002 20009 Dichtung Kuehlmittelpumpe/Kurbelgehaeuse
g00003 01004 Zwischenflansch Zylinderkurbelgehaeuse/Getriebe
g00003 01005 Schraube Zwischenflansch
g00004 01006 Kurbelwellenlagerdeckel
g00004 01007 Schraube Kurbelwellenlagerdeckel
g00004 01011 Schraube Lagerdeckel seitlich
g00005 01013 Abschlussdeckel hinten
g00005 01014 Schraube Abschlussdeckel hinten
g00006 01041 Zylinderdeckel seitlich
g00006 01042 Dichtung Zylinderdeckel
g00006 01043 Schraube Zylinderdeckel
g00007 01044 Verschlussdeckel vorn
g00007 01045 Dichtung Verschlussdeckel vorn
g00007 01046 Schraube Verschlussdeckel
g00008 01015 Stiftschraube Zylinderkurbelgehaeuse/Steuergehaeuse
g00008 01047 Steuergehaeuse
g00008 01048 Dichtung Steuergehaeuse/Kurbelgehaeuse
g00008 01049 Schraube Steuergehaeuse
g00008 01068 Wellendichtring Nebenantrieb Steuergehaeuse
g00009 01050 Steuergehaeusedeckel
g00009 01051 Dichtflaeche/Dichtung Steuergehaeusedeckel
g00009 01052 Schraube Steuergehaeusedeckel
g00009 01063 Verschlussschraube Steuergehaeusedeckel
g00010 01053 Verschlussdeckel Steuergehaeuse
g00010 01054 Dichtung Verschlussdeckel Steuergehaeuse
g00010 01055 Schraube Steuergehaeuseverschlussdeckel
g00010 01066 Dichtung Verschlussdeckel Steuergehaeuse Nebenantrieb
g00011 01060 Stoesselkammerdeckel
g00011 01061 Dichtung Stoesselkammerdeckel
g00011 01062 Schraube Stoesselkammerdeckel
g00012 01082 Motorbelueftungsgehaeuse/Kombitraeger
g00012 01083 Dichtung Motorbelueftungsgehaeuse
g00013 01009 Abdeckung Oelwanne hinten
g00013 01101 Oelwanne Motor
g00013 01102 Schraube Oelwanne
g00013 01103 Oelwanne Unterteil
g00013 01104 Dichtung Oelwanne
g00013 01105 Dichtbeilage Oelwanne/Hauptlagerdeckel
g00013 01108 Gewindeeinsatz Verschlussschraube Oelwanne
g00013 18325 Oelstandschalter/Geber Oelwanne
g00013 18328 Dichtbeilage Oelstandgeber/Schalter - Oelwanne
g00014 01107 Verschlussschraube Oelablass
g00014 01109 Dichtung Verschlussschraube Oelablass
g00015 01110 O-Ring Fuehrungsrohr Oelmessstab
g00015 01111 Fuehrungsrohr Oelmessstab
g00015 01114 Halterung Fuehrungsrohr Oelmessstab
g00015 01115 Flansch Fuehrungsrohr Oelmessstab
g00016 01110 O-Ring Fuehrungsrohr Oelmessstab
g00016 01111 Fuehrungsrohr Oelmessstab
g00016 01112 Oelmessstab Motor
g00016 01113 Gummikappe Oelmessstab
g00016 01114 Halterung Fuehrungsrohr Oelmessstab
g00016 01115 Flansch Fuehrungsrohr Oelmessstab
g00017 01116 Oelsaugrohr Oelabsaugung
g00017 01117 Halterung Oelsaugrohr
g00018 01201 Zylinderkopf
g00018 01202 Zylinderkopfdichtung
g00018 01203 Zylinderkopfschraube
g00018 01207 Kuehlmittelverteiler Zylinderkopf
g00018 01215 Anschlussstutzen Zylinderkopf
g00018 01222 Verschlussschraube Oelkanal Zylinderkopf
g00018 01225 Dichtring Flansch Zylinderkopf vorn
g00018 01241 Zylinderkopf rechts
g00018 01242 Zylinderkopfdichtung rechts
g00018 01243 Zylinderkopf links
g00018 01244 Zylinderkopfdichtung links
g00019 01227 Verschlussdeckel vorn
g00019 01228 Radialdichtring Verschlussdeckel vorn
g00019 01229 Untere Abdichtung Verschlussdeckel vorn
g00020 01231 Schutzhuelse
g00020 01232 Dichtring Schutzhuelse
g00021 01241 Zylinderkopf rechts
g00021 01242 Zylinderkopfdichtung rechts
g00021 01243 Zylinderkopf links
g00021 01244 Zylinderkopfdichtung links
g00023 01301 Zylinderkopfhaube
g00023 01302 Dichtung Zylinderkopfhaube
g00023 01303 Buegel Zylinderkopfhaube
g00023 01304 Schraube/Mutter Zylinderkopfhaube
g00023 01312 Abdeckung Zylinderkopfhaube
g00023 01321 Zylinderkopfhaube rechts
g00023 01322 Dichtung Zylinderkopfhaube rechts
g00023 01323 Zylinderkopfhaube links
g00023 01324 Dichtung Zylinderkopfhaube links
g00024 01306 Verschlussdeckel Oeleinfuellung
g00024 01307 Dichtung Verschlussdeckel Oeleinfuellung
g00025 01308 Oelabscheider
g00025 01309 Abdichtring Oelabscheider
g00026 01321 Zylinderkopfhaube rechts
g00026 01322 Dichtung Zylinderkopfhaube rechts
g00027 01323 Zylinderkopfhaube links
g00027 01324 Dichtung Zylinderkopfhaube links
g00028 01402 Oeleinfuellstutzen
g00028 01406 Halter Oeleinfuellstutzen
g00028 01407 Schraube Oeleinfuellstutzen
g00029 01411 Betriebsstundenzaehler
g00029 01412 Dichtung Betriebsstundenzaehler
g00029 01413 Antriebswelle Betriebsstundenzaehler
g00029 01414 Zaehlwerk Betriebsstundenzaehler
g00029 01415 Traeger Betriebsstundenzaehler
g00029 01416 Entlueftungsschlauch Betriebsstundenzaehler
g00030 01501 Nockenwellengehaeuse
g00030 01502 Dichtflaeche/Dichtung Nockenwellengehaeuse
g00030 01504 Verschlussdeckel Nockenwellengehaeuse
g00031 03003 Auswuchtscheibe/Gegengewicht
g00031 03004 Schraube Auswuchtscheibe/Nabe/Gegengewicht
g00032 03005 Schwingungsdaempfer Kurbelwelle
g00032 03006 Schraube Schwingungsdaempfer
g00033 03203 Pleuelschraube
g00033 03204 Mutter Pleuelschraube
g00034 03102 Pleuellager
g00034 03205 Pleuellagerdeckel
g00035 03304 Kolbenbolzen
g00035 03305 Sprengring Kolbenbolzen
g00035 03307 Buchse Kolbenbolzen
g00036 03311 Oelspritzduese Kolbenkuehlung
g00036 03312 Verschlussschraube Oelspritzduese Kolbenkuehlung
g00037 05003 Nockenwellenlager
g00037 05004 Schraube/Mutter Nockenwellenlager
g00038 05021 Nockenwelle rechts
g00038 05022 Nockenwelle/Schwinghebel rechts
g00039 05023 Nockenwelle links
g00039 05024 Nockenwelle/Schwinghebel links
g00040 05031 Nockenwelle Auslass
g00040 05032 Axial-Fixierung Nockenwelle Auslass
g00040 05033 Pass-Stift Nockenwelle Auslass
g00040 05034 Nockenwellenrad Nockenwelle Auslass
g00040 05035 Zahnrad Nockenwelle Auslass
g00041 05036 Zahnrad Nockenwelle Einlass
g00041 05037 Nockenwelle Einlass
g00041 05038 Axial-Fixierung Nockenwelle Einlass
g00041 05039 Pass-Stift Nockenwelle Einlass
g00042 05040 Steuerkolben Nockenwellenverstellung Einlass

g00042 05041 Feder Steuerkolben Nockenwellenverstellung Einlass
g00042 05060 Steuerkolben Nockenwelle Einlass links
g00042 05061 Feder Steuerkolben Nockenwelle Einlass links
g00042 05080 Steuerkolben Nockenwelle Einlass rechts
g00042 05081 Feder Steuerkolben Nockenwelle Einlass rechts
g00043 05044 Kettenrad Nockenwellenverstellung
g00043 05045 Deckel Kettenrad Nockenwellenverstellung
g00044 05040 Steuerkolben Nockenwellenverstellung Einlass
g00044 05041 Feder Steuerkolben Nockenwellenverstellung Einlass
g00044 05042 Flanschwelle Nockenwellenverstellung
g00044 05043 Stellkolben Nockenwellenverstellung
g00044 05044 Kettenrad Nockenwellenverstellung
g00044 05045 Deckel Kettenrad Nockenwellenverstellung
g00044 05046 Anker Nockenwellenverstellung
g00044 05047 Magnet Nockenwellenverstellung
g00044 05048 Positionsgeber Nockenwellenverstellung
g00044 05049 Befestigungsmutter Nockenwellenverstellung
g00044 05050 Nockenwellenverstellung
g00044 05060 Steuerkolben Nockenwelle Einlass links
g00044 05061 Feder Steuerkolben Nockenwelle Einlass links
g00044 05062 Flanschwelle Nockenwellenverstellung links
g00044 05063 Stellkolben Nockenwellenverstellung links
g00044 05064 Kettenrad Nockenwellenverstellung links
g00044 05065 Deckel Kettenrad Nockenwellenverstellung links
