FIONA

Freight & Invoice Operations Network Application

- Diploma Thesis -

Submitted in fulfillment of the requirements for the
Diploma in Computer Science

by

Stefan Dederichs
Andreas Röhncke

Supervisors:
Prof. Dr. Klaus G. Troitzsch
Dipl.-Inf. Daniel Schmidt

Dept. of Computer Science, Institute for IS Research
University of Koblenz-Landau, Campus Koblenz

Koblenz, September 2006

Hereby we declare having created this entire thesis with none more than the sources and media mentioned.

The first chapter was primarily written by Andreas Röhncke,
Stefan Dederichs is mainly responsible for chapter two.
The remaining chapters were composed in joint effort.

We consent to adding this document to the university library's collection.
We object to publishing it via the Internet.

Koblenz, September 30, 2006

.....	.....
Stefan Dederichs	Andreas Röhncke

Transportation is a precise business.

Frank Martin

Contents

Chapter 1

Introduction

The following chapters give an overview over this document, the information system it accompanies and the background history that led to their realization.

1.1 FIONA

This diploma thesis is to be seen in the context of logistics information systems, more precisely in the sector of transport booking and accounting.

FIONA is an acronym for Freight & Invoice Operations Network Application. The name roughly summarizes the scope of the application and the project in which it was created.

It was one of several projects conducted in cooperation of the Institute for Information Systems Research (www: [Uni06a]) at the University of Koblenz-Landau, Campus Koblenz (www: [Uni06b]) with Tenneco Inc (in the following called Tenneco). The other projects are

- confluentic, which examines the interrelations between inventory and transport cost in logistics business to reduce transport expenses (cf. [KLS05]),
- confluentic WEB, preparing the porting of confluentic to a web-based application fitting into the Tenneco IT environment (cf. [Dro06]),
- PakMan, a packaging management system currently not in use (cf. [Her05] and [Mün05]) and
- CaTIS, the Carrier and Tariff Information System providing the routing and pricing information for FIONA (cf. [Kop06]).

Tenneco is one of the world's leading suppliers of Emission Control (EC)[1] and Ride Control (RC)[2] products. With 19,000 employees in nearly 80 manufacturing and engineering facilities in 24 countries on 6 continents and US$ 4.4 billion of revenues, Tenneco is a Fortune 500 company (cf. [Ten05a] and [For06a] or [For06b]). More than 30 original equipment manufacturers (OEM) and approximately 500 After Market (AM) customers are served by Tenneco all over the world. Moreover, Tenneco is further expanding in new markets e. g. China and Eastern Europe, following its customers into these geographical sectors.

These facts pose a heavy challenge on corporate logistics departments and management because of the tremendous amount of transports that are necessary between Tenneco plants as well as between Tenneco and suppliers or customers. Especially the freight booking processes itself, with planning, booking, calculation of prices, invoicing, controlling and statistics has to be well and reliably organized. This forms the background of the project's considerations.

The intention was to provide an information system that supports Tenneco in processing and monitoring freight bookings in daily business. This includes the booking and updating of concrete shipments, distribution of transport cost and several reporting functionalities.

Different solutions were examined in the course of the project and eventually the first version of an Internet application that streamlines the existing business process was developed. The core of this tool is a central database allowing collected shipping information to be queried and changed online and in real time.

The software is characterized by its main benefits for Tenneco's logistics department: All parties involved in a shipment (sender, carrier, receiver, controller) have immediate access to the booking data, so that booking information and parameters like the load composition can easily be exchanged and updated. Transport costs are portioned automatically and the results can be used by the carrier to create invoices for the paying sites and by the paying sites to verify the invoices. Shipping data can be queried and exported in several ways to simplify its transfer to analysis and invoicing facilities.

Members of the project team at the university were Stefan Dederichs and Andreas Röhncke, supervised by Prof. Dr. Klaus G. Troitzsch and Dipl.-Inf. Daniel Schmidt. At Tenneco, the project manager was Fabrice Pacolet, sponsored by Thomas Klein and Lawrence Chow, all members of the Tenneco

[1] Tenneco markets Emission Control products under highly recognized brands like Walker, Gillet and Fonos.

[2] Ride Control products Tenneco manufactures are sold primarily under the known Monroe brand, as well as Clevite Elastomers and Fric Rot.

logistics management department and the project's initiators. Further on, Tenneco nominated several IT managers to support the technical part.

This thesis describes the development of FIONA using the course of the project (described in Chapter 1.2) as basic plot. After presenting the analysis of the encountered situation, the "traditional" booking workflow and the evaluated alternative approaches, an overview on the operational and technical requirements is given. FIONA's architectural design and its final implementation are discussed in the subsequent chapters. The document is rounded out with a look at possible future enhancements to further optimize the freight booking and invoicing process.

The finished FIONA application prototype was handed over to Tenneco's IT department to go live in an exemplary production environment. Further software development and enhancements now are incumbent on the Tenneco development staff.

1.2 Project structure

The project was divided into two main phases which in turn consisted of different activity blocks, accompanied by several working documents. The outline is shown in Fig. 1.1: activities are depicted by ellipses, resulting documents or document groups by rectangular boxes.

The first part comprised the analyzing and conceptualizing efforts, followed by the implementation phase. The phases and activities were not clearly delimited. There was plenty of interaction, especially between the design and programming activities: As indicated by the diagonal separation line, implementation work already started during design, and the analysis was not considered complete before a significant part of the programming had been done. The mask design of the web interface at an early project stage, for instance, was very helpful in discovering the desired workflow and data sets of the target system. Furthermore, new aspects were identified regularly by the logistics supervisors during the implementation phase, so that additional analysis runs were necessary to integrate the newly raised requirements into FIONA (see Chapter 3).

The initial phase started with an in-depth requirements analysis including the evaluation of several alternative solutions and the decision for a completely new information system. Chapter 2 describes this part of the project. All the collected information on required data, business process flows, user groups etc. was sorted and written down in a requirements specification document (see Appendix D) which was discussed and revised in multiple iterations before it was signed by the Tenneco project members.

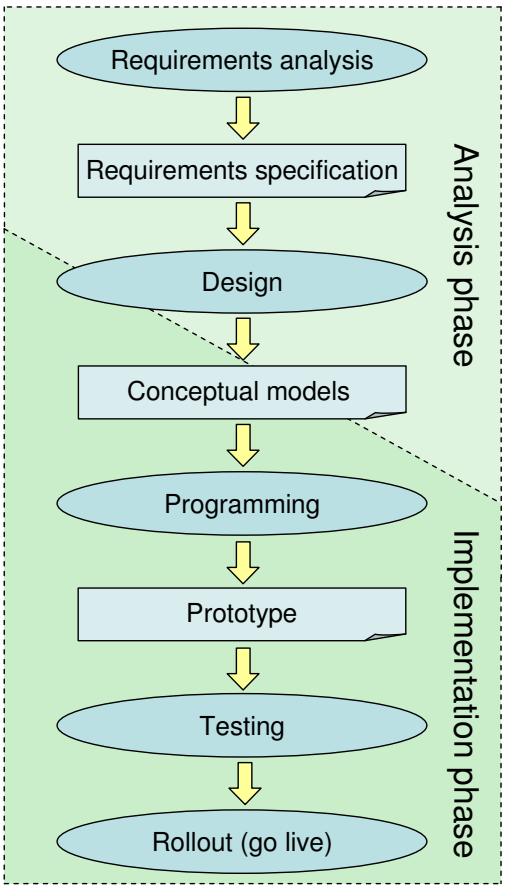

Figure 1.1: Project structure

The gathered knowledge about the relevant processes and required functionality then entered the design activities (see Chapter 4). Results were the first conceptual models, especially the E/R Diagram defining the relational database model. It was developed consolidating the specific needs of FIONA and CaTIS with regard to their intended cooperation and the avoidance of data redundancy. Close contact to the Tenneco project group ensured that all required data and its structure were considered. The design of the web interface was pushed early to have a common visual foundation for all further communication. Many issues and data fields to be included could be identified discussing the masks. They were also used for a first presentation to the future users.

The conceptual models set the course for the database and application code. With proceeding implementation, regular resumptions of the design activities were made. This way, the evolving system structure was successively

refined and incorporated the change requests of the project's supervisors. The most important implementation decisions are described in Chapter 5.

After testing the software, the university's project team provided the responsible development supervisor in Tenneco's IT with the latest technical information, customization and installation hints (see Appendix E). The application was then handed over for the last adaptations to make it run in the destined environment.

The rollout in a representative sector of the Tenneco environment was postponed due to resource shortages. It is planned for shortly after the finishing of this thesis and its outcome can thus not be dealt with here.

Chapter 2

Requirements analysis

The project's first part comprised the analysis of Tenneco's present situation in the affected sector and the determination of desired achievements.

2.1 Approach

To find out which issues triggered the need for a freight booking and invoicing system and which functionality was required, several meetings and conference calls with varying attendees were held. Tenneco participants came from different divisions, namely corporate logistics, operations and IT to make sure all aspects of the project were covered.

The meetings with logistics supervisors aimed at getting an overview over the company and the logistics processes in place to date. They helped in becoming aware of interrelations between Tenneco, its customers, suppliers and carriers and pointed out the sectors with the highest need for improvement. These were

- the freight transport booking process itself,
- the coordination of load allocation,
- the awareness of free truck space,
- the calculation of partial amounts to pay,
- the propagation of pricing and utilization figures.

Results of the meetings were incorporated into specification proposals and diagrams by the project team. Mind maps (cf. [Ill06]) were employed in this early phase to collect and structure the issues and data to be dealt with.

Fig. 2.1 shows a part of the data field collection;[1] the complete mind maps are included in the appendix to this document (see Appendix C). The findings were presented in the following meetings. This in turn led to a revision of the generated documents and was repeated until a settled state was achieved. All key requirements of the software were assessed iteratively during workshops of this kind.

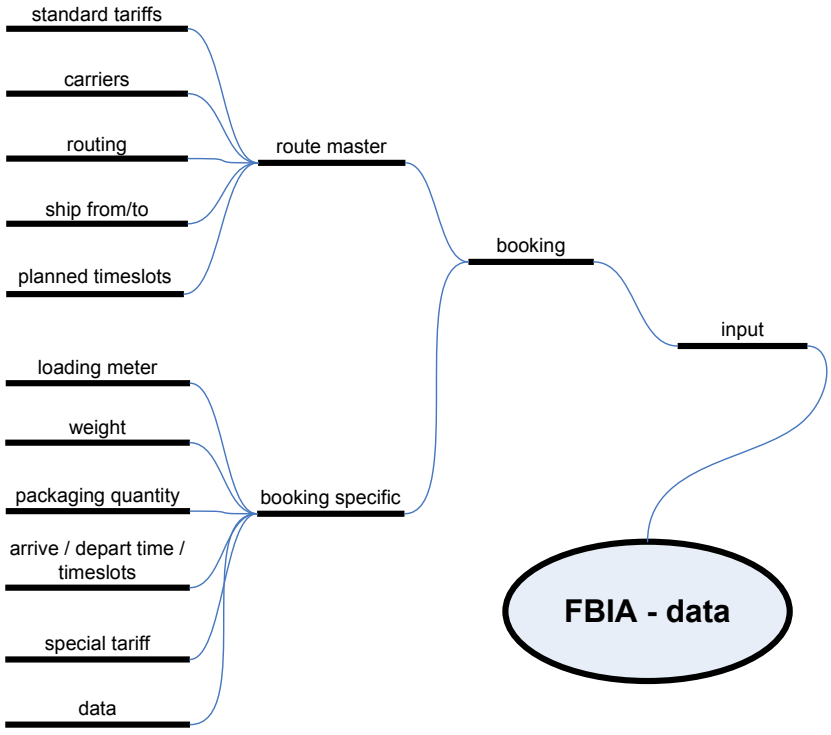

Figure 2.1: Mind map excerpt — Taken from the collection of required data to be handled by the system.

Operations people were involved following the need of having the actual users in the project team. They provided detail knowledge about internal proceedings, gave hints about exceptional cases to be considered and were given presentations of the user interface and benefits to-come. Since their offices are scattered over Europe, most of their participation took place via telephone and remote presentations.

[1]The acronym FBIA — Freight Booking & Invoice Audit — served as the working title referring to the project before the development of a new application and its name were decided.

Tenneco's IT staff was to be kept informed about the development status to prepare them for the handover at the end of the project. They also assisted in verifying the necessity of a new development as opposed to extending any existing information system like those based on SAP or AS/400 or buying an out-of-the-box solution (see Chapter 2.4.2). The development department based in the USA provided its collection of directives on good code and design which is mandatory for all Tenneco software projects (Chapter 3.2). This included a rigid list of platforms, languages and versions to utilize for implementation as well as sample code to illustrate some of the preferred techniques. Adhering to these development regulations reduced the complications when moving the application from the university to Tenneco IT and ensured the availability of hands-on knowledge required to maintain the software later on.

2.2 Encountered situation

An essential point when analyzing the relevant transporting environment and the resulting requirements was the categorization of different methods of transport. With that done, identifying shipment business cases, their treatment and the consequences came into focus.

2.2.1 Transport Types

The Transport Types and related pricing models for shipments occurring in Tenneco transport can be classified as follows[2]. The list is prioritized and reflects the significance in Tenneco logistics and the need for information system support in decreasing order:

1. Full Truck Load (FTL)
   (a) Pure FTL
      i. Single trip
      ii. Round trip
   (b) Route (multi-stop FTL)
      i. Non-round
      ii. Round

[2]Note that the organization of categories here differs slightly from the one in the requirements specification document (see Appendix D): The different flavors of FTL delivery are grouped together, since they can be treated equally on a technical level.

   (c) Milk Run
      i. Inbound
      ii. Outbound
2. Groupage
3. Premium freight
4. Distribution
5. Special

FTL

The carriers in a Full Truck Load (FTL) scenario are paid for a complete dedicated truck regardless of its fill rate. This type of contract is usually not as expensive as filling a truck when paying per load unit (LTL, less than truck load), however, empty space on the truck corresponds with unrewarded expenses and should be avoided.

The shown subcategories of FTL are of logical business case nature. The underlying accounting concept built on a dedicated truck is the same.

Pure FTL means that the truck goes from location A to location B without intermediate loading/unloading stop. This happens either as single trip or, if the journey back is included in the price, as round trip.

A Route is defined as FTL setup with at least one intermediate loading/unloading stop: Not all cargo is shipped from start to end of the trip, some of it is picked up or discharged at one or several stops on the way. As in pure FTL, this may or may not include the trip back to the origin location. The term Route is reserved for so-called inter-company (ICY) shipments. These are transport scenarios with all involved parties belonging to Tenneco. Routes are the result of efforts to optimize truck fill rates and reduce expenses for separate load-unload relations by combining them, if geographically appropriate.

Milk Runs also describe the journey of a truck visiting multiple stops. Here external sites are involved. On an inbound Milk Run goods are collected in sequence at suppliers' locations to unload them at the end of the trip at one or a small number of Tenneco locations. Outbound Milk Runs cover the opposite direction: The truck collects cargo at one or few Tenneco sites and delivers them in sequence to multiple customer locations. It may happen that a Milk Run forms a closed circle, i. e. start and end site are identical. This is necessary when empty packaging used to store the goods on the transport has to be picked up before or brought back after the trip.

Unless otherwise stated, the term refers to the inbound variant when used throughout this document.

Groupage

Groupage identifies inbound delivery in LTL mode, only the actual load to be shipped is taken into account to determine the price. The transport itself is organized in a black-box-oriented way: Tenneco orders the delivery of a certain amount of goods from an external supplier for a specific time slot. The forwarder takes care of the vehicle as well as the logistics infrastructure and optimization to realize the negotiated price. Due to its expensiveness, Tenneco aims at reducing this model's usage more and more in favor of Route and Milk Run installations.

Premium freight

Express delivery in urgent cases is referred to as Premium Freight. The means of transport depends on the sort of cargo and the amount of time that is left to complete the delivery. Ordering such a transport usually requires a special permission by a logistics supervisor and is most often used in scenarios where delayed arrival of goods results in drastic contractual penalties.

Distribution

A transport mode which is analog to Groupage except for its outbound direction is often referred to as Distribution: Goods have to be transferred from Tenneco to external customers. The organization of the transport itself is left to the carrier. Other taxonomies exist that file Distribution under Groupage and just distinguish between in- and outbound. For the business environment affected by FIONA it was, however, identified as a separate type with less significance than Groupage.

Special

The group of more exotic transport modes contains ocean and air freight as well as various combined setups. The pricing structures of such concepts tend to get very complex. Many of them are tailor-made solutions matching individual cases.

The characterization of transport and tariff models given here is very rough and geared to the context of this thesis. For an in-depth analysis from a more tariff- and contract-oriented perspective, see [Kop06].

FIONA only deals with shipments on Routes and (inbound) Milk Runs. They form a representative sector of Tenneco's logistics with significant potential for streamlining. Consequently, the propositions presented in the following chapters will be restricted to these two types — with the addition that Pure FTL may be interpreted as a Route with only the start and end locations, so concepts presented for Routes are transferable without major changes (see Chapter 3.1 for details about the selection of covered concepts).

Most other Transport Types can, however, be broken down into a combination of shipment pattern and pricing model as well. This facilitates transport organization and further extensions of the system as the composition of shipment scenarios is naturally similar. Only the tariff part needs to be changed when extending the operative scope.

