Final Report For Publication

P I S C E S
IN-96-SC.1004

Project Co-ordinator: Fraser Williams Logistics Ltd.

Partners: Van Ommeren Agencies Rotterdam BV

Liverpool John Moores University

Project Duration : 1st January 1997 to 31st January 2000

Date: **26 September 2000**

PROJECT FUNDED BY THE EUROPEAN COMMISSION UNDER THE TRANSPORT RTD PROGRAMME OF THE 4TH FRAMEWORK PROGRAMME

Table Of Contents

0. Partnership

0.1 Companies

Fraser Williams Logistics (FWL)

A major international software company with 30 years experience providing software and systems to the European and Worldwide Transportation Industry, whose role in this project was the overall project coordinator and to be responsible for delivering a system with the other consortium members.

Organisation	**Fraser Williams Logistics Ltd.**
Address	Port of Liverpool Building, Pier Head, Liverpool, L3 1BY,United Kingdom
Contact Name	Dr. William Temple
Telephone	+44–151-2275995 Fax +44–151-2360845
Email	Bill.temple@fwlogistics.co.uk.

Liverpool John Moores University (LJMU)

LJMU is the fourth largest educational establishment in the UK. The LJMU team has internationally recognised expertise in the development of innovative support systems for industry - in particular the development of tools for supporting scheduling and planning activities that exploit artificial intelligence. As the sole academic partner on the project, LJMU will play an important role in the dissemination of the principles and science derived from the project.

Organisation	**Liverpool John Moores University**
Address	2 Rodney Street, Liverpool, L3 3AF, UK
Contact Name	Professor Gerald Kelleher
Telephone	+44-151-2312104 Fax +44-151-20744594
Email	F.N.Arshad@livjm.ac.uk

Van Ommeren Agencies (VO)

Van Ommeren Agencies was part of the Royal Van Ommeren Group, a diversified global transportation and tank storage company established over 150 years ago.

Following the merger between Royal Van Ommeren and Royal Pakhoed, Royal Vopak has evolved. Royal Vopak is the world's largest distributor of chemicals.

Both companies activities are focused on representing the commercial, operational and port interests of shipping lines worldwide and have a long history of providing various logistics services to the chemical and oil industry.

The PISCES project has enabled Van Ommeren and now Royal Vopak to offer shipping lines, importers and exporters a European logistics organization for the movement of containerised freight with efficient and cost effective European inland transport and improved information flows.

Organisation	**Royal Vopak**
Address	10 Westerlaan, Rotterdam, C016 CK, Netherlands
Contact Name	Mr. Peter Den Breejen
Telephone	+31-10-4642262 Fax +31-10-4642261
Email	peter-den.breejen@vopak.com

0.2 Project Supervision

Organisation	**European Commission, Directorate General for Energy and Transport**
Address	Rue de la loi 200, B-1049 Brussels
Contact Name	Mr. Patrick Mercier-Handisyde

1. Executive Summary

1.1 The PISCES System

As road transport becomes more expensive, the need for environmentally effective solutions is sought. It is apparent that competitiveness in such a climate is only possible if systems are in place which enable smooth transfer of information between the parties involved. The PISCES project addressed the problem of providing a framework in which information can be supplied to the users, the people sending freight (the companies, ports, railways etc.) and the receivers of the goods.

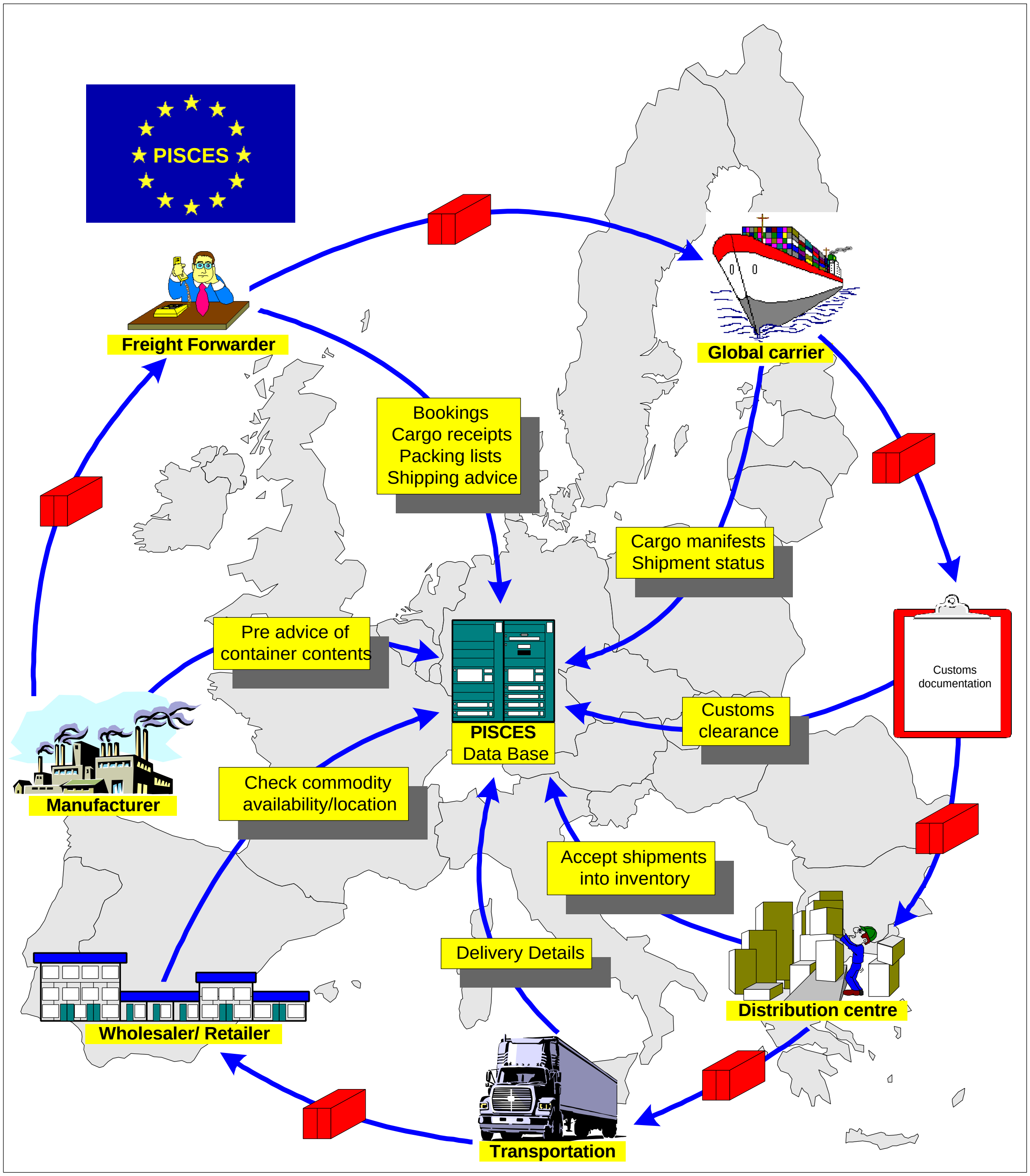
PISCES is an integrated system for intermodal transport. It enables the complete transport chain to be scheduled and managed as a single process, without the need for all the participants to adopt the same information system or even the same information standards. PISCES exploits a constraint based, easy to use, high availability system to make its users more efficient, profitable and service oriented.

The PISCES project focused on the real time exchange of operation critical information via the World Wide Web (WWW). The application resides on top of the network, mailboxes, EDI translators and mappers. PISCES was designed to be usable on top of the existing Port Community systems if required, or on a stand-alone basis independent of any network. All separate site users of PISCES are able to pool and store relevant information. Access is provided on a secure 'need to know' basis depending on client access rights. PISCES provides high-flexibility, configurable triangulation of container traffic. This is achieved via genetic algorithms and constraint satisfaction problem based technology which exploits the early provision of reliable information to provide improved triangulation performance, in terms of both user satisfaction and improved economy.

1.2 Components Of the PISCES System

PISCES utilises a database which houses information about clients e.g. manufacturer, consignee, shipper and cartage type. PISCES is designed to extract information from any current client database – they are not restricted to using a prior database system. Information from the database is only made accessible to those participants who have access rights. PISCES has five levels of security.

Having made a booking, the client may want further levels of control about transport management of their goods. This may include having specific information about time constraints imposed by particular routes, e.g. especially important in the delivery of perishable goods or if there is a strike by any of the transport trains, or if a cargo depot simply wants to ensure that containers are optimally used. In this case, the clients can have access to the PISCES scheduling system for dealing with container triangulation.



Graphical Interpretation of the PISCES concept

2. Objectives Of The Project

2.1 Project Specification

This project addressed the problem, within the freight management industry, of the efficient and flexible integration of multimodal transport chains. This project focused on the difficulties of providing information through the transport chain to enable the efficient implementation of logistic systems supporting the movement of goods through the chain. The project now demonstrates that intermodal transport can be managed more effectively, is more attractive and can be focused on more environmentally benign and sustainable modes if systems are in place which enable smooth transfer of information between the parties involved. The project was concerned with the management of information within the transport chain so that effective decisions can be taken concerning the management of that chain. In particular we are addressed the problem of information provision to the users of the chains in the sense of the people sending freight (the "suppliers"), the companies moving those goods (the shipping companies, ports, railways, ...) and the receivers of the goods. In essence the current "flow" of consignments through the intermodal transport chain is effectively only a sequence of stages, and at each of these stage the cargo is halted while it is re-marshalled, re-loaded, and allocated "new" control references in new systems. PISCES enables a transport chain to be scheduled and managed as a single process, over a range of modes and with different providers, without the need for all the participants to adopt the same information system or even the same information standards. The project was concerned most directly with intermodal chains between ports and other modes of transport (demonstrations being restricted to target transport chains connecting European hinterland consignments to Western seaports, (typically Rotterdam, Antwerp and Hamburg), terminal services and selected deep sea carriers. Basic commodity flows that can demonstrate the feasability and benefits of the concepts include automotive components, chemicals (food grade and industrial) and foodstuffs, including wine. Other projects have sought to address these issues through the provision of better information via the provision of information technology. Most of these earlier initiatives have sought to have access to vested interest "own" information to which they are currently not privileged to receive, or they have used IT which is intrusive or costly to many of the participants. In the worst cases projects have required all parties to adopt a proprietary additional system or application. For all these reasons very limited success has been achieved to date.

The PISCES project was based on a neutral or negative impact strategy for all participants; namely it would not impose a significant extra burden on existing IT setups, it would not require publicising confidential information, and it would not give rise to extra operations. To this end the project only accesses information which is already available in (perhaps) another environment in another format and processing it anonymously and confidentially into a format which is usable by the other partners in the chain. It is likely that this approach could become the de facto standard for information management of intermodal transport following the positive feedback from members of the exploitation board.

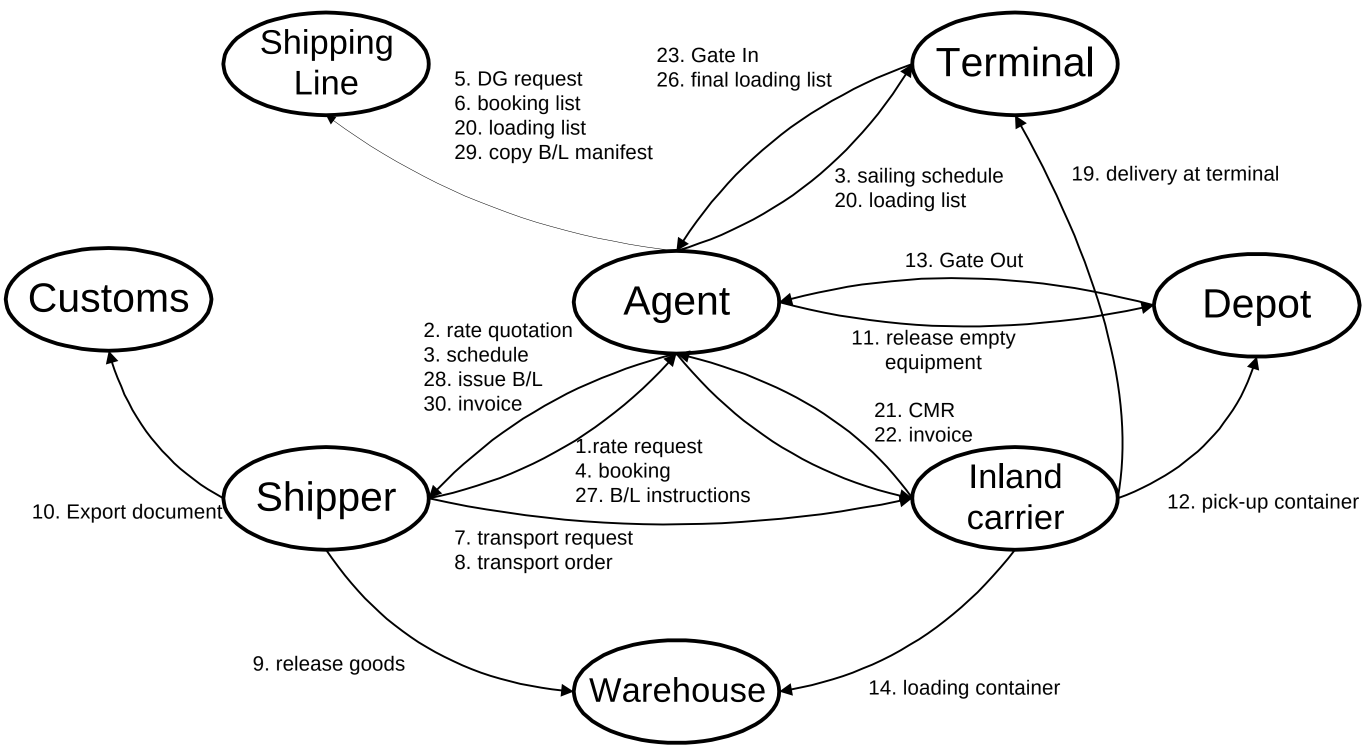
The project was biased towards the practical realisation of solutions to these issues and the provision of demonstrations of the technology developed. It is clear that the general issue described here - the overall problem of information transfer within intermodal transport chains - is too big to be tackled within one project with any degree of certainty. The project therefore concentrated on an important subset of these chains (through and either side of the port) and contained partners with expertise in all elements of that chain and in the provision of information systems to support the chain.

The research component of the project was the need to provide an underpinning information framework, which would reach the objectives outlined above. The contention was that constraint based data modelling - which models information by defining the constraints or parameters within which it must be instantiated - is a technology that could be successfully used in this area. Given the need for data interchange and transfer between and within the system and the need to keep information on a "need to know" , minimal basis constraints were are the natural way of expressing and working with the information needed. This has not been tried before and, whilst the technology is well understood and robust (it has been successfully fielded in large scale industrial scheduling and planning systems) further work was needed to realise the potential of this approach within the intermodal information chain. The project was designed to provide that work and provide demonstrations of the viability of the technology and the concept.

2.2 Business Flow - Van Ommeren

The PISCES system is set up to improve the information exchange and integration within multi-modal transport chains. By doing this the control over the actual transport chain is being enhanced; for example by accurate planning of human resources and equipment at a container terminal.

In order to show how the PISCES system fits into day-to-day transportation and where there is exchange of information between different transport links, we will describe and explain the different screens of the PISCES system according the transport chain from purchase to delivery of goods.
The main information flow between the main parties involved in the transport of cargo is as follows:



Export

The transportation of goods starts with a trade transaction; different parties come together and agree on the selling/buying of goods. The details, as purchase order number and specifications of the goods are to be found on the ***purchase order*** screen. The purchase order screen is set up for trade transactions and is mainly used by trading parties. It allows them accessing the data through various search criteria; in that way the data is very quickly and easy accessible. Both the buying and the selling parties have complete insight into the exact details and specifications of the shipment and the current status of the transaction at all times.

After the sales transaction the goods need to be transported from the suppliers factory/warehouse to the receivers of the goods. The transport is arranged either via the intervention of a forwarding company or directly via a shipping line (or its agent), for which a freight is negotiated. The departure of the ocean going vessel is looked up at the schedule screen (***Schedule***); because PISCES is continuously updating the latest schedules are available. The schedules show all the ports (and dates of arrival and departure) which the vessels will call at.

After booking, the Shipper receives a booking reference (job reference) from the shipping line, which is linked with the consignment (***consignment screen***). In the consignment screen all the shipping details are shown together with all the details of the shipment, like the name of the vessel, the container number and, after shipment, the Bill of Lading number. After actually booking the container, the Shipper makes sure that his purchase order number is linked with the booking reference from the shipping line, in order to give the shipping line access to the consignment details and the Shipper/Consignee to the shipping details. After booking, the container(s) are released at a depot by using the ***Release Empty Containers*** screen; the depot is informed for which booking reference how many containers are going to be picked-up. Normally this is done through a fax or a telephone call. With this feature the depots can check 24 hours a day whether the booking reference is correct.

Either the shipping line or the Shipper arranges the pick-up of the goods. From the ***jobs awaiting delivery*** screen the shipping line knows which shipments still need to be picked-up, for which he can commission a road haulier. At the same time the road haulier can see which jobs need delivery and can apply for those jobs, which means optimizing the use of his trucks. The same information about the deliveries can be used for the ***transport scheduler***. All the jobs awaiting delivery can be centrally processed and organized. The direct advantages and opportunities from this scheduler are obvious: minimizing the total amount of ‘ton/kilometers’ by reducing the number of trucks needed and avoiding empty running.

For the actual transportation the container, in which the goods are carried, is being picked-up at a depot, stuffed at the Shipper’s premises and then returned to the container terminal, it is also possible that some additional pre-carriage is required like barge or rail transport. Because of the information available in PISCES, the pre-carriage is combined and bundled with other transports. The container terminal, where the ocean-going vessel

departs, can check all the incoming containers with the instructions within PISCES (***terminal arrivals***).
The system is, for example, capable of generating alerts in case that the containers have not been delivered in time. This kind of alerts allows preventive measurements.

Because all the container statuses and movements are frequently recorded, the shipping lines have the opportunity to control their container stocks in a dynamic way. Together with the transport ***scheduler*** the container stocks can be managed as circumstances require and minimizing re-positioning of containers. The ***Depot Container Levels*** screen is used for assigning minimal stock levels for all depots.

The scheduler allows re-calculating of the situation in short periods of time, for example when special types of containers are required in short notices or when large amounts of containers are needed at an inland depot.
Another advantage could be the possibility to give a forecast of the container stocks, because the destination of all the incoming containers is known in advance.

Finally, the container is loaded on board of the ocean-going vessel and heading for its destination. The involved parties (depots, terminals, etc.) report all these moves into the PISCES system. As a result the goods can always be traced in the ***consignment-tracking*** screen; the latest status and the place of the shipment is available at any time. This screen gives an overview from all the different transport legs. From loading of the container till the stripping at the consignee’s place. It allows a quick check of shipments by the use several search criteria; for example by booking reference or by container number. After shipment, the shipping line (’s agent) can easily use the PISCES system to retrieve the necessary information to draw up the required documentation as for example the Bill of Lading, the shipping manifest or the customs documentation. The Bill of Lading and manifest information are themselves fed into PISCES to inform the Port of Discharge about shipments to be expected.

Import:

For the import side of the transport chain, the functionalities of PISCES work more or less the same. Instead of feeding the system with data, the information can now be used for delivering the goods to the Consignee. The receiving agent can easily retrieve the manifest data from PISCES and use it for documentation purposes and arranging the delivery of the goods. The consignee has consulted the system and is well aware of the arrival of his shipment. He has been checking the latest status in the consignment-tracking screen or is informed by an automatically drawn up Arrival Notice. The terminal in the port of discharge is well in advance informed about the containers to be discharged. Thus, by using the PISCES system all the different parties have their information available in time and complete.