g00044 05066 Anker Nockenwellenverstellung links
g00044 05067 Magnet Nockenwellenverstellung links
g00044 05068 Kegelfeder Nockenwellenverstellung links
g00044 05069 Befestigungsmutter Nockenwellenverstellung links
g00044 05070 Nockenwellenverstellung links
g00044 05080 Steuerkolben Nockenwelle Einlass rechts
g00044 05081 Feder Steuerkolben Nockenwelle Einlass rechts
g00044 05082 Flanschwelle Nockenwellenverstellung rechts
g00044 05083 Stellkolben Nockenwellenverstellung rechts
g00044 05084 Kettenrad Nockenwellenverstellung rechts
g00044 05085 Deckel Kettenrad Nockenwellenverstellung rechts
g00044 05086 Anker Nockenwellenverstellung rechts
g00044 05087 Magnet Nockenwellenverstellung rechts
g00044 05088 Kegelfeder Nockenwellenverstellung rechts
g00044 05089 Befestigungsmutter Nockenwellenverstellung rechts
g00044 05090 Nockenwellenverstellung rechts
g00045 05051 Nockenwelle Auslass links
g00045 05053 Pass-Stift Nockenwelle Auslass links
g00045 05054 Nockenwellenrad Nockenwelle Auslass links
g00046 05057 Nockenwelle Einlass links
g00046 05059 Pass-Stift Nockenwelle Einlass links
g00047 05060 Steuerkolben Nockenwelle Einlass links
g00047 05061 Feder Steuerkolben Nockenwelle Einlass links
g00048 05062 Flanschwelle Nockenwellenverstellung links
g00048 05063 Stellkolben Nockenwellenverstellung links
g00048 05064 Kettenrad Nockenwellenverstellung links
g00048 05065 Deckel Kettenrad Nockenwellenverstellung links
g00048 05066 Anker Nockenwellenverstellung links
g00048 05067 Magnet Nockenwellenverstellung links
g00048 05068 Kegelfeder Nockenwellenverstellung links
g00048 05069 Befestigungsmutter Nockenwellenverstellung links
g00048 05070 Nockenwellenverstellung links
g00049 05071 Nockenwelle Auslass rechts
g00049 05073 Pass-Stift Nockenwelle Auslass rechts
g00049 05074 Nockenwellenrad Nockenwelle Auslass rechts
g00050 05077 Nockenwelle Einlass rechts
g00050 05079 Pass-Stift Nockenwelle Einlass rechts
g00051 05080 Steuerkolben Nockenwelle Einlass rechts
g00051 05081 Feder Steuerkolben Nockenwelle Einlass rechts
g00052 05082 Flanschwelle Nockenwellenverstellung rechts
g00052 05083 Stellkolben Nockenwellenverstellung rechts
g00052 05084 Kettenrad Nockenwellenverstellung rechts
g00052 05085 Deckel Kettenrad Nockenwellenverstellung rechts
g00052 05086 Anker Nockenwellenverstellung rechts
g00052 05087 Magnet Nockenwellenverstellung rechts
g00052 05088 Kegelfeder Nockenwellenverstellung rechts
g00052 05089 Befestigungsmutter Nockenwellenverstellung rechts
g00052 05090 Nockenwellenverstellung rechts
g00053 05101 Nockenwellenrad
g00053 05102 Schraube Nockenwellenrad
g00054 05103 Zwischenrad Steuerkette
g00054 05104 Zwischenradwelle/Schraubenrad
g00054 05105 Schraube Zwischenrad
g00054 05112 Lagerbuchse Zwischenradwelle
g00055 05106 Umlenkrad Steuerkette
g00055 05113 Lagerbuchse Umlenkrad
g00056 05121 Gleitschiene Steuerkette
g00056 05122 Lagerbolzen Gleitschiene
g00057 05123 Spannschiene Steuerkette
g00057 05124 Lagerbolzen Spannschiene
g00057 05125 Belag Spannschiene
g00058 05131 Kettenspanner
g00058 05132 Dichtung Kettenspanner
g00059 05211 Ventilfeder
g00059 05212 Innere Ventilfeder
g00059 05213 Aeussere Ventilfeder
g00059 05214 Ventilfederteller
g00060 05301 Schwinghebel/Kipphebel
g00060 05302 Buchse Schwinghebel/Kipphebel
g00060 05303 Schwing-/Kipphebelachse
g00060 05304 Schwing-/Kipphebellagerbock
g00060 05305 Schraube Schwing-/Kipphebellagerbock
g00060 05306 Federklammer/Sicherungsring Kipphebelachse
g00060 05307 Passscheibe Kipphebelachse
g00060 05321 Spannfeder Kipphebel
g00060 05325 Verschlussschraube Oelkanal Kipphebellagerbock
g00061 05303 Schwing-/Kipphebelachse
g00061 05306 Federklammer/Sicherungsring Kipphebelachse
g00061 05307 Passscheibe Kipphebelachse
g00062 05304 Schwing-/Kipphebellagerbock
g00062 05305 Schraube Schwing-/Kipphebellagerbock
g00062 05325 Verschlussschraube Oelkanal Kipphebellagerbock
g00063 05315 Einstellschraube Kugelbolzen
g00063 05317 Kontermutter Einstellschraube
g00064 05322 Ringfeder
g00064 05323 Verbindungsnippel Feder
g00065 07001 Einspritzpumpe
g00065 07002 Gehaeuse Einspritzpumpe
g00065 07004 Ueberstroemventil/Drossel Einspritzpumpe
g00065 07006 Druckventilhalter Druckleitung
g00065 07007 Druckventil Einspritzpumpe
g00065 07008 Dichtung/Dichtring Einspritzpumpe
g00065 07009 Schraube Einspritzpumpe
g00065 07010 O-Ring Oelversorgung Einspritzpumpe
g00065 07011 Einspritzpumpenregler
g00065 07012 Gehaeuse Einspritzpumpenregler
g00065 07013 Regelstange Einspritzpumpe
g00065 07014 Regulierwelle Einspritzpumpenregler
g00065 07015 Verstellhebel Einspritzpumpenregler
g00065 07016 Anschlag Verstellhebel Einspritzpumpenregler
g00065 07018 Unterdruckdose/Stoppdose Einspritzpumpe
g00065 07020 Einspritzpumpenelement
g00065 07023 Nockenwelle Einspritzpumpe
g00065 07024 Keil Nockenwelle Einspritzpumpe
g00065 07027 Lager Nockenwelle Einspritzpumpe
g00065 07041 Kraftstoffpumpe Einspritzpumpe
g00065 07042 Saugventil/Druckventil Kraftstoffpumpe Einspritzpumpe
g00065 07044 Handfoerderpumpe Einspritzpumpe
g00065 07046 Dichtung Federraumdeckel Einspritzpumpe
g00065 07065 Regelweggeber Einspritzpumpe
g00065 07161 Leitungssatz Einspritzpumpe
g00065 07162 Mengenstellwerk Einspritzpumpe
g00065 07513 Rueckzugfeder Regulierung Einspritzpumpe
g00065 18123 Schmieroelleitung Einspritzpumpe
g00066 07006 Druckventilhalter Druckleitung
g00066 07007 Druckventil Einspritzpumpe
g00067 07011 Einspritzpumpenregler
g00067 07012 Gehaeuse Einspritzpumpenregler
g00067 07013 Regelstange Einspritzpumpe
g00067 07014 Regulierwelle Einspritzpumpenregler
g00067 07015 Verstellhebel Einspritzpumpenregler
g00067 07016 Anschlag Verstellhebel Einspritzpumpenregler
g00068 07015 Verstellhebel Einspritzpumpenregler
g00068 07016 Anschlag Verstellhebel Einspritzpumpenregler

g00069 07023 Nockenwelle Einspritzpumpe
g00069 07024 Keil Nockenwelle Einspritzpumpe
g00069 07027 Lager Nockenwelle Einspritzpumpe
g00070 07041 Kraftstoffpumpe Einspritzpumpe
g00070 07042 Saugventil/Druckventil Kraftstoffpumpe Einspritzpumpe
g00071 07102 Einspritzpumpenantriebsgehaeuse/Traeger
g00071 07104 Schraube Antriebsgehaeuse
g00072 07111 Spritzversteller
g00072 07115 Kurvenscheibe Spritzversteller
g00073 07122 Kupplungshaelfte
g00073 07123 Schraube Kupplungshaelfte
g00074 07151 Verteilereinspritzpumpe
g00074 07155 Ruecklaufventil Verteilereinspritzpumpe
g00074 07156 Zulaufventil Verteilereinspritzpumpe
g00074 07157 Druckregelventil Verteilereinspritzpumpe
g00074 07158 Verstellventil Verteilereinspritzpumpe
g00074 07160 Abgleichwiderstand Verteilereinspritzpumpe
g00075 07201 Einspritzventil/Einspritzduese
g00075 07205 Isolierhuelse Einspritzventil
g00076 07202 Duesenhalter
g00076 07203 Druckmutter Duesenhalter
g00076 07263 Verschluss-Stopfen Duesenhalter
g00077 07233 Kraftstoffmengenteiler
g00077 07236 Dichtring Mengenteiler
g00078 07234 Luftmengenmesser