2.2.2 Process

The traditional workflow for processing a delivery of goods or containers from booking the truck to sending out the invoices is best illustrated with a concrete shipping scenario. "Route 1" is a typical Trip Scheme transports are conducted on regularly (Fig. 2.2). A Trip Scheme refers to a fixed sequence of stops a truck visits during delivery. In FIONA it is taken as the master data for a general setup of a Route, Milk Run etc. The term was chosen over "Route" as the general term for route master data because the latter might be confused with Tenneco multi-stop Routes described in Chapter 2.2.1.

A truck serving "Route 1" picks up load for Wissembourg, Fameck and Gent in Valencia. It first stops in Wissembourg (connection A), unloading what is destined for Wissembourg or Gent. Transferring the goods to Gent (connection D) is done on another Trip Scheme. This practice is referred to as cross-docking: Wissembourg is a cross-dock for goods and empty packaging to Gent. It was introduced to further optimize truck utilization rates and to get rid of redundant load-unload relations which were scattered over different Routes or pure FTL Trip Schemes. The complete delivery path from Valencia to Gent is realized by joint use of two independent Trip Schemes, so no direct connection is necessary.

The remaining load is forwarded on to Fameck (connection B). Additional goods or empty packaging may be picked up at one of the stops on the way to be delivered back to Valencia (connection C), this results in an optional back trip for the Trip Scheme.

Which site is responsible for ordering the transport depends on the specific Trip Scheme, no general rule set is given. On "Route 1" a transport is initiated by Valencia. The loading space distribution among the different loading and unloading sites of the Trip Scheme is not fixed as well.

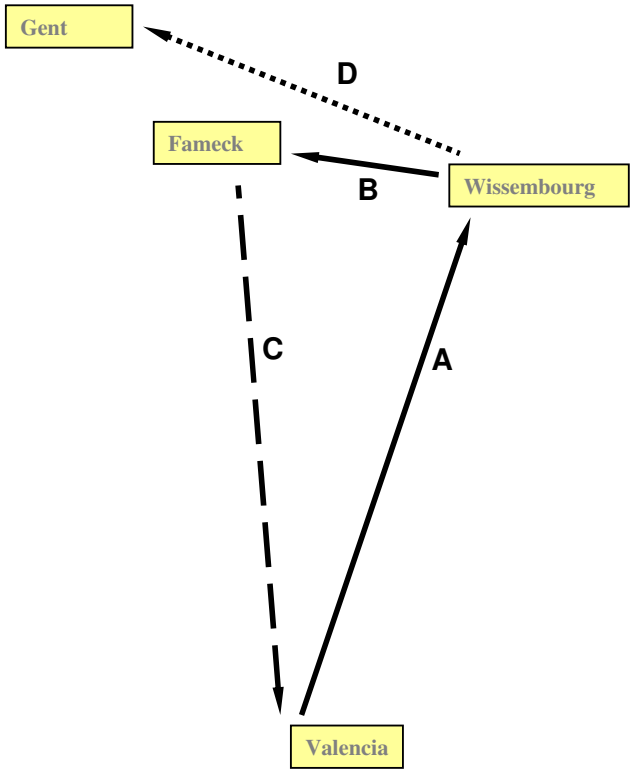

Figure 2.2: Route 1 — Going from Valencia to Wissembourg and Fameck. Goods for Gent are cross-docked at Wissembourg.

In the previous logistics environment it had to be negotiated in direct communication. This usually happened via telephone. Load decreases were often not announced at all, resulting in wasted free space on the truck.

The participating locations share delivery cost. To understand the portioning, the load has to be thought of as split into several parts, each assigned to a load-unload relation on the Trip Scheme. On "Route 1", this could e. g. be:

- 2 ldm from Valencia to Wissembourg
- 7 ldm from Valencia to Fameck

Each of these parts has a portion of the delivery cost assigned to it, which is paid by one of the participating locations.

The price for delivery is assigned to the load fractions in different ways, depending on the type of Trip Scheme. On Routes only the proportion of load and load sum is relevant. This leads to the following payment formula:

$$price_i = tariff * \frac{load_i}{\sum_{i=1}^{n} load_i}$$

The price of load fraction i is proportional to its size in comparison to the total freight on the delivery. On Routes the unit of measurement for cargo is usually the loading length. It is expressed in loading meters (ldm).

For Milk Runs the distances of the load-unload relations are additionally taken into consideration. The resulting calculation instruction is:

$$price_i = tariff * \frac{load_i * distance_i}{\sum_{i=1}^{n} load_i * distance_i}$$

The cost fraction is determined by the ratio of the load * distance product. Cargo on Milk Runs is usually measured by loading weight. Here the unit of choice is kg. The distance of a load-unload relation can be seen as the distance in km between its loading and unloading stop. However, it is also possible to use an arbitrary figure to achieve an artificial weighting of relations.

The computed cost for the load partitions has to be assigned to a location. Generic rules that specify which site pays for a delivery exist, but are sometimes set aside. They demand that

1. the delivery of goods is paid by the receiver,
2. the delivery of empty packaging is paid by the owner, no matter whether they are sending or receiving it,
3. in case of faults the cost is paid by the responsible party.

When a truck had run, it was usually the carrier who collected the necessary information on the load composition as well as on the timing. The data collection and subsequent partitioning of the cost were supported by several template documents Tenneco provided the carrier with. The most important ones were the MS Excel KPI file (Fig. 2.3) and a distributable MS Access database (Fig. 2.4).

The KPI files were utilized for accounting on Routes. They contained a line for each delivery on the Route with shipping date, truck no., arrival and departure times, load information for each site, the full price and the calculation of the locations' payment. They were to be updated in a two-day interval.

The MS Access database was intended for Milk Run organization. It allowed a rough setup of Milk Run location sequences and the creation of delivery records with an indication of the load per receiver. Three different

Route 1 Valencia - Alsace Truck Loading Metre Registration

Full Truck => 13,5 *Ldm*

Route 1: Valencia => Wissembourg (+Gent) => Fameck => Valencia

Depart Date	Start Site	Depart-Time	Trip Type: M=Mixed S=Single	Truck No (Input by carriers)	Loading: B=Benipri S=Sylla	Arrival Date	Time-Slots: Arrive-Wismbrg	Depart-Wismbrg	Arrive-Fameck	Arrive-Valencia	Ldm From Valencia: To WIS	To GENT (via WIS)	To FAM	Ldm To Valencia: From WIS	Fm GENT (via WIS)	From FAM	KPI: Total	Fill rate	Tariff Code	Payment by (example): WIS	Gent	FAM	VAL
CW14	*(Add lines for extra trucks or remove lines if trucks NOT running)*																						
Mo 04-avr.-05	VAL	13:10	M	CCE 8423	B + S	06. Apr 05	08:00	10:00	12:00?		1,5	4,0	4,0				9,5	70%	€ 1.672	264	704	704	0
Tu 05-avr.-05	VAL	13:00	M	CCE 7305	B + S	07. Apr 05	08:00	09:30	12:00		1,5	5,5	4,0				11,0	81%	€ 1.672	228	836	608	0
Tu 05-avr.-05	FAM	00:00	S	CCE 9227	S	06. Apr 05				22:00				0,0	0,0	13,0	13,0	96%	€ 1.356	0	0	0	1356
Tu 05-avr.-05	VAL	17:30	S	CCE 3126	S	06. Apr 05			21:00				13,0				13,0	96%	€ 1.399	0	0	1399	0
We 06-avr.-05	VAL	12:45	M	CCE 7216	B + S	08. Apr 05	08:00	09:00	11:30		3,5	6,0	4,0				13,5	100%	€ 1.672	433	743	495	0
We 06-avr.-05	FAM	13:30	S	CCE 7207	S	10. Apr 05				22:00				0,0	0,0	13,0	13,0	96%	€ 1.356	0	0	0	1356
Th 07-avr.-05	VAL	00:00	M	CCE 8129	B + S	11. Apr 05	08:00	09:00	11:30		1,5	6,5	4,0				12,0	89%	€ 1.672	209	906	557	0
Fri 08-avr.-05	VAL	12:45	M	CCE 5686	B + S	12. Apr 05	08:00	08:50	11:30		1,5	7,0	4,0				12,5	93%	€ 1.672	201	936	535	0
Fri 08-avr.-05	FAM	13:00	S	CCE 8278	S	11. Apr 05				22:00				0,0	0,0	13,0	13,0	96%	€ 1.356	0	0	0	1356
CW15	*(Add lines for extra trucks or remove lines if trucks NOT running)*																						
Mo 11-avr.-05	VAL	13:15	M	CCE 7207	B + S	13. Apr 05	00:00	00:00	00:00		1,0	7,0	3,0				11,0	81%	€ 1.672	152	1064	456	0
Mo 11-avr.-05	FAM	13:00	S	CCE 7216	S	13. Apr 05				22:00				0,0	0,0	13,0	13,0	96%	€ 1.356	0	0	0	1356
Tu 12-avr.-05	VAL	13:15	M	CCE 8465	B + S	14. Apr 05	08:00	09:15	12:30		2,0	6,0	4,0				12,0	89%	€ 1.672	279	836	557	0
We 13-avr.-05	VAL	13:45	M	CCE 5679	B + S	15. Apr 05	08:30	09:30	12:30		3,0	5,5	4,0				12,5	93%	€ 1.672	401	736	535	0
We 13-avr.-05	FAM	16:30	S	CCE 5686	S	17. Apr 05				22:00				0,0	0,0	13,0	13,0	96%	€ 1.356	0	0	0	1356
Th 14-avr.-05	VAL	13:30	M	CCE 8525	B + S	18. Apr 05	08:30	09:20	12:30		2,0	5,0	4,0				11,0	81%	€ 1.672	304	760	608	0
Th 14-avr.-05	FAM	00:00	S	CCE 7207	S	17. Apr 05				22:00				0,0	0,0	13,0	13,0	96%	€ 1.356	0	0	0	1356
Fri 15-avr.-05	VAL	13:30	M	CCE 9252	B + S	19. Mai 05	08:00	10:00	12:00		2,0	5,0	4,0				11,0	81%	€ 1.672	304	760	608	0
CW16	*(Add lines for extra trucks or remove lines if trucks NOT running)*																						
Mo 18-avr.-05	VAL	12:15	M	CCE 3126	B + S	20. Apr 05	09:00	10:15	13:00		2,0	4,0	4,0				10,0	74%	€ 1.672	334	669	669	0
Mo 18-avr.-05	FAM	15:00	S	CCE 3548	S	20. Apr 05				22:00				0,0	0,0	13,0	13,0	96%	€ 1.356	0	0	0	1356
Tu 19-avr.-05	VAL	12:00	M	CCE 8273	B + S	21. Apr 05	08:45	10:00	12:45		3,0	5,5	4,0				12,5	93%	€ 1.672	401	736	535	0
Tu 19-avr.-05	FAM	13:00	M	CCE 5679	S	21. Apr 05				22. Apr				0,0	0,0	13,0	13,0	96%	€ 1.356	0	0	0	1356
We 20-avr.-05	VAL	12:00	M	CCE 3548	B + S	22. Apr 05	09:00	10:00	13:00		0,0	4,5	4,0				8,5	63%	€ 1.672	0	885	787	0
We 20-avr.-05	VAL	16:00	M	CCE 1123	S	22. Apr 05			08:00		0,0	0,0	13,0				13,0	96%	€ 1.672	0	0	1672	0
We 20-avr.-05	FAM	14:30	S	CCE 5708	B	24. Apr 05				22:00				0,0	0,0	13,0	13,0	96%	€ 1.356	0	0	0	1356
Th 21-avr.-05	VAL	12:30	M	CCE 5579	B + S	25. Apr 05	08:00	09:15	12:30		1,5	8,0	4,0				13,5	100%	€ 1.672	186	991	495	0
Th 21-avr.-05	VAL	12:00	M	CCE 8369	S	25. Apr 05			13:30				13,0				13,0	96%	€ 1.672	0	0	1672	0
Fri 22-avr.-05	VAL	12:00	M	CCE 7218	B + S	26. Apr 05	08:00	09:30	12:30		2,0	5,0	4,0				11,0	81%	€ 1.672	304	760	608	0
Fri 22-avr.-05	VAL	17:00	M	CCE 8516	B + S	26. Apr 05	08:15	10:30	18:00				13,0				13,0	96%	€ 1.672	0	0	1672	0
Fri 22-avr.-05	FAM	13:30	M	CCE 1123	B	24. Apr 05				22:00				0,0	0,0	13,0	13,0	96%	€ 1.356	0	0	0	1356

Figure 2.3: A KPI file for Route 1

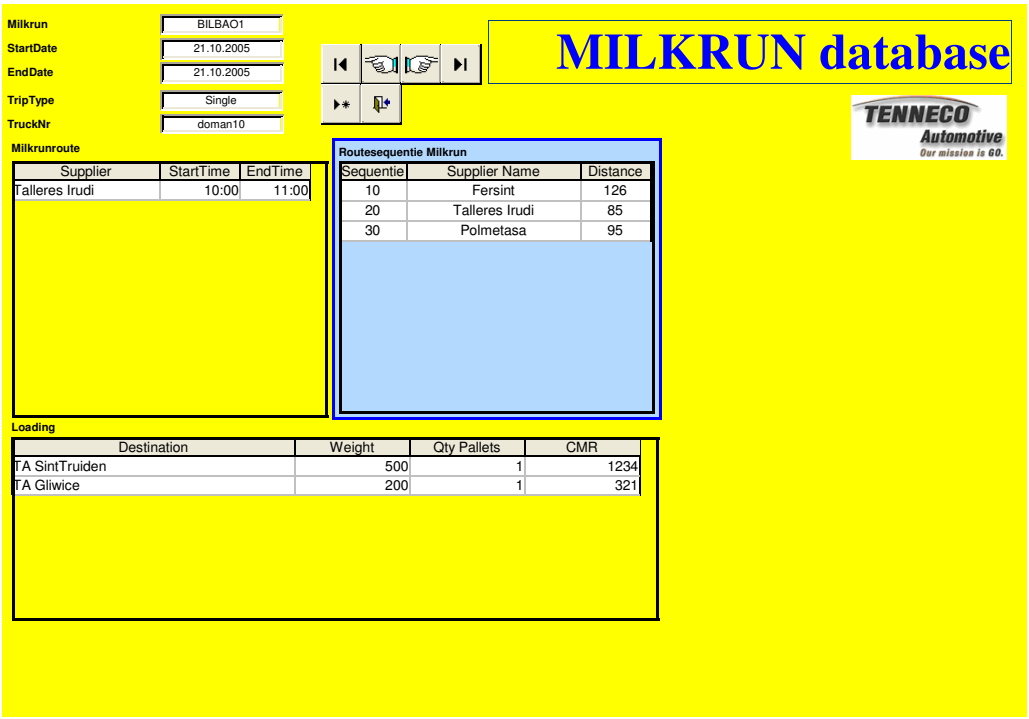

Figure 2.4: The Access Milk Run database

reports could be generated on the data entered. These were an invoice report on the accumulated payments for a certain month and location, a report on the truck fill rate for a certain month and Milk Run and an overview cost report. The overview report was statically created and a new template had to be added for each new Milk Run.

After collecting the information in the appropriate file, it was distributed manually to all parties involved. Invoices for the paying sites were created and sent out regularly, based on the calculated figures.

Tenneco's invoice clearing has been outsourced to AIMS Logistics (see Chapter 2.4.1). Thus, all invoices in the FIONA context go through this provider's working cycle.

2.3 Concerns

The observed shipment handling process entailed several complications and disadvantages. Overcoming them was the motivation that led to the FIONA project and the resulting system. The following paragraphs explain these issues.

In the analyzed environment transport booking was done in the way that was either requested by the carrier or had been negotiated for the specific service contract. The possibilities ranged from ordering single trucks per phone or fax to regular runs arranged once and repeated until cancellation. The latter was common practice especially on Routes.

As they were completely detached from this booking part of the logistics workflow, the pricing and statistics files were not directly generated from transport orders. They relied on a manual transfer of booking records after delivery, unless the carrier chose to use the files directly when the truck was booked. Since especially larger shipment companies use their own information systems to manage incoming bookings, this was rather improbable. A disruption in the data processing chain typically causes loss of time and precision. In addition, it gave rise to a certain reluctance to promptly and accurately fill out the provided files.

Loading and booking information was not commonly available before trucks had run and the completed documents had been distributed by the carrier. There was no real-time access to shipment information. Moreover, no central storage of such data was installed at all. This rendered changes of shipments or loading shares very laborious due to the lack of knowledge about fill rates and the missing communication platform. Negotiations in case of load collisions had to be organized and synchronized by the involved parties on their own account. Inefficient use of truck space was one immediate result. Another was the discrepancy between expected and actual payload: ICY sites often filled a truck with unannounced parts or less goods than originally ordered. In extreme cases an additional truck would have to be ordered when there were additional goods to be shipped on time and the truck seemed full, because one site did not announce a load decrease.

The booking data files varied in structure and file format, which made booking records from different Trip Schemes hard to consolidate. A central monitoring on data accuracy or statistics facilities for an overall controlling were not available. The checking of invoiced amounts against the load partitioning took various different shapes, often involving hand-written load or order lists. A commonly agreed verification process could not be identified during the analysis phase.

2.4 System placement

This chapter outlines the landscape of information systems relevant to the booking and invoicing process and explains the decision for a new development.

2.4.1 Existing systems

The first section simply introduces the different available information systems that were examined during the analysis of the project's environment.