After all the freight has been paid the containers is released to the receiver. By using the *Bill of Lading Numbers* screen the exact details of the receivers are entered after which, for example, notify messages are sent. Once again the agent can check which containers still await delivery (jobs awaiting delivery). Via PISCES the containers are released at the terminal, this avoids waiting truck drivers. Similar as for export all the container movements are stored in PISCES; the shipping lines can use the same tracking and tracing module to retrieve and monitor their equipment. It is possible from the PISCES system to create events to ensure that containers are being collected and returned in time. In future developments, using information from PISCES it will be possible to generate the necessary customs documents, declaration may be made easier by electronic means, also it will be possible for Port authorities to be informed of the Dangerous Goods on the incoming vessel.

Alerts & events:
The basic idea of the alerts and events system is that most companies in the logistic chain are handling many transports simultaneously and will have no possibility to check the status of each individual shipments.

Some companies are handling thousands of shipments daily and it will be extremely labour intensive to monitor all shipments.

In fact we are of the opinion that most companies are only interested to know the exceptions, for this reason we have identified in the Pisces system the alerts & event possible and concentrated of the alert of ‘consignment information is received out of sequence’. This allows the parties involved to concentrate on the problem and take the appropriate corrective actions.

Security of information:
Security of information is absolute requirement as some information processed in the Pisces has a high degree of confidentiality.

Pisces has two levels of security:

External security (access to the system)

It need no further explanation that only authorized users are allowed access to the system. To guarantee correct use of the Pisces system a login procedure was developed, which is a combination of a user code and password.

Each user has only access if the correct user code and password is used.
After the correct password is used the Pisces system recognizes the different type of users and per individual type of user only relevant information is released.

Internal security (different information for different users)
The Pisces system is a multi-user database, which allows easy communication between the various parties in the transport chain. As described above this has many benefits but at the same we have realised that not all information should be available for all users. The central database contains confidential information on volumes, commodity descriptions, etc which can be very sensitive information.

For example most exporters are interested to have the information of his competitor, what quantities is the competition shipping to certain areas. Who is doing business with whom?

Although this type of information is available in the system it is not available for all users.

For this reason we have selected the information which was processed in the system. Based on our in-house experience and some discussions with exporters and importers we selected this information. For each item we considered the necessity for including this information in the system. Less information means fewer risks.

We furthermore decided to avoid unnecessary risks and not included very confidential information, such as prices, and commodity values.

For example a trucking company is receiving all relevant transport data, such as loading address, terminal, weight of cargo etc. but the trucking company will have no access to the purchase order data.

All users have only limited access to the Pisces system and only the required information is shown.

3. Means used to achieve the Objectives

The prime objectives of the project were :

a. To reduce the costs to all parties concerned with freight transport / distribution.

b. To reduce the environmental impact.

c. To enhance the service to the buyers of transport services.

To solve these objectives, a two-fold approach was taken :-

a. The iterative design of the project using the latest technology available, thus constantly revisiting the technology used for the system and optimising were feasible.

b. The on-going involvement of industrial partners, thus providing the business knowledge and approach for the system.

The overall outcome is the PISCES system, an integrated system for intermodal transport. It enables the complete transport chain to be scheduled and managed as a single process, without the need for all participants to adopt the same information system or even the same information standards. PISCES exploits a constraint based, easy to use, high availability system to make its users more efficient, profitable and service oriented.

4. Scientific And Technical Description Of The Project

4.1 PISCES Database

4.1.1 Introduction

In the initial stages of the project it was envisaged that EDI links would play a major role in extracting information from the system. However, following the initial research required for the prototype system, and recognising the developing reach of the Web, it was agreed that the PISCES prototype system would be developed as an Internet based track and trace logistics system which would communicate to external systems using EDI and E-mail technology. Thus the key features of the prototype PISCES system were defined to
be

- Internet connectivity / transmission
- Confidentiality Of Information e.g Security
- Consignment Track And Trace Facility
- Partner Profile
- Efficient Scheduling

Following the prototype development and the evolution in technology during the course of the project, the functionality of the system was extended in order to deliver a pilot system which was used by Van Ommeren for a trial period at their offices in Rotterdam. This additional functionality included extensive data entry by using other technologies, communicating to external systems using EDI and E-mail facilities, and processes to generate events / alerts message at defined statuses of the logistics cycle where applicable.

This document is aimed at enabling the reader to understand the technology behind the PISCES system and the tools used in order to produce the optimum system solution at the time. It will discuss each aspect of the system – Internet technology, database technology, security of information and finally the functionality of the system in order to create the final solution.

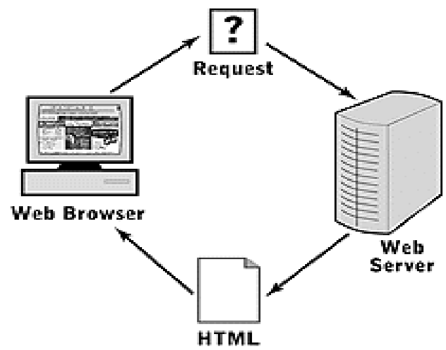
4.1.2 Technology Used In PISCES

4.1.2.1 Internet Connectivity & Communication

The Web greatly reduces the costs of administering and maintaining applications, while allowing for a thin, low-cost client.

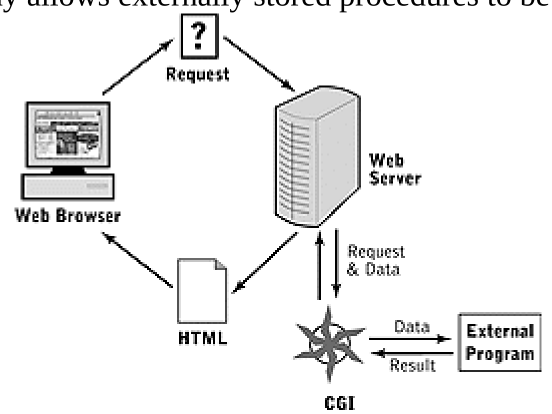
Fraser Williams has currently used ORACLE tools for Web deployment.

Traditional Web technology for comparison with the PISCES systems is as follows.



Traditional Web Architecture

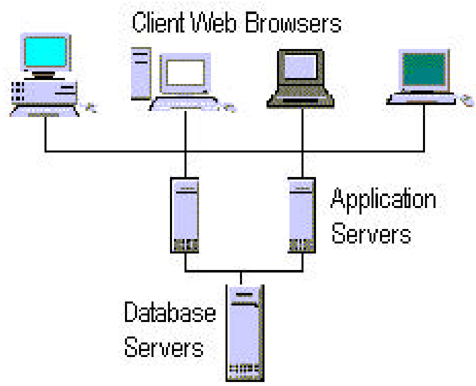
One development of static HTML has been to introduce CGI (the common gateway interface). This gateway allows externally stored procedures to be run on request.



CGI Web Architecture

Most current server-based and client /server applications assume that the user is directly connected to the Application Server, and that the server is aware of the user and is keeping track of what that user is doing. This is a state-oriented architecture. In order to use such an application on the Web, the CGI program must contain code to keep track of the user, all requests associated with that user, what information was sent to the user, and what information the user sent to the application. All this is required to create the illusion of a state-oriented environment.

The development of the Web Server to include Web implementation, application logic and processing led to three distinct tiers being created. This is known as three-tier architecture.



Three-tier architecture

Web applications are deployed and maintained on centralised Application Servers, from which they download to end users' Web browsers at runtime. To roll out a Web application, the end users requires the application's URL. This distribution method reduces the time, cost, and complexity of deploying applications to a large or geographically dispersed end user base, all without installing application software on their desktop machines.

"Thin client" architecture. The only client-side requirement is a Web browser (must be Java enabled). Any end user with a browser can run Web-enabled applications. This greatly reduces processor and memory requirements for end users' desktop machines.

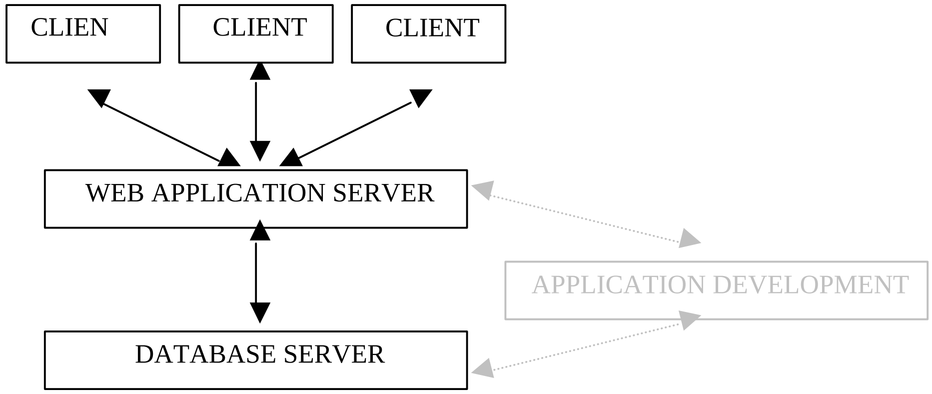
4.1.2.2 PISCES Internet Technology

The PISCES system required technologies that would enable any client to access PISCES given a modern, Java-enabled browser, an Internet connection and a valid PISCES login.

During the course of the project, great advances in available technologies have been observed and these have been exploited to enable a deployment of the PISCES model that is both user friendly and efficient.

In the early stages of the project during the prototype development, Designer 2000 (ver 2.1) was used to produce PL/SQL programs that dynamically generated HTML when executed.

A PL/SQL cartridge was installed on the Web Server to enable execution of these



programs via the WWW, using the Java-enabled HTML browser.
It is on the middle tier that connection to the database and security interrogations takes place (Reference Security section)

Designer 2000 Web Server Generator:

Designer 2000 was, for our purposes, a GUI for defining applications, and subsequently generating 3GL PL/SQL code which would run on a Database Server. The main features used were as follows: -

Reverse engineering of relevant schemas
Development of application modules
Generation of Web applications

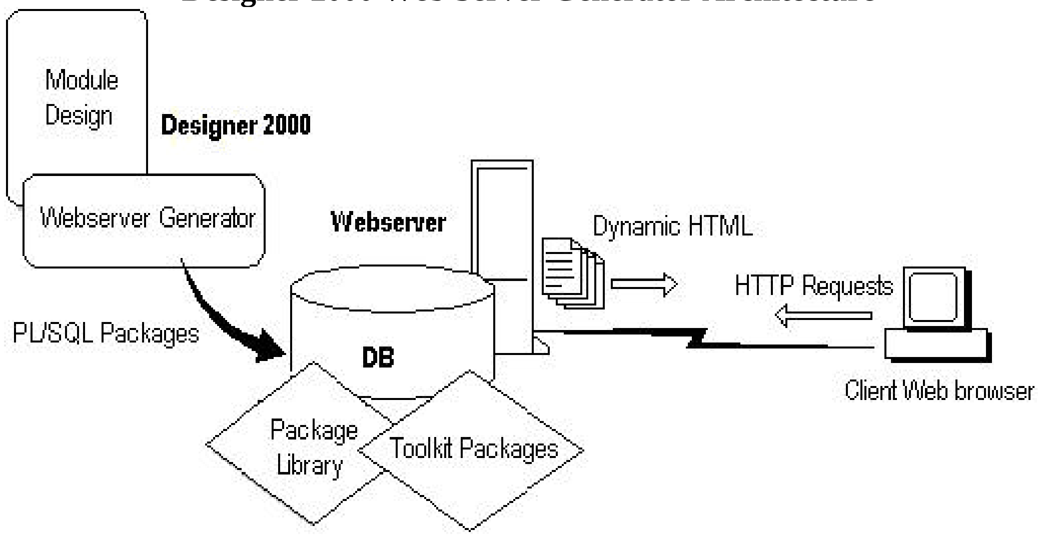
Using this development path, a client therefore only needs to enter a URL that points to a program on the Web Server. It is here on the middle tier that connection to the database and security interrogations takes place.

Within the Designer 2000 repository, generated applications based on module and database design specifications were recorded.

The main input to the generation process was a module design specification recorded using the Designer/2000 Design Editor. PL/SQL routines stored in the database determined the modules, records and columns that can be accessed.
Preferences were input to determine the general look and feel of the generated application and were customised to suit particular requirements.

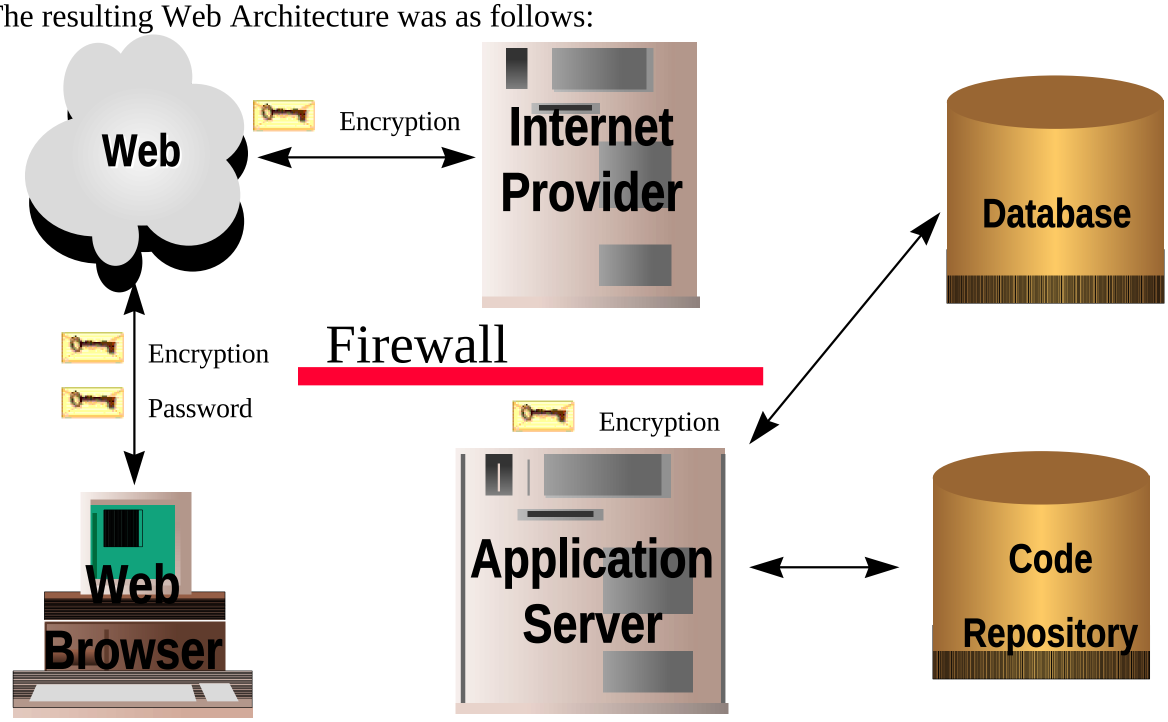
In the system, relationships are dynamically checked at run-time to ensure PISCES operations change with the evolving nature of partners and their relationships. Using PL/SQL was the fastest way to manipulate data in an oracle database.

Designer 2000 Web Server Generator Architecture



For data manipulation, generated Web Server applications made use of PL/SQL API packages that provided insert, update, delete and lock procedures for each application table. These packages were generated and installed once the table design was complete, prior to generating the Web Server application which was then run via the Oracle Web Application Server.

The resulting Web Architecture was as follows:



PISCES Internet technology

The Runtime Engine communicated directly with the database through SQL*Net. This was the most efficient method of accessing an Oracle Database. The PL/SQL syntax generated was the most efficient means of accessing this data.

Only very basic HTML / JavaScript could be produced using this method. When PL/SQL packages needed to be modified to incorporate additional features, it was done by placing markers in Designer 2000 that will be placed in the generated code.

JavaScript and some Java was inserted into the generated HTML but because this design is essentially HTML based, it was fast, efficient and required an extremely thin client (browser only). This design was ideal for the broad client base that PISCES requires and many PISCES modules were produced using this method.

A set of values that specify how the PL/SQL Agent is to connect to the Oracle 8 Server to fulfil an HTTP request was configured in a DAD (Database Access Descriptor). Each PL/SQL Agent had to be associated with a DAD. The information in the DAD included the username (which also specifies the schema and the privileges), password, connect-string, error log file, standard error message, and the language to be used.

The Application Server was used to service requests from browsers and store externally referenced files such as templates and graphics. These procedures received cookie/variables/parameters from the browser requests. The output of these procedures was sent back as HTML/JavaScript to the browser requesting it via the Application Server.

Thus summarising the above technology: -

- The Browser would send requests for URLs to the Application Server (parameters / variables known as cookies are sent with these requests)

- The Application Server received requests for information from the thin client, and subsequently invoked application modules to service the client requests. In the case of PISCES, these are PL/SQL modules.

- The Database Server responded to data requests from the Application Server.

This initial HTML based system relied on the traditional WWW approach where clients submitted a request, the request was processed by the server, and the response sent back. This stateless interaction enabled simple database interaction however where more complex inserts / updates were necessary, a more modern approach was needed and as the specification of PCs improved and new communication protocols became available, other options were investigated.

4.1.2.3 New Technologies Available

Due to the improvement in technology since the start of the project, we investigated what were the feasible options to use for the PISCES project in the timescale available. These were defined to be Designer 2000, Oracle Forms and Cognos Web.
In each case, we identified the advantages and disadvantages for using each option.

4.1.2.3.1 Designer 2000 Web Server Generator

Advantages

The use of preference sets enables the fast generation of more simple modules with little coding necessary. This leaves more development time for writing the PL/SQL necessary for the more complex modules (see Disadvantages)

Developers experienced with Designer 2000 and PL/SQL do not need to learn Java, JavaScript, or any other new language.

Developed to specifically satisfy the requirement from customers to provide "shopping basket" applications on the Web, Web Server Generator facilitates this type of application very easily via search methods and form submits.

The HTML / JavaScript generated can be run on different platforms. The look and feel of the application will not differ for users with different platforms / browsers.

Advantages can be gained from other tools from the Oracle suite, especially administration tools such as Reports Queue Manager.

Disadvantages
Only very basic HTML / JavaScript can be produced using this method. The PL/SQL packages need to be modified to incorporate additional features.

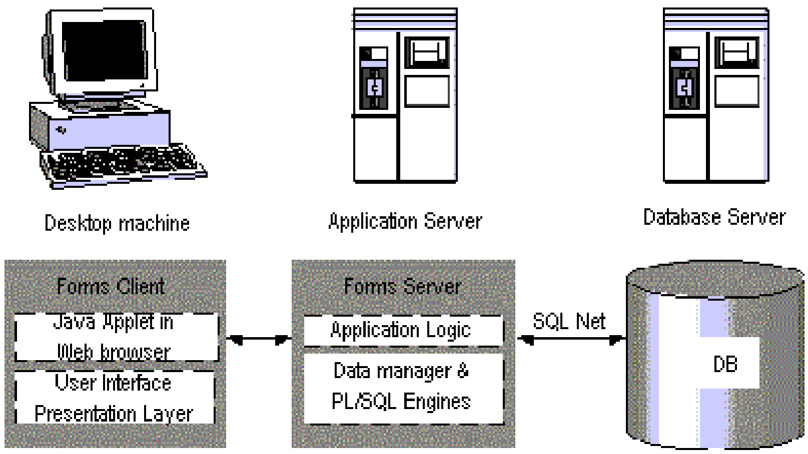
The linking of modules is intended to be implemented using a hierarchical structure. Each module is therefore generated independently and the links have to be coded manually (by making amendments to the PL/SQL packages).

Inputting large amounts of data via modules generated in this way is not feasible. The resulting application is HTML based and therefore does not offer cross screen validation in the way that consignment entry on the freight system does. Records **can** be created via these modules but they would have to be created via one screen of input.

4.1.2.3.2 Oracle Forms

How it works

Oracle Forms implements a unique architecture by deploying a Forms client and a Forms Server.



Forms Client/Server Web Archetecture

The Forms Client

The Forms Client is a Java applet downloaded at runtime from an Application Server to an end user's Web browser. This module displays the form's user interface and manages interaction between end users and the Forms Server. The Forms Client receives interface commands from the Forms Server and translates them into interface objects for the end user. Some interface events handled by the Forms Server Runtime Engine in a client/server implementation (such as typing characters in a text field, or moving around a dialog) occur only on the Forms Client in the Web implementation, with no interaction with the Forms Server runtime engine.