g00078 07238 Geber Luftmengenmesser
g00079 07230 Gemischregler
g00079 07269 Gummilager Gemischregler
g00080 07209 Halter/Schlauchschelle Kraftstoffleitung
g00080 07215 Kraftstoffleitung Vorlauf
g00080 07217 Kraftstoffleitung Ruecklauf
g00080 07228 Kraftstoffleitung Pumpenpaket
g00080 07245 Kraftstoffleitung Pumpenpaket Fahrzeugboden
g00080 07247 Dreifachschelle Kraftstoffleitung
g00080 07264 Kraftstoffleitung Einspritzpumpe Vorlauf
g00080 07266 Kraftstoffleitungen
g00080 07270 Dichtring Kraftstoffleitung
g00081 07275 Waermetauscher Kraftstoffvorwaermung
g00081 07278 Thermostat Kraftstoffvorwaermung
g00081 07279 Halter Kraftstoffvorwaermung
g00081 07280 Dichtung Kraftstoffvorwaermung
g00081 07281 Vorwaermgeraet/Kraftstoffvorwaermung
g00081 07282 Schlauch/Leitung Kraftstoffvorwaermung
g00082 07291 Einspritzventil LH
g00082 07292 Halter Einspritzventil LH
g00083 07293 Kraftstoff-Verteilerrohr LH
g00083 07294 Sieb Kraftstoff-Verteilerrohr LH
g00083 07295 Ventileinsatz Kraftstoff-Verteilerrohr LH
g00084 07301 Klappenstutzen
g00084 07302 Dichtung Klappenstutzen
g00085 07312 Zusatzluftschieber
g00085 07320 Dichtung Zusatzluftschieber
g00086 07406 Schwimmer
g00086 07407 Schwimmernadelventil/Ventilsitz
g00087 07410 Vergaserdeckel
g00087 07411 Dichtung Vergaserdeckel
g00088 07434 Vergaserflansch
g00088 07435 Schraube Vergaserflansch
g00089 07501 Motorlauf-Beanstandung
g00089 07505 Motorlauf-Beanstandung kalter Motor
g00089 07506 Motorlauf-Beanstandung warmer Motor
g00090 07503 Umlenkhebel Regulierung Motor
g00090 07504 Halter Umlenkhebel
g00091 07511 Lagerbock Regulierwelle
g00091 07512 Regulierwelle
g00092 07517 Regulierhebel
g00092 07518 Schlepphebel Regulierung
g00093 07602 Vergaserdeckel
g00093 07606 Dichtung Vergaserdeckel
g00093 07607 Schrauben Vergaserdeckel
g00094 07603 Schwimmernadelventil/Ventilsitz
g00094 07604 Schwimmer
g00095 07612 Drosselklappenpotentiometer
g00095 07613 Kupplung Drosselklappenpotentiometer
g00096 01031 Verschlussflansch Kraftstoffpumpe
g00096 09001 Kraftstoffpumpe
g00096 09003 Dichtung Kraftstoffpumpe
g00096 09004 Dichtflansch Kraftstoffpumpe
g00096 09006 Rueckschlagventil Kraftstoffpumpe
g00096 09007 Schutzhaube Kraftstoffpumpe
g00096 09010 Kraftstoffpumpe Vorfoerderung
g00096 09011 Handpumpe zur Kraftstoffpumpe
g00096 09012 Druck-/Saugventil Kraftstoffpumpe
g00096 09013 Halter Handpumpe zur Kraftstoffpumpe
g00096 09018 Dichtring Kraftstoffpumpe
g00096 09021 Kraftstoffhandpumpe Filter
g00097 09011 Handpumpe zur Kraftstoffpumpe
g00097 09013 Halter Handpumpe zur Kraftstoffpumpe
g00097 09021 Kraftstoffhandpumpe Filter
g00098 09101 Luftfilter
g00098 09102 Luftfiltereinsatz
g00098 09103 Dichtring Luftfilter
g00098 09104 Gummilager/Schraube/Mutter Luftfilter
g00098 09106 Dichtring unten
g00098 09135 Halter Luftfilter
g00098 09136 Schnellverschluss Luftfilter
g00099 09125 Saugrohr/Luftfuehrungsrohr
g00099 09126 Halter Saugrohr
g00100 09131 Oelbadluftfilter
g00100 09132 Einsatz Oelbadluftfilter
g00100 09133 Dichtring Oelbadluftfilter
g00101 09138 Ansaugkamin
g00101 09139 Halter Ansaugkamin
g00101 09140 Regenkappe Ansaugkamin
g00102 09151 Wartungsanzeige Trockenluftfilter
g00102 09156 Leitung Wartungsanzeige Luftfilter
g00102 13061 Dichtring Lagerdeckel/Flansch
g00103 09175 Luftmassenmesser LH
g00103 09176 Zwischenstueck Luftmassenmesser LH
g00104 09201 Kraftstoff-Filter
g00104 09202 Kraftstoff-Filtereinsatz
g00104 09203 Dichtring Kraftstoff-Filter
g00104 09204 Schraube Kraftstoff-Filter
g00104 09205 Schraube Kraftstoff-Filter
g00104 09206 Halter Kraftstoff-Filter
g00104 09207 Kraftstoff-Feinfilter
g00104 09208 Kraftstoff-Vorfilter
g00104 09209 Entlueftungsschraube Kraftstoff-Filter
g00104 09210 Schraube Halter
g00104 09212 Deckel Kraftstoff-Filter
g00104 09213 Gehaeuse Kraftstoff-Filter
g00105 09206 Halter Kraftstoff-Filter
g00105 09210 Schraube Halter
g00106 09220 Kraftstoff-Vorfilter mit Wasserabscheider
g00106 09221 Kraftstoff-Filtereinatz, Wasserabscheider
g00107 09301 Abgasturbolader
g00107 09302 Dichtbeilage Abgasturbolader/Auspuffkruemmer
g00107 09303 Schraube/Mutter Lader
g00107 09305 Dichtung Lader/Flanschkruemmer
g00107 09322 Abgasrohr Abgasturbolader
g00107 09323 Luftansaugrohr Turbolader
g00107 09324 Muffe Luftansaugrohr Abgasturbolader
g00107 09340 Mischgehaeuse Abgasturbolader
g00107 09373 Kolbenring Turbolader
g00107 09374 Lagerbuchse Turbolader
g00107 09375 Dichtungsbuchse Turbolader
g00107 14107 Flansch Abgasturbolader
g00107 14108 Metallschlauch Abgasturbolader
g00107 14109 Rohr Zwischenstueck Abgasturbolader
g00107 14110 Flansch Auspuff Abgasturbolader
g00107 14111 Schlauchschelle Abgasturbolader
g00107 14123 Stuetze Auspuffkruemmer/Abgasturbolader
g00107 14327 Umluftventil Abgasturbolader
g00107 18124 Oelzulaufleitung Abgasturbolader
g00108 09310 Anschlussstueck Ladeluftleitung
g00108 09311 Dichtring Anschlussstueck
g00109 09312 Ansaugleitung
g00109 09313 Dichtring Ansaugleitung

g00110 09318 Ladeluftrohr
g00110 09319 Halter/Stuetze Ladeluftrohr
g00111 09323 Luftansaugrohr Turbolader
g00111 09324 Muffe Luftansaugrohr Abgasturbolader
g00112 09503 Duesenhalter Flammstartanlage
g00112 09504 Dichtung Duesenhalter
g00113 09506 Flansch Flammgluehkerze
g00113 09507 Dichtung Flansch Flammgluehkerze
g00114 13001 Luftpresser
g00114 13002 Zylinderfussdichtung Luftpresser
g00114 13003 Schraube/Mutter Zylinder Luftpresser
g00114 13004 Zylinderkopfdichtung Luftpresser
g00114 13005 Schraube/Mutter Zylinderkopf Luftpresser
g00114 13006 Druckventil Luftpresser
g00114 13007 Ventilscheibe Luftpresser
g00114 13008 Saugventil Luftpresser
g00114 13009 Schraube Luftpresser
g00114 13010 Lagerdeckel/Flansch Luftpresser
g00114 13011 Riemenscheibe Luftpresser
g00114 13012 Keilriemen Luftpresser
g00114 13013 Antriebsrad Luftpresser
g00114 13014 Kurbelwellenlager Luftpresser
g00114 13015 Pleuel Luftpresser
g00114 13016 Pleuellager Luftpresser
g00114 13017 Pleuelschraube Luftpresser
g00114 13018 Zylinderkopf Luftpresser
g00114 13019 Kurbelwelle Luftpresser
g00114 13020 Luftpressergehaeuse
g00114 13021 Traeger Luftpresser
g00114 13022 Schraube Traeger Luftpresser
g00114 13023 Zwischenflansch Luftpresser/Lenkhelfpumpe
g00114 13024 Kolben Luftpresser
g00114 13025 Sprengring Kolben Luftpresser
g00114 13026 Kolbenring Luftpresser
g00114 13027 Kolbenbolzen Luftpresser
g00114 13028 Pleuelbuchse Luftpresser
g00114 13031 Saugleitung/Schlauch Luftpresser
g00114 13033 Winkelanschluss Luftpresser
g00114 13034 Abdichtring Kurbelwelle Luftpresser
g00114 13035 Zylinder/Laufbuchse Luftpresser
g00114 13037 Schraube Antrieb Luftpresser
g00114 13040 Ventilfeder Luftpresser
g00114 13041 Anschlussschraube Schmieroelleitung Luftpresser
g00114 13045 Halter Luftpresser oben
g00114 13046 Stuetzstange Halter Luftpresser oben
g00114 13047 Schraube Luftpresserhalter oben
g00114 13050 Spannrolle Keilriemen Luftpresser
g00114 13051 Spannvorrichtung Keilriemen