Schenkernet

Schenkernet (cf. [Sch06c]), a system developed and hosted by Schenker AG ([Sch06b]), offers an online collaboration platform which is capable of Internet freight booking. Tenneco uses it with Schenker and a Spanish carrier as a solution to create the missing link between suppliers, carriers and Tenneco in Ride Control (OE as well as AM). In these inbound shipment scenarios, Schenkernet provides a dedicated view on the bookings matching the actor's role. A supplier is restricted o his supply relation, a carrier only works in his dedicated countries.

Apart from the web interface, there is a connection between Tenneco's relevant AS400 installations (see below) and the system. Schenkernet also provides an e-mail service and an Instant Messaging service allowing communication between the different parties.

An issue with Schenkernet is the suppliers' extensive freedom of initiating calls for transport, which imposes undesirable expenses on Tenneco: suppliers often split single deliveries into smaller shipments and e. g. send a truck three times a week, where only one shipment is expected. They also have the possibility to change the Transport Type of a booking to Premium Freight to compensate production delays on their side without Tenneco's consent. As Tenneco pays for the shipments in the present environment, this can quickly lead to cost explosions on the logistics sector.

Schenkernet is also offered as self-hosted software solution independent of Schenker, putting the licensee in charge of operation.

Transwide

Transwide (cf. [Tra06]) is a comprehensive Internet platform for freight booking and related activities. Its functionality is encapsulated in modules, which allow being combined according to the customers' needs. It follows a service provider paradigm and is therefore completely hosted by the Transwide company.

Many carriers are connected to it, ensuring a wide range of instant cooperation. Integration of existing information systems is made possible by a variety of interfaces. Additional connections can be implemented if agreed in an individual contract. The company also offers further individual adaptations of the system.

There are no plans to use this system in Tenneco logistics at present (see Chapter 2.4.2).

Tenneco express shipment database

All logistics staff throughout Tenneco have to request a supervisor's permission before booking an express transport.

Tenneco uses an internal software based on their Lotus Notes environment (cf. [IBM06a]) to conduct these approvals of premium Freight Shipments. Premium Freight requests have to include an estimation of the delivery cost in advance. The software provides a reference number for shipment identification, which is mandatory for the corresponding invoices since October, 2005. This way, all Premium Freight spendings can be traced back to the business case that originally caused them.

The database is utilized on a global level, but offers no facilities for interaction with any ERP system or connectivity to suppliers, customers or carriers.

Rybnik freight booking tool

Another application based on Lotus Notes is employed in the analyzed freight booking environment: The Tenneco plant in Rybnik, Poland uses it internally to manage its transports which are handled by the local carrier Pamtrans (cf. [PTS06]).

The software is capable of attaching SAP identification numbers to stored data but has no further connection to this information system.

AS/400-based software

Several Tenneco locations use ERP Systems or similar applications based on IBM's AS/400 (Application System 400). AS/400, in the current version known as "System i", is a computing system consisting of dedicated hardware and a specific operating system (see [IBM06b]).

These installations are considered deprecated in the Tenneco environment and are bound to be replaced with advancing SAP deployment.

SAP

Tenneco utilizes and extends the use of ERP software provided by the SAP AG (cf. [SAP06]). Increasing coverage of sites is a major plan for the coming years. The system is not used for transport booking or carrier system

integration but one of its tasks is the management of information about demand for goods and resulting orders. This data in turn is the prerequisite for transportation requests and is transfered from the SAP system for example to SupplyWEB for further processing.

SupplyWEB

SupplyWEB is a Supply Chain Management (SCM) software by Infor Global Solutions ([Inf06]). It "integrates information from disparate ERP systems used by various suppliers to create a single, universal, real-time view of supply chain operations for suppliers and internal users" (cf. [Inf04]).

Tenneco's suppliers access it via the web interface which can, along with other supplier-relevant services and the SupplyWEB training, be reached through the company's Internet portal for suppliers (at [Ten06]). The suppliers are provided with data filled by Tenneco's ERP systems, especially from the SAP environment. Part-level purchase order information based on standard delivery notes is available as well as track and trace information.

The use of SupplyWEB is still being consolidated and different trainings are conducted by Tenneco.

AIMS Logistics

AIMS Logistics (cf. [AIM06]) is a company that offers its customers individual freight payment auditing and processing along with a growing portfolio of related add-on services.

Although this is a provider service and no information system in software, it is mentioned in this chapter, because it was an important factor when possible alternatives to developing a new system were examined (see Chapter 2.4.2).

Tenneco has outsourced its invoice auditing and clearing to this company, so that all invoices in the FIONA project domain pass the validation process there. To prepare AIMS for this task, Tenneco preprocesses all new tariffs negotiated with a carrier and forwards the refined information in MS Excel files to AIMS.

Apart from regular reports provided by AIMS via e-mail, an Internet database can be queried for controlling purposes. The web interface allows exporting the results to MS Excel files which are then employed for internal reporting and plausibility checks by Tenneco logistics supervisors. Unique client numbers identify the different sites involved in the transaction that originated an invoice.

Timing is a major concern with the controlling data coming from AIMS. It usually is accessible around two months after the shipment, which is insufficient for realtime reporting and fast reactions to possible deficiencies.

2.4.2 Gap analysis

Before the decision in favor of developing a completely new application was made, the existing information systems and services described above were examined. The goal was to find out if one of the existing solutions was already capable of overcoming the presented issues (Chapter 2.3) or if it was possible to efficiently extend one of them to do so.

Of course, many of the demands, that shaped the solution to find, had to be already formulated or anticipated to prepare this evaluation. Yet, the requirements specification document itself was oriented towards FIONA after the decision was made and is therefore dealt with in Chapter 3).

Schenkernet quickly became out of the question because its focus lies on the sector of online collaboration and it lacks the essential cost splitting functionality (see Chapter 3.1). It also provides no opportunity to manage different tariff models, especially for different carriers.

As a solution hosted by a service provider that did not have any business connection with Tenneco so far, Transwide was rejected by business decision on supervisor level. An in-depth analysis, however, would have most likely shown that it could have covered most — if not all — of the functionality needed, but at the cost of a transaction-based pricing model.

The Tenneco express shipment database serves a completely different purpose: Its sole functionality is the ex ante authorization of Premium Freight transport bookings. This is a feature that might be a later extension to the system in demand, but covers none of the key issues.

The freight booking tool for Rybnik is just an isolated application matching the specific needs of the local plant–carrier constellation it was developed for. Accordingly, it was judged ineligible, as well as the AS/400-based systems that are to be superseded by ERP software on the basis of SAP.

SAP, SupplyWEB and AIMS required a more detailed investigation. A list of key demands was set up, to help compare the three candidates with a proposed new system in a Gap Analysis (Fig. 2.5).

The percentage values indicate the estimated fulfillment or availability of the corresponding feature in the specified solution. They are not to be taken verbatim, but show a tendency for realization effort to make the solution fit the requirements. A high value means that the functionality is nearly reached, while a low value denotes high implementation effort.

No.	function	FIONA	Sup.Web	AIMS	SAP
1	Internet-based (access from any internet-capable PC)	100%	100%	75%	50%
2	Derivation of required packaging from material order	0%	70%	0%	70%
3	Processing & monitoring of freight shipment bookings (daily business)	100%	50%	0%	30%
4	Trip Schemes are predefined (optimized daily business)	100%	50%	0%	30%
5	Connection to tariff DB (latest changes of carriers or tariffs integrated)	100%	0%	100%	30%
6	Automated E-mail notification	100%	50%	0%	30%
7	Easy to negotiate changes of order or shipment	100%		0%	
8	Common visibility of freight data while in transit	100%		0%	
9	Transparency on truck usage before, during and after delivery	100%		30%	
10	Automatic invoice split based on agreed truck usage calculation	100%	0%	0%	30%
11	Avoidance of manual loading info distribution	100%		30%	
12	Pre-audited invoices	100%		0%	
13	Visualization of shipments not yet invoiced (=> accruals)	100%		0%	
14	Archive cleared booking records	100%	20%	100%	30%
15	Query booking records	100%	30%	100%	30%
16	Easy to consolidate freight costs	100%		100%	
17	Integration with Confluentic freight planning	100%		50%	
	Average:	**94%**	**41%**	**34%**	**37%**

Figure 2.5: Gap analysis chart

Entries for SupplyWEB and SAP are empty if Tenneco failed to provide information on the possibility of extension and related workload. These fields are not included in the average values, but are considered unrealizable or high effort for the following remarks.

A proprietary development naturally matches the business demands, as it has to be developed with their realization being the goal. Thus, the fulfillment for FIONA is given as 100%, except for item 2: Deducing the sort and amount of required packaging containers from a list of ordered supplies is a general issue in logistics. It requires a set of nomenclature and technical description for containers which is used uniformly by all parties involved. Then a relation associating parts and their packaging units with possible or, better, the most efficient containers has to be set up. This would allow, accepting a certain inaccuracy, the determination of packaging to dispatch for a planned shipment.

Since neither the deduction rules, nor the fundamental container taxonomy exist at a stage that is valid for a sufficiently comprehensive part of Tenneco, the realization was not a realistic goal for the diploma thesis. In addition, this functionality lies out of the project's main focus: a treatment of bookings on part level is not required to accomplish the demanded streamlining of the freight booking and invoicing process. SupplyWEB and SAP already handle information about ordered goods and therefore are ahead on this sector.

Building an Internet application was a key requirement crucial to easy realtime access to booking information for all parties (item 1). The readiness

for this feature decreases from SupplyWEB over AIMS to SAP: SupplyWEB is an Internet application, AIMS offers a database query web interface and the SAP present installations would need Internet enhancements, which were possible but not implemented when the analysis took place.

The overall scenario of daily freight booking business, based on an initial setup of Trip Schemes (items 3 & 4) would be partly supported by SupplyWEB, which already handles relations between Tenneco and its suppliers. AIMS offered no adequate facilities at all and SAP required high implementation effort to add the required functionality.

A connection to the CaTIS tariff database developed in parallel (see [Kop06]), to ensure access to the latest and most accurate carrier and tariff data had to be taken care of (item 5). SupplyWEB had no preparations for this, while AIMS offered integration of customer databases by individual agreement. SAP again would have required substantial extending for this as well as for integrating an e-mail notification component (item 6). AIMS did not offer any possibility for e-mail notification, while SupplyWEB already partly included such facilities.

Features that belong to the domain of cooperation functionalities and tracking in-transit information (items 7, 8, 9 & 11) were not offered by the three existing solutions. AIMS only provides online access to shipment information after the delivery and the issuing of invoices.

Any processing of invoices prior to their transmission (items 10, 12 & 13) was also not intended or realizable with reasonable effort in the existing environment. Especially the automated calculation and splitting at the time of the booking or load allocation, which enables auditing and calculation with future expenses (accrued figures) was missing here.

Data preservation and query functionality to be consulted for controlling purposes (items 14, 15 & 16) was already included in AIMS: there shipments can be identified via the collected invoices. SupplyWEB and SAP could also be extended to provide such information, as they store purchasing records which can be related to the corresponding deliveries. However, this would have caused significant implementation effort in both cases.

Similar to the integration of CaTIS (see above), the freight planning tool Confluentic (see Chapter 1.1) also had to be considered to provide matching data exchange mechanisms if necessary. SupplyWEB and SAP had no facilities here. AIMS, as mentioned before, offered data exchange with arbitrary systems, if agreed by contract. Yet, its focus was different from what was needed for freight planning optimization, it was especially not designed for the association with installed Trip Schemes and lacked promptness in the delivery of shipping information.

The results showed that SupplyWEB, AIMS and SAP did not provide

the required functionality at the time of the gap analysis.

AIMS had to be sorted out due to its differing focus and the fact that it is a solution completely based on a service provider contract.

SupplyWEB did not offer enough adaptation possibilities, individually implemented extensions were not part of its design or the contract with Tenneco.

SAP would have most likely allowed being enhanced to the required extent, but the programming would have taken plenty of time (as did the development of the new system) and possibly the assistance of additional external SAP specialists. Such an extension would also have affected the existing installations of SAP systems, imposing a serious risk on the project, while a new system can be more freely developed and deployed without any impact. The sites throughout the company where SAP solutions are not yet in place could have been given access via the web interface, but the exchange of fundamental master data would still have to be accomplished through auxiliary constructions. In addition, the development at a university in the course of a diploma thesis is usually significantly cheaper than the binding of internal development resources. There is usually only sparse additional capacity available during growing and optimization phases of a company and the purchasing and incorporation of external staff tends to immoderately raise cost.

Next, the detailed requirements were aligned with the plan to develop a new system and used as guidelines for the following project phases.

Chapter 3

Requirements specification

During the analysis, the scope of the project was narrowed down to FTL shipments. They form a representative transport domain and required the most expansion and information system support at the time of the project's start. The other Transport Types had been examined and could enter the concept to allow future extensions (cp. Chapter 2.2.1).

Later in the designing and implementation phases, the business supervisors demanded to explicitly exclude Pure FTL, although it was already covered with a processing that was analog to the one used for Routes. In order to fulfill this requirement, additional exclusion logic had to be added to various sections of the system.

The Milk Runs included in the planning for the project's pilot phase are inbound only, but the system scope covers both directions of this Transport Type.

The proposed target scenario was divided into several use cases to help identify the different user groups, partial processes and interaction patterns. Fig. 3.1 shows the resulting diagram.

The visualization of responsibilities and actions was helpful in formulating a base process for the system (see Chapter 4.1) and in dividing it into different modules (Chapter 4.2). It also was a good starting point for the organization of the requirements specification.

The actual requirements specification document is attached in the appendix on CD (Appendix D), the following chapters give a high-level overview and address the general interrelations and especially the changes in scope or realization that were made after the document had been settled.

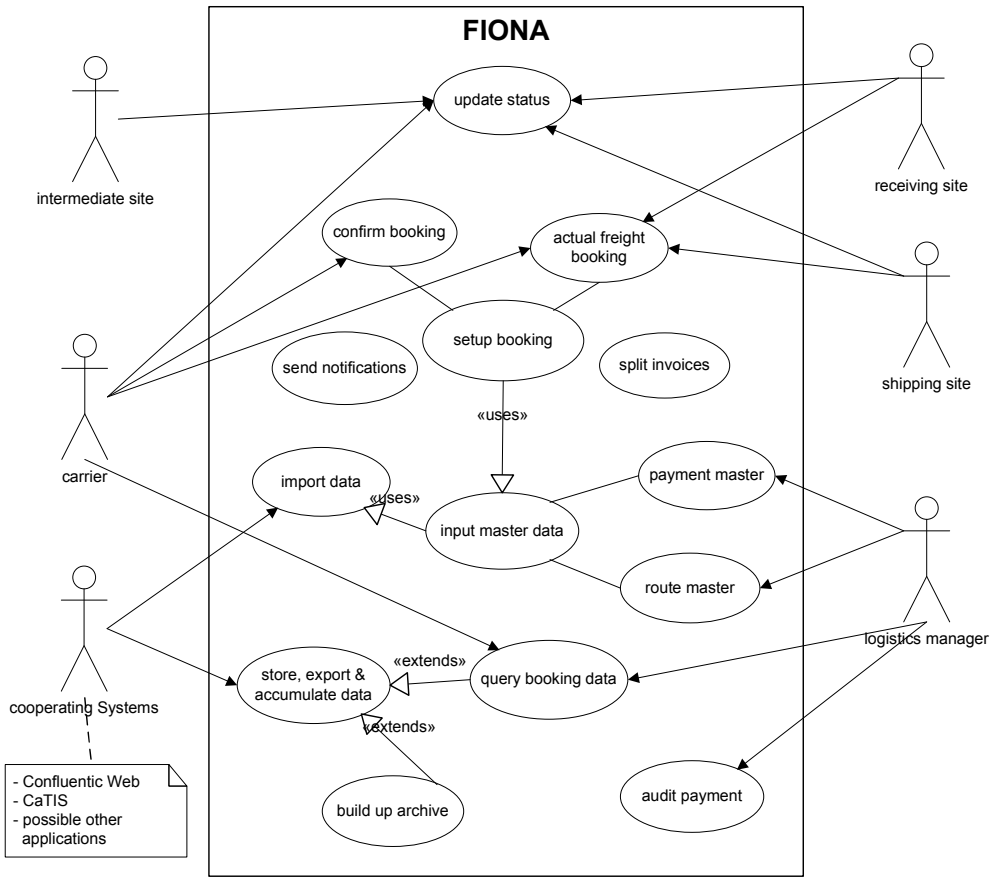

Figure 3.1: Global Use Case diagram

3.1 Functional Goals

The functional goals were implicated by the business issues Tenneco had with their existing freight booking and invoicing process and together formed the envisioned concept to improve the situation.

3.1.1 Logical user groups

Access control with different authorization levels is a key requirement for web-based multi-user information systems like FIONA. It protects critical data from unauthorized manipulation or viewing and also helps the users to focus on the data relevant to them. Any information that is of no interest in their context can be suppressed.

All required user categories were identified by going theoretically through the relevant booking and invoicing processes. They could be distinguished

by their relation to the company, focus of interest and responsibilities. According to their roles in the transport environment, the users have to be presented different portions of the information that is generated during the logistics process.

FIONA Administrators

Tenneco's logistics managers are in charge of the operational introduction of the system, organize Trip Schemes and open them for freight booking by the involved parties. In the analyzed environment they also work on the statistics sector. They are the contact persons for the tariff announcements of the carriers as well as the invoice auditing services of AIMS (see Chapter 2.4.1) and oversee the transport coordination. Thus, a user group combining these top-level tasks had to be created.