The Forms Client contacts the Forms Server when users perform high-level operations (such as accepting or cancelling a dialog) and other operations (such as checking a checkbox or navigating between fields). These operations involve validation processing and cause default and user-defined triggers to fire.

The Forms Server

The Forms Server runtime engine (and all application logic) is installed on Application Servers, not on client machines. All trigger processing occurs on Database and Application Servers, while user interface processing occurs on the Forms Client.

The Forms Server contains:

- **The Listener.** The Forms Server Listener initiates the Forms Server runtime session and establishes a connection between the Forms Client and the Forms Server Runtime Engine.

- **The Runtime Engine.** The Forms Server runtime engine is a modified version of the Forms 5.0 Runtime Engine, with user interface functionality redirected to the Forms Client. It handles all form functionality except UI interaction, including trigger and commit processing, record management, and general database interaction.

To start and run a Form Builder application on the Web, end users use a Java-enabled Web browser to access a URL. The following sequence occurs automatically:

- The browser is validated by the server as appropriate from the communication. If the browser is invalid, appropriate download sites are offered.

- The HTML page and the Forms Client applet are downloaded from the Application Server to the user's browser.

- The Forms Client sends a request to the Forms Server Listener.

- The Listener contacts the Forms Server runtime engine and connects to a Forms Server runtime process (either by starting a new process, or by connecting to an existing process). If included in the HTML page, Forms command-line parameters (such as form name, user ID and password, database SID, menu name, and so on) and any user-defined Form Builder parameters are passed to the process by the Listener.

- The Listener establishes a direct socket connection with the Runtime Engine, and sends the socket information to the Forms Client. The Forms Client then establishes a direct socket connection with the Runtime Engine. The Forms Client and Runtime Engine then communicate directly, freeing the Listener to accept start-up requests from other end users. The Forms Client displays the application's user interface in an applet window outside the main window of the end user's Web browser.

Advantages

Extends and makes use of the traditional strength of Oracle's client/server development suite. Any existing client/server application can therefore be deployed without changing the form definition (.FMB file). You can run the same Forms executable (generated .FMX file) in client/server mode or on the Web.

The Forms Client is:

- **Generic.** You are not required to deploy a separate Java applet for each form you wish to deploy on the Web.

- **Dynamic.** The Forms Client dynamically reacts to the current form at runtime, requesting and displaying only the information and user interface elements necessary to represent the current state of the application at any given time.

- **Feature-rich.** The Forms Client supports all user interface widgets and tools available in a client/server implementation. Due to Java object standards, the look and feel of some Forms widgets will vary only slightly when deployed on the Web.

- **Thin.** At start-up, only those class files necessary to render the initial state of an application are downloaded to the end user's machine. Additional class files are downloaded dynamically (as needed) to support additional UI functionality.

Developers experienced with Developer/2000 do not need to learn Java, JavaScript, or any other new language.

Forms is designed in accordance with Oracle's NCA. Developer/2000 applications run as NCA-compliant Web cartridges.

Forms uses Java to map GUI widgets to their native counterparts on other platforms. The look and feel of widgets will differ only slightly between end user platforms (Windows, Mac, Motif, and so on).

Can easily be interfaced with Reports Server via Reports cartridge or Web CGI. In this way, URLs can be associated with objects in a report. Users can navigate to other Web pages or launch other reports that provide more detailed or related data.

Development is performed using Forms, which has been an Oracle flagship product for many years. Forms development skills are widely available and easily attainable for developers with Oracle experience.

Advantages can be gained from other tools from the Oracle suite, especially administration tools such as Reports Queue Manager or by using Designer 2000 to generate the .FMB file without the use of Forms

The same application can be deployed on a traditional client server network as on the Internet. In this way a non-Web enabled product could be developed in parallel with the Internet application.

Since we are using an Oracle Database, the Runtime Engine communicates directly with the database through SQL*Net. This is the most efficient method of accessing an Oracle Database.

There is a reduction in Web traffic as not all user actions are sent to the Forms Server for validation.

With bandwidths increasing exponentially in today's IT market, the disadvantage of downloading the Forms Client applet will be increasingly insignificant.

Disadvantages

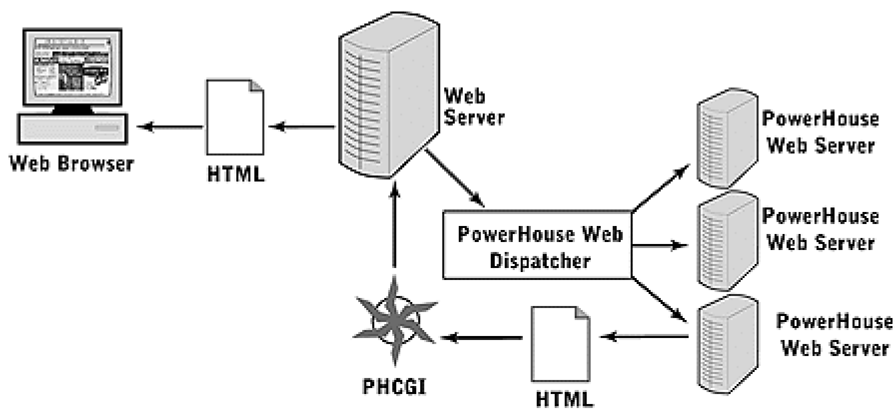
The Forms Client applet is typically around 500k and must be downloaded by any browser requesting a cartridge or even non-cartridge page. With present transfer rates this is impractical for attracting casual surfers.

4.1.2.3.3 Cognos Web

How it works

The existing QUICK "engine" can be used in the Web context by making each request from the user's browser a self-contained transaction. This removes the need to track the user's state within the application, and allows a single QUICK process appear to be multi-user.

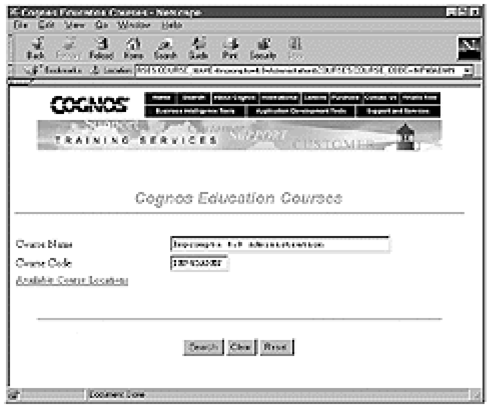
Conceptually, you can imagine a QUICK "Web" process as always waiting at the "Screen Id=" prompt. Providing the name of the screen causes it to be loaded. Then the appropriate action command is specified, followed by any necessary data. Following this, any results are displayed on the screen and eventually sent back to the browser. Finally, the screen exits and is unloaded. The QUICK process is now back at the "Screen Id=" prompt, waiting for the next request.



Each HTML template may contain two different layouts, a "Record Form" and a "Report Form."

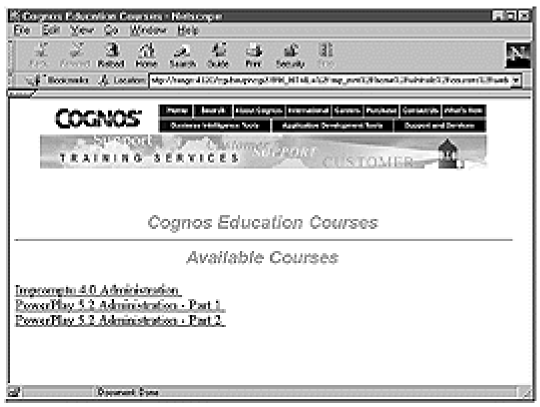
The "Record Form"

The Record Form presents data in a "fill-in-the-blanks" layout. The user enters or changes data by tabbing between or clicking on fields. Buttons are provided for the available actions. DESIGNER procedures are listed in a drop-down list with a "Go" button next to the list.



The "Report Form"

The Report Form is the default layout used to display the results of a search. There is a link for each record found based on the search criteria. Clicking on a link brings up the Record Form that displays the corresponding data.



Advantages

Using PowerHouse Web in conjunction with the PowerHouse 4GL we could build Web applications using robust, tried-and-tested, tools already used within the organisation to build our core business applications.

The HTML / JavaScript generated could be run on different platforms. The look and feel of the application will not differ for users with different platforms / browsers.

Although a new application, it is likely to be more refined, mature and stable due to its strong links with the familiar PowerHouse syntax.

Developed to specifically satisfy the requirement from customers to provide “shopping basket” applications on the Web, Cognos Web facilitates this type of application very easily via search methods and form submits.

Generation of applications relies on pre-written HTML template files. By providing alternate templates, alternative generation (e.g. alternative languages) can be achieved.

To obtain apparently state oriented applications, Cognos employ QUICK. QUICK is a state-oriented program, and all current PowerHouse applications are state-oriented.

QUICK is state-oriented because it knows about the user, what they have done, and what they are currently doing. The Web is stateless, there is no persistent connection between the browser and the server however by using QUICK applications we could appear to maintain the state of connections.

The architecture of Cognos Web allows for PowerHouse Web Dispatcher processes to be started on any PowerHouse Series 8 server. These processes can communicate with each other regardless of the operating system they are running on: UNIX, Windows NT Server, OpenVMS etc.

This enables the use of virtually any configuration. For example, we could use Windows NT Server as the Web Server, have one PowerHouse Web application running on an OpenVMS machine, and another PowerHouse Web application on a UNIX machine. Alternatively, we could run our Web Server on UNIX and our PowerHouse Web applications on Windows NT Server machines.

Disadvantages

Has been slow to emerge on the market with Cognos Server Web Edition currently only at release 1.0

Only very basic HTML / JavaScript can be produced using this method. The code would need to be modified to incorporate additional features.

Cognos concede that Cognos Web does not attempt to provide full windows functionality. When this is needed, they recommend using Axiant 4GL with the Microsoft ActiveX control on a LAN/WAN. In this way, Windows-based applications can run on a server with Windows Terminal Server accessed from client's Web browsers. Cognos believe this is sufficient as complex Windows applications are not usually required on the Web. When this complexity **is** required, they argue it is for a limited number of trained, known users. Citrix MetaFrame could be used to obtain wider distribution.

Inputting large amounts of data via modules generated in this way is not feasible. The resulting application is HTML based and therefore does not offer cross screen validation in the way that consignment entry on the freight system does. Records **can** be created via these modules but they would have to be created via one screen of input.

With QUICK there is a single process per user until the user leaves the application. On the Web, a user might have their browser connected all day long, but only occasionally actually perform any inquiries or processing. This would demand excessive resources from the server if there was a single QUICK "session" for each Web user.

Common browser commands like Previous Page and Next Page are local to the browser and have no QUICK equivalents.

PowerHouse Web Server must have an HTML template in order to generate output in HTML format. An HTML template is an HTML page that contains PowerHouse substitution tags.

You can add your own scripts to the templates to perform extra functions however, you must ensure that the data stream sent from the browser to the Web Server is what PowerHouse Web expects to process. This could restrict the flexibility when implementing requirements.

Building a Web application with PowerHouse is not as simple as taking the existing application and putting it through a "recompile-and-go" process, unless the existing application fits the discrete business transaction model. Because our PowerHouse applications rely on the state-driven nature of the permanently connected terminal environment, they could not simply be recompiled for deployment to the Web.

Summarising

The following table compares features of the three development methods described in this document:

	Web Server Generator	**Oracle Forms**	**Cognos Web**
Previously used / skill set present.	☑	✘	☑
Uses Oracle database.	☑	☑	☑
Uses PL/SQL – most efficient for Oracle Databases.	☑	☑	✘
Uses SQL*Net – most efficient for Oracle Databases.	☑	☑	✘
Quickly generates simple modules.	☑	☑	☑
Generates complex modules with no coding necessary.	✘	☑	✘
Generic across platforms.	☑	☑	☑
Interfaces naturally with Oracle tools	☑	☑	✘
Enables data entry across screens.	✘	☑	✘
Parallel non-Web enabled application.	✘	☑	✘
Utilises all Windows GUI widgets	☑	☑	☑
Thin client – no software / applets required.	☑	✘	☑
Well established / developed product.	☑	☑	✘
Minimal Web traffic.	✘	☑	✘
Makes use of existing code.	**?**	**?**	☑
Maintains business logic from current applications.	✘	✘	☑
Easily enable alternate generation (e.g. languages).	☑	☑	**?**
Support for Unix/NT/OpenVMS	☑	☑	☑
Releases idle user processes to increase resources	☑	☑	✘

Table to compare development methods

In reviewing the options, it was agreed that Internet based Oracle Forms was the sensible route to use, as it is already a well known tool worldwide.

4.1.2.4 Oracle Forms Technology In PISCES

This second complimentary path was extensively used to develop additional modules for the pilot system which was to be used by Van Ommeren for a trial period at their offices in Rotterdam.
The Oracle Forms route also meant that the original development was not redundant and would be used in the pilot system for the track and trace modules.

In order to use Oracle Forms, it was achieved by installing a plug-in to the client browser, thus more processing could be completed on the client during these complex inserts / updates. The Microsoft Java Virtual Machine is one such plug-in made available to exploit this kind of client side processing however this was in it's infancy and contained errors. Oracle's JInitiator can replace the Microsoft Java Virtual Machine and was found to provide a reliable infrastructure for such transactions to take place for the PISCES system.

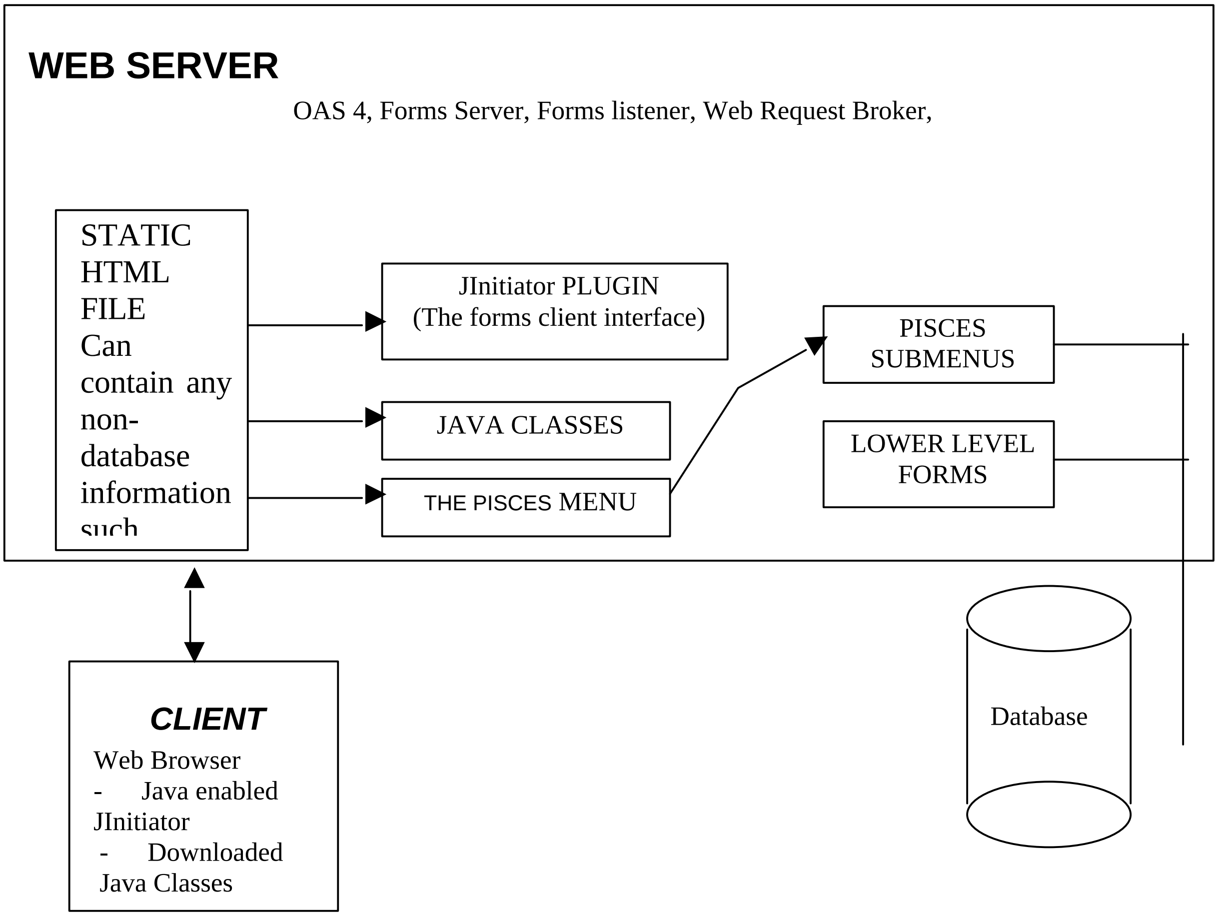
Using JInitiator, the middle tier software is supplemented with Oracle Forms Server, Forms listener, the forms executables and JInitiator. A new Oracle proprietary communication protocol was used to make socket connections between the middle tier and the client. To the client this connection appears to be persistent and therefore allows states of transactions to be maintained between forms. Validation of items within forms can be performed with less server side interaction due to processing within JInitiator.

We used a static implementation for the Webforms. This means that the PISCES front-end form had to have an HTML file on the middle tier.

The HTML file is static (i.e. not dynamically generated as with Designer 2000). A virtual directory on the Web Server points to the physical directory that the HTML file resides. It is the Web Server, the virtual directory and the HTML filename that form the URL in the form http://machine_name/virtual_directory/static_html_file

The machine name can be any registered name so could easily be registered as WWW.PISCES.NET or similar.

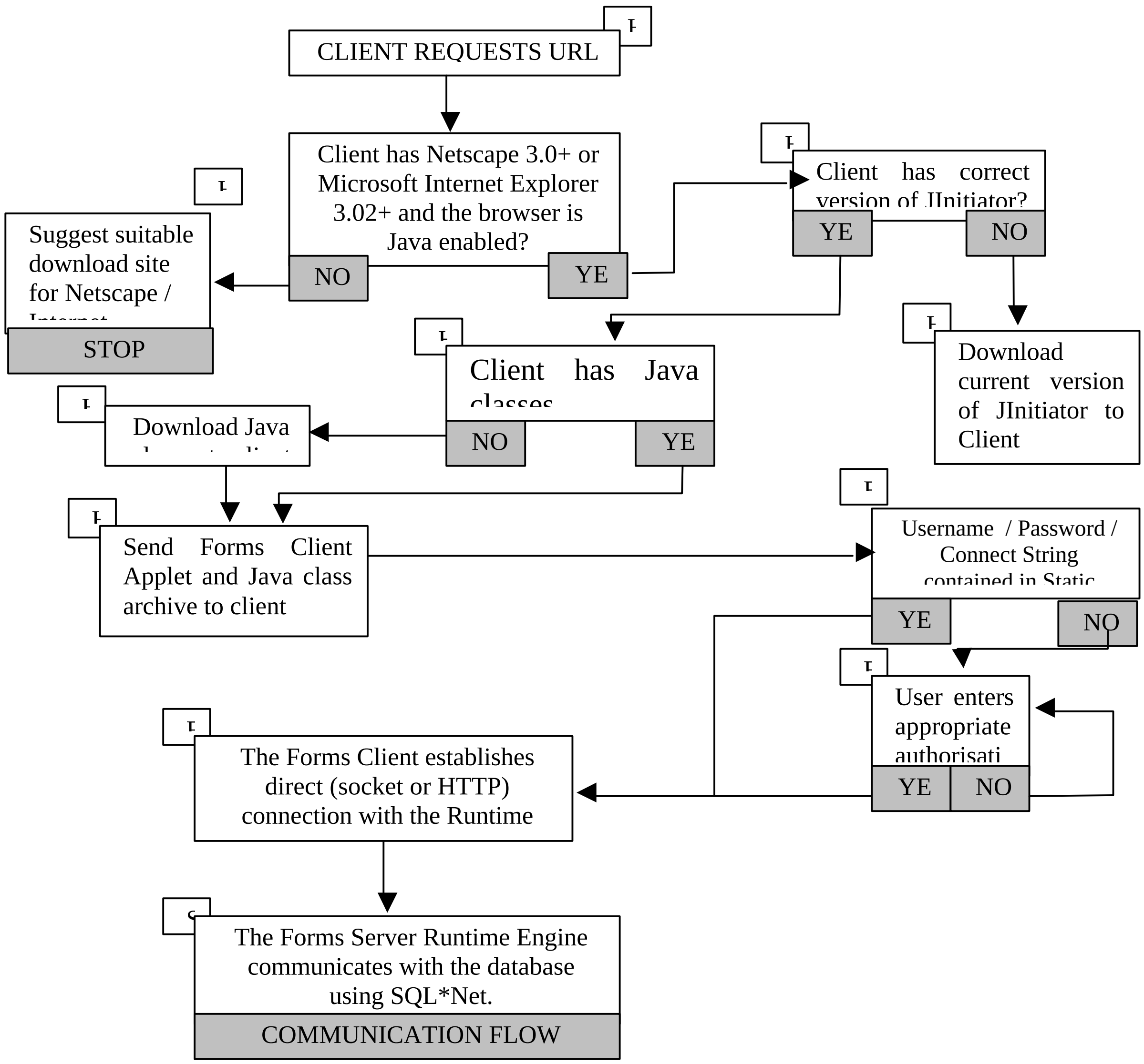
In the PISCES system, by calling the static HTML file, the user invokes JInitiator and a top-level form. This form can subsequently open other forms that reside in the same virtual directory as the top-level form. Once the client has the top level form open, normal client server hierarchy of the forms can be used.