g00114 13060 Luftfilter Luftpresser
g00114 13063 Zwischenstueck/Leitung Oelruecklauf Luftpresser
g00114 13065 Befestigungsflansch Lager Luftpresser
g00114 13066 Schraube Deckel/Flansch Luftpresser
g00114 18122 Schmieroelleitung Luftpresser
g00114 18136 Ruecklaufleitung Luftpresser
g00114 20029 Kuehlmittelruecklaufleitung Luftpresser
g00114 20030 Kuehlmittelvorlaufleitung Luftpresser
g00114 50213 Kuehlmittelschlauch Luftpresser
g00115 13021 Traeger Luftpresser
g00115 13022 Schraube Traeger Luftpresser
g00116 13024 Kolben Luftpresser
g00116 13025 Sprengring Kolben Luftpresser
g00117 13019 Kurbelwelle Luftpresser
g00117 13034 Abdichtring Kurbelwelle Luftpresser
g00118 13045 Halter Luftpresser oben
g00118 13046 Stuetzstange Halter Luftpresser oben
g00118 13047 Schraube Luftpresserhalter oben
g00119 13012 Keilriemen Luftpresser
g00119 13050 Spannrolle Keilriemen Luftpresser
g00119 13051 Spannvorrichtung Keilriemen
g00120 13010 Lagerdeckel/Flansch Luftpresser
g00120 13066 Schraube Deckel/Flansch Luftpresser
g00121 14001 Saugrohr
g00121 14002 Saugrohr-Oberteil
g00121 14003 Saugrohr-Unterteil
g00121 14004 Dichtung Saugrohr
g00121 14005 Stiftschraube/Schraube/Mutter Saugrohr
g00121 14017 Vorwaermdeckel Saugrohr
g00121 14019 Heizgitter Saugrohr
g00122 14007 Saugkruemmer
g00122 14009 Schraube/Mutter Saugkruemmer
g00123 14101 Auspuffkruemmer
g00123 14102 Dichtung Auspuffkruemmer
g00123 14103 Stiftschraube/Schraube/Mutter Auspuffkruemmer
g00123 14113 Abschirmblech Auspuffkruemmer
g00123 14123 Stuetze Auspuffkruemmer/Abgasturbolader
g00123 14124 Dehnungskoerper Auspuffkruemmer
g00123 14131 Auspuffkruemmer vorn
g00123 14132 Auspuffkruemmer hinten
g00123 14135 Auspuffkruemmer links
g00123 14136 Auspuffkruemmer rechts
g00123 14395 Auspuffrohr zwischen Auspuffkruemmer und Partikel/Abgasfiltera
g00124 14202 Drosselklappe Motorbremse
g00124 14203 Welle Drosselklappe Motorbremse
g00124 14209 Buchse Drosselklappenwelle
g00125 14201 Motorbremse
g00125 14202 Drosselklappe Motorbremse
g00125 14203 Welle Drosselklappe Motorbremse
g00125 14204 Motorbremszylinder
g00125 14205 Winkel-/Regulierhebel Motorbremse
g00125 14206 Gestaenge Motorbremse
g00125 14207 Kugelkopf/Pfanne Motorbremse
g00125 14208 Klappenstutzen Motorbremse
g00125 14209 Buchse Drosselklappenwelle
g00125 14210 Anschlag Motorbremse
g00125 14216 Halter Motorbremszylinder
g00125 14245 Tastschalter Motorbremse
g00125 14246 Kontroll-Leuchte Motorbremse
g00125 14247 Drehzahlschalter Motorbremse
g00126 14242 Abdichtring Stoessel Konstantdrossel
g00126 14243 Stoessel Konstantdrossel
g00127 14301 Luftpumpe Abgas
g00127 14307 Luftfiltereinsatz Luftpumpe Abgas
g00127 14319 Elektromagnetische Kupplung Luftpumpe Abgas
g00127 14320 Keilriemen Luftpumpe Abgas
g00127 14321 Riemenscheibe Luftpumpe Abgas
g00127 14322 Halter Luftpumpe Abgas
g00127 14328 Flansch Kupplung Luftpumpe Abgas
g00128 14314 Dichtung Abgasrueckfuehrungsventil
g00128 14315 Abgasrueckfuehrventil
g00129 14334 Unterdruckdose Abgasklappe
g00129 14335 Abgasklappe
g00129 14336 Dichtung Abgasklappe
g00130 14344 Spannungsrelais Luftpumpe
g00130 14349 Elektrische Luftpumpe
g00130 14350 Gummipuffer Luftpumpe
g00131 14354 Partikel-/Abgasfilteranlage
g00131 14356 Magnetventil Partikel-/Abgasfilteranlage
g00132 14361 Unterdruckdose Drucksteuerklappe
g00132 14362 Drucksteuerklappe
g00132 14363 Betaetigung Drucksteuerklappe
g00133 14354 Partikel-/Abgasfilteranlage
g00133 14356 Magnetventil Partikel-/Abgasfilteranlage
g00133 14370 Partikel-/Abgasfilter
g00133 14371 Kontroll-Leuchte Partikel-/Abgasfilteranlage
g00133 14372 Druckschalter Partikel-/Abgasfilteranlage
g00133 14373 Druckschalter Kontrolle Partikel-/Abgasfilteranlage
g00133 14375 Pneumatisches Absperrventil Partikel-/Abgasfilteranlage
g00133 14376 Steuergeraet Partikel-/Abgasfilteranlage
g00133 14377 Elektrische Leitung Partikel-/Abgasfilteranlage
g00133 14383 Einspritzduese Partikel-/Abgasfilter
g00133 14385 Oelleitung Partikel-/Abgasfilteranlage
g00133 14395 Auspuffrohr zwischen Auspuffkruemmer und Partikel/Abgasfiltera
g00134 14372 Druckschalter Partikel-/Abgasfilteranlage
g00134 14373 Druckschalter Kontrolle Partikel-/Abgasfilteranlage
g00135 14390 Dosiereinrichtung
g00135 14391 Dosiereinrichtung Kugelventil

g00136 14393 Muffe Metallschlauch am Auspuffrohr
g00136 14394 Metallschlauch vorn am Auspuffrohr
g00137 14401 Verschlussdeckel Vorratsbehaelter
g00137 14402 Dichtring Verschlussdeckel Vorratsbehaelter
g00138 14421 Vollstrombrenneranlage
g00138 14422 Partikelfilter Vollstrombrenner
g00138 14424 Kontrollampe Vollstrombrenner
g00138 14425 Elektrische Leitung Vollstrombrenner
g00138 14427 Drucksensor Vollstrombrenner
g00138 14428 Steuergeraet Vollstrombrenner
g00138 14430 Russbrenner
g00138 14431 Ueberhitzungsschalter
g00138 14433 Regelfuehler
g00138 14434 Gluehkerze Vollstrombrenner
g00138 14435 Brennkammer
g00138 14436 Flammfuehler Vollstrombrenner
g00138 14437 Absperrventil Vollstrombrenner
g00138 14439 Magnetventil Vollstrombrenner
g00138 14440 Dosierpumpeneinheit Vollstrombrenner
g00138 14443 Druckluftleitung Vollstrombrenner
g00138 14445 Wasserabscheider Vollstrombrenner
g00138 14446 Gluehkerzenrelais Vollstrombrenner
g00138 14448 Stromregelrelais Vollstrombrenner
g00138 14449 Vorwiderstand Vollstrombrenner
g00138 14451 Parallelwiderstand Vollstrombrenner
g00138 14452 Brennstofftank Vollstrombrenner
g00138 14453 Brennstoffilter Vollstrombrenner
g00138 14454 Brennstoffpumpe Vollstrombrenner
g00138 14456 Druckminderer Vollstrombrenner
g00138 14457 Brennstoffleitung Vollstrombrenner
g00139 14441 Halter Luftpumpe Vollstrombrenner
g00139 14442 Luftpumpe Vollstrombrenner
g00140 15001 Starter
g00140 15002 Rollenfreilauf Starter
g00140 15003 Welle/Kupplung Starter
g00140 15004 Ritzel Starter
g00140 15005 Schraube/Mutter Ritzel
g00140 15006 Magnetschalter Starter
g00140 15007 Kohlenbuerste Starter
g00140 15008 Schraube/Mutter Starter
g00140 15009 Flansch Starter
g00140 15010 Zwischenflansch Starter
g00140 15012 Erregerwicklung Starter
g00140 15013 Anker Starter
g00140 15015 Einrueckhebel Starter
g00140 15016 Lager/Lagerbuchse Starter
g00140 15017 Dichtring Starter
g00141 15004 Ritzel Starter
g00141 15005 Schraube/Mutter Ritzel
g00142 15101 Drehstromgenerator
g00142 15102 Diode Drehstromgenerator
g00142 15103 Lagerdeckel Drehstromgenerator
g00142 15104 Kohlenbuerste Drehstromgenerator
g00142 15105 Anschluss Drehzahlmesser Drehstromgenerator
g00142 15106 Schraube Lagerdeckel Drehstromgenerator
g00142 15108 Fluegelrad Drehstromgenerator
g00142 15109 Stator Drehstromgenerator
g00142 15110 Luefter Drehstromgenerator
g00142 15111 Riemenscheibe Drehstromgenerator
g00142 15112 Mutter Riemenscheibe Drehstromgenerator
g00142 15113 Traeger Drehstromgenerator
g00142 15114 Schraube/Mutter Traeger Drehstromgenerator
g00142 15115 Spannschraube Drehstromgenerator
g00142 15117 Keilriemen Drehstromgenerator 8-Zylinder