FIONA Administrators are responsible for the creation of new users, they administer Trip Schemes by entering required master data and default information. They evaluate the booking statistics and extract necessary operating figures to enable controlling after shipments have been completed.

Trip Scheme Owners

A new shipment is ordered by a person at the location responsible for the underlying Trip Scheme. There usually is one site on a Trip Scheme that is in charge of this, but no general rule for the assignment could be identified. Therefore, the appointment of this Trip Scheme Owner had to be designed to allow individual configuration for each Trip Scheme.

In addition, the initiating user can be internal or external to Tenneco. In some cases an agreement is in place, allowing even the carrier to start a transport. This had to be taken into account when setting up the user groups for the system.

Trip Scheme Members

Trip Scheme Members, as the Owner group, are situated in locations that are part of a Trip Scheme. These users have less rights, meaning that they are not allowed to call for new shipments. On existing bookings, they basically change their part of the truck's load allocation.

This group is a mixture of both, Tenneco internal users and employees of supplier and customer companies. Its composition depends on the Transport Type of the associated Trip Scheme.

Carriers

Carriers need to have immediate access to new booking records, they have to be able to send the trucks based on the information given by the system. An additional booking confirmation mechanism improves the handling of planned and agreed shipments.

In order to split the invoices correctly, carriers have to be presented with the calculated cost portions. Accumulation is required to allow invoicing for arbitrary periods of time.

Since transport companies usually have the relevant tracking & tracing information, status updates to shipment records can most conveniently be done by this user group as well.

3.1.2 Invoice splitting

After the first approach of having varying cost splitting algorithms for the different Transport Types, further workshops showed that Tenneco's logistics supervisors committed themselves to the cost calculation algorithm used in Milk Run scenarios. Accordingly, the appendant formula had to be integrated as the standard calculation rule for cost splitting:

$$price_i = tariff * \frac{load_i * distance_i}{\sum_{i=1}^{n} load_i * distance_i}$$

However, the extensibility of the calculation functionality, allowing to have algorithms added or changed by the development department, still had to be ensured.

Due to the absence of universally valid regulations, the decision which site in a load-unload relation has to pay for a shipment had to be implemented as configurable individually for each of these relations (see Chapter 2.2.2).

Paying sites and Administrators had to be given access to the calculated figures to allow the previewing of prices before invoices are issued and to support their auditing afterwards.

3.1.3 Data export

To simplify the process of invoicing, a file export of the location-based cost information had to be provided.

As MS Excel is already widely used in various sectors of the target environment — it is utilized to announce tariffs, serve controlling activities and support the propagation of new business concepts in Tenneco — the output file format was already predestined.

An agreement exists between Tenneco and its carriers, defining an electronic format for the exchange of invoices. The columns in the export file had to be designed to correspond with this pattern. Fig. 3.2 shows the agreed set of fields for electronic invoices.

Field name	Description	Type	Length	Dec.
currency	ISO currency code	char	3	0
mastinv_no	Master invoice no.	char	25	0
inv_date	Invoice date	date	8	0
pronumber	Shipment number	char	25	0
prodate	Shipment date	date	8	0
pieces	Amount of pieces	num	12	2
weight	Weight	num	12	2
mode	Transport mode (Ground, Air..)	char	10	0
serv_level	Service level (Normal, Premium...)	char	35	0
serv_Categ	Service category (Box, Packaging...)	char	35	0
consignor	Consignor name	char	35	0
origin	Consignor zip	char	10	0
org_cty	Consignor country	char	2	0
org_city	Consignor city	char	20	0
consignee	Consignee name	char	35	0
dest	Zip of destination	char	10	0
dest_cty	Country of destination	char	2	0
dest_city	City of destination	char	20	0
billed	Shipment billed amount	num	14	2
vat	Vat amount	num	14	2
fee1_id	Transport cost	char	10	0

Figure 3.2: Tenneco electronic invoice format

The complete list could not be covered, since not all of the relevant information is available to FIONA, e. g. the invoice date is neither determined by the export date, nor known to the system in any other way.

Because the existing KPI files (see Chapter 2.2.2) had to be replaced by FIONA not only by means of web interface representation, but also in the form of distributable files, a KPI file export from the system had to be created in addition to the invoice data output. This export is Excel-based as well and had to be prepared to provide the same set of information as the original files.

The actual realization of the export functionalities is dealt with in Chapter 4.1.6.

3.1.4 System integration

To avoid data redundancies and facilitate cooperation, the course in database design had been set for a single database scheme combining all new Tenneco logistics projects which are realized at the University of Koblenz-Landau. The first projects affected by this decision were CaTIS, FIONA and confluentic WEB.

As the data sets processed by confluentic WEB and FIONA barely overlap and the former is a merely conceptual Diploma Thesis with downstream development by Tenneco IT, the database implementation itself was hardly affected by this project. During design, several columns of shared tables and commonly used terms had to be considered to make sure the database model of confluentic, redesigned by confluentic WEB, could be seamlessly integrated into the new scheme.

By contrast, CaTIS was developed concurrently with FIONA and a significant amount of data is shared by both systems: CaTIS provides a major part of the required master data. Therefore, the design and implementation of the two projects had to be conducted in close cooperation. This affected basically the access to current transport pricing information, along with contract and carrier data. Information about vehicle types is shared as well.

Yet, functionality for system-internal manual adjustment (overriding) of prices had to be included in FIONA as a second instance to keep flexibility and deal with missing or incorrect tariff information.

An additional requirement, that was only raised late in the implementation process, was to allow that finished shipments are affected by the back dating of tariff changes. These will be done via tariff revisions in CaTIS (cf. [Kop06]) and seem to be common practice in Tenneco's logistics environment. The inclusion of this possibility to change settled booking records retroactively required the reworking of several parts of FIONA's concept.

3.1.5 E-mail notification

To keep the system's users up-to-date and help them react on time, an e-mail notification component had to be created as part of the software. It has to announce important booking updates, time-critical information and error messages that require a user's reaction. Examples for relevant cases are the creation and the decline of a new shipment request. For plain reporting a presentation via the web interface or a file export were preferred.

As a minimum processing effort for the user had to be achieved, the notification messages should only contain summarized information in a compact form. In addition, a distinction according to the user groups' different scopes

of interest was required, so that they receive only information that is vital in their context. Flooding the users with irrelevant data tends to make the essential information perish and had to be avoided.[1]

The initial plan to implement a part of FIONA's e-mail notification feature using a detached observing process (e. g. a cron job, see [TT04]) to provide time-interval or deadline triggering of alerts was discarded. An event-based implementation as part of the systems business logic proved sufficient for the first version of the software.

The actual concept of notification with the organization of user groups and trigger events is described throughout Chapter 4.1.

3.2 Technical Goals

This chapter outlines the technical requirements, that were mainly driven by the IT department of Tenneco.

The base premise for the system's technical realization was to ensure that the system can be maintained by Tenneco's development division. A complete handover was intended from the beginning on. Therefore, the software implementation had to be conducted in compliance with Tenneco's software development directives.

The software engineering division based in the USA provided the set of Java design guidelines ([Ten05b]) and coding standards ([TA 05])[2].

They demand a multi-layer architecture for web applications. The Model View Controller (MVC) oriented approach (see Chapter 4.2.1) is build around the use of the Jakarta Apache Struts framework (www: [Apa06b]). This implies the use of Java Server Pages (JSP) for the display processing that performs the composition of the web interface and Java for the business layer, including all process logic (www: [Sun06b] and [Sun06a]). The demanded means of persistent data storage was an Oracle database (www: [Ora06b]). All query and data manipulation logic had to be set up in the database itself to be invoked by the applications's business logic.

The user account validation had to be realized via the facilities that were already established in the Tenneco information system environment. These are an Active Directory (AD) for internal users and Active Directory Application Mode (ADAM), AD's lightweight version, for the connection of external users (cf. [Mic05a] and [Mic05b]). For FIONA, these external users can

[1] A high "signal–noise ratio" is desirable to achieve acceptance and effectiveness with notifications or warnings. (cf. [Nie03] and [Car03]).

[2] The Tenneco Automotive Java Coding Standards are a corporate revision of Sun Microsystems' Java Code Conventions [Sun99].

be employees of carrier, supplier or customer companies. The task was to prepare the software for the integration of the two authorization systems. It was not possible to completely implement the integration mechanisms, since the university was not given access to these systems and a matching test system was not available on-site.

Tenneco uses a WebSphere Application Server by IBM to host its web applications and log4j to output system messages for debugging and monitoring purposes (www: [IBM06c] and [Apa06a]).

The detailed list of platforms, languages and versions to be employed comprises the following entries:

- IBM WebSphere Application Server 5.1
- Java Version 1.3.1
- Jakarta Struts 1.1
- Log4j 1.2.8
- Oracle 9.205 JDBC (classes12.zip)
- Oracle 9i Database

Extensive source code documentation was an obvious additional key requirement to assist future developers in understanding the structures, concepts and processes behind the software. The code documentation is attached on CD (see Appendix G).

To ease further use and adaptation, possible extensions and additional functionality had to be taken into consideration during development. Especially localization capabilities that allow different languages, currencies, units of measurement etc. had to be implemented in a modular way. This will enable later projects to extend or change these quickly.

Together with the functional requirements, the mentioned technical demands set the course for the project's software design.

Chapter 4

System design

The design phase of the project partially overlapped the development phase, as many aspects, especially exceptional cases, only surfaced when business supervisors were confronted with concrete realization ideas by the project team (see also Chapter 1.2). This showed that the approach of alternating analysis work and result presentations was a good choice.

FIONA's operating model is derived from a proposed process that results from organizing the booking-related activities identified while analyzing and fixing the requirements (see Chapter 3). This in turn influenced the architectural layout of the system.

4.1 Main process

Actions performed by the users of FIONA in the logistics environment can be arranged in sequence to represent the logical workflow of shipment preparation, execution and post-processing. Fig. 4.1 demonstrates this structure.

It shaped the design of the software and was also the base for several presentations held during project reviews.

Three phases can be identified, that subdivide the process: Before daily business activities start, the system has to be configured with elementary master data. This comprises basic user setup as well as the preparation of the booking environment.

The next part of the operational model includes all regular logistics activities that usually accompany the shipment of goods and empty packaging, from ordering transports and their confirmation to changing load or entering actual arrival times.

To further process the data collected while executing this daily work, it can then be aggregated using different statistics functionalities.

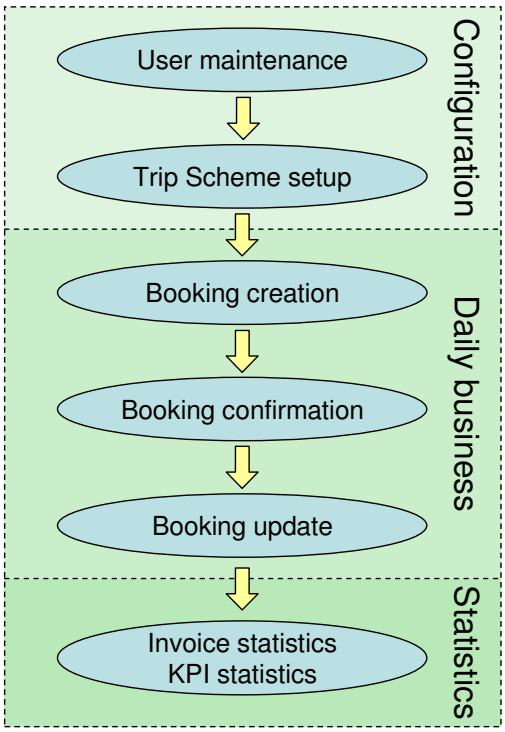

Figure 4.1: FIONA main process

The tasks to be performed along this process chain are described in detail in the following sections.

4.1.1 User maintenance

User maintenance comprises adding users to the system as well as the account configuration that determines their responsibility in the transport process.

The introduction of new user accounts is a task that depends on several factors. A first distinction has to be made according to the future user's company membership. As mentioned in the description of technical goals (Chapter 3.2), external users have to be authenticated via Tenneco's ADAM while internal employees are present in the company's AD.

Authentication

Integrating these systems led to a setup that does not store the basic user accounts, especially the passwords, in FIONA. Instead, the system validates them by accessing the matching service and stores only additional FIONA-specific information, like the mandatory e-mail address, in its own database.

This entails going through a complex decision process when a new account is added to FIONA. An excerpt from the corresponding decision table is shown in Fig. 4.2.

	R3	R4	R7	R8
Conditions				
Internal user	Y	N	Y	N
User exists in AD/ADAM	Y	Y	N	N
Actions				
Add to ADAM				X
Add to FIONA	X	X		X
Error: not in AD			X	

Figure 4.2: User insertion — decision table excerpt

Internal users that already exist in the AD (R3) can simply be added to the FIONA database. This is usually the case as user accounts for all major information systems in the Tenneco environment are managed there via the central AD. If the username cannot be found in the AD (R7), it will not be added to FIONA since the application does not have sufficient access rights to insert it into Tenneco's AD first. The user has to be given an AD account by some dedicated management tool first.

For external users the proceeding is different: If the user exists in the ADAM, it can also simply be added to FIONA's database (R4). But since new users might be given access over ADAM for the first time when they are added to FIONA, they have to be added to ADAM as well if they do not yet exist there (R8). FIONA offers routines that call the corresponding Tenneco functions to realize this action.[1]

When an account is added to FIONA, it has to be associated with an authorization level that determines the activities the user is allowed to execute.

User groups

The requirements specification yielded a catalog of four logical user groups (see Chapter 3.1.1). On process level, these groups required a reorganization because of incompatibilities in their scopes. A system of groups and roles was the result.

[1] As mentioned in Chapter 3.2, the integration had to be completed by Tenneco's corporate developers.

The new user groups are defined on application-level and allow the same user being added to only one of them. At present, FIONA offers four groups to choose from:

- FIONA Administrators
- External users
- Internal users
- Carriers

The user group with the highest level of permission is the FIONA Administrator group. It directly corresponds to the logical group of the same name.

Internal and external users both operate on a lower, the standard user level. They are only distinguished by their company membership and the resulting authentication means. Internal users are usually situated in ICY locations, external users work for suppliers or customers. This distinction also allows later extensions of the system if display restriction becomes necessary to prevent external users from accessing certain confidential information. Users from both groups are associated with their (internal or external) location on setup.

Carrier users are external as well, but need to be given a different view on the system as they have to receive and confirm bookings and create invoices. In addition, they are not connected to a site but to a carrier company. Thus, an additional group was created to cover these requirements. This group also matches the logical user group of the same name.

In contrast to the resulting system-wide authorization mechanism, some permissions have to be defined on a more fine-grained level.

User roles

The user roles provide for user rights that can be fixed individually for each Trip Scheme. At present, one of two user roles can be given to a user:

- Trip Scheme Owner
- Trip Scheme Member

As indicated by the names, these roles realize the demands that were defined when the corresponding logical user groups were identified. Owners are responsible for the booking of shipments, while members only work on their part of existing bookings.

A user's role for a given Trip Scheme is determined when the Trip Scheme is prepared for usage.

4.1.2 Trip Scheme setup

Setting up a Trip Scheme means enhancing an existing Trip Scheme with the information that is required to conduct shipments on it via FIONA. FIONA Administrators are responsible for this task.

The Trip Schemes are constructed from basic master data which is created via initial imports from CaTIS. The existing raw data consists of a sequence of locations that are visited along the trip and a list of associated vehicle types and tariffs that determine the Transport Type and pricing.

FIONA does not allow bookings on a Trip Scheme before it has been enriched with all necessary processing information.

First, time limits and user groups or roles responsible for entering real timing and real loading information are set.

The tariffs that Trip Scheme Owners will be allowed to use for bookings have to be chosen from the list of all associated tariff – truck type combinations. FIONA's own query logic is employed to retrieve the tariff information directly from the database, as the data preparation is FIONA-specific and the development of CaTIS was at a different point, when the functionality was required. Yet, no redundant information is kept or created for this purpose, the same set of records contains the relevant information for both applications.

The list of visited locations can then be supplemented with default arrival times which are suggested to the booking user. This is also the stage at which cross-docked locations can be added to a Trip Scheme.

Next, the load-unload relations between the specified locations are defined. For each created relation, the FIONA Administrator has to enter a distance that will be used for price splitting (see Chapter 3.1.2). Additional input comprises the determination of the paying site (loading or unloading location) and the type of cargo (goods or empty packaging). Default loading amounts for each of the relations can also be added, if desired.

The last step is adding the users that are supposed to work on the Trip Scheme. By definition, FIONA Administrators have the right to create and edit bookings on all Trip Schemes, so they cannot be added. External and internal users can be chosen if their location is part of the trip's sequence. They have to be given the appropriate role when assigning them to the Trip Scheme.

Carrier users are automatically granted access to bookings which were created using one of their tariffs, so they usually do not have to be explicitly added. Contract setups which allow the carrier to book a truck are the exception to this rule: in this case, carrier users may be added to a Trip Scheme — they will automatically be given the Trip Scheme Owner role.

4.1.3 Booking creation

The Trip Scheme Owner chooses from the list of Trip Schemes he is allowed to book on, and is then presented a new booking, filled with the default timing and loading information from setup.

In the ideal case, only the shipping date has to be entered. If more than one tariff is available to the user, they can select the desired one. Again, the tariff and vehicle information is queried using FIONA's own data access logic.