It is the static HTML file which determines which level of JInitiator to use, provides a logon screen to the PISCES database, loads up the Java classes needed and calls the Java applet and first top level form. The HTML file also determines what communication protocol to use (sockets). After that, the developed forms can be placed in the forms virtual directory for access by other forms.

Using this method, an application with one top-level form, need only have one static html file and therefore one URL. A top-level form will typically therefore be a menu form for providing access to lower level forms.

The communication and process flows involved are as outlined :



H

HTTP (Hypertext transfer protocol).
This protocol is used to establish the initial stages of communication.
Forms Messages.

F

This is a proprietary protocol.
This communication takes place on port 9000 (configurable) and is encrypted.

S

SQL*Net
is used for communication between the Forms Server Runtime Engine and the database server. This takes place on port 2521 (configurable) and is not encrypted.

The modular design of PISCES allows intuitive navigation between related areas whilst providing an easily scalable system

PISCES provides an integrating platform to enable the current information exchange systems already in use to be more widely accessible and to improve efficiency in data exchange. The functionality includes a discrete system architecture that supports documentation, in/out gate, traffic optimisation, warehouse management, transport management, load allocation, route scheduling and distribution.

4.1.2.5 Client Hardware Configuration

In order for the client to use the system, they will be required to have the following hardware.

PC (Pentium 90 MHz or better processor.),
12 MB free hard disk space (recommended 20 MB).
48 MB system RAM minimum.
Netscape Navigator 3.0 or later and Microsoft Internet Explorer 3.02 or later.

4.1.2.6 Database

4.1.2.6.1 Database Technology

What is a database ?

A database is a structured collection of data, which may be manipulated to select and sort desired items of information. For example, an accounting system might be built around a database containing details of customers and suppliers. In larger computers, the database makes data available to the various programs that need it, without the need for those programs to be aware of how the data are stored.

A database-management system (DBMS) program ensures that the integrity of the data is maintained by controlling the degree of access of the applications programs using the data. Databases are normally used by large organisations with mainframes or minicomputers.

Oracle Database

The PISCES database was constructed using Oracle utilizing the well proven Oracle7 relational database server, this, with its proven technology, allows the easy storage, manipulation, retrieval, and sharing of many types of data - including business records, documents, messages, images, audio, and video - in any type of application.

An Oracle database allows the secure and reliable delivery of powerful applications to any number of users, over any type of network, no matter what hardware and software they use.

The database server is the key to solving the problems of information management. In general, the server reliably manages a large amount of data in a multi-user environment in order that many users can concurrently access the same data. All this must be accomplished while delivering high performance. The database server will also prevent unauthorised access and provide efficient solutions for failure recovery.

An Oracle Server provides efficient and effective solutions with the following features:

4.1.3 Client/server (distributed processing) environments

To take full advantage of a given computer system or network, Oracle allows processing to be split between the database server and the client application programs. The computer running the database management system handles all of the database server responsibilities while the workstations running the database application concentrate on the interpretation and display of data.

4.1.4 Large databases and space management

Oracle supports the largest of databases, potentially terabytes in size. To make efficient use of expensive hardware devices, it allows full control of space usage.

4.1.5 Many concurrent database users

Oracle supports large numbers of concurrent users executing a variety of database applications operating on the same data. It minimises data contention and guarantees data concurrency.

4.1.6 High transaction processing performance

Oracle maintains the preceding features with a high degree of overall system performance. Database users do not suffer from slow processing performance.

4.1.7 High availability

Oracle is fully capable of working 24 hours per day with no down time to limit database throughput. Normal system operations such as database backup and partial computer system failures do not interrupt database use.

4.1.8 Controlled availability

Oracle can selectively control the availability of data, at the database level and sub-database level. For example, an administrator can disallow use of a specific application so that the application's data can be reloaded, without affecting other applications.

Openness, industry standards

Oracle adheres to industry accepted standards for the data access language operating systems, user interfaces, and network communication protocols. It has been evaluated by the U.S. Government's National Computer Security Centre (NCSC) as compliant with the Orange Book security criteria; the Oracle7 Server and **Trusted Oracle7** comply with the C2 and B1 Orange Book levels, respectively, as well as with comparable **European ITSEC security criteria**. Oracle also supports the Simple Network Management Protocol (SNMP) standard for system management. This protocol allows administrators to manage heterogeneous systems with a single administration interface.

4.1.9 Manageable security

To protect against unauthorised database access and use, Oracle provides fail-safe security features to limit and monitor data access. These features make it easy to manage even the most complex design for data access.

4.1.10 Database enforced integrity

Oracle enforces data integrity, "business rules" that dictate the standards for acceptable data. As a result, the costs of coding and managing checks in many database applications are eliminated.

4.1.11 Distributed systems

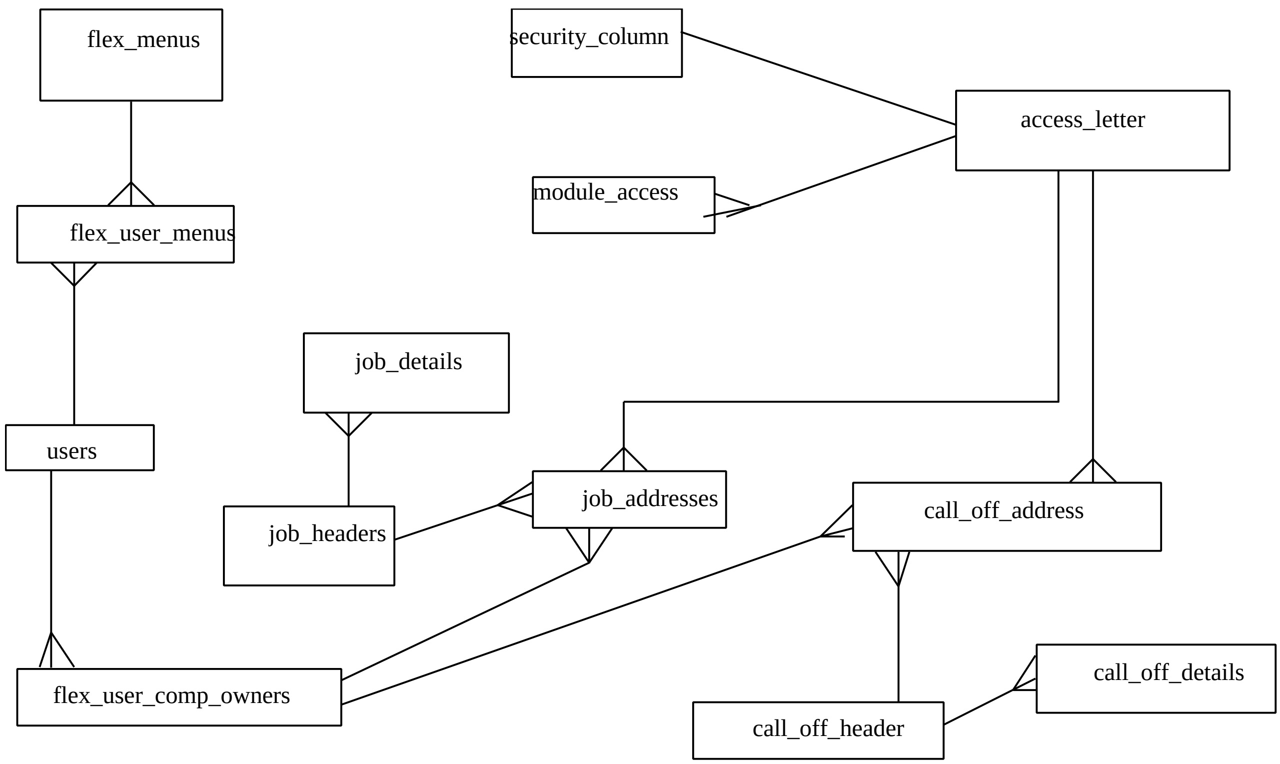
For networked, distributed environments, Oracle combines the data physically located on different computers into one logical database that can be accessed by all network users. Distributed systems have the same degree of user transparency and data consistency as non-distributed systems, yet receive the advantages of local database management. Oracle also offers the heterogeneous option that allows users to access data on some non-Oracle databases transparently.

4.1.12 The PISCES Database

Oracle software is ported to work under different operating systems. Applications developed for Oracle can be ported to any operating system with little or no modification. The Oracle database was configured to reflect the PISCES data model as shown below.

The data model consists of business data objects - tables which are inter-related by elements in each data table, hence the term ‘relational database’.

Fraser Williams Fraser Williams designed the database to present a data model of general applicability. As such the data model reflected the business requirements within the logistics chain which were applicable to the business needs of Van Ommeren and companies which have direct business links with Van Ommeren.



The PISCES Data Model

The Fraser Williams data model has certain distinct advantages, it was designed in such a manner that it is extremely flexible and contains a very hierarchical structure. It is envisaged that the data model could be easily extended to include other areas of the business logistics chain as shown :

4.1.13 Security Technology In PISCES

Security was an important issue for the PISCES project as the information in the database has a high degree of confidentiality.

Security in PISCES can be grouped into two categories -

External Security. At the connectivity level, it is important that only selected users have access to the system. An organisation should not be able to access data that belongs to a different organisation. Simply, users that are not registered, should not be able to enter the system.

Internal Security. At the database level, data owners are to be considered. Different privileges are given to different users, that are only able to access the data they own.

In order to manage the security, several objectives have been achieved :

- **Access to screens, and navigation between screens has to be controlled.**
- **Elements on screens are secure, down to item level.**
- **Security in screen queries is provided.**

Within the PISCES system , five levels of security have been defined :-

- Application Security - This restricts users to options which are permissible to the user type assigned to them.

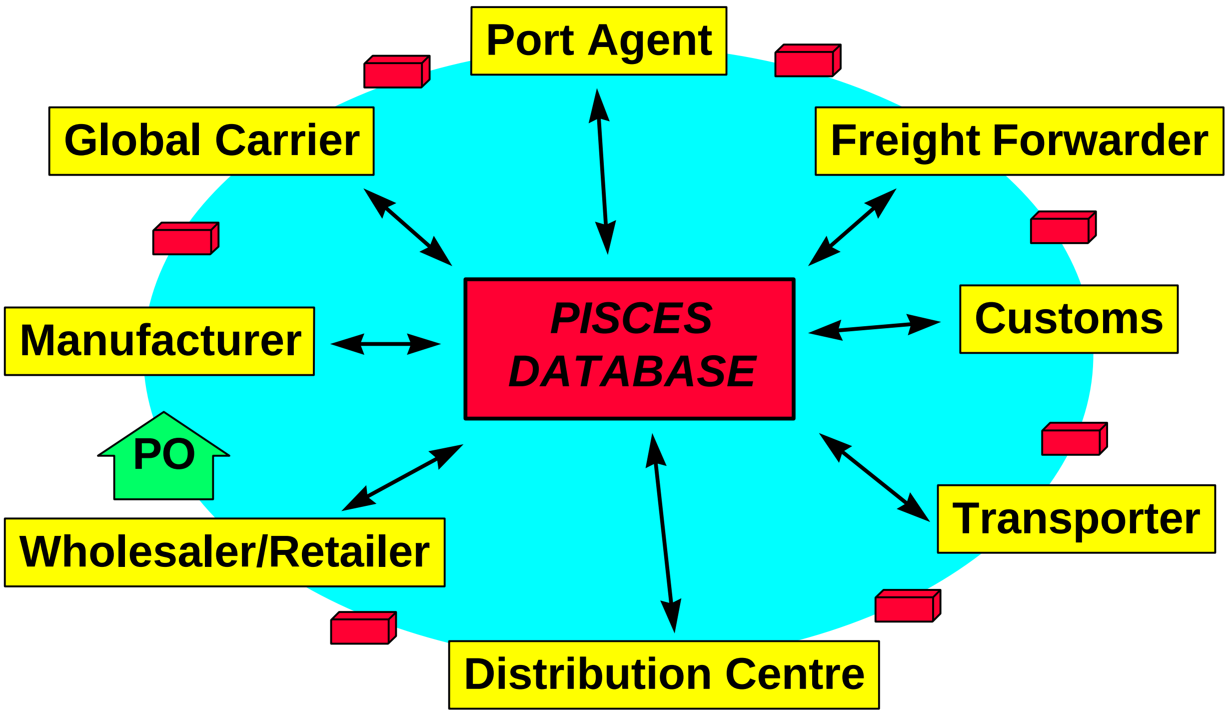
- Program Security - Programs can only be executed if the user has a role that enables them to execute that program.

- Module Security - Filters the records returned from the database. If the user has an association with this module then records are allowed.

- Record Security - A filter on each record matched. If the user has an association with a particular record, then the record is returned.

- Column Security - A user must have an acceptable access type (relative to that record) in order to see sensitive columns.

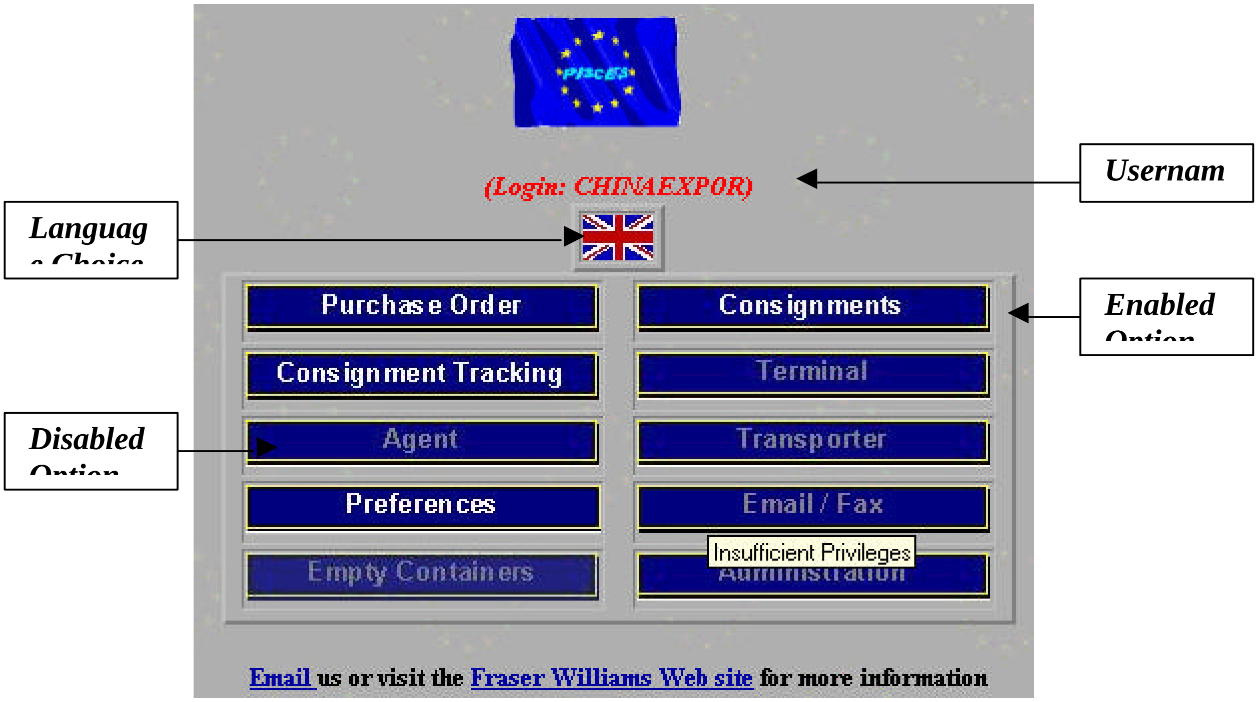
These various security aspects required for PISCES were implemented using a combination of PL/SQL procedures and features inherent within the Oracle DBMS as follows :



4.1.13.1 Application Level Security

This is set within the administration module, by assigning a user type to each user login id. For each user type a defined set of menu options are defined. These are set up in a flexible manner thus enabling the restriction of options to selected users.

Currently within the PISCES system, the user login of 'CHINAEXPOR' is defined as user type 'shipper' and thus has the following main menu displayed :



Main menu (original system)

4.1.13.2 Program Level Security

This is maintained within the Oracle DBMS. Execution rights to the PL/SQL programs are granted / revoked within ORACLE by calls from the PISCES administration module. A valid entry in the Flex_Menus table will create a role for that module. An entry in Flex_User_Menus will allocate that role to a particular user. The subsequent housekeeping required for synonyms and execute rights must be handled by the administrator either manually or via a batch process.

Example entry in Flex_Menus

Example entry in Flex_User_Menus

4.1.13.3 Module Level Security

This is controlled by settings within the PISCES administration module (module access). This determines three security features:-

Example entry in Module Access

- Only assigned modules are given menu options at the main menu screen.
- Only assigned modules are given navigation buttons to point to them.
- If an unassigned module is executed (theoretically impossible), no records are returned as the module security forms the first filter on all records returned.

4.1.13.4 Record Security

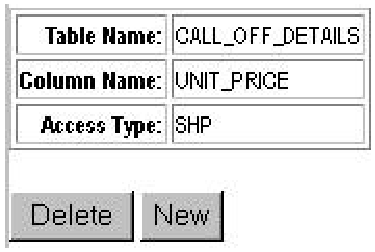
Each record matching the query criteria is checked for relationship with current user. If the user has a valid relationship with the record in the relevant context, then the record is returned.

Example entry in Access Letter.