g00142 15118 Spannbuegel Drehstromgenerator
g00142 15119 Befestigung Spannschraube Drehstromgenerator
g00142 15120 Kabelstecker B+ Drehstromgenerator
g00142 15121 Entstoerkondensator Drehstromgenerator
g00142 15124 Keilriemen Doppeldrehstromgenerator
g00142 15127 Lager Drehstromgenerator
g00142 15141 Belueftungsschlauch/Rohr Drehstromgenerator
g00142 15142 Schutzblech Drehstromgenerator
g00142 15143 Deckel Luftansaugung Drehstromgenerator
g00142 20213 Keilriemen Kuehlmittelpumpe/Drehstromgenerator
g00143 15103 Lagerdeckel Drehstromgenerator
g00143 15106 Schraube Lagerdeckel Drehstromgenerator
g00144 15111 Riemenscheibe Drehstromgenerator
g00144 15112 Mutter Riemenscheibe Drehstromgenerator
g00145 15113 Traeger Drehstromgenerator
g00145 15114 Schraube/Mutter Traeger Drehstromgenerator
g00146 15115 Spannschraube Drehstromgenerator
g00146 15119 Befestigung Spannschraube Drehstromgenerator
g00147 15201 Zuendverteiler
g00147 15202 Kappe Zuendverteiler
g00147 15204 Verteilerlaeufer
g00147 15210 Schutzhaube Zuendverteiler
g00147 15222 Antrieb Zuendverteiler
g00147 15225 Druckstueck Zuendverteiler
g00148 15305 Glueh-/Zuendkerze
g00148 15306 Zuendkerzenstecker
g00149 15301 Zuendkabel
g00149 15316 Halter Zuendkabel
g00149 15317 Abdeckung Zuendkabel
g00150 18001 Oelpumpe
g00150 18002 Oelpumpengehaeuse
g00150 18003 Schraube Oelpumpe
g00150 18004 Oelpumpenrad
g00150 18005 Saugrohr/Saugkorb/Sieb Oelpumpe
g00150 18006 Antriebswelle Oelpumpe
g00150 18007 Deckel Oelpumpe
g00150 18008 Schraube Deckel Oelpumpe
g00150 18010 Halter Saugkorb Oelpumpe
g00150 18011 Oelpumpenrad
g00150 18012 Lagerbuchse/Druckstueck Oelpumpe
g00150 18018 Verschlussstopfen Oelpumpenantrieb
g00150 18019 Ausgleichstueck Oelpumpe
g00150 18020 Schraube Saugrohr Oelpumpe
g00150 18021 Oelueberdruckventil Oelpumpe
g00150 18024 Schraube Halter Saugkorb
g00150 18025 Dichtung Saugrohr Oelpumpe
g00150 18031 Antriebsrad Oelpumpe
g00151 18007 Deckel Oelpumpe
g00151 18008 Schraube Deckel Oelpumpe
g00151 18019 Ausgleichstueck Oelpumpe
g00152 18010 Halter Saugkorb Oelpumpe
g00152 18024 Schraube Halter Saugkorb
g00153 18013 Rollenkette
g00153 18032 Spannbuegel Rollenkette
g00154 18005 Saugrohr/Saugkorb/Sieb Oelpumpe
g00154 18010 Halter Saugkorb Oelpumpe
g00154 18020 Schraube Saugrohr Oelpumpe
g00154 18024 Schraube Halter Saugkorb
g00154 18025 Dichtung Saugrohr Oelpumpe
g00155 18101 Oelfilter
g00155 18102 Traeger Oelfilter
g00155 18103 Oelfiltereinsatz
g00155 18104 Dichtring Oelfilter
g00155 18105 Mittelschraube Oelfilter
g00155 18106 Gehaeuse Oelfilter
g00155 18107 Oelablassschraube Oelfilter
g00155 18108 Dichtung Oelfilter/Kurbelgehaeuse
g00155 18109 Schraube Oelfiltertraeger
g00155 18110 Nebenstromoelfilter
g00155 18116 Thermostat Oelfilter
g00155 18117 Stiftschraube/Mutter Oelfiltergehaeuse
g00155 18127 Schraubstutzen Oelfilter
g00155 18129 Dichtring Schraubstutzen
g00155 18130 Deckel Oelfilter
g00155 18134 Deckel Oelfiltergehaeuse
g00155 18135 Dichtung Oelfiltergehaeuse
g00155 18145 Umgehungsventil Oelfilter
g00155 18146 Rueckschlagventil Oelfilter
g00155 18147 Wechselpatrone Oelfilter
g00155 18203 Dichtung Oelkuehlergehaeuse/Oelfilter
g00155 18204 Schraube/Mutter Oelkuehlergehaeuse/Oelfilter
g00155 20511 Oelfilter
g00156 18102 Traeger Oelfilter
g00156 18109 Schraube Oelfiltertraeger

g00157 18114 Oeldruckgeber
g00157 18115 Oeldruckgeber mit Warnkontakt
g00157 18138 Schraubstutzen Oeldruckgeber
g00157 18139 Abdichtring Schraubstutzen
g00158 18106 Gehaeuse Oelfilter
g00158 18117 Stiftschraube/Mutter Oelfiltergehaeuse
g00158 18134 Deckel Oelfiltergehaeuse
g00158 18135 Dichtung Oelfiltergehaeuse
g00159 18127 Schraubstutzen Oelfilter
g00159 18129 Dichtring Schraubstutzen
g00160 18138 Schraubstutzen Oeldruckgeber
g00160 18139 Abdichtring Schraubstutzen
g00161 18121 Oelleitung/Schlauch Oelkuehler Vorlauf
g00161 18140 Oelleitung/Schlauch Oelkuehler Ruecklauf
g00161 18201 Luftoelkuehler/Oelkuehler
g00161 18202 Oelkuehlergehaeuse
g00161 18203 Dichtung Oelkuehlergehaeuse/Oelfilter
g00161 18204 Schraube/Mutter Oelkuehlergehaeuse/Oelfilter
g00161 18205 Oelkuehlereinsatz
g00161 18206 Dichtung Oelkuehlereinsatz
g00161 18207 Schraube/Mutter Oelkuehlereinsatz
g00161 18211 Oelkuehlerdeckel
g00161 18212 Dichtung zwischen Oelkuehlerdeckel
g00161 18213 Schraube Oelkuehlerdeckel
g00161 18214 Dichtung Oelkuehler/Kurbelgehaeuse
g00161 18215 Schraube Oelkuehler Kurbelgehaeuse
g00161 18216 Oelueberdruckventil Oelkuehler
g00161 18217 Filtereinsatz Oelkuehler
g00161 18218 Filtergehaeuse Oelkuehler
g00161 18219 Dichtring Filtergehaeuse Oelkuehler
g00161 18220 Dichtung Oelkuehler
g00161 18225 Schraubstutzen Oelkuehler
g00161 20041 Kuehlmittelleitung/Schlauch Oelkuehler Vorlauf
g00161 20042 Kuehlmittelleitung/Schlauch Oelkuehler Ruecklauf
g00162 18205 Oelkuehlereinsatz
g00162 18206 Dichtung Oelkuehlereinsatz
g00162 18207 Schraube/Mutter Oelkuehlereinsatz
g00163 18211 Oelkuehlerdeckel
g00163 18212 Dichtung zwischen Oelkuehlerdeckel
g00163 18213 Schraube Oelkuehlerdeckel
g00164 18218 Filtergehaeuse Oelkuehler
g00164 18219 Dichtring Filtergehaeuse Oelkuehler
g00165 18241 Oel-Kuehlmittel Waermetauscher
g00165 18242 Dichtring Waermetauscher
g00166 18312 Oeleinfuellstutzen
g00166 18313 Deckel Oeleinfuellstutzen
g00166 18332 Dichtung Oeleinfuellstutzen
g00167 18311 Oelbehaelter
g00167 18329 Schauglas Oelvorratsbehaelter
g00168 18322 Oelstandgeber mechanisch Oelbehaelter
g00168 18333 Dichtung Oelstandgeber Oelbehaelter
g00169 20001 Kuehlmittelpumpe
g00169 20003 Welle Kuehlmittelpumpe
g00169 20004 Lager Kuehlmittelpumpe
g00169 20005 Fluegelrad Kuehlmittelpumpe
g00169 20006 Gleitdichtring Kuehlmittelpumpe
g00169 20007 Radialdichtring Kuehlmittelpumpe
g00169 20009 Dichtung Kuehlmittelpumpe/Kurbelgehaeuse
g00169 20010 Schraube Kuehlmittelpumpe
g00169 20013 Deckel Kuehlmittelpumpe
g00169 20014 Dichtringhalter Kuehlmittelpumpe
g00169 20015 O-Ring Dichtringhalter Kuehlmittelpumpe
g00169 20016 Schraube Dichtringhalter Kuehlmittelpumpe
g00169 20020 Dichtung/O-Ring Kuehlmittelpumpe
g00169 20024 Auslaufstutzen Kuehlmittelpumpe
g00169 20025 Schraube Auslaufstutzen Kuehlmittelpumpe
g00169 20210 Schraube Riemenscheibe
g00169 20211 Riemenscheibe Kuehlmittelpumpe
g00169 20212 Nabe Riemenscheibe Kuehlmittelpumpe
g00169 20213 Keilriemen Kuehlmittelpumpe/Drehstromgenerator
g00170 20014 Dichtringhalter Kuehlmittelpumpe
g00170 20015 O-Ring Dichtringhalter Kuehlmittelpumpe
g00170 20016 Schraube Dichtringhalter Kuehlmittelpumpe
g00171 20002 Gehaeuse Kuehlmittelpumpe
g00171 20019 Schraube Gehaeuse Kuehlmittelpumpe
g00172 20024 Auslaufstutzen Kuehlmittelpumpe
g00172 20025 Schraube Auslaufstutzen Kuehlmittelpumpe
g00173 20031 Stutzen Kuehlmittelruecklauf
g00173 20032 Dichtung Stutzen Kuehlmittelruecklauf
g00173 20033 Schraube Stutzen Kuehlmittelruecklauf
g00174 20210 Schraube Riemenscheibe
g00174 20211 Riemenscheibe Kuehlmittelpumpe