If necessary, the load for all associated load-unload relations, as well as the planned arrival times, can be adjusted while setting up the booking. Adding an additional comment is also possible. Overloading the truck is prevented using the maximum load specified in the vehicle type. The price portions for all load-unload relations are calculated immediately when the booking is stored. A preview calculation functionality is available that allows checking the prices before saving.

After finishing the creation of a new booking, FIONA sends out a notification e-mail to the responsible carrier and all users associated with the underlying Trip Scheme. If the booking is late, the e-mail will be specially marked and also sent to the FIONA Administrators.

A booking is considered late when the time between the shipping time and the booking's creation falls below the booking lead time agreed between Tenneco and the carrier. The booking lead time is part of the contract information saved in CaTIS — if it is not set, FIONA assumes a default buffer of 24 hours. The shipping time is determined by the shipping date and the arrival time for the first location. A late booking requires an explicit confirmation from the user creating it.

4.1.4 Booking confirmation

Once a booking has been created, it appears in the list of new bookings for the associated carrier. From there, it can be confirmed or declined. When a carrier confirms a booking, it is marked accordingly in the user interface and, in addition, can no longer be deleted. The intention is to provide a means for the carriers to announce their agreement and to guarantee that a delivery will be performed.

A shipment that has not been confirmed can still be executed, as it is available to all responsible parties without restriction. It was a business requirement to ensure that trucks can run, although a carrier may fail to use the confirmation functionality.

If the carrier refuses to accept a transport assignment, they can use the decline function to mark the booking appropriately and enter an explanatory statement. The booking will stay in the system for controlling purposes, but changing it will not be possible.

The decline event triggers a notification e-mail to all users associated with the booking's Trip Scheme. A copy goes to the members of the FIONA Administrator group.

4.1.5 Booking update

Updating a booking comprises all changes to an existing booking except for confirmation and decline. The different types of activities will be described in supposable chronological order.

Generally, each change to a booking that happens before the shipping date triggers a notification e-mail to all associated users. Exceptions or extensions to this rule will be specified where existent.

Deletion vs. revocation

A booking that has not yet been confirmed can be deleted by the responsible Trip Scheme Owner or a FIONA Administrator. This function is disabled on confirmation and replaced by the revoking operation. In contrast to a deleted booking, a revoked booking will stay in the system. It is, similar to the effect of the decline feature, marked accordingly and locked to prevent further editing.

Both methods of canceling a shipment are obviously only available before the shipping date. Completed booking records may not be deleted from the system for controlling and tracking reasons.

When a booking is deleted or revoked, the corresponding notification e-mail to all associated users will be sent also to the carrier. If a late deletion or revocation occurs, the proceeding is analog to that of late booking creations (see Chapter 4.1.3): The e-mail is specially marked and an additional copy is sent to the FIONA Administrators. A cancellation lead time agreed in the contract is used for both, deletion and revoking; FIONA's backup value is again 24 hours.

Load changes

The editing functionality for existing bookings allows changing the truck's load allocation before the shipping date arrives. On which load-unload re-

lations users can edit the load amount is determined by their group and role.

FIONA Administrators can change the load allocation of all shipments. Trip Scheme Owners are allowed to edit all loading information for each booking on a Trip Scheme they own. Trip Scheme Members only have write access to a relation if it is paid by the location they are associated with.

As for the booking creation, the cost splitting is done automatically when saving, but can be previewed via a dedicated calculation functionality.

Status updates

To serve tracking and tracing purposes, FIONA offers a status indication for each booking record. As the business requirements did not allow to demand entries that could block the shipment execution if they were not provided, the status functionality does not trigger any action.

However, it can help to observe and and categorize bookings if used consequentially. This is supported by search functionalities that support filtering by booking status.

The current range of possible status values comprises four items:

- Booked
- CMR-checked
- In transit
- Delivered

The restriction to these values was a business requirement, but the list can easily be augmented by adding entries to a master data table (see Appendix F).

All users associated with the corresponding Trip Scheme, responsible carriers and FIONA Administrators can change a booking's status. The decision who will be obliged to do so depends on the transport constellation. As most carriers already track their deliveries to some extent, having them enter this data is a convenient option.

Real timing and loading information

After the shipping date, a shipment record can be enriched with real timing and loading information to allow identifying discrepancies between the booked and the actual data.

Which users are responsible for these entries is configured during Trip Scheme setup (Chapter 4.1.2). The editors for real arrival times and loading are defined separately, each by choosing from the following alternatives:

- No real timing/loading information is entered
- Carriers enter it
- Trip Scheme Owners enter it
- Trip Scheme Members (and Owners) enter it

In all cases FIONA Administrators are allowed to edit both, real timing and loading data.

As invoices are expected to be based on the actual data, the transition from booked to real loading information is crucial to cost splitting.

This challenge is handled using a copy mechanism: Since only the booking (i. e. planned) values can be set before the shipping date, they are silently copied to the fields for the actual values. With the arrival of the shipping date, the booked information is closed and only real data will be edited. Thus, prices can always be calculated using the real loading information entries. If the controlling feature is utilized, they will finally contain the actual values. The booked data remains available for evaluation purposes. Fig. 4.3 illustrates this procedure, which is also applied to the timing information.

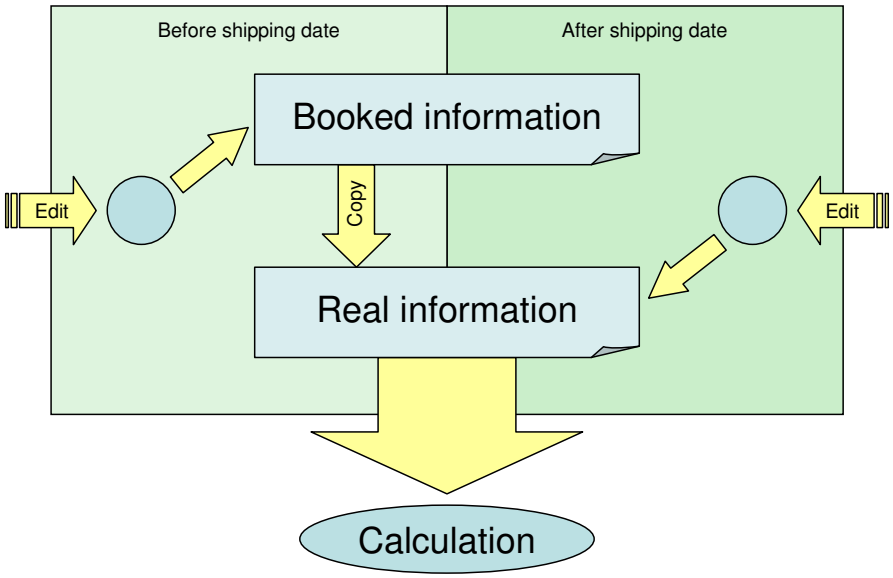

Figure 4.3: Booked vs. real information

The time slot for entering the actual information after the shipping date is defined when setting up the Trip Scheme. Obviously, the final invoices should only be created when the configured editing period is over, to ensure they are not based on data that is subject to change.

4.1.6 Statistics

FIONA's statistics functionalities can be divided into two parts. The first serves invoicing purposes, based on Tenneco's electronic invoice format, the second aims at internal controlling and process optimization and is derived from the KPI files (see Chapter 3.1.3).

Invoice statistics

The accumulation of prices to be invoiced is available to all users of FIONA.

Export data	Invoice field	Comment
Tariff	-	
Carrier	-	
Currency	currency	
Truck No.	pronumber	
Shipping date	prodate	
Loading length	-	
Loading weight	weight	
Loading volume	-	
Price type	serv_level	N=normal, X=express, B=back trip
Load location	consignor	Carrier adds Fiscal Entity
Load zip	origin	
Load Country	org_cty	
Load city	org_city	
Unload location	consignee	Carrier adds Fiscal Entity
Unload zip	dest	
Unload Country	dest_cty	
Unload city	dest_city	
Billed Amount	billed	
-	mastinv_no	n/a in FIONA
-	inv_date	n/a in FIONA
-	pieces	n/a in FIONA
-	mode	Ground for all FIONA shipments
-	serv_Categ	n/a in FIONA
-	vat	n/a in FIONA
-	fee1_id	n/a in FIONA

Figure 4.4: FIONA invoice export — field mapping

Carriers can query the results of the transport cost distribution on all Trip Schemes they offer tariffs for. Trip Scheme Owners and Members can check the sums to be invoiced for their locations. FIONA Administrators have access to all invoicing data.

The information is queried in a location-based manner for a configurable period of time via the web interface. The presented data can then be exported to files for further processing in MS Excel.

The file format used for the exporting functionality is an Excel-oriented flavor of comma-separated values (CSV) files.[2] It contains a line with headers and the field delimiter is localization-dependent (see Chapter 5.7). This method was chosen because of the tight project schedule, which did not allow an in-depth evaluation and integration of the proprietary MS Excel file format (.xls). However, a CSV-file opened in Excel can conveniently be saved as .xls file. In addition, it has the advantage to be read by various other applications as well, which enhances the integration possibilities.

An overview of the exported information, mapped to the electronic invoice fields (see Chapter 3.1.3), is given in Fig. 4.4.

KPI statistics

Replacing the KPI files is already realized in FIONA with the immediate calculation of the cost portions for each location and making them available online to all affected parties in real time.

An additional export that includes the information from the old KPI files serves further statistical processing and presentation tasks.

				Valencia		Valencia to Edenkoben			Total						
Shipping	Carrier	Truck no.	Truck type	Arrival	Departure	Length	Weight	Volume	Length	Weight	Volume	Fill rate	Currency	Price	Edenkoben
15.09.06	Ewals		Standard	A 19:00	A 10:00	5			13	0	0	96%	EUR	1800	608,11
15.09.06	Ewals		Standard	A 19:00	A 10:00	5			13	0	0	96%	EUR	1800	608,11
15.09.06	Ewals		Standard	A 20:00	A 10:00	5			13	0	0	96%	EUR	1800	608,11
15.09.06	Ewals		Standard	A 20:01	A 10:00	5			13	0	0	96%	EUR	1800	608,11
15.09.06	Ewals		Standard	A 20:02	A 10:00	5			13	0	0	96%	EUR	1800	608,11
15.09.06	Ewals		Standard	A 20:03	A 10:00	5			13	0	0	96%	EUR	1800	608,11
15.09.06	Ewals		Standard	A 20:04	A 10:00	5			13	0	0	96%	EUR	1800	608,11
16.09.06	Ewals		Standard	A 08:00	A 10:00	5			13	0	0	96%	EUR	1800	608,11
16.09.06	Ewals		Standard	A 08:00	A 10:00	5			13,6	0	0	100%	EUR	2000	634,52
17.09.06	Ewals		Standard	A 08:00	A 10:00	5			13	0	0	96%	EUR	1800	608,11
17.09.06	Ewals		Standard	A 08:00	A 10:00	5			12	0	0	88%	EUR	2000	757,58
17.09.06	Ewals		Standard	A 08:00	A 10:00	5			13	0	0	96%	EUR	1800	608,11
17.09.06	Ewals		Standard	A 08:01	A 10:00	5			13	0	0	96%	EUR	1800	608,11
17.09.06	Ewals		Standard	A 08:02	A 10:00	5			13	0	0	96%	EUR	1800	608,11
17.09.06	Ewals		Standard	A 08:03	A 10:00	5			13	0	0	96%	EUR	1800	608,11
19.09.06	Ewals		Standard	A 08:00	A 10:00	5			12	0	0	88%	EUR	2000	757,58

Figure 4.5: FIONA KPI export — formatted excerpt

[2] A general introduction to this non-standardized format can be found at [Cre06].

The KPI export is only available to FIONA Administrators, as they are responsible for the controlling activities on the logistics sector that is covered by the system. Each file contains KPI information for one Trip Scheme and is, like the invoice export, created out of booking data from an adjustable period of time.

The above statements concerning the file format are also applicable to the KPI export: CSV is employed to ensure easy composition and a wide range of use. Necessary formatting can easily applied to the files using Excel.

An excerpt from a generated KPI file (with additional formatting) is shown in Fig. 4.5. The export is available for all Trip Schemes covered by FIONA.

4.2 Architecture

The base structure of the software's architecture was determined by the Tenneco design guidelines (see Chapter 3.2). A MVC-oriented multi-tier system[3] based on Jakarta Struts and an Oracle database had to be set up. The database was refined in multiple iterations to be able to represent all shipment and invoicing scenarios. The next sections describe the system's architecture with an increasing level of detail.

4.2.1 Layer model

Seven layers form the data processing framework of the software. It is shown in Fig. 4.6. The first two are imposed by the MVC pattern and, except for a part of the View, based on the Struts framework. The remaining tiers are derived from common best practices and technical optimization. They form the MVC's Model.

View

The View presents the information system to the user. It ideally shows preformatted information matching the user's scope of interest and authorization level. This part of FIONA is realized via the web interface. Implementing different types of Views to exist in parallel is common practice, yet not required for the given system. Input is accepted by the view allowing the user to interact with the Controller.

[3]For an introduction to the Model View Controller pattern see [BMR+96], pp. 125ff.

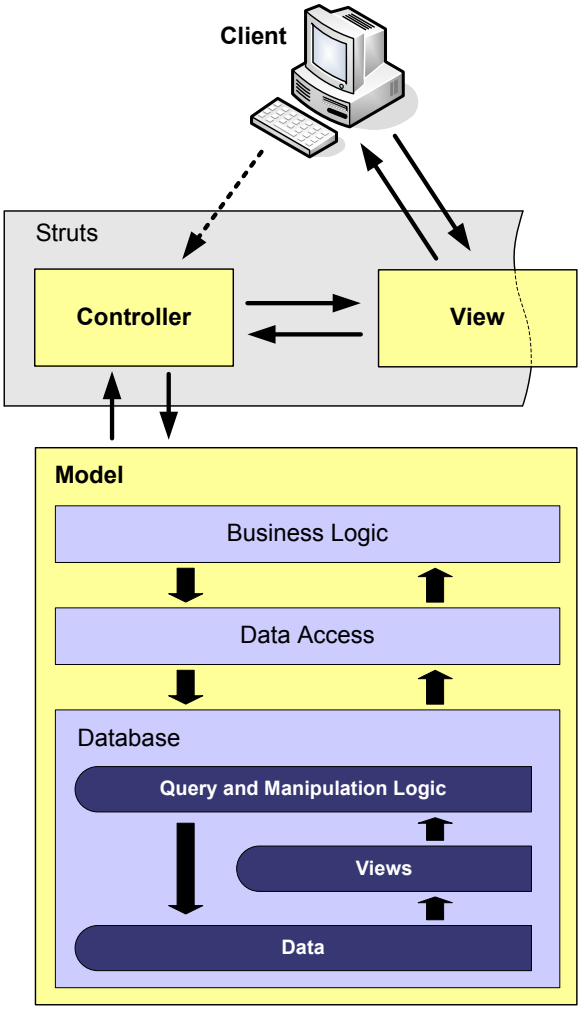

Figure 4.6: Layer model

Controller

The Controller is responsible for propagating changes to the underlying Model and, thus, indirectly accepts the client's requests. Syntax validations of user input are executed here, so non-matching data entries do not reach the Business Logic at all. In the realized architecture, the Controller is also the instance that fills the View with data to display. Chapter 5 describes its interaction with the View and the layers underneath with focus on Struts' role in this process.

Business Logic

FIONA's actual functionality is provided by the business layer. There, the logical validation of input is performed and the booking information is processed. The transport cost splitting calculation is executed in this layer and e-mail notifications are triggered. The tier was implemented completely in Java.

Data Access

The Business Logic's connection to the database is provided by the Data Access Layer which was also coded in Java. It accepts query, storing and deletion calls and converts given data into or from the appropriate database-compatible format. However, no query logic is implemented in this tier. This was a requirement by the Tenneco development department. Just the translation of the Business Layer's data reading or writing requests to database function calls is performed here.

Query and Manipulation Logic

Actual queries and data manipulation functionality are implemented in the uppermost layer of the database. This part is realized in the Oracle database itself, using stored PL/SQL packages (cf. [LK02], p. 529ff). They offer functions that allow being called by the Data Access Layer and contain the core SQL statements.

Terms or values required on database level that are common to all of the applications involved (currently CaTIS, confluentic WEB and FIONA) are stored in a central utility package to avoid redundancies. The agreed value representation for true/false flags is an example for these shared constants. It was required because Oracle does not support fields of boolean type in a table.

Views

Yet, for reading purposes, tables containing the relevant data are in most cases not accessed directly. Instead, the information passes through an additional layer: The views, as set of stored queries, can themselves be accessed like tables. They provide an abstraction from the actual database model and allow the query logic to be independent of changes in it. In addition, common data conversions can be handled there. Date formatting is a typical example for this approach that was also used in FIONA.

Data

This layer comprises the persistent data. Changes are only made via the Data Manipulation logic. Referential integrity is ensured by the logic validation in the Business Logic or, for bulk deletion of multiple related records, in the corresponding package function. This is used e. g. for deleting a booking with all corresponding information. The database model is presented in the next chapter.

4.2.2 Database model

As mentioned in Chapter 3.1.4, the database model was designed to be shared by multiple applications. The E/R diagram in Fig. 4.7 shows the tables that are accessed by FIONA either directly or via its views. The complete diagram is attached on CD (see Appendix F).

The colors indicate the intended system scope of the tables. Yellow tables are meant for common use, tables for CaTIS are of orange coloring and blue expresses usage by FIONA only. Gray symbolizes, that a table is destined for future use, and, thus, only conceptual so far. The lighter shades of the utilized colors mark reference tables (n:m relationships).