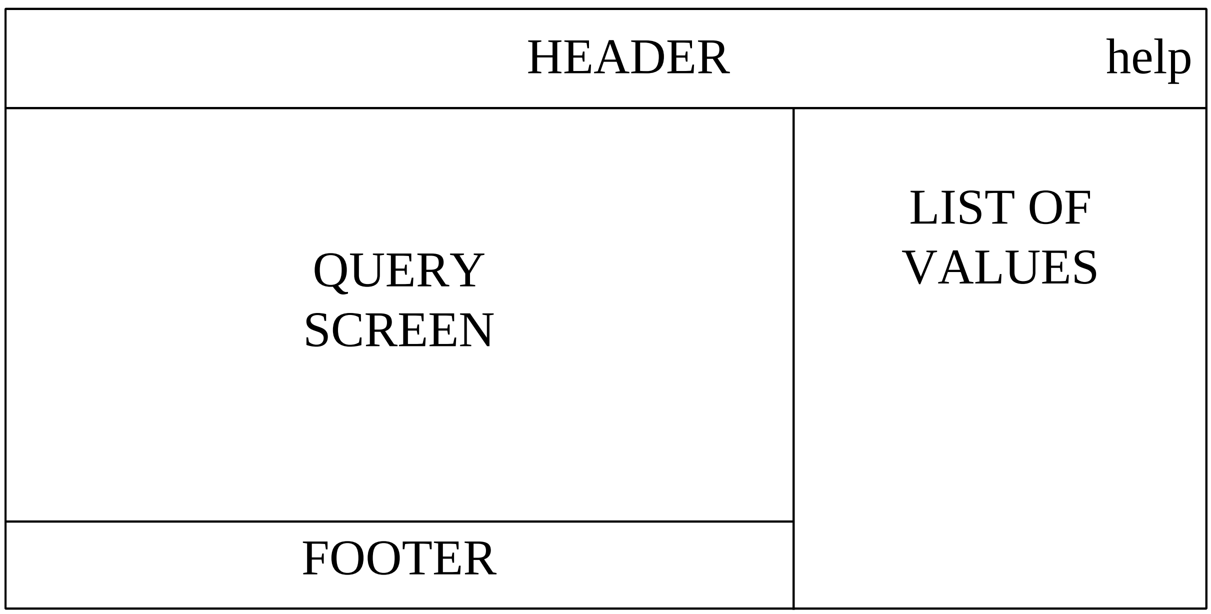
An access string is generated for each record and an access string is held for each user. Where matches are found, records are returned.

4.1.13.5 Column Security

The administration module also holds details for sensitive columns so that data will only be disclosed to parties with an authorised relationship.

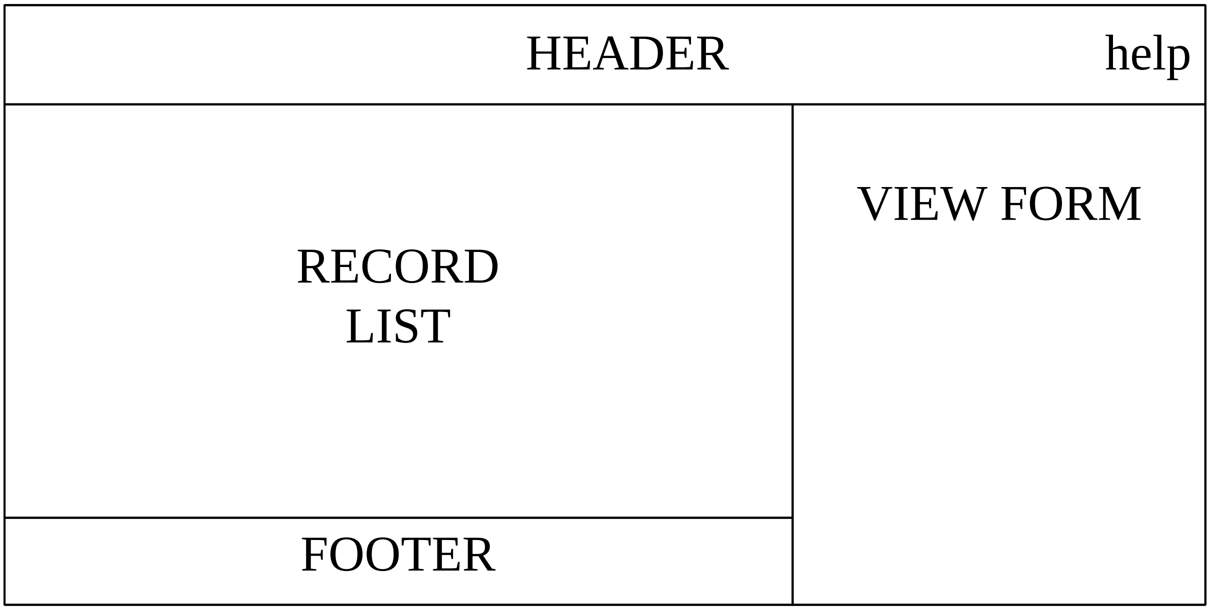


Example entry in Column Security



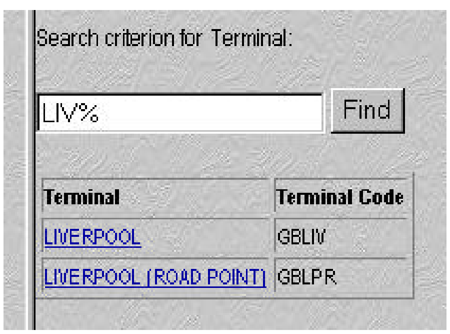
Layout of initial query screen

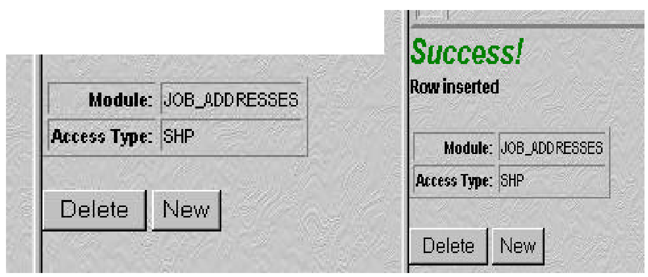
Similarly, the record list and view form (returned following a query) contains mainly dynamically generated HTML.



Layout of record list and view form

The view form of the screen can also be used to display lists of values (pick lists) and for update / delete frames.





4.1.14 System Functionality In PISCES

4.1.14.1 Functionality

The system functionality has been derived from extensive research and analysis of the Van Ommeren business model.

The functions covered by the PISCES system address the following areas of the logistics chain :

- **Purchase Order** – Facility to search and track purchase order data within the system, including drill down to individual stock item level.

- **Consignments –** Enables the flexible retrieval of master consignment data within the Pisces database, including drill down to Consignment Tracking.

- **Consignment Tracking –** Facility to track a consignment within the system given any valid reference. The logistic chain for a given consignment may consist of a number of separate journey legs, each of which may be tracked.

- **Agent –** Facility to locate opportunities for transporters and allow on-line booking of jobs to a haulier.

- **Transporter –** Facility for terminal staff to anticipate work loads and record arrivals

- **Bill Of Lading Information –** Enables the user to either view the bills of lading given a voyage reference or view the bills of lading for a specific voyage and journey leg.

- **Release Empty Containers –** Facility allows the agent to inform the depot of the number of containers to be released against a specified booking reference.

- **Booking Containers –** Enables users to view the depot cods and container types for a specified bill of lading and discharge point.

- **Schedule –** This options allows the entry of information relating to the sailing list and has a link to the consignment screen which show information of the voyage

- **Pool Levels –** this is a standing data maintenance screen where the container levels for each container type within a depot are defined. These figures are updated in background tasks in the PISCES system.

- **Container Types**- this screen allow a list of container types to be defined, this data is then used for validation in other modules in the PISCES system.

- **Administration** – Allows on-line updating of administration features by a privileged super user, e.g. security access to menus.

- **Generate the Schedule file for the scheduling and triangulation process –** This option generates and exports a file to the scheduler system to provide efficient scheduling information for selected consignments.

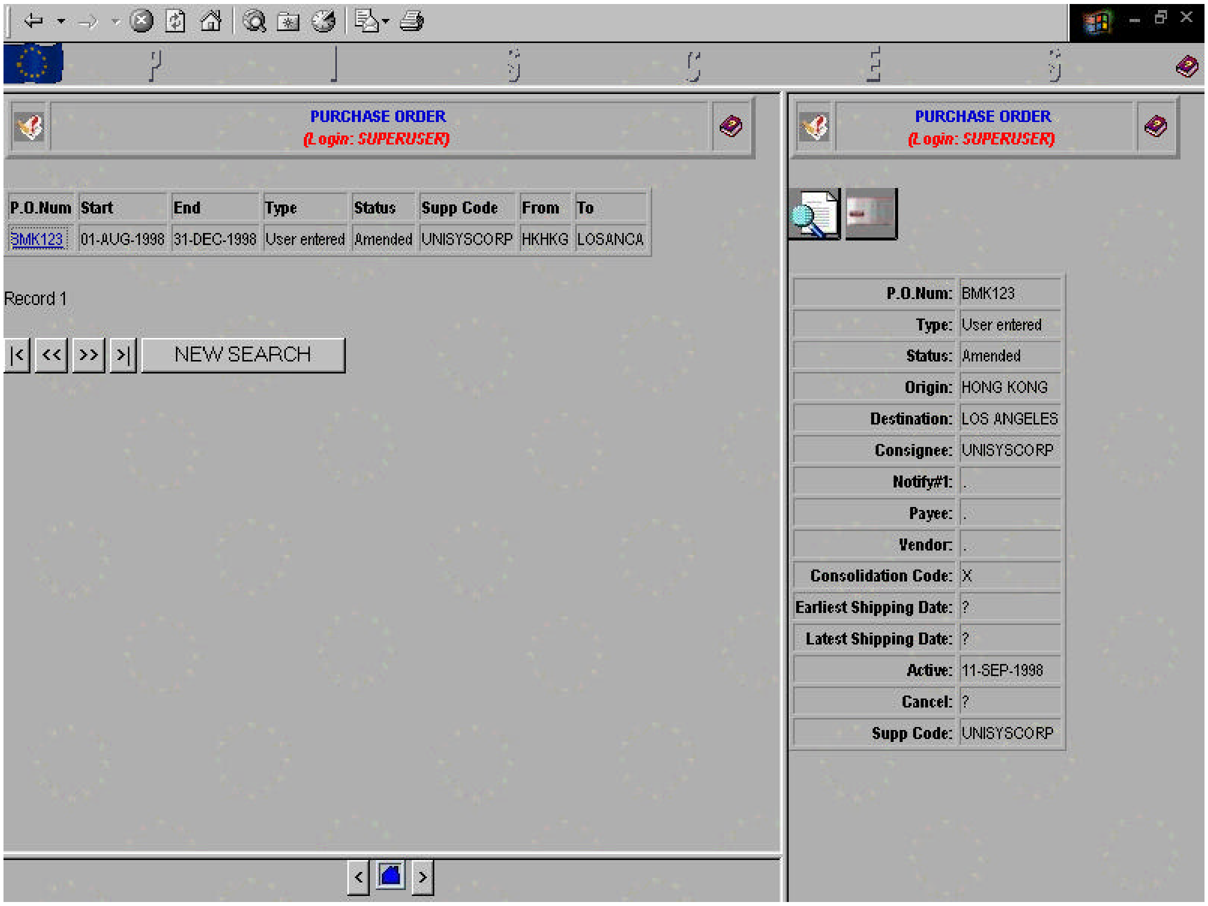
The system will allow flexible cross-linking of modules - whereby the purchase order module can be accessed from the consignment module provided the user has the approved security access to view purchase order information.

As defined in the previous section, the access to the information within the database is restricted to the specific / relevant users. The system allows distinct user logins which are password protected to see distinct menu options. These menu options will allow find / update on a restricted subset of records that will in turn will show restricted column access to the data within each record. This multi-layer security shell is always active when a user is interrogating the system, but is also highly flexible and configurable. Such configuration is a high level privileged role, which would be granted at System Administrator level.

The following is a sample of the screens in the PISCES system, for a more detailed view of the screens reference 'User Guide To The PISCES System'

4.1.14.1.1 Purchase Order Module

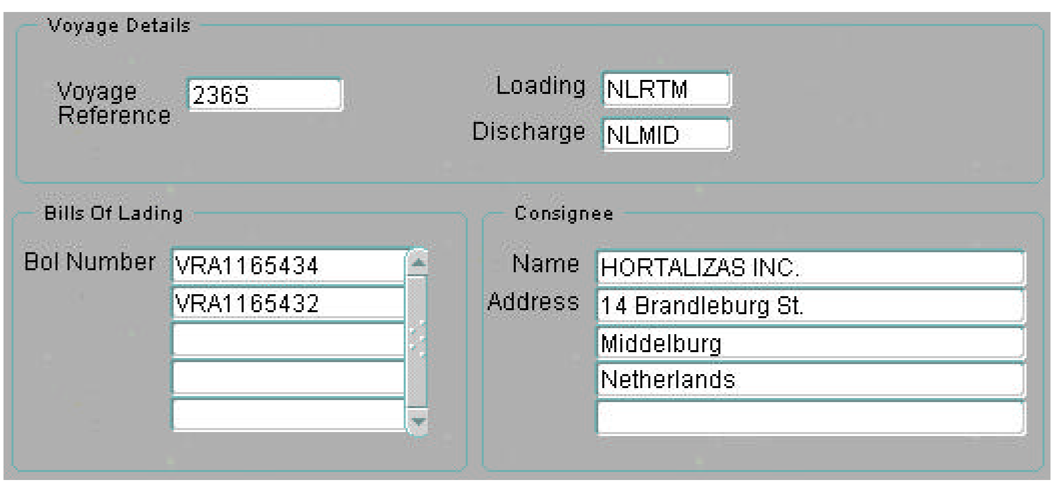
This screen allows trade transactions to be viewed by both the buying and selling parties, giving them complete insights into their shipments at all times.



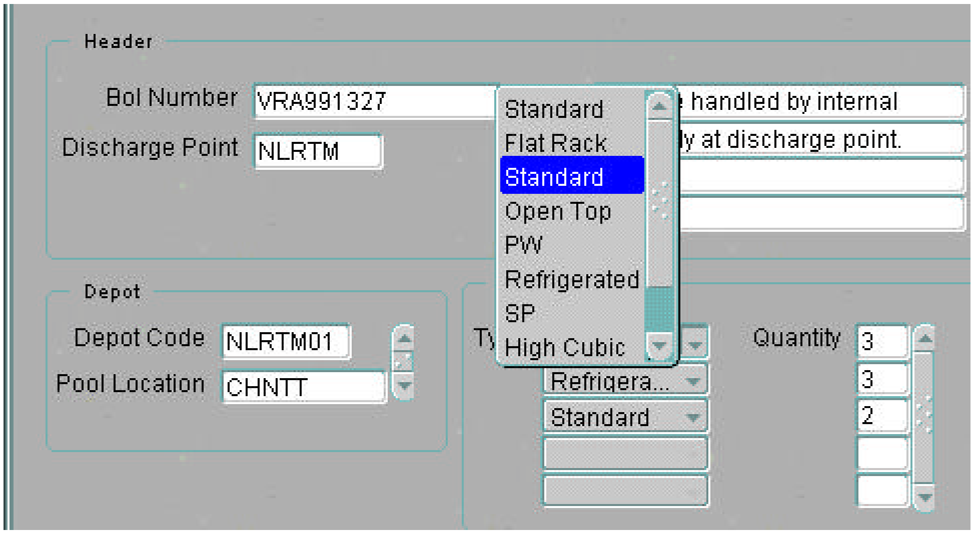
Links are provided to the Purchase order details and the Consignment module

4.1.14.1.2 View Bills Of Lading Information Module

This module is available for the input or query of records. Records store data and relationships for voyage references, voyage legs, bills of lading and their consignees.



4.1.14.1.3 Booking Containers Modules



4.1.14.1.4 Partner Profile

The partner profile set-up plays an important role in the system as to the type of information retrieved by a user. Within the Pisces system, it is possible to define a set of partners for each user login. Throughout the system, the partner code is used against each job / consignment / containers. When a user accesses the system there is validation against the address_type for the job against that for the menu option selected. Thus, for a user to access a record, the partner code must be assigned to him, the address_type for that partner code must match that for the menu option being selected.

4.1.14.2 Communication To External Systems-EDI And E-mail Facilities

In order to communicate with the Van Ommeren data systems EDI and E-mail processing facilities were selected for the PISCES system.

4.1.14.2.1 EDI (Electronic Data Interchange)

EDI (Electronic Data Interchange) is a process that involves the computer-to-computer exchange of information in a structured manner. It is sent in a manner that allows automatic processing with no manual intervention.

4.1.14.2.2 EDI Transfer File Mechanism

In order that files can be received, processed into the PISCES database, the Xfmonitor - External Interface Monitor mechanism is used. This facility allows the user to maintain and control the different types of information that are being passed in or out of the system.

The information provided here enables the External Interface Monitor (XFmonitor) to identify the file type, and process it accordingly, were :

- **Type** - the type of the interface file as denoted by the file name prefix.
- **Send / Receive** - denotes the direction of the message i.e. incoming or outgoing.
- **Method** - indicates how the message is sent / received, e.g. EDI, FTP.
- **Source** - the specification of the file to be processed.
- **Object** - the specification of the file when it has been processed.

The XFmonitor process is divided into 10 cycles, numbered 0 to 9. Cycle(s) allows the user to specify the cycles on which the XFmonitor searches for the file, so that a file which is rarely processed need not be searched for during every cycle.

4.1.14.2.3 Starting and Stopping the XF Monitor

The external interface monitor may be started or stopped by choosing the required option from the external interfaces menu.

4.1.14.3 E-Mail Processing

E-mail (Electronic mailing) processing within the PISCES system has enabled the system to inform members involved in the logistics chain who may not necessarily be accessing the PISCES system of important information relating to consignments which they need to be aware of - Alerts e.g delay of shipment.

4.1.14.4 Communications In The PISCES System

For the pilot system, it was agreed to utilise the EDI transactions already established between Van Ommeren's trading partners and to have the relevant messages routed directly to the PISCES data base, in addition to their original destination. On receipt of an EDI message the system extracts the information contained within the message and updates the relevant files with the requisite information.

Following analysis of the EDI messaging available at Van Ommeren, it was feasible to develop the PISCES system development to receive EDI messages for the following areas of the business chain :

Booking Information - this message contain the information relating to the initiation of a consignment booking.

Bill Of Lading - this message contains information relating to a consignment into the PISCES system . The type of data stored in this message is the booking reference, the number of packages, package types etc.

Consignment Movement Message – provides the PISCES system with the status of the consignment, thus reducing the number of input screens required in the PISCES system to ensure that the consignment is of the correct status and reducing user error.

4.1.14.5 Events / Alerts Processing

Events / Alert processing within the PISCES system enables messages to be generated at specified stages of a consignment in the logistics chain. These messages are system configurable; thus users of the system will only receive messages specific to themselves. These are received by the user either by e-mail or fax (this is also system configurable).

Following the analysis with Van Ommeren, the following Events and Alerts were identified as being beneficial to users in the logistics chain :-

Events

- Confirmation Of Loaded Containers Onboard
- Confirmation Of Discharged Containers
- Arrival Of The Container : Arrival Notice
- Release Of The Container

Alerts

- Container Status Out Of Sequence
- Receipt Of Hazardous Cargo
- Missing Containers To Be Shipped At Terminal
- Delay Of The Shipment
- Start Of Demurrage / 30 Days Demurrage
- Unavailability Of Containers
- Containers Which Have Not Yet Assigned For Inland Transport

Within the pilot system, we concentrated on the alert - 'Container Status Out Of Sequence' as the PISCES system was receiving information of the consignment status through the EDI processing mechanism, thus we could be ensured of receiving such data and testing the principle of triggering an alert if incorrect information was received or not entered. This was satisfactorily proven, giving us the confidence that the other Events /Alerts identified (as listed above) could be implemented quite easily.

4.1.15 Conclusions of the Database Development

The PISCES system was developed to consider qualitative and quantitative improvement within intermodal transport. We have explored a real-world example provided by Van Ommeren Agencies Rotterdam BV.

The potential benefits demonstrated from using a system such as PISCES would include:

- Reduced costs, achieved by better utilisation of resources, more optimised loads and close matching of shipments to priorities.

- Less unproductive movements, less empty running for the positioning of containers and improved “imbalance” ratios.

- Less environmental impact through more routing toward rail and barge, fewer vehicle running miles, less movements of empty units for positioning and reduced vehicle waiting and queuing times.

- Enhanced services to buyers of transport services via better planning information, enhanced track and trace and more reliable times.