g00174 20212 Nabe Riemenscheibe Kuehlmittelpumpe
g00175 20222 Spannrolle
g00175 20223 Riemenscheibe Spannrolle
g00175 20224 Schraube/Mutter Spannrolle
g00175 20226 Traeger Spannrolle
g00175 20227 Lager Spannrolle
g00175 20228 Antrieb Spannrolle
g00175 20235 Keilriemen (Spannrolle zum Schwingungsdaempfer)
g00175 20236 Keilriemen (Spannrolle zur Luefterscheibe)
g00175 20241 Spannrolle
g00175 20245 Schraube Spannrolle
g00176 20231 Antriebswelle
g00176 20234 Lager Antriebswelle
g00177 20229 Spannhebel Luefterachse
g00177 20233 Mutter Spannhebel
g00178 20242 Hebel Riemenspanner
g00178 20243 Feder Riemenspanner
g00178 20244 Stossdaempfer Riemenspanner
g00178 20246 Lagerzapfen Riemenspanner
g00178 20247 Spannstange Riemenspanner
g00178 20248 Spannmutter Riemenspanner
g00178 20249 Riemenspanner
g00178 20250 Gummilager Riemenspanner
g00178 20252 Halter Riemenspanner/Zentralschraube
g00178 20253 Umlenkrolle Riemenspanner
g00178 20259 Zentralschraube Riemenspanner
g00178 20260 Hebel Feder Riemenspanner
g00179 20252 Halter Riemenspanner/Zentralschraube
g00179 20259 Zentralschraube Riemenspanner
g00180 20214 Luefterlagerbock
g00180 20216 Schraube Lagerbock/Traegerplatte
g00180 20254 Luefterlagerbock
g00180 20255 Nabe Luefterlagerbock
g00180 20256 Lager Luefterlagerbock
g00180 20257 Schraube Luefterlagerbock
g00181 20243 Feder Riemenspanner
g00181 20260 Hebel Feder Riemenspanner
g00182 20261 Halter Stossdaempfer Keilriemendaempfung
g00182 20262 Schraube Stossdaempferhalter
g00183 20203 Zentralschraube Luefterkupplung
g00183 20221 Luefterkupplung
g00184 20301 Luefterkupplung
g00184 20323 Lager Luefterkupplung
g00185 20515 Gelenkwelle
g00185 20516 Flansch Gelenkwelle
g00186 20517 Hydraulikpumpe Heckkuehlanlage
g00186 20518 Halter Hydraulikpumpe
g00187 23052 Lagerdeckel
g00187 23053 Befestigungsbolzen/Mutter Lagerdeckel
g00188 23055 Abtriebsrad
g00188 23057 Mutter Abtriebsrad
g00189 23056 Abtriebswelle
g00189 23058 Rollen-/Kugellager Abtriebswelle
g00190 23059 Abtriebsflansch
g00190 23060 Mutter Abtriebsflansch
g00190 23061 Abdichtring Abtriebsflansch
g00191 13023 Zwischenflansch Luftpresser/Lenkhelfpumpe
g00191 23062 Lenkhelfpumpe sekundaer
g00191 23101 Lenkhelfpumpe
g00191 23102 Gehaeuse Lenkhelfpumpe
g00191 23103 Lenkhelfpumpenwelle
g00191 23104 Abdichtring Lenkhelfpumpe
g00191 23105 Rotor Lenkhelfpumpe
g00191 23106 Flanschnabe/Riemenscheibe Lenkhelfpumpe
g00191 23107 Mutter Flanschnabe/Riemenscheibe Lenkhelfpumpe
g00191 23108 Mengenregelventil Lenkhelfpumpe

g00191 23109 Ueberdruckventil Lenkhelfpumpe
g00191 23110 Schraubstutzen Lenkhelfpumpe
g00191 23111 Schraube Lenkhelfpumpe
g00191 23123 Spannschraube Lenkhelfpumpe
g00191 23124 Keilriemen Lenkhelfpumpe
g00193 23121 Traeger Lenkhelfpumpe
g00193 23122 Schraube Traeger Lenkhelfpumpe
g00200 01026 Verschlussdeckel Nockenwelle
g00200 01029 Nockenwellendeckel mit Tragarm
g00200 01030 Schraube Nockenwellendeckel
g00201 01022 Schraube/Hahn Kuehlmittelablass
g00201 01034 Dichtung Schraube Kuehlmittelablass
g00201 01039 Leitung Kuehlmittelablass
g00201 18208 Kuehlmittelablassschraube
g00202 01003 Zylinderlaufbuchse
g00202 01017 Abdichtung Zylinderlaufbuchse Bund
g00202 01020 Dichtring Laufbuchse (Tombak)
g00203 03001 Kurbelwelle
g00203 03007 Riemenscheibe Kurbelwelle
g00203 03010 Abdichtring Kurbelwelle vorn
g00203 03011 Abdichtring Kurbelwelle hinten
g00203 03019 Gewindebohrung Kurbelwelle/Schwungrad
g00203 03023 Mitnehmer Schwungrad/Kurbelwelle
g00203 03025 Druckring Kurbelwelle
g00203 03026 Zwischenstueck/Riemenscheibe Kurbelwelle
g00203 03104 Anlaufscheibe Kurbelwelle/Passlager
g00204 03002 Kurbelwellenrad
g00204 03027 Scheibenfeder/Zylinderstift Kurbelwellenrad
g00205 03021 Schwungrad
g00205 03030 Segmentmagnet Schwungrad
g00206 03101 Hauptlager
g00206 03102 Pleuellager
g00206 03103 Hauptlager/Pleuellager
g00206 03201 Pleuelstange
g00206 03202 Pleuelbuchse
g00206 03205 Pleuellagerdeckel
g00207 03301 Kolben
g00207 03303 Oelabstreifring
g00208 05001 Nockenwelle
g00208 05002 Nockenwelle/Schwinghebel
g00208 05005 Oelrohr Nockenwelle/Lager
g00208 05006 Scheibenfeder/Stift Nockenwelle
g00208 05007 Verschlussdeckel Nockenwelle
g00208 05008 Axial-Fixierung Nockenwelle
g00208 05011 Antriebsrad auf Nockenwelle
g00208 05013 Druckscheibe Nockenwelle
g00208 05014 Schraube Druckscheibe/Anlaufscheibe
g00208 05015 Anlaufscheibe Nockenwelle
g00208 05021 Nockenwelle rechts
g00208 05022 Nockenwelle/Schwinghebel rechts
g00208 05023 Nockenwelle links
g00208 05024 Nockenwelle/Schwinghebel links
g00208 05031 Nockenwelle Auslass
g00208 05032 Axial-Fixierung Nockenwelle Auslass
g00208 05033 Pass-Stift Nockenwelle Auslass
g00208 05034 Nockenwellenrad Nockenwelle Auslass
g00208 05035 Zahnrad Nockenwelle Auslass
g00208 05036 Zahnrad Nockenwelle Einlass
g00208 05037 Nockenwelle Einlass
g00208 05038 Axial-Fixierung Nockenwelle Einlass
g00208 05039 Pass-Stift Nockenwelle Einlass
g00208 05051 Nockenwelle Auslass links
g00208 05053 Pass-Stift Nockenwelle Auslass links
g00208 05054 Nockenwellenrad Nockenwelle Auslass links
g00208 05057 Nockenwelle Einlass links
g00208 05059 Pass-Stift Nockenwelle Einlass links
g00208 05071 Nockenwelle Auslass rechts
g00208 05073 Pass-Stift Nockenwelle Auslass rechts
g00208 05074 Nockenwellenrad Nockenwelle Auslass rechts
g00208 05077 Nockenwelle Einlass rechts
g00208 05079 Pass-Stift Nockenwelle Einlass rechts
g00210 05103 Zwischenrad Steuerkette
g00210 05104 Zwischenradwelle/Schraubenrad
g00210 05105 Schraube Zwischenrad
g00210 05106 Umlenkrad Steuerkette
g00210 05111 Steuerkette
g00210 05112 Lagerbuchse Zwischenradwelle
g00210 05113 Lagerbuchse Umlenkrad
g00210 05121 Gleitschiene Steuerkette
g00210 05122 Lagerbolzen Gleitschiene
g00210 05123 Spannschiene Steuerkette
g00210 05124 Lagerbolzen Spannschiene
g00210 05125 Belag Spannschiene
g00210 05131 Kettenspanner
g00210 05132 Dichtung Kettenspanner
g00211 05104 Zwischenradwelle/Schraubenrad
g00211 05112 Lagerbuchse Zwischenradwelle
g00212 05202 Auslassventil
g00212 05204 Ventilabdichtung Auslass
g00213 05201 Einlassventil
g00213 05205 Ventilabdichtung Einlass
g00214 05201 Einlassventil
g00214 05202 Auslassventil
g00214 05203 Ventile insgesamt
g00214 05204 Ventilabdichtung Auslass
g00214 05205 Ventilabdichtung Einlass
g00214 05211 Ventilfeder
g00214 05212 Innere Ventilfeder
g00214 05213 Aeussere Ventilfeder
g00214 05214 Ventilfederteller
g00216 05312 Kugelbolzen
g00216 05315 Einstellschraube Kugelbolzen
g00216 05317 Kontermutter Einstellschraube
g00216 07051 Kaltstartventil
g00216 07056 Duese Startpilot
g00216 07057 Pumpe Startpilot
g00216 07058 Behaelter Startpilot
g00216 07059 Leitung Startpilot
g00216 07060 Halter Startpilot
g00216 07061 Leitungssatz Startpilot
g00216 07062 Schalter Startpilot
g00216 07064 Kompressor Startpilot
g00218 07113 Antriebsrad Einspritzpumpe/Spritzversteller
g00218 07114 Schraube Antriebsrad