The CaTIS tables are included in this description because the query logic accessing them is part of FIONA in this first version of the system (see Chapter 4.1.2).

User storage

A user is either associated with a location or a carrier. Which of the two connections exists is determined by the user's group: all non-carriers have to be assigned to a location (see also Chapter 4.1.1).

The user group is set using a flag directly in the `FIONA_user` table. This realizes the system-wide validity of the corresponding access rights. By contrast, the user roles are attached to the user's association to a Trip Scheme (`FIONA_ts_user_role` and `FIONA_tripScheme_user` respectively). Thus, a role and — for future extensions — even multiple roles can be assigned to a user for each Trip Scheme individually during master data setup.

TripScheme vs. FIONA_tripScheme

In the first instance, Trip Schemes exist as a sequence of locations to be visited during a delivery (see Chapter 2.2.2). This information is also required by CaTIS and will be by confluentic WEB. It is kept in the commonly used

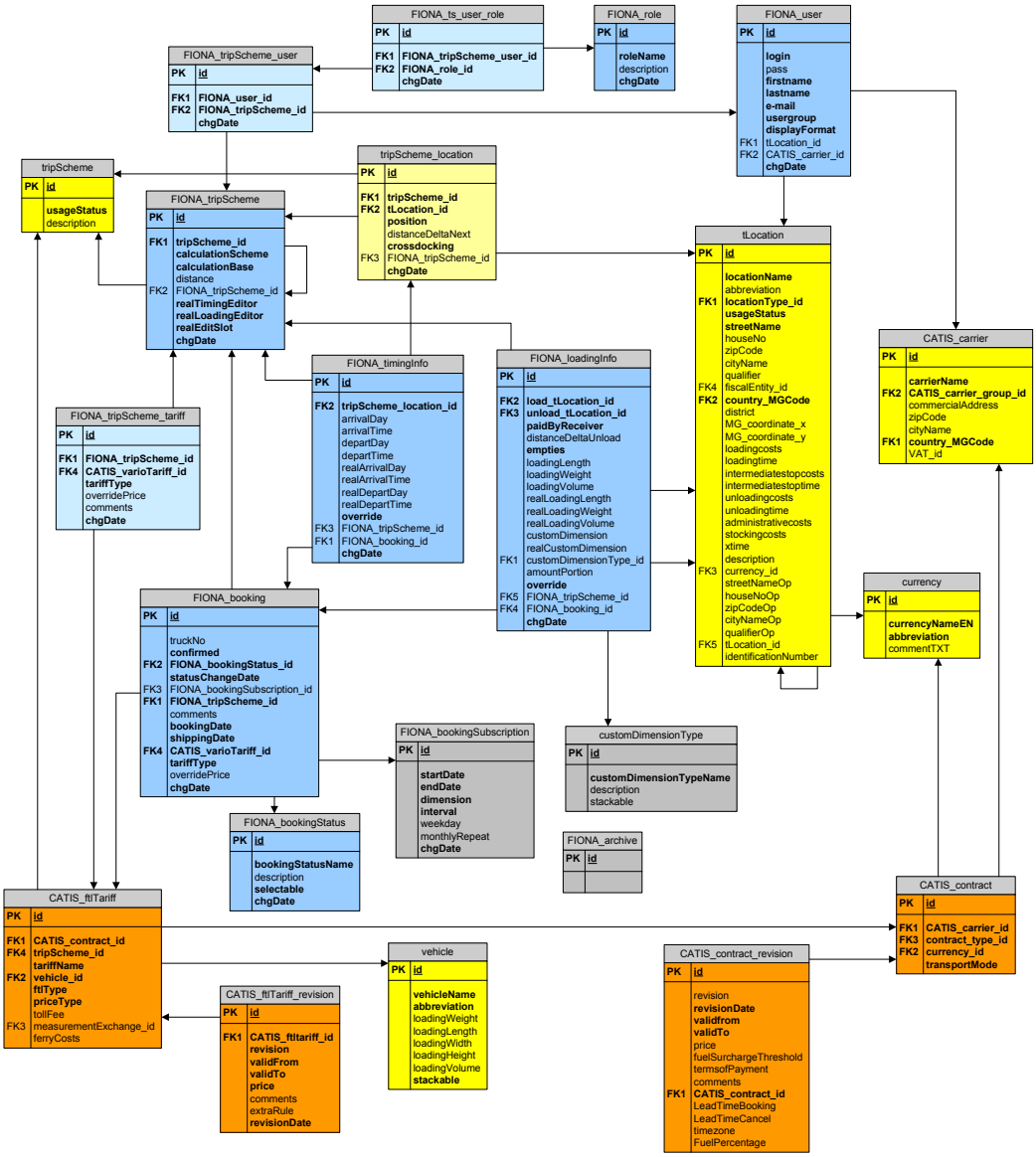

Figure 4.7: Database model — Tables accessed by FIONA

`tripScheme` table. The second table, `FIONA_tripScheme`, had to be introduced, because the perception of a Trip Scheme by the logistics staff has significant differences.

The Trip Schemes listed in FIONA are dependent on the tariff: the same trip can be a single forward delivery as well as a back trip to another delivery, resulting in different pricing and usage scenarios. To fulfill this requirement, FIONA needs to offer separate Trip Schemes for the same location sequence.

Accordingly, FIONA-specific information, like the calculation base (loading length, weight or volume) is stored in `FIONA_tripScheme` and linked to the actual underlying Trip Scheme. Forward and back trips will have separate entries in FIONA's table, the latter associated with a back trip tariff. The tariff's type is indicated by the `priceType` in `CATIS_ftlTariff`.[4]

Connecting the forward part to a back trip is enabled by a relation of FIONA's Trip Scheme table to itself.

Trip Schemes do not have a unique name that can be used to represent them in the system, as can be seen in the diagram. Although the introduction of this identifier was highly recommended to allow the importing and changing of master data (cf. [Kop06]), it was strictly rejected by the logistics supervisors: The concept of unified Trip Scheme names as prefix to the corresponding tariff names was deemed overly complicated.

Instead, a Trip Scheme has to be identified by the name of the associated tariff. This entails the requirement of having the same name entered for each tariff that is related to it and has the same Transport Type variant — even across different carriers. The evident drawbacks and risks caused by this redundancy in naming were accepted without comment.

The Trip Schemes to be set up in FIONA are found according to the tariff and distinguished by the name. When they are initialized, the related entries to `FIONA_tripScheme` are made. Mistakes in tariff names will consequently cause a split into multiple Trip Schemes.

Each FTL tariff has a connection to the entry in `tripScheme` it determines the price for. The reference table `FIONA_tripScheme_tariff` stores which of the available tariffs have been set up by the FIONA Administrator to be used for bookings.

Location sequence

The position of each location on a Trip Scheme is kept in the reference table (`tripScheme_location`) that connects the two. For cross-docked locations,

[4]The `priceType` flag indicates the variant of the given FTL tariff. Besides "normal" (forward) and "back", there is also "express", which does not cause a new Trip Scheme but is treated as special execution of a forward trip.

a new entry is added to this table by FIONA. It can be distinguished by the correspondent flag and, if applicable, will be ignored by the other applications that share the database.

Timing and loading information

Both, the default timing and loading information and the corresponding individual values for each booking, are stored in the `FIONA_timingInfo` and `FIONA_loadingInfo` tables, respectively. Default entries have their `override` flag set to false and a relation to the Trip Scheme. For the booking entries the flag is set to true and the record links to `FIONA_booking` instead. Booked and real values are stored in the corresponding fields.

Each loading information record also represents the whole load-unload relation by connecting two locations and defining the assigned distance as well as the configured paying site (see Chapter 4.1.2). This allows changing the relations on a Trip Scheme (default entries) without affecting existing bookings.

To satisfy the demand for retroactively changed tariffs (Chapter 3.1.4), the loading information entries for bookings do not store their resulting price but the portion of it as a decimal factor. This enables a fast output of the correct price by simply multiplying the stored fraction with the price from the latest tariff revision (`CATIS_ftlTariff_revision`) matching the shipping date or the manual override price, especially for statistics purposes.[5]

Override price

Transport prices entered manually (see Chapter 3.1.4) during Trip Scheme setup are stored in the relation `FIONA_tripScheme_tariff` which provides the connection between a bookable tariff and FIONA's Trip Scheme. Thus, each associated tariff can be overridden individually.

Again, the existing shipments are protected from master data changes by copying: if existent, the corresponding override price is stored in each new booking record upon creation.

4.2.3 Modules

Using the facilities of the Struts framework, FIONA's upper layers were subdivided into different modules. This approach serves for more convenient development and a well-organized structure of the software. The partitioning

[5]For FIONA's export functionality this multiplying is directly implemented in database views.

is based on the main process (see Chapter 4.1) and comprises the following modules:

- Default
- User
- Master
- Booking & statistics

Each of the modules contains the View and Controller part of a coherent section of the system and is defined by a combination of activities (cp. Chapter 5.1).

The default module handles the login and logout functionality. User setup and management are covered by the user module, while all other master data manipulation is carried out in the master module. The fourth module is responsible for all booking-related activities and, as they are closely related, also for statistics creation and export.

Switching between the modules happens transparently for the user by choosing the desired menu entry. The current module just appears as a segment in the web-interface's URI path — except for the default module which represents the application's root "folder".

Chapter 5

Implementation

Most of the implementation to realize the design and provide the required functionality was organized top-down for the different modules. The mask layouts were implemented early to serve as communication means when aligning the design with the logistics supervisors (see Chapter 1.2). These static masks were successively ported to the functional Struts-based pages. The controlling and business logic underneath were added and then completed with the database-related facilities.

During the implementation activities, it became obvious that a significant part of the development work had to cover the creation of auxiliary infrastructure. Elements of what can be described as "inner framework" were created. These auxiliary constructions allow being reused throughout the application and facilitate extending it. By establishing generic default structures and practices, they also provide an orientation help to the developers who will adopt the software maintenance.

In the course of these programming efforts, many required system-internal functions had to be implemented that are included in current versions of Struts. However, a transition to such a release of the framework was prohibited by the restrictive platform list in Tenneco's development guidelines (see Chapter 3.2).

The most important mechanisms created during development will be described in this chapter. Understanding them requires knowledge about the system's structure (see Chapter 4) and a basic insight into the employed framework. Therefore, a short introduction to the concepts and mechanisms that constitute Jakarta Struts is given first.

5.1 Excursion: Struts — A brief outline

As mentioned in the design description, the user interface of FIONA and its input/output logic are almost completely based on the features provided by Jakarta Struts. This open source framework focuses on supporting the developer with facilities to transfer data from the web page to the Controller level and back. The involved items and the corresponding data flow are described in the next section, followed by an overview of the utilized configuration method.

For a comprehensive presentation of Struts and detailed explanations of all mechanisms described here, please refer to [HDFW03] and [Cav04].

5.1.1 Structure

Fig. 5.1 illustrates the interplay of the different elements that constitute the completely Java-based framework.

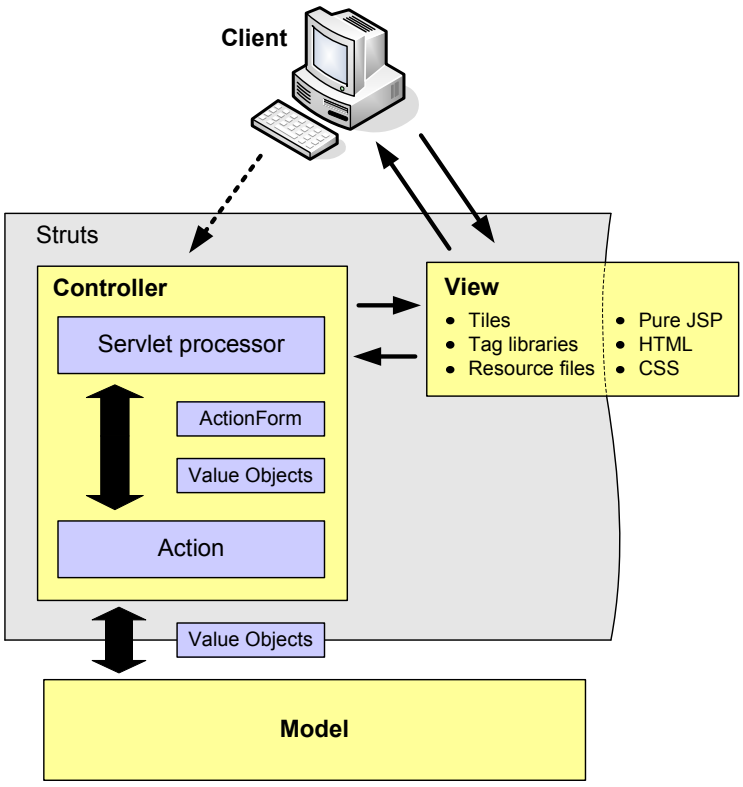

Figure 5.1: Layer model — Struts

The web interface is composed of JSP pages. Their programming is ba-

sically performed using libraries of special tags that ensure the generation of HTML code which can be processed by Struts. They allow the utilization of standard web page controls, such as radio buttons or text fields, and provide additional functionalities, like conditional source code sections. Thus, the need for writing pure JSP code when preparing the pages is reduced to a minimum.

To provide enhanced possibilities for inclusion and reuse of page fragments on coarse-grained (e. g. headers, footers) as well as fine-grained level (e. g. reoccurring groups of buttons) a customizable template system is available. These "tiles" are configured using XML files and support various different scenarios of modular page construction, including nested templates and inheritance of contents.

Free strings to be displayed to the user (e. g. text field labels) are kept outside of the page source, in dedicated human-readable resource files. This enables easy adaptation of the user interface and is also a prerequisite for internationalization (see Chapter 5.7).

The Struts-specific tags in the JSP source code are supplemented by conventional HTML elements and can be formatted with regular Cascading Style Sheets (CSS) or standard format attributes.

User input to an Internet application is usually accepted via forms presented on the web pages. Struts offers special support for these forms. The servlet processor accepts the requests, that contain the input data and automatically creates and fills a dedicated object with the information received. This ActionForm is a Java representation of the web form, that usually contains properties of the same name, along with matching getter and setter methods.

At this stage of the process, an integrated syntax checking facility can be put into place. Using the ActionForm, it validates the user's entries according to a set of configurable white lists. If the validation fails, the process will not be continued. Instead, the input page will be redisplayed with appropriate error messages, if desired.

When the ActionForm has been filled with acceptable values, it is forwarded to the appropriate Action. This object is responsible for triggering all follow-up activities. Here, the user's entries are available to the programmer and can be extracted for further processing.[1]

The information related to a business domain is usually grouped together to form different objects. In the context of FIONA, bookings and locations

[1] In many cases, the ActionForm's code is created by the developer as well, but the functionality is restricted to input, output and conversion of data. The process is not influenced.

are examples for such entities. A common practice for their representation is the creation of Value Objects (VOs). These are usually simple objects that basically contain several properties and the corresponding getter and setter methods.

In the Action, the VOs can be created or manipulated using data from the ActionForm, before they are forwarded to the Model. There, the Business Logic (see Chapter 4.2.1) is responsible for the actual processing. The Action ensures that the correct operation in the Business Layer is triggered and in turn receives the result of the call. Other approaches exist, that forward the ActionForm itself to the next layer. This is sometimes done because the ActionForms are often similar to the required VOs. However, they tend to offer more information and additional functionality than required.

Depending on the outcome, another Action might be called or a direct forward to the next page can be executed. Actions are also the instance that changes the contents of an ActionForm to display a result to the client. The transfer of this data from the ActionForm back to the web page form is again executed by the servlet processor.

5.1.2 Configuration

An XML format is used to store the configuration of a Struts application. Since their structure can easily be translated into a diagram, many development environments allow the visual editing of Struts configuration files. Each module (see Chapter 4.2.3) has its own configuration file.

The module settings describe the possible sequences of Actions and pages that occur when working in the application. Thus, each diagram illustrates a section of the business workflow in a step-by-step manner.

It is a common practice to frame each page with an Action that prepares and shows it and another Action that accepts the input from it. The accepting Action calls the showing Action of the next page, when data processing has finished. That way, page preparation (e. g. loading data for a selection box) and input handling are separated. With correct setup this also prevents unintended re-execution of Actions when a page is reloaded.

To open a page or pass the execution on to the next Action, an ActionForward is returned by each Action. The servlet processor maps this information to the target by consulting the configuration file. Global ActionForwards are available module-wide, the local ones are only valid for the associated Action.