4.2 The PISCES Scheduler

To overcome the problem of commercial vehicles travelling excessively, attempts have been made to use IT tools for assisting with planning and co-ordination of vehicles routes, jobs etc. However, these have been typically through the use of isolated IT tools that cannot solve large scale problems, fail to address important constraints, cannot balance partially conflicting objectives, do not react to dynamics, and, cannot interact with the user in a timely and meaningful way. Existing tools are inadequate, or too expensive. Development and maintenance of tailor-made solutions are too costly, partly because of the software bottleneck. Recent technological advances in constraint programming, global optimisation, and applications modelling have enabled us to remedy these shortcomings. The PISCES project draws upon the latest technological advances in the development of a rapidly re-configurable, generic software tool for dynamic and optimised vehicle routing. The system contains a comprehensive, software library based on constraint programming and genetic algorithm, with algorithms for the modelling and resolution of a wide range transport problem such as Pick-up and Drop of containers and vehicle routing problems, the central problem in logistic management. To overcome the software bottleneck, the system includes an application-modelling tool, with facilities for automated configuration of bespoke decision support systems. The selected optimisation approach is genetic algorithm techniques that have shown remarkable performance on computationally hard, combinatorial optimisation problems. Empirical investigation using prototype versions of the systems shows good performance, both on standard vehicle routing benchmarks from Operations Research, and real-life cases from industrial partners.

This section of the report will describe the Routing Scheduler of the PISCES system, and show how it will improve transportation logistics performance with knock-on effects on the environment and the economy.

4.2.1 Aim and Introduction to the Scheduler

The PISCES scheduling system demonstrates the benefits of capitalising internet technology and Artificial Intelligence techniques to provide an integrated system for Intermodal Transport for freight management. The aim is to support the process of planning, implementing and controlling the efficient flow and storage of containers, services and related information from point of origin to point of destination (including inbound, outbound, internal and external movements) for the purpose of conforming to customer requirements.

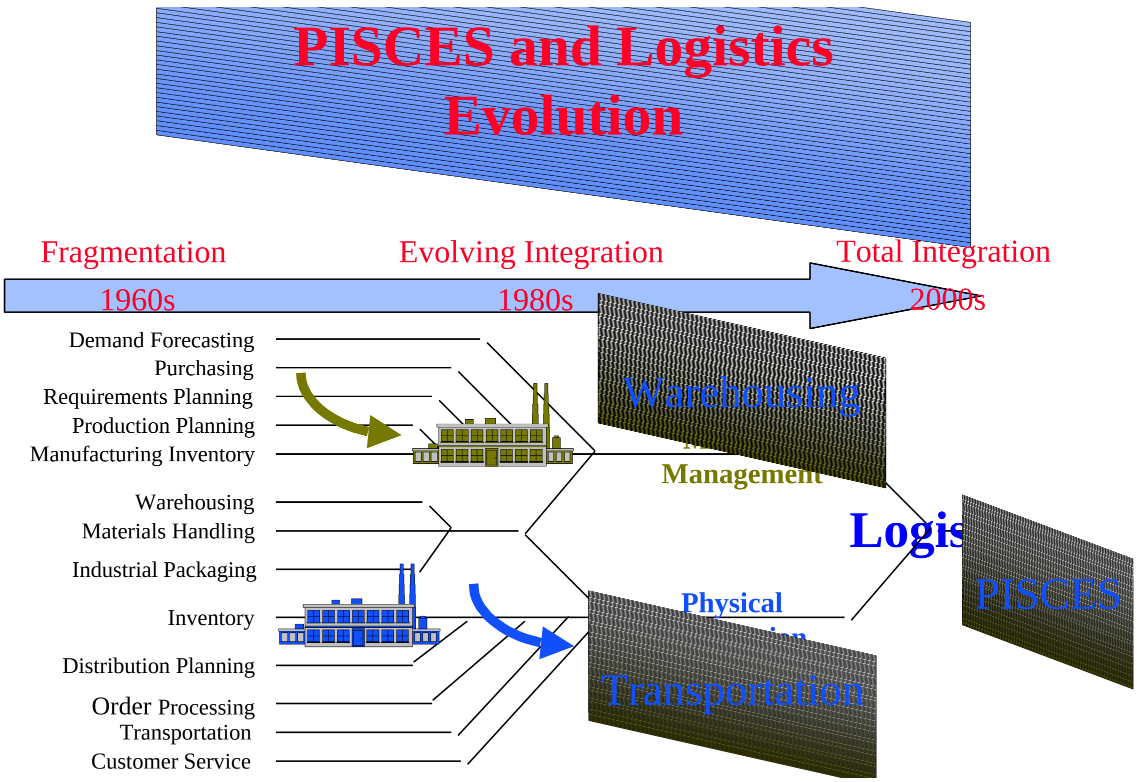


Figure 1 PISCES and Logistics Evolution

In the 60's the logistic chains supply was fragmented in several independent functions, whilst the 80's saw the beginning of an evolving integration through two main logistics function; the material management through the warehousing function, and the physical distribution through the transportation function. In the new millennium, projects such as PISCES will enable the total integration of logistic functions. Figure 1 describes the evolution of logistic integration sustained by PISCES.

The latest information technology (i.e. internet) and application of Artificial Intelligence, multi-users database system and computer security techniques enable better integration. This is based on a neutral impact strategy for the users with no significant extra burden on existing IT set-ups. Additionally it does not require publicising confidential information, and will not give rise to additional operations. It allows the complete chain to be scheduled and managed as a single process, without the need for all participants to adopt the same information system or even the same information standards.

Intermodality is another aspect of the PISCES project. It aims to demonstrate that intermodal transport can be managed more effectively when integration is possible. The project provides a discrete system architecture which supports the functionality of documentation, booking, in/out gate, traffic optimisation, warehouse management, transport management and route scheduling taking into account the different modes of transport.

With the advent of the global market, it is expected that trade will increase at a rapid pace. Transportation, which is already vital, will become even more strategic to industries. The purpose of transportation logistics is to design, organise and manage movement of commodities to meet cost requirements, environmental requirements and, customer service demands. Logistic systems must comply with regulations on traffic, laws on labour, and other constraints. Increasingly addressing, environmental problems (i.e. gas emission, noise, and waste) are becoming important. Implementing and managing a logistics system is complicated, for large corporations as well as for small and medium-sized enterprises. There is a general need for IT solutions that will assist in the design and management of the logistics chain.

The operation of a fleet of commercial vehicles, i.e., container transport or fleet management, imposes co-ordination of a number of complex tasks. For instance, in a transportation terminal scenario, the daily local distribution is the form of pickup and delivery tasks are dynamic. Each request must be assigned to a vehicle or to a batch of containers, or, rather, one of several tours that a given vehicle will perform during the day. Several types of constraints must be adhered to in this allocation process. Each transportation request will typically have a time window, i.e., a time interval within which the request must be serviced. Containers (Vehicles) may have different capacities in terms of weight, volume, length etc.. Some shipments may need special containers (vehicles), for instance due to dangerous chemicals or special temperature requirements. There will be time frames and geographical constraints on tours, including constraints on start and end points. The fleet is bound to operate within a transportation network with traffic restrictions and variations in average speed (due perhaps to congestion). There are labour restrictions that originate from legislation and trade union agreements. Moreover, the organisation strives towards a healthy economy, a high level of customer service, and to an increasing degree, a reputation for addressing inherent environmental problems in transportation in the best way possible.

In transportation, there are important co-ordination tasks at various levels in the control hierarchy. At the highest, strategic level, there is the design and reconfiguration of a transportation network. In the middle, at the tactical level, an important task is to determine the optimal transportation capacity according to demand, i.e., fleet dimensioning. At the lowest level, a typical task is the dynamic, day-to-day dispatching of transportation requests, for instance at the planning office for a distribution centre, as described above. For all these tasks, several criteria and constraints need to be taken into account. Note, that all these tasks may be regarded as optimisation problems. The field of Operations Research (OR), has addressed such problems since its birth during World War II.

Focussing on co-ordination at the lower levels, as exemplified by container transport or the dynamic fleet management does not simply rely on local decision rules. To achieve good performance at a global level, a detailed operational plan must be constructed, and maintained throughout the day when unexpected events occur. Arriving transportation requests, delays due to traffic situations, and driver illness are examples of events that may call for dynamic revision of operational transportation plans. Detailed routing plans will contain information for loading, dispatching, and sequencing of visits for each vehicle in the fleet.

The planning task may be broken down into three sub-tasks:

1. The allocation of a given transportation request, i.e., a pick-up or delivery, to a specific container/vehicle.
2. The triangulation/sequencing of visits for all containers/vehicles.
3. The use of alternative mode of transport if available and cost-justified, i.e. train/barge
4. For each pair of consecutive visits, find the cheapest path in the road network.

The overall goal is to arrive at a plan, which is:

- executable, i.e., does not violate given constraints
- optimal with respect to several global criteria, e.g., overall transportation cost, customer service, and environmental concerns.
- Use of intermodal transport as much as possible

These four sub-tasks contain a number of semi-independent, discrete choices. The theoretical number of alternative routing plans grows exponentially with the number of requests and the fleet size. Even for small fleet sizes and a moderate number of transportation requests, the planning task is highly complex. Hence, it is not surprising that human planners soon get overwhelmed, and must resort to simple, local rules for vehicle dispatching. There is a huge potential for improvement.

In OR, the planning problem described above is called the Vehicle Routing Problem (VRP). The VRP comes in many guises, depending on the structure of the transportation operations, fleet sizes, presence of capacity constraints, time constraints, geographic constraints, traffic regulations, customer preferences, and other, idiosyncratic constraints. The standard, mathematical programming approach in OR has failed to provide effective methods to solve large instances of the VRP, except for a few, well-structured cases.

In the past few years, human planning in fleet management has been assisted by the introduction of transport planning IT systems on the market. Few such systems contain functionality for the actual construction and maintenance of executable and optimised routing plans. The Dutch newsletter *Transport and Logistic* has conducted two surveys of IT solutions for transportation logistics. In 1991 it was concluded that VRP systems were not as cost-effective as humans. The November 1993 survey showed that VRP software had evolved to a point where it would give good results, but only for relatively simple, regular problems. This study also concluded that available software was still not flexible enough for solving the wide variety of VRP problems that arise in reality.

When faced with a logistics problem including a Vehicle Routing Problem (VRP), end users typically have two alternative solutions today:

1. Use an off-the-shelf software package. This is a good solution if the problem can fit in one of the predefined problems of the package. If it is not the case, it can be very difficult, if at all possible, to adapt to the final user needs.
2. Develop an ad-hoc program or buy consultancy from a software company. Although a specific problem can be solved quite well, this development results in software that can rarely be re-used and typically has a large price tag.

The technology used in current approaches shows sharp performance degradation when faced with problems involving more than 500 customers, a relatively small number for real-life applications. Recent advances in methods for solving hard, constrained optimisation problems such as the vehicle routing problems, and the general increase of computer performance, have removed technological barriers for a successful introduction of IT support in transport planning. Moreover, developments in telecommunication, geographical information technology, and EDI standardisation, as well as the emergence of electronic road maps and address registers, have paved the way for a successful introduction of sophisticated decision-support tools for vehicle routing in industry. However, a successful introduction of such systems heavily depends on the presence of necessary computer infrastructure, knowledge, and organisational support within the company.

The PISCES project uses the latest advances in constraint programming, global optimisation, and applications modelling. The development has lead to a generic, C-JAVA based tool for the development of bespoke VRP decision-support systems.

4.2.2 Transport Planning and the Vehicle Routing Problem

The definition of the Vehicle Routing Problem (VRP) is as follows: A number of sites must be visited by a number of vehicles. This should be co-ordinated, generally to balance a number of criteria, for instance, total distance travelled and delays to customers. There are variants of this problem involving capacity, temporal, technological, and geographical constraints.

Economic area	Application
Car industry	spare parts distribution
Commodity delivery	oil, natural gas, concrete
Food transportation	large retailers or small shops
Food delivery to individuals	milk (in the UK), frozen food
Retail	appliances delivery.
Health	medicines to pharmacies
Press	newspapers and magazines
Banking industry	delivery of cash to banks and ATM, collection of money from stores and malls
Public sector	domestic dustbins; public dustbins, street cleaning, winter gritting
Manufacturing	organisation of the moves of a fleet of transportation robots, optimisation of the moves of the arm of a melting robot.
Industry	supply of parts and goods among different sites
Agriculture	collection of animals, milk, cereals, delivery of animal food
Transportation industry	Pickup and delivery companies

Table 1. Applications of VRP

The VRP is an important formulation which captures problems at several logistics management control levels. Examples included in Table 1 above are the design and reconfiguration of a transport network, and the day-to-day management of pickup and delivery. Conventional Linear Programming models are highly effective in solving strategic, high-level transportation network design problems where a static view is adequate. However, in industrial environments that exhibit dynamics and non-regular constraint types, the VRP is a far more adequate formulation. The increasing emphasis on time, cost reduction, and concerns and regulations related with social and environmental factors, makes the presence of dynamics and non-regular constraints the rule rather than the exception in European industries of today.

The VRP, as depicted in Figure 2, comes in many guises. Vehicles, containers or drivers may not be interchangeable. Time windows may be associated with customers. In pickup and delivery of goods there are loading constraints and often a significant number of customers and a complex routing problem. In the case of repair or installation there are technological constraints: a given service can only be provided by certain vehicle/engineer pairings and the service time will vary. In repairman problems a good assignment may be crucial, whereas the detailed routing may not be as important.

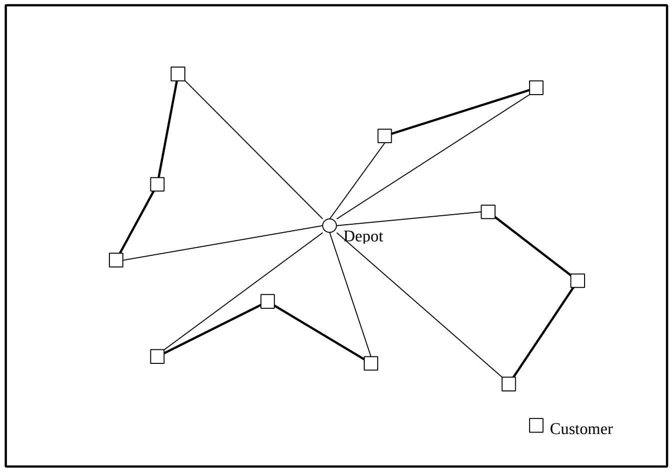


Figure 2. A Vehicle Routing Problem

In summary, VRP problems are ubiquitous. They appear at various levels, with different objectives, constraint types, and size. Improvements in logistic performance through improved capabilities VRP decision support tools will have huge economic and environmental impacts.

Besides being one of the most important problems in practical terms, the VRP (even in highly idealised formulation) is also one of the most difficult problems to solve. This is partly due to its combinatorial nature. Its simplified companion problem, the Travelling Salesperson Problem (TSP), where only one person has to visit all the customers, has been studied since the early days of Computing and OR. TSP is a so-called NP-hard problem [Whitley 1991], which means solving such problem could take exponential time in the worst case.

It is believed that one may never find a technique that will guarantee optimal solutions to large instances in reasonable time for such problems. The VRP is even more complex. One must concentrate on techniques that will produce high quality solutions in limited time. Precisely because of the complexity and generality of the VRP problem, advances have been incremental.

In addition to the theoretical combinatorial complexity issue of the VRP problem, there are also more pragmatic issues related with the development and implementation of VRP tools in industry. To provide useful support for real-life industrial problems, such tools must be based on real requirements from industry. Underlying techniques must be capable of addressing a variety of constraints, optimising multiple criteria, reacting adequately to dynamics, as well as support meaningful user interaction. A VRP decision support module must of course be integrated in the overall company IT infrastructure, which typically includes Oracle Database, GIS, Internet, EDI and other communication systems.

4.2.3 Triangulation Problem

The Pickup and Drop problem is a variation of the VRP [Savelsbergh 1995], that involves pickup and drop-off of goods at different customer locations. In this problem a set of containers are to be transported between terminals, depots, and clients. Those containers may be loaded or empty, and their transportation may be achieved by truck, rail, or barge, or by combination of these different modes of transport. The transport between the different locations is performed by the collection of containers (Pickup), delivery of containers (Drop), and positioning of containers (to keep balanced, the distribution of containers between depots and/or terminals). Within the current logistic management chain, the different transport operations are performed as follows:

- *Delivery*, containers are loaded at the terminal or at the depot in order to serve a customer, they are unloaded at the delivery point, and brought back empty to the initial location (terminal or depot).
- *Collection*, containers are brought empty to the pickup point, where they are loaded and brought back laden to the depot or the terminal.
- *Positioning*, empty containers are moved from a terminal or depot to other terminals or depots, to ensure balanced distribution of containers

In a Pickup and Drop problem, a set of routes must be constructed to satisfy transportation requests. Each requests specifies the load size, the pickup location, the delivery location, and time windows for the pickups and deliveries. Every load must be transported by one mode of transport, from its origin to its destination without transhipment at other locations. Each request has typically a time window; a priority and a mode of transport specified (road, rail, or barge). Containers may have different capacities in terms of weight, size, type or volume. Some shipments may need special containers, for instance dangerous chemicals or refrigeration requirements. There are time frames and geographical constraints on tours, including constraints on start and end points.

The problem is to minimise the number of transports and/or the total distance to service all the customers, without violating the constraints. The aim is to improve the current operations, by reducing the distance covered by empty containers. This is achieved by triangulation of containers between depots, terminals and clients. The processes are as follows:

- *Delivery*, look for an opportunity to collect new goods from a client (possibly the current one) to a depot or a terminal within the vacinity, prior to returning.
- *Pickup*, look for an opportunity to serve a delivery point close to the collection point where they are loaded and brought back laden to the depot or the terminal.
- *Positioning*, look for an opportunity to serve a delivery close to the positioning point.

The problem of Pickup and Drop with Triangulation is thus to find routes for the containers, minimising the empty running between terminals, depots, and clients, whilst respecting the constraints. To solve this problem, we use a combination of Constraint Satisfaction Problem (CSP) [Frost 1994] [Kelleher 1996] and Genetic Algorithm (GA) [Goldberg 1989] based techniques.

4.2.4 Models and Techniques for Transport Planning

This section provides an overview of state-of-the-art technologies relevant to dynamic transportation planning problems that involves the reactive routing and scheduling of a fleet of vehicles in response to dynamically changing transportation demands. Specifically we focus on a new class of complex transportation planning problems, that are referred to as the "Triangulation (Pick-up and Drop) Problem with Multiple Port and/or Depot, and Intermodality" (the Triangulation Problem in the following sections). This problem is dynamic, and while it is representative of a number of practical transportation problems, it does not appear to have been the object of prior studies. This does not means that techniques proposed for simpler routing and scheduling problems could not be brought to bear on this problem. Indeed, a number of techniques developed in the fields of vehicle routing and scheduling, including reactive techniques proposed in the constraint-directed scheduling and manufacturing scheduling literature [Kelleher et al. 1993,1994], appear quite relevant. They can be adapted and/or combined with other approaches to design effective solutions to the Triangulation problem.

4.2.4.1 A Taxonomy of Models and Techniques

This section briefly reviews major VRP models studied in the literature, introducing some basic terminology and identifying sources of complexity associated with dynamic transportation problems. It proceeds with the introduction of a simple taxonomy that is used to categorise techniques discussed in this section.

4.2.4.1.1 A Taxonomy of Models

In its canonical form, the VRP requires the design of a set of minimum cost routes originating and terminating at a central depot for a fleet of vehicles that has to service a set of clients with known demands. The problem can be easily seen to be NP-hard. For instance, the single uncapacitated vehicle version of the problem with objective of minimising total travel time reduces to a TSP, a well-known NP-hard problem [Lenstra et al. 1981].

Comprehensive surveys of VRP models and techniques include [Psaraftis 1980]. In many practical situations, assumptions made in the canonical VRP formulation are too simplistic and a variety of additional considerations need to be taken into account. The following is a brief summary of more complex modelling assumptions considered in the literature:

- **Time window**: In the VRP with Time Window (VRPTW), customers impose earliest/latest possible service time windows. Examples of similar constraints in the Triangulation Problem include a container's batch earliest ready-to-move time or a latest acceptable arrival time at port/depot/customer location. A good review of techniques for VRPTWs is provided in [Solomon et al. 1988].