g00218 07121 Kupplung zwischen Einspritzpumpe/Antrieb
g00219 07502 Verbindungsstange/Reguliergestaenge
g00219 07519 Kugelbolzen/Kugelpfanne Reguliergestaenge
g00220 09161 Druckregelklappe Mischgehaeuse
g00220 09162 Unterdruckdose Druckregelklappe
g00221 09127 Gummimanschette Luftansaugung
g00221 09146 Halter Gummimanschette
g00222 09308 Ladeluftrohr
g00222 09309 Dichtung Ladeluftleitung
g00223 09506 Flansch Flammgluehkerze
g00223 09507 Dichtung Flansch Flammgluehkerze
g00223 09508 Flammgluehkerze
g00224 09304 Flanschkruemmer/Abgasrohr
g00224 09306 Schraube/Mutter Flanschkruemmer
g00225 14382 Be- und Entlueftung
g00225 14384 Verteiler Be- und Entlueftungsleitung
g00226 13003 Schraube/Mutter Zylinder Luftpresser
g00226 13035 Zylinder/Laufbuchse Luftpresser
g00227 13004 Zylinderkopfdichtung Luftpresser
g00227 13005 Schraube/Mutter Zylinderkopf Luftpresser
g00227 13018 Zylinderkopf Luftpresser
g00230 20201 Luefter
g00230 20202 Schraube Luefter/Luefterkupplung
g00230 20203 Zentralschraube Luefterkupplung
g00230 20214 Luefterlagerbock
g00230 20216 Schraube Lagerbock/Traegerplatte
g00230 20217 Luefternabe Welle/Lager
g00230 20218 Mutter Luefterlagerung
g00230 20220 Abdichtring Nabe Luefterspannrolle
g00230 20221 Luefterkupplung
g00230 20229 Spannhebel Luefterachse
g00230 20233 Mutter Spannhebel
g00230 20254 Luefterlagerbock
g00230 20255 Nabe Luefterlagerbock

g00230 20256 Lager Luefterlagerbock
g00230 20257 Schraube Luefterlagerbock
g00230 20301 Luefterkupplung
g00230 20323 Lager Luefterkupplung
g00231 20311 Deckel Viskoseluefter
g00231 20312 Dichtung Deckel Viskoseluefter
g00231 20313 Schraube Deckel Viskoseluefter
g00232 20303 Antriebsscheibe Viskoseluefter
g00232 20307 Gehaeuse Viskoseluefter
g00232 20309 Feststellvorrichtung Viskoseluefter
g00232 20310 Gumminabe Viskoseluefter
g00232 20314 Abdichtring Viskoseluefter
g00233 20506 Oelkuehler Luefterantrieb
g00233 20510 Oelbehaelter Luefterantrieb
g00233 20514 Hydraulikpumpe Luefterantrieb
g00234 20507 Oelkuehler Retarder
g00234 20508 Halter Oelkuehler
g00235 09157 Abdeckhaube Luftansaugung
g00235 09171 Staubabscheider Luftansaugung
g00235 09172 Wasserabscheider Luftansaugung
g00236 09501 Brenner Flammstartanlage
g00236 09502 Duese Flammstartanlage
g00236 09505 Gluehkerze Flammstartanlage
g00236 09510 Magnetventil Flammstartanlage
g00236 09511 Kraftstoffleitung Flammstartanlage
g00236 09512 Steuergeraet Flammstartanlage
g00236 09513 Temperaturgeber Flammstartanlage
g00236 09514 Relais Flammstartanlage
g00236 09515 Betaetigungsschalter Flammstartanlage
g00236 09516 Kontroll-Leuchte Flammstartanlage
g00236 09517 Widerstand Flammstartanlage
g00236 09518 Leitungssatz Flammstartanlage
g00236 09519 Temperaturschalter Flammstartanlage
g00300 01008 Stiftschraube Motortraeger
g00300 24001 Motortraeger vorn
g00300 24002 Schraube/Mutter Motortraeger vorn
g00301 24003 Tragarm vorn
g00301 24004 Mutter/Schraube Tragarm vorn
g00302 24005 Motorlager vorn
g00302 24006 Schraube/Mutter Motorlager vorn
g00302 24021 Hydraulisches Motorlager vorn
g00302 24022 Traeger Motorlager vorn
g00302 24023 Schrauben Traeger Motorlager vorn
g00302 24024 Kuehlrohr hydraulisches Motorlager vorn
g00303 24011 Motorstossdaempfer
g00303 24012 Traeger Motorstossdaempfer
g00303 24013 Schraube Traeger
g00303 24015 Gummipuffer Motorstossdaempfer
g00303 24016 Schraube Stossdaempfer
g00303 24116 Halter Motorstossdaempfer
g00304 24012 Traeger Motorstossdaempfer
g00304 24013 Schraube Traeger
g00305 24022 Traeger Motorlager vorn
g00305 24023 Schrauben Traeger Motorlager vorn
g00306 24021 Hydraulisches Motorlager vorn
g00306 24024 Kuehlrohr hydraulisches Motorlager vorn
g00307 24101 Motortraeger/Stuetze hinten
g00307 24102 Schraube/Mutter Motortraeger hinten
g00308 24103 Motorlager hinten
g00308 24104 Schraube/Mutter Motorlager hinten
g00308 24105 Konsole Motorlager hinten
g00308 24106 Schraube Konsole hinten
g00309 24105 Konsole Motorlager hinten
g00309 24106 Schraube Konsole hinten
g00310 24108 Zugstange
g00310 24109 Zugstangenkopf
g00310 24110 Halter Zugstange
g00311 47001 Kraftstoffbehaelter
g00311 47002 Ablassschraube Kraftstoffbehaelter
g00311 47010 Einfuellstutzen Kraftstoffbehaelter
g00311 47011 Einfuellverschluss Kraftstoffbehaelter
g00311 47012 Abdichtung Einfuellverschluss
g00311 47013 Schliesszylinder Einfuellverschluss
g00311 47014 Einsatz Einfuellstutzen Kraftstoffbehaelter
g00311 47019 Manschette Kraftstoffbehaelter
g00311 47020 Gummimanschette um Einfuellstutzen
g00311 47021 Konsole Kraftstoffbehaelter
g00311 47022 Schraube Konsole
g00311 47023 Spannband Kraftstoffbehaelter
g00311 47024 Mutter Spannband
g00311 47025 Unterlage Kraftstoffbehaelter
g00311 47026 Verschlussstopfen Kraftstoffbehaelter
g00311 47109 Kraftstoffruecklaufschlauch Kraftstoffbehaelter
g00312 47003 Kraftstoff-Filter/-Sieb
g00312 47004 O-Ring Kraftstoff-Sieb
g00313 47006 Tauchrohrgeber/Tankgeber
g00313 47007 Dichtring Tauchrohrgeber/Tankgeber
g00313 47008 Schwimmer Tankgeber
g00313 47009 Flansch Tauchrohrgeber
g00314 47010 Einfuellstutzen Kraftstoffbehaelter
g00314 47014 Einsatz Einfuellstutzen Kraftstoffbehaelter
g00314 47020 Gummimanschette um Einfuellstutzen
g00315 47011 Einfuellverschluss Kraftstoffbehaelter
g00315 47012 Abdichtung Einfuellverschluss
g00315 47013 Schliesszylinder Einfuellverschluss
g00316 47015 Tankeinfuellrohr
g00316 47016 Schlauch Tankeinfuellrohr
g00316 47017 Schlauchbinder Tankeinfuellrohr
g00316 47027 Kraftstoffrueckschlagklappe Tankeinfuellrohr
g00317 47021 Konsole Kraftstoffbehaelter
g00317 47022 Schraube Konsole
g00318 47023 Spannband Kraftstoffbehaelter
g00318 47024 Mutter Spannband
g00319 47030 Zusatzkraftstoffbehaelter mit Einfuellstutzen
g00319 47031 Zusatzkraftstoffbehaelter rechts
g00319 47032 Kraftstoff-Verbindungsschlauch/-rohr
Zusatzkraftstoffbehaelte
g00319 47033 Absperrhahn Zusatzkraftstoffbehaelter
g00319 47034 Halter Kraftstoff-Verbindungsschlauch/-rohr
g00319 47042 Entlueftungsschlauch Zusatzkraftstoffbehaelter
g00319 47043 Betaetigungshebel Absperrhahn
g00320 47032 Kraftstoff-Verbindungsschlauch/-rohr
Zusatzkraftstoffbehaelte
g00320 47034 Halter Kraftstoff-Verbindungsschlauch/-rohr
g00321 47033 Absperrhahn Zusatzkraftstoffbehaelter
g00321 47043 Betaetigungshebel Absperrhahn
g00322 47101 Kraftstoff-Ausgleichbehaelter
g00322 47102 Kraftstoffschlauch Ausgleichbehaelter
g00323 47103 Be-/Entlueftungsleitung
g00323 47104 Ventilsystem Be-/Entlueftung
g00323 47108 Schlauch Be-/Entlueftung
g00323 47142 Leitung Be-/Entlueftung
g00323 47152 Entlueftung
g00324 47116 Kraftstoffkuehler
g00324 47117 Halter Kraftstoffkuehler
g00325 47112 Kraftstoffruecklaufleitung Fahrgestell
g00325 47118 Hohlschraube Kraftstoffruecklaufleitung
g00326 47105 Regenerierventil Kraftstoffverdunstung
g00326 47141 Thermoventil Kraftstoffverdunstung
g00326 47144 Regenerierleitung Kraftstoffverdunstung
g00326 47145 Pneumatikleitung Kraftstoffverdunstung