Fig. 5.2 shows an extract from FIONA's user management module. In the example — which illustrates changing the master data of existing FIONA user accounts — the global ActionForward `userSearch` leads to the showing Action of the corresponding page. The page (`.fiona.user.userSearch`) is

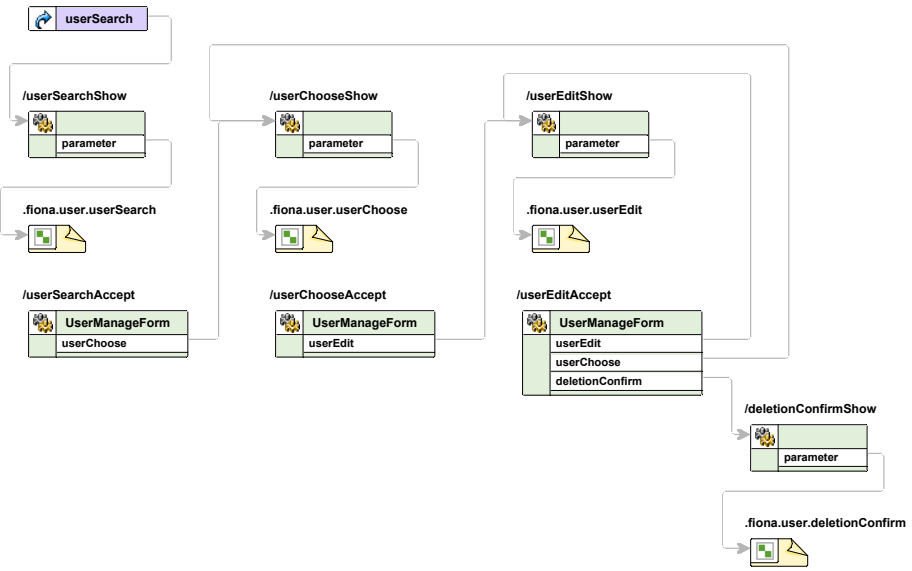

Figure 5.2: Struts configuration excerpt (user module)

depicted by a tile symbol, since each page in FIONA is constructed with tiles[2] to allow the reusing of JSP fragments — especially the header, footer and menu. Submitting the search page calls the associated accepting Action. The input data is stored in an ActionForm called `UserManageForm`.

Next, the user can choose one of the accounts returned by the search and is forwarded to the editing page. Depending on their activities on this page, the accepting Action might simply reopen it with an error message (which is not part of the configuration), redisplay the search result or forward to the confirmation page for account deletion. Multiple local ActionForwards in `userEditAccept` are in place to enable this behavior. Alternatively, the global ActionForward could be returned. The confirmation handling configuration is not shown the excerpt.

[2]Except for the JSPs that create the content of export files.

5.2 Heavy-weight ActionForms

Many of the ActionForms that were created for FIONA contain additional logic that streamlines the data extraction and the composition of required Value Objects.

As the most of FIONA's functionalities span multiple pages, the relevant ActionForms were designed to hold their entries across page requests. They have to be capable of storing values that belong to different views on a business object, ranging from summarized presentations of complex structures to detailed property lists for a single VO.

To support the handling of Value Objects in the Actions, various helper methods were created, especially for VOs that have to be chosen from a list to be edited and put back in place. A typical example is the setup of a Trip Scheme: the Trip Scheme contains a list of locations that can be individually edited to enter arrival and depart times. Thus, the corresponding ActionForm stores the list, but also offers properties to temporarily store a location's individual fields when it has to be displayed on the editing page. The servlet processor uses these properties to fill the JSP and to write back the submitted user input.

Reading and writing the location data in the Actions is supported by the following functions that are offered by the ActionForm. The location currently chosen by the user is identified by its index in the list of locations. The getter/setter pair

```
public String getCurLocationIndex();

public void setCurLocationIndex(String curLocationIndex);
```

provides the access to this index. The setter is also used by Struts' servlet processor to tell the form about the location index that was picked by the user on the web form (usually by activating a radio button).

The accessors that read and write the list of locations and its elements exist, but are not listed here in order to facilitate reading.
A private helper method

```
private void fillLocationFields(BaseLocationVO location);
```

facilitates filling the properties in the ActionForm with the information from an arbitrary location's VO. Acting vice versa is also required:

```
private BaseLocationVO extractLocationFields();
```

reads all relevant properties from the ActionForm and uses them to create and return a complete location.

As especially the currently chosen location is of interest to the processing, it can be returned with a getter that simply uses the index from above to return the appropriate location entry from the list:

```
public BaseLocationVO getCurLocation();
```

Combining the previous functions leads to a pair of helpers that significantly facilitate Action development:

```
/**
 * puts the property values of the currently
 * chosen location (from location list) into
 * the corresponding properties on form level.
 * Calls fillLocationFields on value of
 * getCurLocation for that purpose
 */
public void flattenCurLocation();
```

fills the ActionForm's location properties with the values of the currently chosen location and

```
/**
 * puts the values of the location properties
 * on form level into the corresponding properties
 * of the currently chosen location (from list
 * locations). Calls extractLocationFields for
 * that purpose.
 */
public void deFlattenCurLocation(boolean dirty);
```

writes them back.
Consequently, the basic scheme for editing the chosen location can be reduced to these steps:

1. In the Action that accepts the choice of location:
   call `flattenCurLocation()`,

2. Forward to the page that allows editing the location,

3. In the Action that accepts the editing page:
   call `deFlattenCurLocation()`,

4. Forward back to the list, to allow choosing the next location, if desired.

This pattern is utilized to organize the management of several object lists throughout FIONA.

5.3 Generic deletion infrastructure

Deleting an object in an application usually follows a fixed scheme:

1. The object to delete is chosen,
2. A confirmation dialog is shown,
3. Depending on the answer to this dialog, the deletion is either performed or canceled.

To avoid implementing a new dialog page and the corresponding Actions for each case that requires a confirmation by the user, a universal infrastructure was prepared.

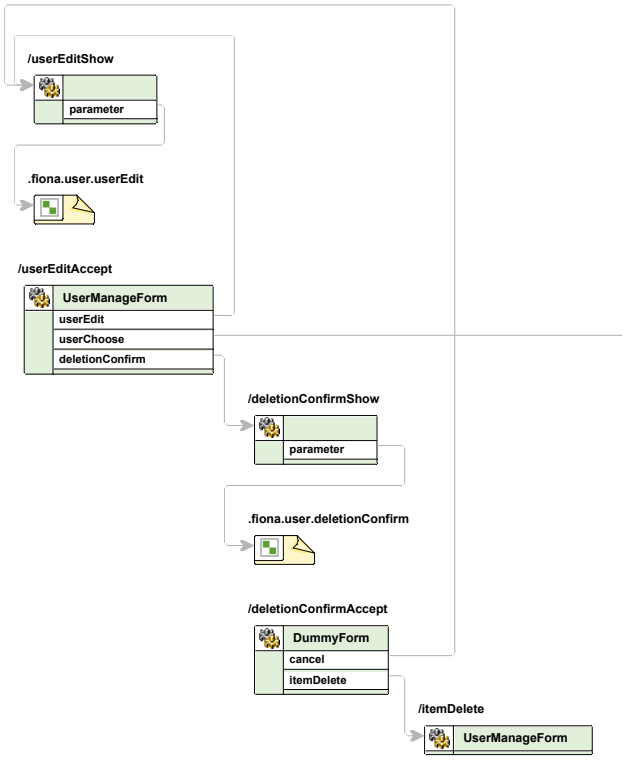

Figure 5.3: Deletion infrastructure (user module)

Fig. 5.3 illustrates the basic process sequence with another excerpt from the user module's configuration file.

When, on the editing page, the user decides to delete the chosen account, the accepting Action first stores all necessary information in the session. This is basically the data required to generate the confirmation message and the matching deletion method's identifier. Then, it returns the local forward to the confirmation page (`deletionConfirm`) which simply shows the message and the buttons to confirm and cancel.

Depending on the response to this page, its accepting Action forwards either to the `itemDelete` Action or the page that is configured to be opened on cancellation.

The `itemDelete` Action, which provides the system's complete deletion functionality, identifies the method to execute by the entry in the user's session. It performs the chosen task and then uses a global forward to reach the next page and display a success message. In most cases, this will be the page where the deleted entry was chosen.

Thus, all activities that require confirmation can be realized by using the same implementing classes. Different target pages to be redirected to on cancellation can be realized by configuring different (logical) pages and Actions that refer to the same implementation.

5.4 Exception handling

Exceptions are propagated from the Data Access layer back up the hierarchy formed by the sequence of calling methods. If they are not handled by appropriate logic on the way, they will eventually be handed over to the originating Action. There, a generic mechanism is activated, that covers all unhandled Exceptions to prevent them from reaching the server-internal logic and crashing the application. It is part of FIONA's base Action implementation and therefore available to all derived classes.

The universal error management logs the Exception's message to a file via the logging facility in place on the server.[3] Next, it performs a user logout to safely end the working session and forwards to an error page that shows the Exception message in a text field to allow convenient copying. From this screen, the user can proceed to the login screen to resume their activities.

[3]The Tenneco list of required platforms specifies log4j for this purpose (see Chapter 3.2).

5.5 Access restriction

Enforcing access restriction regulations (cp. Chapter 3.1.1) on Action level is a key feature to the application's security mechanisms. Each Action class provides a list of the user groups (see Chapter 4.1.1) that are entitled to execute it.

Upon the call to an Action, the group of the user that initiated the execution is read. The processing is only continued if the user group is found in the list, otherwise a forward to the login page is performed.

The checking functionality is provided by FIONA's basic Action class implementation. This centralized verification mechanism facilitates the integration of new functionalities into the user group concept: a new Action class that is added to FIONA only needs to provide this list property with the desired groups in it.

The described mechanism is complemented by access restriction on page level. The base tile for pages that require authorization checks the user's status and does not generate the source code unless the user is logged in. Unauthorized calls are also redirected to the login screen.

5.6 Multi-user synchronization

As the pilot version of FIONA will be supervised by only two or three administrators with mostly disjoint working sectors, a synchronization of master data editing was not included.

The users involved in daily business, however, will be working on shared data. Two users loading the information of the same booking and changing the load could run into a conflict: If both started editing the data before each saves their changes, the updates of the person saving first will be overwritten.

To prevent this data loss, a synchronization mechanism was put in place that covers all conflict-prone activities for users that are not in the FIONA Administrator group. Fig. 5.4 summarizes their activities.

Note that storing a new booking cannot produce a collision, because no existing data is changed.

The synchronization functionality makes use of the fact that each of these activities affects the table which stores the bookings in the database.[4] A change date field was added to this table (as to any other table) that is set

[4]Since a booking's status and the load allocation are edited on the same web form, a change of only the load will also cause the status (which is set in the booking table) to be updated.

Process	Activity	Conflicts
Create booking	Preview calculation	
Create booking	Save	
Confirm/decline booking	Save	X
Edit booking	Preview calculation	
Edit booking	Save	X
Edit booking	Delete	X
Edit booking	Revoke	X
Statistics	Show	
Statistics	Export	

Figure 5.4: Synchronization — Conflicts for non-master-data activities

to the date and time of the booking's creation. With each update, this time stamp is refreshed.

Whenever a booking is retrieved from the database, the change date is attached to it using a dedicated property in the corresponding Value Object. All VOs in FIONA inherit this field from their base class to support future extensions of the synchronization mechanism.

Once set, the property is immutable. If the booking is written back to the database, which happens during all of the conflicting activities, the data manipulation logic (see Chapter 4.2.1) first compares the incoming time stamp with the one currently present in the database. If the values are equal, the data can safely be written. If they differ, the booking has been changed by another user in the meantime. In this case, the database logic will not perform the changes, but return a synchronization error message instead.

When it receives this message, the Data Access layer creates a dedicated Synchronization Exception. This Exception will traverse all layers and reach the Action that triggered the aborted database update (see Chapter 5.4). Upon catching this type of Exception, all Actions involved with conflicting activities call a function from their base class that prepares a message telling the user that their data was not updated due to a saving collision and suggests reloading the record. The event is also written to the server's log file, if logging of info-level messages is activated. This is helpful for monitoring the synchronization and deciding whether extended synchronization measures have to be taken, as the system grows.

The responsible Action will then forward the user to a preceding page that allows loading the updated records from the database.

5.7 Internationalization

Since FIONA's users operate in various countries and a future expansion to American users may come into focus, enabling the user interface for internationalization was required.
The affected parts of the system are

- labels and messages,
- number formats,
- units of measurement,
- export formats.

Currencies for pricing were not included in the localization efforts, as all tariffs have to be displayed in the currency of the related carrier contract and conversions were not included in the pilot version's scope.

The capability of displaying messages and labels in the user's language is integrated in the Struts framework. The resource files that are utilized to keep these strings (see Chapter 5.1.1), can be provided in different languages. Struts automatically chooses the resource file for the browser's locale or the default file if no match is found (refer to [HDFW03], pp. 409ff for details).

All other conversions are based on the user's display format, which can be individually configured by a FIONA Administrator. Currently the system offers European or American format.

Internally, all values are stored in European format. Special getter and setter methods (display helpers, cf. [HDFW03], p. 176) in Value Objects and, if unavoidable, in ActionForms handle the conversion for input and output: the values are translated when they are written to or read from the web page.

Number format conversion is restricted to switching the decimal separators between the European and American locales. Thousands separators are neither displayed nor accepted by FIONA.

For values entered by the user, a comma as well as a dot marking the beginning of the decimal part is accepted, regardless of the configured display format. Output data will use the former for the American and the latter for European users.

Units of measurement are covered by the same display helpers: they are handled in the European format internally and converted on output and input, if the user uses the American format. The conversion factors are stored centrally as constants to allow easy adjustment. The corresponding labels are organized similarly: for each unit label, a European and an American version exist. Their displaying is toggled appropriately.

Unit conversion covers the items listed in Fig. 5.5.

Unit	European	American
Distance	km	miles
Loading length	ldm	t.b.d.
Loading Weight	kg	t.b.d.
Loading Volume	cm^3	t.b.d.

Figure 5.5: Internationalization — Units of measurement

As mentioned in Chapter 4.1.6, the CSV format accepted by MS Excel also has locale-specific differences: apart from the inherently different decimal separator, the field delimiter is also changed. When the decimal separator becomes a comma, the delimiter is changed to a semicolon — in contradiction to the format's name.

Again, the user format from the setup determines which version is used for the statistics export.

Chapter 6

Conclusion

With finishing the pilot version's implementation, FIONA was handed over to Tenneco's IT department. This chapter recalls the course of the project and highlights the key results. It is completed with a look at the software's possible future development and usage scenarios.

6.1 Review

A short walk through the project recounts the different phases and findings on the way to the finalized application.

6.1.1 The project

The environment of the project was defined by a pivotal sector of Tenneco's logistics operations. Booking shipments and tracking their execution are combined in a process that is crucial to accurate production and delivery. The according payment management and data collection serve the awareness of key expenses and, thus, are a prerequisite for cost reduction.

The set of related activities was determined and analyzed in a series of several workshops. During the analysis, the drawbacks of the existing process were identified and approaches to their solution set the course for all further activities.

A key aim was to unify and integrate various detached methods of conducting the booking and invoicing process. Different inhomogeneous documents were to be replaced by a central platform for all parties involved. The operational workflow had to be enhanced with a dedicated software tool to be first employed on a representative sector.

FIONA was developed after comparing possible alternatives that range from complete solutions by third-party sellers to the adaptation of existing information systems.

The interlinked processes of analysis and design provided insight into many fields and principles that determine the corporate as well as the inbound and outbound logistics activities of a worldwide supplier company. In turn, the presentation of design and development approaches led to the identification of several process-internal circumstances by Tenneco, that were not obvious before.

Implementing the software within a short time and ensuring that it can be maintained by Tenneco's development department required the preparation of elementary framework-like structures. These facilitate extensions of the software and ensure a coherent arrangement of the different components and functionalities.

Rolling out and evaluating the software in the pilot environment, which comprises a Route setup (see Chapter 2.2.1), is to be conducted by Tenneco in the weeks after the finishing of this thesis.

How the created system supports and streamlines the target process is presented in the next section.

6.1.2 Outcome

Centralized Internet-based processing of transport information in FIONA brings along several benefits that support resolving the identified logistics issues (see Chapter 2.3).

Booking trucks online, with the input of load composition attached to it, renders any additional order documents needless. All parties involved can access the data as soon as it is stored in the system. This simplifies information exchange as well as changing shipment parameters.

Because the information on truck usage is available to all loading and unloading sites, the solution supports negotiating and updating the allocation of loading space. Overloading the truck via the system is prevented by consulting vehicle type definitions. A synchronization mechanism coordinates concurrent changes. These factors can significantly improve the truck utilization ratio.

Manually creating and distributing pricing or statistics reports will no longer be necessary. The invoice splitting calculation is provided by the software and can now serve as basis for the invoicing. Data accumulation for controlling purposes can be executed directly on the system's data. Standard exports for invoices and KPI files are already in place.

The immediate common visibility of tariff portions to be invoiced has further advantages. Invoices sent to the different sites can be audited by each of the sites using the data from the application. Future expenses (accruals) can be closely estimated when the bookings have been entered in the system. As soon as the load allocation is consolidated, the figures will be exact.

Collecting actual timing and loading information after the shipments have been completed is beneficial to controlling activities. Having the carriers enter the real loading information is not only convenient, it can also prove helpful when locations' loading statements tend to be inaccurate. If, in addition, the locations enter the actual timing, the carrier's reliability can be monitored as well.

Additional confirmation functionality allows the carrier to assure the Tenneco sites of the delivery execution and prevents bookings from being deleted without a corresponding revocation record.

Integration with the concurrently developed Carrier and Tariff Information System (CaTIS, cf. [Kop06]) and the upcoming confluentic WEB (cf. [Dro06]) promise further process optimization and extension opportunities.

6.2 Perspectives

Due to the fact that FIONA is a pilot version, it basically provides the set of features required for testing it under real conditions in daily business. An additional collection of extensions was planned and partially designed for future releases of the software. Because of the project's time restrictions, these extensions could not be implemented.

Testing the prototype in the current state of implementation will be sufficient for evaluating the basic concept, which is the main goal of the first deployment stage. Broadening the application area to include more Routes and several Milk Runs will be the next step.