- **Capacitated Vehicles**: Servicing a customer/location involves picking-up and delivering goods or people, vehicle capacities restrict the number of customers/locations or the number of demands that can be serviced by a given vehicle [Osman 1992]. For example, in the Triangulation Problem, truck/train/barge can carry particular number of containers, depending on the mode of transport and configuration of this mode of transport (truck usually one, barge and train, a large volume depending on their size).

- **Multiple Depots**: In a number of environments, not all vehicle routes originate from the same depot. Work in VRPs with Multiple Depots includes [Haouari 1990]. In the Triangulation Problem domain, container movements may originate from numerous ports/depots.

- **Constraints on Lengths and/or Duration of Vehicle Tours**: Some VRP formulation also allows for constraints on the lengths and/or duration of vehicle tours. For example, in the Triangulation Problem, such considerations may arise from constraints on the range of a given mode of transport, distance, restrictions on the drivers' duty day (the maximum numbers of hours a driver is allowed to work without an extended rest).

- **Multiple Types of Vehicles**: In many actual VRPs, not all vehicles are the same and some vehicles can only service some customer's requests. In the Triangulation Problem we have two aspects, one is linked with the type of containers, the other one the possibility of intermodal transport and the alternative/combined use of truck/train/barge.

- **Pickup and Delivery Problems**: These problems arise when a vehicle is required to pickup an entity (e.g., a passenger, a parcel or a container) at one location and then deliver it to another location. In this case, the pickup location must precede the drop-off location. Thus, Pickup and Delivery are equivalent to VRPs with the addition of precedence constraints between Pickup and Drop-off locations. Early work on the Pickup and Delivery Problem has been reported in [Wilson 1977]. Single vehicle Pickup and Delivery Problem include [Psaraftis 1980]. Techniques for the multiple vehicles Pickup and Delivery Problems, which are more relevant to PISCES, include those described in [Dumas 1989]. It should be noted that multiple vehicle Pickup and Delivery Problems does not capture the added complexity of many practical dynamic transportation problems which require matching demands with appropriate origin or destination points. In other words, there may be more than one pickup locations (e.g. when the customer will accept delivery of a commodity from any depot), or there may be more than one acceptable drop-off location to choose from (for example an empty container can be delivered to any suitable depots/ports).

- **Dynamic Models**: Static formulations assume that customer demand is known ahead time (i.e. model assuming "advance reservations") and that vehicle routes have to be optimised once and for all (e.g. School Bus problem). In contrast, in dynamic models, new customer requests are eligible for immediate consideration [Bertsimas 1993] and in general require revisions of already established routes and schedules. For example, the Triangulation Problem is an instance of Pickup and Delivery Problem that includes a mix of both advance and immediate reservations.

It should be clear that the Triangulation Problem is in many ways more complex than VRPs traditionally discussed in the literature. Although, a number of techniques proposed for simpler problems can in fact be extended, refined and combined to support realistic dynamic transportation planning and scheduling problems.

4.2.4.2.2 A Taxonomy of Techniques

As the general VRP is NP-hard, attempts at developing optimal solutions to variations of these problems have been limited to fairly simple models and problems of relatively small size [Assad 1988]. The focus is on more pragmatic approaches that have emphasised the development of heuristic procedures capable of supporting the dynamic requirements of the Triangulation Problem.

Heuristic search procedures developed for the VRP and related routing and scheduling problems can be categorised along a number of dimensions. In particular three basic dimensions can be identified:

- **Constructive versus Iterative/Repair Procedures**: Constructive techniques build solutions by incrementally instantiating decision variables (e.g. assignment of containers to transport order) until a complete feasible and satisfactory solution is obtained. In contrast, iterative procedures evolve in a (search) space of complete solutions (i.e. solutions in which all decision variables have been instantiated), using transformation operators to move from one complete solution to another until a feasible and satisfactory solution has been found or until some predetermined condition becomes satisfied (e.g. a time limit ha been reached). Simple examples of transformation operators in the Triangulation Problem could include reassigning a container from one transport order to another or modifying the itinerary of combinations of transport orders. A number of constructive and iterative procedures have been proposed in the literature to solve various routing and scheduling problems. They include simple one-pass constructive procedures which stop as soon as a first satisfactory solution has been obtained [Jaw 1986] as well as more expensive constructive procedures such as enumerative procedures described in [Smith 1992]. Iterative/repair procedures developed for routing and scheduling problems include Simulated Annealing procedures [Laarhoven 1992], Tabu search procedures [Osman 1992] and genetic algorithm procedures [Whitley 1991]. As it turns out, generative and iterative techniques are not necessarily incompatible but can sometimes be viewed as complementary: simpler constructive approaches can provide a basis for quickly producing new though potentially highly sub-optimal solutions while iterative techniques offer a way to spend additional time post-processing these solutions to improve their quality.

- **Informed versus Brute Force Search**: involves looking at the amount of analysis performed by the procedure before making a decision. At one extreme of the spectrum we have brute-force techniques such as branch-and-bound or simulated annealing, which attempt to rapidly cover wide areas of the search space but spend very little time deciding which area to explore next. At the other extreme sit more sophisticated techniques, which emphasise more selective (and hence slower) exploration of smaller areas in the search space [Smith 1995]. In between, some techniques rely on constraint propagation to quickly prune unpromising areas of the search space but spend little time differentiating between remaining alternatives [Smith 1992].

- **Adaptative versus Fixed Search Procedures**: The adaptability of a search procedure refers to its ability to gather information as it searches for a solution and exploit this information to improve search performance. This is also referred to as the ability of a procedure to 'learn' as it goes. Learning can be used in many different ways to enhance search efficiency. Tabu search, for example, [Osman 1992] relies on simple short-term and long-term memories to dynamically adapt its behaviour. Learning mechanisms have also been developed to enhance the performance of fast brute force search techniques such as Simulated Annealing [Nakakuki 1994] or to determine how to best select among a set of repair operators [Miyashita 1994].

Clearly, these three dimensions are not the only ones along which one can classify routing and scheduling techniques. In the following sections, it maybe necessary to refine this taxonomy or consider aspects of proposed solutions that are orthogonal to the dimensions identified above. Nevertheless, this simple 3-dimensional taxonomy provides a useful starting point for differentiating among the many techniques discussed in the following section.

4.2.4.1 A Selective Overview of Dynamic Rescheduling Techniques

While considerable efforts have been expended developing optimal algorithms and heuristic procedures for the static version of the VRP and its many variants, work on dynamic versions of these problems has been rather scarce. In the beginning of his 1988 survey on dynamic vehicle routing problems [Psaraftis 1988], Psaraftis noted: "*for all the explosive growth in the vehicle routing literature over the past several years…, very little has been published on dynamic variants of the vehicle routing problems. Of the 62 references cited in [Golden 1987], only three include phrases such as 'dynamic', 'real-time' or 'on-line' in their titles*".
Even though the body of research in dynamic vehicle routing problems has grown significantly since the time these observations were made, dynamic aspect still occupy a relatively small fraction of the overall research in VRP.

Requirements of dynamic problems differ from those of static ones in a number of ways (e.g. "real-time" requirements, higher criticality of near-term decisions, open-ended problems, uncertain requirements, etc.). As a result, solutions to static problems are often difficult to adapt to dynamic situations. When it come to building solutions to dynamic problems, it is important to distinguish between procedures that rely on constructive techniques and those that rely on iterative one:

- **Dynamic Rescheduling using Constructive Techniques**: the process begins with an incomplete or even empty solution and constructs the missing elements of the solution. Typically, this process proceeds through a path, which does not include infeasible solutions. When relying on constructive search procedures, reactive functionalities require technique that help determine which part of the solution to undo before invoking the constructive procedure to rebuild a new solution. This is a critical barrier to adapting constructive techniques to rescheduling tasks. Extreme variant of this approach involves freezing all decisions variables associated with those parts of the solution that have already been executed, while rebuilding a brand new solution for the remaining decision variables [Psaraftis 1980]. Rebuilding brand new solution in this fashion raises two concerns. One is that in large-scale domains, such as the Triangulation Problem, the computational requirements of such an approach could be prohibitive. By the time a new solution has been constructed, additional contingencies may have occurred, rendering the new solution obsolete. Another and more profound concern is that in situations where it is possible to build a brand new solution each time a contingency occurs, this approach may still be undesirable because it introduces too many disruptions. Often it is preferable to restrict solution revisions to small parts of the solution (e.g. to avoid creating confusion, account for difficulties in communicating new solutions in real-time or difficulties in adapting to new solutions). Below a number of approaches are discussed that have been proposed to determines which part of a routing/scheduling solution to revise, including matchup approaches [Bean 1991], conflict propagation approaches [Sadeh 1994a] and truth maintenance approaches [Kelleher 1994].

- **Dynamic Rescheduling using Iterative Repair Techniques**: Iterative repair techniques traverse a path of complete solutions, both feasible and infeasible, eliminating constraint violations and improving the quality of the solution. In theory, these techniques can directly support reactive capabilities, provided one can ensure they eventually extricate themselves from infeasible regions and converge to a feasible solution. Examples of such techniques include genetic algorithm such as the described in [Thangiah 1993], simulated annealing such as the one described in [Zweben 1991], or constraint-directed repair procedures such as the ones described in [Miyashita 1994]. Iterative improvement techniques that do not allow for infeasible solutions can still be used to re-optimise solutions. This happens when favourable contingencies occur that make the problem easier and offer opportunities for improving the quality of the existing solution (e.g. cancellation of a request, addition of a new resources, duration of a triangulation is shorter than expected, etc). In the face of contingencies that invalidate an existing solution (e.g. a transportation resource, driver/mode of transport/ containers, becoming unavailable for some period of time), iterative techniques require heuristics that can decide which part of the solution to undo, just like constructive techniques.

Instances of each approach are further discussed below.

4.2.4.1.1 Dynamic Rescheduling Using Constructive Techniques

4.2.4.1.1.1 Rebuilding New Solutions from Scratch

When relying on a constructive scheduling procedure, the most obvious approach to dynamic rescheduling involves rerunning the procedure from scratch while freezing decision variables corresponding to activities that have already been executed or can be no longer replanned (e.g. because they are already executing or because of difficulties in replanning them).

An example of a technique based on this approach is described in [Psaraftis 1980] for the single vehicle Pickup and Dropoff problem with many-to-many immediate requests. The technique is based on a dynamic programming solution to the static version of the problem. Under the dynamic scenario, the technique is generalised by (1) imposing constraints that reflect the current situation when the contingency occurs (i.e. customers already serviced and customers currently on board of the vehicle), and (2) rebuilding a new solution that satisfies these added constraints.

In complex dynamic environment such as the Triangulation Problem, such an approach, at least in its pure form, is impracticable, due to the significant solution modifications it would continuously introduce (e.g. continuous changes in transport assignments and times would make effective control impossible). However, a modification of this approach reported in [Carnegie Group 1995] appears to be promising. This *commitment-constrained* technique uses the most recent solution as a constraint on the new solution. This minimises the differences between the new solution and the preceding solution.

4.2.4.1.1.2 Insertion Techniques

The next most obvious approach to handling dynamic events such as the arrival of a new order is to incrementally insert new orders into existing triangulated routes. A similar approach can be used when requests are cancelled: simply deleting the corresponding legs from existing triangulated routes. This approach can be simpler and faster than one that rebuilds new solutions. It was first considered by Wilson et al. [Wilson 1977]. More recent work using insertion techniques is described in [Jaw 1986]. In particular, Kikuchi and Rhee describe an insertion procedure for the many-to-many pickup and drop-off problem with multiple vehicles, time windows and advance reservations [Kikuchi 1989]. Their approach builds vehicle routes one vehicle at time, using a two-step process in which an initial vehicle route is constructed and then additional requests are inserted in the vehicle route. The approach uses a tree search technique to identify the maximum number of requests that can be inserted in a given vehicle route. This amounts to a local optimisation procedure where vehicles are considered one by one and requests are optimally assigned to the vehicle under consideration. Prior to inserting a request, the procedure checks for feasibility using time window updating mechanisms of the earliest/latest start time propagation found in constraint-directed scheduling systems (e.g. [Minton 1981]). Like other insertion approaches, this technique can be generalised to handle dynamic requests. The new requests are simply inserted into existing vehicle routes, and new routes are initiated when none of the existing routes can accommodate a new request. Insertion techniques are generally quite fast, e.g. [Jaw 1986].

However, when used in dynamic contexts where new requests are incrementally inserted into existing solutions, insertion techniques are likely to produce highly sub-optimal solutions. This is because these techniques do not take advantage of re-optimisation opportunities, namely opportunities to produce better solutions by re-sequencing legs already assigned to vehicle routes or reassigning requests to different vehicles. As all constructive approaches, this problem can be alleviated, using local re-optimisation procedures to post-process the resulting solution. This can be done using local interchange and reassignment operators [Psaraftis 1986], as discussed in the following section about iterative improvement/repair technique.

4.2.4.1.1.3 Partial Revision

Insertion/deletion procedures are relatively narrow in scope. They are mainly intended to handle the arrival of new requests or the cancellation of existing ones. (it is possible that they may react to contingencies involving a loss of capacity by deleting legs that relied on the list capacity and assigning them to different vehicle /or the same vehicle but at a different time.

Short of rebuilding new solutions each time a contingency occurs, a more general approach involves rebuilding a portion of the existing solution. Clearly, reconstructing a large portion of the existing solution is likely to yield a new high quality solution (assuming a good constructive procedure), but it can also be time consuming. In the extreme case, this approach actually reduces to rebuilding an entirely new solution. Generally, when deciding how much of the current solution and which specific part(s) of the solution to be rebuilt, a number of factors need to be considered:

solution feasibility: when new conditions need to be accommodated (e.g. new requests with tight time windows or an important loss of capacity), undoing too few decisions may cause the resulting problem to be infeasible;

disruptions: as already noted, important changes in the current solution can be detrimental even if a-priori resulting in a ‘better’ solution. This is due to the added difficulty in communicating the new solution to all the staff involved, difficulties in adapting to the requirements of the new solution, etc. In general, there is a trade-off between obtaining a highly optimised solution and maintaining stability in the system.

solution quality: a large enough number of decisions should be undone to provide sufficient opportunities for re-optimisation (e.g. opportunities to come up with new sequencing decisions on a given vehicle route or different vehicle assignment).

real-time requirements: the computation time necessary to rebuild a solution should not exceed the amount of time available.

Below, different approaches are examined to determine which part of a scheduling/routing solution to revise, given a new situation.

i : The Matchup Scheduling Approach:

Bean et al. describe a “matchup” procedure to determine which part of a production schedule to rebuild, given contingencies such as machine breakdowns [Bean 1991]. They consider a manufacturing facility in which a set of jobs needs to be processed on a set of resources with some degree of processing compatibility. Different jobs have different release dates at which they become available for processing and, different due dates by which they need to be completed. Tardiness costs are incurred for finishing a job past its due date and earliness costs (e.g. inventory costs) for finishing it too early. Jobs in this model require two resources, a machine and a tool. The machines are subject to stochastic breakdown, requiring that the schedule be revised. Additionally, switching production from one job to another on a given resource requires a set-up with a duration, which may be sequence-dependent (i.e. duration may depend on the previous job processed using the same resource).

Matchup scheduling assumes that, given a long enough scheduling horizon and a small enough disruption (e.g. a small enough machine breakdown), it is possible to find a *matchup time* beyond which an optimal solution to the scheduling problem before the disruption occurred remains optimal with the disruption. However, rather than looking for an optimal solution, which is typically impractical, match-up scheduling looks for satisfactory solutions. A solution is considered satisfactory when the sum of tardiness and inventory costs of all jobs on any given machine is below some acceptable value. Given a contingency involving one or several machine breakdowns, match-up scheduling picks a matchup time and rebuilds new schedules for each machines involved up to the selected matchup time. If the resulting machine schedules are not considered satisfactory, the procedure increments the value of the match-up time by some pre-specified duration and tries again. If in the process of revising machine schedules, the match-up time comes to exceed some predetermined value, the procedure attempts to reallocate jobs across machines so as to reduce the number of set-ups and achieve a better balance between the loads of the different machines.

The procedure has the merit of being relatively fast, provided the matchup times are properly selected and the underlying constructive procedure used to rebuild new solutions is not too time consuming. Selecting too tight matchup time can potentially result in a number of iterations before the procedure finally finds a new satisfactory solution. Selecting too large a matchup time could force the procedure to revise a large portion of the solution and, hence, could also prove computationally expensive. Ideally, selection of a matchup point should account for the severity of the disruption at hand, the amount of time available to come up with a new solution, and the tightness of the existing solution or more specifically the ease/difficulty with which the new disruption could be absorbed by the existing solution. For instance, a breakdown on bottleneck resource is likely to be more difficult to absorb than one on an under-utilised resource. Unfortunately, these issues are not explicitly addressed [Bean 19991].

ii : The Conflict Propagation Approach:

This was first introduced within the Micro-Boss scheduling system [Sadeh 1994]. This approach is somewhat similar to matchup scheduling but is more selective in the way it determines which part of the solution to rebuild. It adapts to the severity of the disruption. Rather than using a common matchup time for all resources affected by a contingency, Conflict Propagation analyses the impact of the disruption on individual operations. The result typically amounts to unscheduling operations with different “matchup times” on different resources or even unscheduling multiple non-contiguous groups of operations on the same resource. The approach can also account for precedence constraints, such as those found between related operations in a manufacturing job. Specifically, given a constraint violation such as a machine breakdown, Conflict Propagation considers dependencies in the original schedule (e.g. precedence constraints between multiple operations) and propagates the conflict at hand until it is totally absorbed by slack and/or re-sequencing opportunities present in the schedule. Conflict Propagation unschedules all the operations it traverses, then rebuilds a new solution for these operations. In the Micro-Boss system, a micro-opportunistic search procedure is used to rebuild the new solution. In general, any constructive procedure could possibly be used, though the quality of the resulting solution will depend on how good a procedure is used. Conflict Propagation is guaranteed to always unschedule enough operations to restore integrity in the solution, though it will sometimes unschedule more than the strict minimum. As it is more selective than match-up scheduling, Conflict Propagation is expected to generally reschedule fewer operations and, does not need multiple trials to identify the set of operations to be rescheduled. The approach can also be complemented with heuristics that selectively unschedule additional reservations to provide more flexibility to the constructive procedure used to rebuild a new solution. Hence, it allows for the construction of a higher quality solutions. See [Sadeh 1993] for examples of such heuristics in the factory scheduling domain.

iii : Truth/Reason Maintenance Approach:

Truth/Reason Maintenance Systems (TMS/RMS) [Doyle 1979] and their breadth-first counterpart, Assumption-based Truth Maintenance Systems (ATMS) [De Kleer 1986], provide yet a alternative approach to determining which part of the solution to modify. The idea here is that during the construction of a solution, one should record the impact of each decision on the domain of possible values of other decision variables. If later contingency invalidates assignments in the current solution, the information can be used to help determine the minimum set of earlier decisions that needs to be undone to restore integrity in the solution. For example, in the Triangulation Problem, one could record the impact of the assignment of new transport orders to a transport network on the time window for pickup and delivery for a particular transport order already assigned to that leg. Say $transport\_order_1$ (i.e. as new transport orders are assigned to the leg, we would record for each one of them whether they caused the time windows for pickup and delivery of $transport\_order_1$ to shrink and, if so, which specific time interval became unavailable). If the transport is delayed to the point that $transport\_order_1$ may no longer be delivered on time to its destination, the recorded information could be used to help to identify transport orders within the same transport network, which if reassigned to other transport networks would allow $transport\_order_1$ to be delivered on time. More complex dependencies between transport networks could possibly be recorded as well. Clearly, recording dependencies and updating them over time as the solution evolves can become rather complex and time consuming (i.e. exponential worst case complexity).

Early work using truth maintenance in reactive scheduling systems include Elleby et al. [Elleby 1988]. Prosser reported using a TMS approach to keep track of dependencies in dynamic single-resource scheduling problems found in the context of a larger distributed scheduling system called DAS [Burke 1989]. Prosser’s approach combines truth maintenance with a simple shallow learning mechanism that can dynamically identify more complex dependencies as conflict arise and help determine which scheduling decisions to undo. More recently, Kelleher et al. have developed a Lazy/Focused RMS approach to tackle factory scheduling problems at Pirelli and airline scheduling problems at Iberia airlines [Kelleher 1994]. In Kelleher’s lazy RMS approach, bookkeeping activities are performed on demand to answer specific queries. The author reports two-order of magnitude speedups over standard ATMS bookkeeping mechanisms.