g00327 47018 Magnetventil Sicherheitseinrichtung Kraftstoffanlage
g00327 47028 Wasserabscheider Kraftstoffanlage
g00327 47106 Lueftungsventil Kraftstoffanlage
g00328 47111 Kraftstoffleitung Fahrgestell
g00328 47112 Kraftstoffruecklaufleitung Fahrgestell
g00328 47118 Hohlschraube Kraftstoffruecklaufleitung
g00329 47201 Gasmischer Erdgasanlage
g00329 47202 Stellmotor Erdgasanlage
g00329 47203 Sensor Zuendzeitpunktregelung
g00329 47205 Steuergeraet Erdgasanlage
g00329 47206 Druckminderer Erdgasanlage
g00329 47207 Absperrventil elektromagnetisch
g00329 47208 Absperrventil mechanisch
g00329 47209 Gasvorratsbehaelter
g00329 47212 Fuellanschluss Erdgasanlage
g00329 47213 Hauptschalter Erdgasanlage
g00329 47214 Druckmanometer Erdgasanlage

g00329 47216 Gasleitung
g00329 47217 Abdeckhaube Gasvorratsbehaelter
g00329 47219 Elektrische Leitung Erdgasanlage
g00329 47220 Kontrolleuchte Erdgasanlage
g00329 47222 Tastschalter Fehlercode Erdgasanlage
g00329 47223 Tastschalter evakuieren Erdgasanlage
g00330 49012 Auspuffleitung vorn
g00330 49013 Dichtung Auspuffleitung vorn
g00331 49033 Halter Auspuff
g00331 49034 Mutter/Schraube Halter Auspuff
g00332 49065 Resonator
g00332 49066 Auspuffrohr Resonator
g00333 49016 Mittelschalldaempfer
g00333 49052 Flanschverbindung Mittelschalldaempfer
g00334 49022 Nach-/Hauptschalldaempfer
g00334 49053 Flanschverbindung Nachschalldaempfer
g00335 49009 Mutter/Schraube Auspuffleitung
g00335 49023 Rohrschelle Auspuffleitung
g00335 49038 Abdeckung Auspuffleitung Dach
g00335 49055 Seitenabstuetzung Auspuffleitung
g00336 49021 Auspuffendrohr
g00336 49037 Regenschutzklappe Auspuffendrohr
g00337 50007 Ablassschraube Kuehler
g00337 50008 Dichtring Ablassschraube Kuehler
g00338 50013 Luefterhaube
g00338 50014 Manschette Luefterhaube
g00339 50011 Kuehlerbefestigung
g00339 50016 Gummilager Kuehlerbefestigung
g00340 50003 Kuehler Kuehlmittelkasten oben
g00340 50004 Kuehler Kuehlmittelkasten unten
g00340 50023 Kuehler Kuehlmittelkasten links
g00340 50024 Kuehler Kuehlmittelkasten rechts
g00341 20111 Temperaturschalter Zusatzluefter
g00341 50041 Geblaese Zusatzluefter
g00341 50042 Schutzgitter Zusatzluefter
g00341 50043 Vorwiderstand Zusatzluefter
g00342 20503 Ausgleichbehaelter
g00342 50101 Kuehlmittelausgleichbehaelter
g00342 50102 Verschlussdeckel Ausgleichbehaelter
g00342 50103 Einfuellstutzen Ausgleichbehaelter
g00342 50111 Halter Kuehlmittelausgleichbehaelter
g00342 50112 Spannschelle Ausgleichbehaelter
g00342 50211 Kuehlmittelschlauch/Leitung Ausgleichbehaelter
g00342 50214 Schlauchschelle Entlueftungsleitung Ausgleichbehaelter
g00342 50216 Entlueftungsleitung/Schlauch Ausgleichbehaelter
g00342 50217 Halteclips Entlueftungsleitung Ausgleichbehaelter
g00343 20502 Kuehlerverkleidung
g00343 50001 Kuehler
g00343 50002 Kuehler Lamellen
g00343 50005 Kuehler Kuehlmittelstutzen
g00343 50006 Kuehlerverschluss
g00343 50007 Ablassschraube Kuehler
g00343 50008 Dichtring Ablassschraube Kuehler
g00343 50017 Seitenblech am Kuehler
g00343 50018 Kuehlerlagerung unten
g00343 50020 Kuehlerabdeckung
g00343 50022 Befestigungsmutter Getriebeoelkuehler/Kuehler
g00343 50026 Dichtbeilage Kuehler
g00343 50201 Kuehlmittelschlauch Thermostat/Kuehler
g00343 50202 Kuehlmittelschlauch Kuehler/Motor
g00343 50203 Ablaufschlauch Kuehler
g00344 50105 Kuehlmittelstandgeber
g00344 50106 Sicherungsring Kuehlmittelstandgeber
g00345 50214 Schlauchschelle Entlueftungsleitung Ausgleichbehaelter
g00345 50216 Entlueftungsleitung/Schlauch Ausgleichbehaelter
g00345 50217 Halteclips Entlueftungsleitung Ausgleichbehaelter
g00346 50251 Elektrische Leitung Regelanlage
g00346 50252 Hauptschalter Regelanlage
g00346 50253 Relais Regelanlage
g00346 50254 Kondensator Regelanlage
g00346 50255 Widerstand Regelanlage
g00346 50257 Schutzkappe Regelanlage
g00347 50301 Ladeluftkuehler
g00347 50302 Schraube Ladeluftkuehler
g00347 50303 Rohrkruemmer Ladeluftkuehler
g00347 50304 Dichtung Rohrkruemmer Ladeluftkuehler
g00347 50305 Schraube Rohrkruemmer Ladeluftkuehler
g00347 50306 Formschlauch Ladeluftkuehler/Rohrkruemmer
g00347 50307 Rohrschelle Formschlauch Ladeluftkuehler
g00348 50303 Rohrkruemmer Ladeluftkuehler
g00348 50304 Dichtung Rohrkruemmer Ladeluftkuehler
g00348 50305 Schraube Rohrkruemmer Ladeluftkuehler
g00349 50306 Formschlauch Ladeluftkuehler/Rohrkruemmer
g00349 50307 Rohrschelle Formschlauch Ladeluftkuehler
g00350 07246 Halter Einspritzleitung Metall
g00350 07248 Halter Einspritzleitung Kunststoff
g00350 07283 Einspritz-Leitung Zylinder 1
g00350 07284 Einspritz-Leitung Zylinder 2
g00350 07285 Einspritz-Leitung Zylinder 3
g00350 07286 Einspritz-Leitung Zylinder 4
g00350 07287 Einspritz-Leitung Zylinder 5
g00350 07288 Einspritz-Leitung Zylinder 6
g00350 07289 Einspritz-Leitung Zylinder 7
g00350 07290 Einspritz-Leitung Zylinder 8
g00351 09119 Gitter Luftansaugung
g00351 09120 Sieb Luftansaugung
g00352 50206 Kuehlmittelschlauch
g00352 50207 Schlauchschelle Kuehlmittelschlauch
g00353 20102 Einsatz Kuehlmittelregler
g00353 20103 Gehaeuse/Deckel Kuehlmittelregler
g00353 20104 Dichtring Kuehlmittelregler/Gehaeuse
g00353 20105 Dichtung Gehaeuse Kuehlmittelregler
g00353 20106 Schraube Gehaeuse Kuehlmittelregler
g00354 18022 Kolben Druckbegrenzungsventil
g00354 18023 Verschlussschraube Druckbegrenzungsventil
g00355 18321 Anzeige-Instrument Oelstand
g00355 18325 Oelstandschalter/Geber Oelwanne
g00355 18326 Warnleuchte Oelstand
g00355 18328 Dichtbeilage Oelstandgeber/Schalter - Oelwanne
g00355 18330 Elektrische Leitung Oelstandgeber/Schalter
g00356 20215 Traegerplatte
g00356 20216 Schraube Lagerbock/Traegerplatte
g00357 14240 Luftleitung Konstantdrossel
g00357 14241 Verschlussschraube Konstantdrossel
g00357 14244 Ventil Konstantdrossel
g00358 14306 Schlauch Lufteinblasung
g00358 14345 Kombiventil Lufteinblasung
g00358 14368 Elektro-Umschaltventil Lufteinblasung
g00359 14386 Diode Schaltsperre
g00359 14387 Relais Schaltsperre
g00360 15215 Thermoventil Zuendverstellung
g00360 15216 Rueckschlagventil Zuendverstellung
g00361 14006 Luftquerrohr
g00361 14008 Dichtung Querrohr/Saugkruemmer
g00362 14240 Luftleitung Konstantdrossel
g00362 14241 Verschlussschraube Konstantdrossel
g00362 14242 Abdichtring Stoessel Konstantdrossel
g00362 14243 Stoessel Konstantdrossel
g00362 14244 Ventil Konstantdrossel
g00365 01071 Schwungradgehaeuse
g00365 03019 Gewindebohrung Kurbelwelle/Schwungrad
g00365 03022 Dehnschraube Schwungrad/Mitnehmer
g00365 03023 Mitnehmer Schwungrad/Kurbelwelle
g00366 07401 Vergaser
g00366 07624 Vergaserflansch
g00367 20501 Kuehler Heckkuehlanlage
g00367 20513 Luefter Heckkuehlanlage
g00367 20517 Hydraulikpumpe Heckkuehlanlage
g00367 20518 Halter Hydraulikpumpe
g00368 23002 Hydraulikpumpe Motorabtrieb vorn
g00368 23011 Schraube Hydraulikpumpe
g00368 23012 Traeger Hydraulikpumpe
g00368 23013 Schraube Traeger
g00368 23014 Riemenscheibe Hydraulikpumpe
g00368 23015 Keilriemen Hydraulikpumpe