Evaluating the first usage period, with special focus on the feedback from daily business users, will provide information on necessary optimizations and the direction in which the system should be further developed.

An overview on possible enhancements is given in the following Chapters. The preparations that were made for some of these extensions consist of Java and JSP stubs[1] or additional database facilities.

[1]These deactivated code sections are listed in the programmers manual (see Appendix E).

6.2.1 Booking reruns

Currently, bookings are created individually: for each shipment the responsible user has to select the relevant Trip Scheme and set up the booking by entering the shipping date and loading information (see Chapter 4.1.3).

In the Tenneco logistics environment, agreements on reoccurring shipments for the same Trip Scheme are common practice. Usually a set of days or a greater interval for repetition is arranged (e. g. every second working day) and the transports will be performed on each of these occasions unless they are explicitly canceled.

One of FIONA's future functionalities could implement these booking subscriptions. An additional database table, `FIONA_bookingSubscription` (cp. Fig. 4.7), was prepared for this purpose. It gives a first impression of the data that will be required. The definition of a rerun should, for instance, offer specifying the start and end date, the dimension (e. g. day or week) and the interval (e. g. every third or fifth).

Further design considerations were made and prepared by providing the JSP source code for the corresponding user interface elements:

Upon the creation of a booking rerun, all resulting occurrences should be generated as standard bookings supplemented with a link to the subscription record. This implies that the end date for the rerun has to be mandatory, but allows editing each of the related bookings individually. This is required because the load composition usually is subject to change.

In case of a deletion or revocation request, the user should be given the opportunity to choose whether only the selected booking or all bookings from the selected one to the end of the rerun should be deleted. This enables canceling single bookings out of a rerun as well as stopping the complete subscription after a certain booking.

6.2.2 Archive

A recommended extension to FIONA that aims at increasing the efficiency when the amount of managed bookings increases, is the archiving functionality. The basic idea is to enable FIONA Administrators to shift older completed bookings out of the working data into a separate archive.

This function should take all business-relevant data from the non-master-data tables and insert it into one or two archiving tables in a flattened form. The source records can then be removed from the database. Data aggregation can support this process and further reduce the amount of records.

Adding this feature will yield multiple benefits: The users will have to handle less data and the amount of bookings exposed to daily business queries

can be restricted to data from a defined period of time. Reducing the record count in the core tables will also significantly increase the performance, especially that of complex statistics prepared on view-level.

Archived information can further be processed by possible retrospective analysis functions, specializing in long-term statistics.

Which data will have to be kept in such an archive has yet to be defined by the logistics supervisors.

6.2.3 Additional statistics

Currently, FIONA provides two types of statistics: the invoice statistics and the KPI file export (see Chapter 4.1.6). Adding further analysis functionality to the system will most probably be required as its operational area increases. Integrating additional statistics features requires only moderate implementation effort: if a file export of the results is sufficient, most of the required logic can be realized on database level. Multiple export files are already supported by the download pages of the user interface.

One suggested analysis function could perform the comparison between booked and actual timing and loading values (see Chapter 4.1.5). The resulting information is important to the logistics supervision department, as it indicates whether shipments arrive on schedule and reveals inaccuracies in the locations' loading habits.

Another possible statistics feature could create and export different reports on location-level information, e. g. comparisons of transport expenses, loading amount or shipment frequencies.

The data collected by FIONA allows the design and implementation of various further analysis facilities.

6.2.4 Extended integration

Since FIONA covers only a part of the logistics process — the ordering of goods that precedes the booking, for instance, is not touched at all — connecting it to other corresponding software suggests itself.

Examples for systems that might be integrated to some extent in a future scenario are: PakMan, SupplyWEB, the SAP-based information systems, confluentic WEB or the electronic invoicing facilities.

As the group of intended users for PakMan (cf. [Her05], p. 65f) overlaps that of FIONA, a connection of the two systems would enable the operations staff to handle data about required or stored packaging and shipment records in conjunction. With FIONA's capability of indicating the delivery of empty

packaging and defining its payer (see Chapter 4.1.2), the booking of space for required containers could be covered.

If, in addition, an application involved in supply chain management on part-level — such as the SAP system or SupplyWEB — was connected, the relation between ordered goods, the appropriate packaging and the required load allocation could be determined. This might even be used to trigger the booking of a shipment as soon as a sufficient amount of goods has been ordered.

Concerning system integration, confluentic WEB operates on the opposite side of the transport process: it is intended to use collected booking records to gain average transport cost values for its calculation to support optimization efforts. This data will instantly be available to confluentic WEB, because it is designed to use the same common database as FIONA. Therefore, the integration of the two systems is already part of the planned information system environment in Tenneco logistics (cf. [Dro06]).

Another promising feature for future integration is FIONA's invoice exporting functionality: the format is already designed to match the electronic invoice format currently used by Tenneco and its carriers. Thus, injecting this information into existing systems or creating invoices directly from FIONA after entering the values for the missing fields (see Chapter 4.4) are possible options.

Appendix A

Utilized software

The following applications were helpful in creating this thesis: OpenOffice.org Impress, OpenOffice.org Calc (cf. [Ope06]), MS PowerPoint, MS Excel, MS Project and MS Visio (cf. [Mic06]) were used to create visualizations and other analysis and project management documents.

Adobe Macromedia Studio 8 (cf. [Ado06]) was utilized to design the user interface.

Eclipse (cf. [The06]) and the plugins Exadel Studio (cf. [Exa06]), Relo (cf. [Mas06]) and Subclipse (cf. [Col06]) formed the development environment to create the Struts and Java code.

The scripts for database setup and logic were written in Oracle SQL Developer (cf. [Ora06a]).

Version control for the projects involved was done in Subversion, the client for accessing it from outside Eclipse was TortoiseSVN (both [Col06]).

The requirements specification and this thesis were created in MiKTEX (cf. [Sch06a]) using TexnicCenter (cf. [Tex06]).

Appendix B

User's manual

Attached on CD.

Appendix C

Mind Maps

Attached on CD.

Appendix D

Requirements specification document

Attached on CD.

Appendix E

Installation checklist & developer hints

Attached on CD.

Appendix F

E/R database model

Attached on CD.

Appendix G

Code documentation

G.1 Javadoc

Attached on CD.

G.2 Struts configuration visualized

Attached on CD.

List of Figures

Glossary

Accrued figure

Synonyms: accrued expense, accrual. An anticipated expense; usually derived from order data, estimations or planning. 28

AD

Active Directory 36

ADAM

Active Directory Application Mode 36

AM

After Market — The market sector comprising spare parts intended for non-original manufacturers/dealers. 7

CSS

Cascading Style Sheets 58

CSV

comma-separated values. 48

Distribution

Outbound black-box delivery with quantity-based tariffs. 16

E/R Diagram

Entity Relationship Diagram — A diagram type used to describe relational database models. 9

EC

Emission Control — The Tenneco product division comprising exhaust systems parts. 7

ERP System

Enterprise Resource Planning System 24

FIONA

Freight & Invoice Operations Network Application 7

FTL

Full Truck Load — Transport Type with fixed price, regardless of the truck's fill rate (dedicated truck). 15

Groupage

Inbound black-box delivery with quantity-based tariffs. 16

HTML

Hypertext Markup Language 58

ICY

Inter-CompanY — refers to Tenneco-internal shipments and locations 15

JSP

Java Server Pages 36

KPI

Key Performance Indicator — A figure serving statistical purposes, e. g. average truck fill rate, annual transport expenses etc. 19

KPI file

In the context of this thesis: an MS Excel file for price calculation and statistics on Tenneco Routes. FIONA extends its area of application to all Transport Types covered by the system. 19

ldm

loading meters 19

LTL

Less than Truck Load — Transport Type with price depending on the truck's fill rate. 15

Milk Run

Transport with a dedicated truck; starts or/and stops at Tenneco, visits multiple external locations. 15

MVC

Model View Controller 36

OEM

Original Equipment Manufacturers — The market sector comprising parts used for automotive production by original manufacturers. 7

Premium freight

Urgent express delivery. 16

RC

Ride Control — The Tenneco product division comprising shock absorber parts. 7

SCM

Supply Chain Management 25

Transport Type

A class of similarly organized shipments, classified according to direction, stops, transport means and pricing model 14

Trip Scheme

A sequence of stops to be visited by a truck. Used as master data to refer to when booking a delivery. 17

URI

Uniform Resource Identifier 56

VO

Value Object 59

XML

Extensible Markup Language 59

Bibliography

[Ado06] Adobe Systems Incorporated. Website: Macromedia - Studio 8. http://www.adobe.com/products/studio, Jun 2006. Last accessed: 2006-09-30.

[AIM06] AIMS Logistics. Website: AIMS Logistics. http://www.aimslogistics.com/, 2006. Last accessed: 2006-09-30.

[Apa06a] Apache Software Foundation. Website: Log4j Project. http://logging.apache.org/log4j/docs/, 2006. Last accessed: 2006-09-30.

[Apa06b] Apache Software Foundation. Website: Struts. http://struts.apache.org/, Oct 2006. Last accessed: 2006-09-30.

[BK02] Luc Besson and Robert Mark Kamen. The Transporter. Europa Corp. et al., 2002. France, USA.

[BMR+96] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal. *Pattern-Oriented Software Architecture: A System of Patterns*. John Wiley & Sons Ltd., Chichester, 1996.

[Car03] Christopher N. Carlson. Information overload, retrieval strategies and Internet user empowerment. In Haddon and Leslie, editors, *Proceedings The Good, the Bad and the Irrelevant (COST 269)*, pages 169–173, 2003. available at: http://eprints.rclis.org/archive/00002248/.

[Cav04] Chuck Cavaness. *Programming Jakarta Struts*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 2 edition, Jul 2004.

[Col06] CollabNet. Website: Tigris.org — Category home: Software configuration management tools. http://scm.tigris.org/, 2006. Last accessed: 2006-09-30.

[Cre06] Creativyst, Inc./Dominic John Repici. Website: How To: The Comma Separated Value (CSV) File Format. http://www.creativyst.com/Doc/Articles/CSV/CSV01.htm, 2006. Last accessed: 2006-09-30.

[Dro06] Jens Drogi. Confluentic WEB — Porting existing Software of "Confluentic 1" into Tenneco's Client-Server-Environment. Diploma Thesis, University of Koblenz-Landau, Koblenz, Oct 2006.

[Exa06] Exadel, Inc. Website: Exadel - Exadel Studio. http://www.exadel.com/web/portal/products/ExadelStudio, 2006. Last accessed: 2006-09-30.

[For06a] Fortune Magazine. THE FORTUNE 500 LIST. *Fortune Magazine*, 153:p. n/a, Apr 2006.

[For06b] Fortune Magazine. Website: Fortune 500 2006. http://money.cnn.com/magazines/fortune/fortune500/, Apr 2006. Last accessed: 2006-09-30.

[HDFW03] Ted Husted, Cedric Dumoulin, George Franciscus, and David Winterfeldt. *Struts in Action — Building web applications with the leading Java framework*. Manning Publications Co., Greenwich, CT, USA, 2003.

[Her05] Timo A. Herborn. PakMan — Packaging Management System — Conceptual Development of a web based Packaging Management System. Master Thesis, University of Koblenz-Landau, Koblenz, Sep 2005.

[IBM06a] IBM Corporation. Website: IBM Lotus Notes. http://www.ibm.com/software/sw-lotus/products/product4.nsf/wdocs/noteshomepage, 2006. Last accessed: 2006-09-30.

[IBM06b] IBM Corporation. Website: IBM System i. http://www.ibm.com/systems/i/, 2006. Last accessed: 2006-09-30.

[IBM06c] IBM Corporation. Website: IBM WebSphere Software. http://www.ibm.com/software/websphere/, 2006. Last accessed: 2006-09-30.

[Ill06] Illumine Limited. Website: The Definition of Mind Mapping. http://www.mind-mapping.co.uk/mind-mapping-definition.htm, 2006. Last accessed: 2006-09-30.

[Inf04] Infor Global Solutions. Press Release: Infor to Provide Supply Chain Software to Tenneco Automotive. http://www.infor.com/cps/rde/xchg/infor/hs/Company_PressReleases_697.htm, Dec 2004. Last accessed: 2006-09-30.

[Inf06] Infor Global Solutions. Website: Infor Global Solutions. http://www.infor.com/, 2006. Last accessed: 2006-09-30.

[KLS05] Ralf Krechel, Lars List, and Daniel Schmidt. confluentic. Diploma Thesis, University of Koblenz-Landau, Koblenz, Mar 2005.

[Kop06] Jürgen Kopper. CaTIS —Carrier and Tariff Information System. Diploma Thesis, University of Koblenz-Landau, Koblenz, Oct 2006.

[LK02] Kevin Loney and George Koch. *Oracle9i: The Complete Reference*. McGraw-Hill/Osborne, New York et al., 2002.

[Mas06] Massachusetts Institute of Technology. Website: Relo — Relationship based Exploration. http://relo.csail.mit.edu, 2006. .

[Mic05a] Microsoft Corporation. Website: Microsoft TechNet — Microsoft Windows Server TechCenter — Active Directory Concepts. http://technet2.microsoft.com/WindowsServer/en/library/77a19ae8-bffe-42ca-a841-3d18ea62dc9b1033.mspx, Aug 2005. Last accessed: 2006-09-30.

[Mic05b] Microsoft Corporation. Website: Microsoft TechNet — Microsoft Windows Server TechCenter — ADAM Concepts. http://technet2.microsoft.com/WindowsServer/en/library/29fb059e-544c-4577-bf7c-ba4b08df48431033.mspx, Aug 2005. Last accessed: 2006-09-30.

[Mic06] Microsoft Corporation. Website: Microsoft Product Information Center - Office. http://www.microsoft.com/products/office, Sep 2006. Last accessed: 2006-09-30.

[Mün05] Thomas Münch. PakMan — Packaging Management System — Implementation of a web-based Packaging Management System for Tenneco Automotive Europe. Bachelor Thesis, University of Koblenz-Landau, Koblenz, Aug 2005.

[Nie03] Jakob Nielsen. Website: Alertbox: Information Pollution. http://www.useit.com/alertbox/20030811.html, Aug 2003. Last accessed: 2006-09-30.

[Ope06] OpenOffice.org. Website: OpenOffice.org: Home. http://www.openoffice.org, 2006. Last accessed: 2006-09-30.

[Ora06a] Oracle. Website: Oracle SQL Developer. http://www.oracle.com/technology/products/database/sql_developer, Aug 2006. Last accessed: 2006-09-30.

[Ora06b] Oracle. Website: Oracle, The World's Largest Enterprise Software Company. http://www.oracle.com, 2006. Last accessed: 2006-09-30.

[PTS06] PTS Pamtrans. Website: Pamtrans - transport i spedycja miedzynarodowa. http://www.pamtrans.pl/, 2006. Last accessed: 2006-09-30.

[SAP06] SAP AG. Website: SAP - Business Software Solutions Applications and Services. http://www.sap.com, 2006. Last accessed: 2006-09-30.

[Sch06a] Christian Schenk. Website: MiKTEX Project Page. http://www.miktex.org, 2006. Last accessed: 2006-09-30.

[Sch06b] Schenker AG. Website: Schenker AG. http://www.schenker.com/index_en/index.html, 2006. Last accessed: 2006-09-30.

[Sch06c] Schenker AG. Website: Schenkernet. http://www.schenkernet.com, 2006. Last accessed: 2006-09-30.

[Sun99] Sun Microsystems, Inc. *Java Code Conventions*, 1999.

[Sun06a] Sun Microsystems, Inc. Website: Sun Developer Network (SDN) — Java Technology. http://java.sun.com/, 2006. Last accessed: 2006-09-30.

[Sun06b] Sun Microsystems, Inc. Website: Sun Developer Network (SDN) — JavaServer Pages Technology. http://java.sun.com/products/jsp/index.jsp, 2006. Last accessed: 2006-09-30.

[TA 05] TA Corporate Technology. *Tenneco Automotive Java Coding Standards*, 2005.

[Ten05a] Tenneco Inc. Website: Tenneco - profile. http://www.tenneco.com/overview/financial.html, 2005. Last accessed: 2006-09-22.

[Ten05b] Tenneco Information Technology. *Java Design Guidelines*. Tenneco Inc., Dec 2005.

[Ten06] Tenneco Inc. Website: Supplier Manual. http://www.tasupplier.com/, 2006. Last accessed: 2006-09-30.

[Tex06] TexnicCenter.org. Website: TexnicCenter.org. http://www.texniccenter.org, 2006. Last accessed: 2006-09-30.

[The06] The Eclipse Foundation. Website: Eclipse.org. http://www.eclipse.org, 2006. Last accessed: 2006-09-30.

[Tra06] Transwide Ltd. Website: Transwide. http://www.transwide.com/en/index.html, 2006. Last accessed: 2006-09-30.

[TT04] The IEEE and The Open Group. *The Open Group Base Specifications Issue 6 — IEEE Std 1003.1: crontab — schedule periodic background work*. The IEEE and The Open Group, 2004 edition, 2004. available at: http://www.opengroup.org/onlinepubs/009695399/utilities/crontab.html.

[Uni06a] Universität Koblenz-Landau. Website: Institute for IS Research. http://www.uni-koblenz.de/FB4/Institutes/IWVI/index_html, 2006. Last accessed: 2006-09-30.

[Uni06b] Universität Koblenz-Landau. Website: Universität Koblenz-Landau, Campus Koblenz. http://www.uni-koblenz.de/index_extern.html, 2006. Last accessed: 2006-09-21.