It appears that, the application of truth maintenance concepts to dynamic scheduling problems has only considered the problem of building feasible solutions.

4.2.4.1.2 Dynamic Rescheduling Using Iterative/Repair Techniques

In contrast to dynamic rescheduling approaches relying on constructive techniques, iterative/repair approaches navigate in a search space of complete, though possibly infeasible solutions. These procedures move from one complete solution to another, while trying to eliminate constraint violations and improve the overall quality of the solution. Several approaches to dynamic rescheduling using iterative/repair techniques can be identified.

4.2.4.1.2.1 Interchange Approaches

Over the years, interchange procedures have been successfully used to solve a number of routing and scheduling problems [Psaraftis 1983]. These procedures are based on the concept of “k-interchange” introduced by Lin and Kernighan in the context of the Travelling Salesman Problem [Whitley 1991]. In general, interchange procedures can be used to re-sequence vehicle trips or reassign trips between multiple vehicle (e.g. lambda-interchange [Osman 1991]). In its simplest form, an interchange procedure iteratively considers possible interchanges in the neighbourhood of the current solution. If a given interchange improves the quality of the solution, it is performed and a new solution is obtained. The procedure can be applied until the solution can no longer be improved (local optimum). Alternatively, it can be run for some pre-determined amount of time. Interchange procedures can be used to post-process solutions obtained using other techniques, either dealing with static or dynamic problems. For instance, a simple insertion procedure can be used to handle incoming requests and the resulting solution can be further optimised, using an interchange procedure. In their simplest form, interchange procedures are only allowed to move from one feasible solution to another. An example of such a procedure is the one developed by Psaraftis for constrained routing problems [Psaraftis 1983]. Sometimes, by allowing the procedure to wander into infeasible regions of the search space, it is possible to eventually reach better solutions. While attractive, this approach is far from straightforward. It requires ensuring that the procedure eventually extricates itself from infeasible regions and converges toward a feasible solution. One approach here involves introducing cost penalties in the objective function to account for constraint violations in the solution. When properly adjusted, these cost penalties can help find feasible solutions, though some are more difficult to remove than others. As a result, interchange procedures may allow for some constraint violations that are easy to get rid of but not others.

Interchange procedures that allow for infeasible solutions can be used to directly support dynamic reactive capabilities. Contingencies causing one or more constraints to be violated introduce cost penalties in the objective function. The iterative procedure then works on trying to get rid of these penalties, moving the solution back into feasible region.

As noted earlier, in their simplest form, interchange procedures generally get stuck in local optima. This can be annoying if the local optimum is infeasible. One way of alleviating this problem is to sometimes allow the procedure to transition to neighbouring solutions that are not as good as the current one in the hope of eventually reaching better solutions. A number of techniques have been developed to do just that. They include Simulated Annealing procedure [Zweben 1991] and adaptive enhancement of simulated Annealing procedures relying on speedup learning mechanisms [Sadeh 1995], Genetic Algorithm [Thangiah 1991], as well as Tabu search procedures [Osman 1992].

4.2.4.1.2.2 Constraint-Directed Repair

Interchange procedures typically rely on brute force search. They attempt to find better solutions and/or eliminate constraints' violations by rapidly covering large areas of the search space, while spending little time deciding which interchange to try next. An alternative approach is to spend more time analysing the problem at hand and deciding how to modify/repair the current solution. This "informed" approach to solution repair, which is more selective in deciding what to do next and tends to cover smaller areas of the search space has been emphasised in constrained-directed repair work, including work on OPIS factory scheduling system [Smith 1994]. Work is this area also includes that of Miyashita et al. in the context of Cabins [Myashita 1993], work by Minton et al. on the Min-Conflict repair heuristic [Minton 1990] and work by Hildum in the context of the DSS system.

In the OPIS scheduling system, a number of metrics are used to analyse flexibility of the solution and opportunities for improvement. Given one or more constraint violations such as those caused by temporary loss of capacity on a resource, the system evaluates these metrics in the vicinity of the conflict to select among a set of alternative repair operators (e.g. whether to right shift the current solution to absorb the conflict at hand, or reassign conflicting operations to other equivalent resources, or re-sequence operations involved in the current conflict, etc… or some combination of the above). Example of useful metrics include resource utilisation in the vicinity of the conflict, fragmentation of reservation on different resources, amount of downstream and possibly upstream slack within different jobs affected by the conflict at hand, variance in these metrics, etc. Based on these metrics, an initial repair operator is selected. Application of the operator may introduce new conflicts, which in turn need to be repaired. The process goes on until all conflicts have been eliminated and a satisfactory solution has been obtained. An agenda-based mechanism is used to keep track of new conflicts as they arise and to help the system sequence the application of multiple repair operators. As with other repair procedures, special care must be taken to ensure that the procedure eventually converges to a feasible solution. A somewhat similar approach is also described in the context of the DSS system, though the author places a higher emphasis on search flexibility, allowing the repair procedure to continuously revise its search strategy in a way reminiscent of the micro-opportunistic search procedures described in [Sadeh 1993].

Miyashita and Sycara have reported using Case-Based Reasoning (CBR) to train a scheduling system to recognise situations where one repair operator is preferable to others [Myashita 1994]. This is used in their Cabins system. Cabins computes metrics similar to those found in OPIS and relies on earlier experience gathered in the form of cases to decide which repair operator to apply next. Given a new domain, this system would have to be retrained, a process, which could possibly be tedious and time consuming.

Relatively little work has been done on the class of problems that involve routing and scheduling of multiple vehicles responding to dynamically changing transport orders. However, a number of techniques developed for related factory scheduling problems (e.g. match-up scheduling) and/or developed in Artificial Intelligence (e.g. conflict propagation, constraint-directed repair) can also be brought to bear on this challenging problem.

Broadly, two main approach can be distinguished which react to contingencies such as the arrival of a new transport order (or its cancellation), a loss in available capacity, a variation in the expected duration of a particular activity (e.g. Transport, container maintenance / repair/ cleaning/tuning). One approach – partial revision – involves identifying ahead of time a group of decision variables affected by the contingency and relying on a constructive technique to rebuild a solution for these decision variables. Examples of such techniques include matchup scheduling, conflict propagation and truth maintenance. A second approach – iterative/repair – relies on technique where complete, though possibly infeasible solutions, is iteratively manipulated using (local) transformation operators until all constraints violations have been removed and a satisfactory solution has been obtained. Examples of this approach include interchange technique and other brute force variations of these techniques such as Simulated Annealing, Genetic Algorithm or Tabu Search, as well-as more knowledge-intensive approaches such as the one implemented in constraint-directed repair techniques.

These approaches are not incompatible. In fact, they can be viewed as complementary. For instance, a solution obtained by using constructive techniques for partial revision can be further optimised using an iterative repair procedure. Similarly, a solution invalidated by dynamic disruptions can be first repaired partially using simple repair operators and then further improved by reconstructing a significant part of the solution.

4.2.5 Approaches Used for Triangulation

The Techniques used to solve the Triangulation Problem are based on Constraint Satisfaction and Genetic Algorithm.

4.2.5.1 Constraints Satisfaction Problems

Constraint Satisfaction Problems (CSP) have been a subject of research in Artificial Intelligence (AI) for many years. The pioneering works on networks of constraints were motivated mainly by problems arising in the field of picture processing. AI research, concentrated on difficult combinatorial problems, which date back to 60's and 70's and it has contributed to considerable progress in constraint-based reasoning. Many powerful algorithms were designed that became a basis of current constraint satisfaction algorithms.

A Constraint Satisfaction Problem (CSP) consist of:

a set of variables X={x1,...,xn},

for each variable xi, a finite set Di of possible values (its domain),

and a set of constraints restricting the values that the variables can simultaneously take.

Note that values need not be a set of consecutive integers (although often they are), they need not even be numeric.

A solution to a CSP is an assignment of a value from its domain to every variable, in such a way that every constraint is satisfied. We may want to find:

just one solution, with no preference as to which one,

all solutions,

an optimal, or at least a good solution, given some objective function defined in terms of some or all of the variables.

Solutions to CSPs can be found by searching systematically through the possible assignments of values to variables. Search methods divide into two broad classes, those that traverse the space of partial solutions (or partial value assignments), and those that explore the space of complete value assignments (to all variables) stochastically.

The reasons for choosing to represent and solve a problem as a CSP rather than, say as a mathematical programming problem are twofold.

- Firstly, the representation as a CSP is often much closer to the original problem: the variables of the CSP directly correspond to problem entities, and the constraints can be expressed without having to be translated into linear inequalities. This makes the formulation simpler, the solution easier to understand, and the choice of good heuristics to guide the solution strategy more straightforward.

- Secondly, although CSP algorithms are essentially very simple, they can sometimes find solution more quickly than if integer programming methods are used.

This section provides basic grounding in constraint satisfaction problems and some of the algorithms used to solve them. In general, the tasks posed in the constraint satisfaction problem paradigm are computationally intractable (NP-hard).

4.2.5.1.1Binarization of Constraints

A constraint can affect any number of variables form 1 to n (n is the number of variables in the problem). It is useful to distinguish two particular cases: unary and binary constraints. Since unary constraints are dealt with by pre-processing the domains of the affected variables, they can be ignored thereafter. If all the constraints of a CSP are binary, the variables and constraints can be represented in a constraint graph and the constraint satisfaction algorithm can exploit the graph search techniques. This is interesting because any constraint of higher arity can be expressed in terms of binary constraints. Hence, binary CSPs are representative of all CSPs.

4.2.5.1.2 Systematic Search Algorithms

A CSP can be solved using generate-and-test paradigm (GT) that systematically generates each possible value assignment and then it tests to see if it satisfies all the constraints. A more efficient method uses the backtracking paradigm (BT) that is the most common algorithm for performing systematic search. Backtracking incrementally attempts to extend a partial solution toward a complete solution, by repeatedly choosing a value for another variable.

Most algorithms for solving CSPs search systematically through the possible assignments of values to variables. Such algorithms are guaranteed to find a solution, if one exists, or to prove that the problem is insoluble. The disadvantage of these algorithms is that they take a very long time to do so.

Generate and Test (GT)

Generate-and-test method originates from the mathematical approach to solving combinatorial problems. First, the GT algorithm guesses the solution and, then, it tests whether this solution is correct, i.e., whether the solution satisfies the original constraints. In this paradigm, each possible combination of the variable assignments is systematically generated and tested to see if it satisfies all the constraints. The first combination that satisfies all the constraints is the solution. The number of combinations considered by this method is the size of the Cartesian product of all the variable domains.

Algorithm GT:

```
gt(Variables,Constraints,Solution):-
   generate(Variables,Solution),
   test(Constraints,Solution).

generate([V::D|RemainingVariables],[V-X|PartialSolution]):-
   select_value(X,D),
   generate(RemainingVariables,PartialSolution).
generate([],[]).

test([C|RemainingConstraints],Solution):-
   test_constraint(C,Solution),
   test(RemainingConstraints,Solution).
test([],_).
```

V::D denotes variable V and its domain D

Disadvantages: The generate-and-test approach is not very efficient because it generates many wrong assignments of values to variables which are rejected in the testing phase. In addition, the generator leaves out the conflicting instantiations and it generates other assignments independently of the conflict. Visibly, one can get far better efficiency, if the validity of the constraint is tested as soon as its respective variables are instantiated. In fact, this method is used by the backtracking approach.

Backtracking (BT)

The most common algorithm for performing systematic search is backtracking. Backtracking incrementally attempts to extend a partial solution that specifies consistent values for some of the variables, toward a complete solution, by repeatedly choosing a value for another variable consistent with the values in the current partial solution.
Backtracking can be seen as a merge of generate and test phases from the GT approach. In the BT method, variables are instantiated sequentially and as soon as all the variables relevant to a constraint are instantiated, the validity of the constraint is checked. If a partial solution violates any of the constraints, backtracking is performed to the most recently instantiated variable that still has alternatives available. Clearly, whenever a partial instantiation violates a constraint, backtracking is able to eliminate a subspace from the Cartesian product of all variable domains. Consequently, backtracking is strictly better than generate-and-test, however, its running complexity for most nontrivial problems is still exponential.

Algorithm BT:

```
bt([V::D|RemainingVariables],Constraints,PartialSolution,Solution):-
    select_value(X,D),
    NewPartialSolution=[V-X|PartialSolution],
    test(Constraints,NewPartialSolution,RemainingConstraints),
    bt(RemainingVariables,RemainingConstraints,NewPartialSolution,Solution).
bt([],[],Solution,Solution).

test([C|RemainingConstraints],PartialSolution,NonTestedConstraints):-
    (can_be_tested(C,PartialSolution)
      -> test_constraint(C,PartialSolution),NonTestedConstraints=Constraints
      ; NonTestedConstraints=[C|Constraints]),
    test(RemainingConstraints,PartialSolution,Constraints).
test([],_,[]).

can_be_tested(Constraint,PartialSolution):-
    variables(Constraint,Variables),
    are_instantiated(Variables,PartialSolution).
```

Disadvantages: There are three major drawbacks of the standard backtracking scheme. One is thrashing, i.e., repeated failure due to the same reason. Thrashing occurs because the standard backtracking algorithm does not identify the real reason of the conflict, i.e., the conflicting variables. Therefore, search in different parts of the space keeps failing for the same reason. Thrashing can be avoided by intelligent backtracking, i.e., by a scheme on which backtracking is done directly to the variable that caused the failure.
A further drawback of backtracking is having to perform redundant work. Even if the conflicting values of variables is identified during the intelligent backtracking, they are not remembered for immediate detection of the same conflict in a subsequent computation. There is a backtracking based method that eliminates both of the above drawbacks of backtracking. This method is traditionally called dependency-directed backtracking and is used in truth maintenance systems. It should be noted that using advanced techniques adds other expenses to the algorithm that has to be balanced with the overall advantage of using them.
Finally, the basic backtracking algorithm still detects the conflict too late as it is not able to detect the conflict before the conflict really occurs, i.e., after assigning the values to all

the variables of the conflicting constraint. This drawback can be avoided by applying consistency techniques to forward check the possible conflicts.

4.2.5.1.3 Consistency Techniques

The late detection of inconsistency is the disadvantage of GT and BT paradigms. Therefore various consistency techniques for constraint graphs were introduced to prune the search space. The consistency-enforcing algorithm makes any partial solution of a small sub-network extensible to some surrounding network. Thus, the inconsistency is detected as soon as possible. The consistency techniques range from simple node-consistency and the very popular arc-consistency to full, but expensive path consistency.

4.2.5.1.4 Constraint Propagation

By integrating systematic search algorithms with consistency techniques, it is possible to get more efficient constraint satisfaction algorithms. Improvements of backtracking algorithm have focused on two phases of the algorithm: moving forward (forward checking and look-ahead schemes) and backtracking (look-back schemes).

The previous sections presented two rather different schemes for solving the CSP: backtracking and consistency techniques. A third possible scheme is to embed a consistency algorithm inside a backtracking algorithm as follows.

As a skeleton we use the simple backtracking algorithm. This incrementally instantiates variables and extends a partial solution that specifies consistent values for some of the variables, toward a complete solution, by repeatedly choosing a value for another variable. After assigning a value to the variable, some consistency technique is applied to the constraint graph. Depending on the degree of consistency technique we get various constraint satisfaction algorithms.

Backtracking

Even simple backtracking (BT) performs some kind of consistency technique and it can be seen as a combination of pure generate & test and a fraction of arc consistency. The BT algorithm tests arc consistency among already instantiated variables, i.e., the algorithm checks the validity of constraints considering the partial instantiation.

As the domains of instantiated variables contains just one value, it is possible to check only those constraints/arcs containing the last instantiated variable. If any domain is reduced then the corresponding constraint is not consistent and the algorithm backtracks to a new instantiation.

The following procedure AC3-BT is called each time a new value is assigned to some variable Vcv (cv is the consecutive number of the variable in the order of instantiating variables)

Algorithm AC-3 for Backtracking

```
procedure AC3-BT(cv)
  Q <- {(Vi,Vcv) in arcs(G),i<cv};
  consistent <- true;
  while not Q empty & consistent
```

```
    select and delete any arc (Vk,Vm) from Q;
    consistent <- REVISE(Vk,Vm)
  endwhile
  return consistent
end AC3-BT
```

The BT algorithm detects the inconsistency as soon as it appears and, therefore, it is far more efficient than the simple generate & test approach. But it has still to perform too much search.

The BT algorithm can be easily extended to backtrack to the conflicting variable and, thus, to incorporate some form of look-back scheme or intelligent backtracking. Nevertheless, this adds some additional expenses to the algorithm and it seems that preventing possible future conflicts is more reasonable than recovering from them.

Forward Checking

Forward checking is the easiest way to prevent future conflicts. Instead of performing arc consistency to the instantiated variables, it performs restricted form of arc consistency to the not yet instantiated variables. Restricted arc consistency must be considered because forward checking examines only the constraints between the current variable and the future variables. When a value is assigned to the current variable, any value in the domain of a "future" variable which conflicts with this assignment is (temporarily) removed from the domain. The advantage of this is that if the domain of a future variable becomes empty, it is known immediately that the current partial solution is inconsistent. Forward checking therefore allows branches of the search tree that will lead to failure to be pruned earlier than with simple backtracking. Note, that whenever a new variable is considered, all its remaining values are guaranteed to be consistent with the past variables, so the checking an assignment against the past assignments is no longer necessary.

```
Algorithm AC-3 for Forward Checking
  procedure AC3-FC(cv)
    Q <- {(Vi,Vcv) in arcs(G),i>cv};
    consistent <- true;
    while not Q empty & consistent
      select and delete any arc (Vk,Vm) from Q;
      if REVISE(Vk,Vm) then
        consistent <- not Dk empty
      endif
    endwhile
```

```
  return consistent
 end AC3-FC
```

Forward checking detects the inconsistency earlier than simple backtracking and, thus it allows branches of the search tree that will lead to failure to be pruned earlier than with simple backtracking. This reduces the search tree and (hopefully) the overall amount of work done. But it should be noted that forward checking does more work when each assignment is added to the current partial solution.

Forward checking is almost always a much better choice than simple backtracking.

4.2.5.1.5 Variable and Value Ordering

The efficiency of search algorithms which incrementally attempts to extend a partial solution depends considerably on the order in which variables are considered for instantiations. Having selected the next variable to assign a value to, a search algorithm has to select a value to assign. Again, this ordering effects the efficiency of the algorithm. Various heuristics exits for dynamic or static ordering of values and variables.

4.2.5.1.6 Reducing Search

The problem of most systematic search algorithms based on backtracking is the occurrence of many "backtracks" to alternative choice, which degrades the efficiency of the system. In some special cases, it is possible to completely eliminate the need for backtracking. Also, algorithms exist which reduce the backtracking by choosing the special variable ordering.

4.2.5.1.7 Heuristics and Stochastic Algorithms

In the last few years, greedy local search strategies have become popular. These algorithms incrementally alter inconsistent value assignments to all the variables. They use a "repair" or "hill climbing" metaphor to move towards more and more complete solutions. To avoid getting stuck at "local maxima" they are equipped with various heuristics for randomising the search. Their stochastic nature generally voids the guarantee of "completeness" provided by the systematic search methods.
The local search methodology uses the following terms:

state (configuration): one possible assignment of all variables; the number of states is equal to the product of domains' sizes

evaluation value: the number of constraint violations of the state (sometimes weighted) neighbour: the state which is obtained from the current state by changing one variable value

local-minimum: the state that is not a solution and the evaluation values of all of its neighbours are larger than or equal to the evaluation value of this state

strict local-minimum: the state that is not a solution and the evaluation values of all of its neighbours are larger than the evaluation value of this state

non-strict local-minimum: the state that is a local-minimum but not a strict local-minimum.

Hill-Climbing

Hill-climbing is probably the most known algorithm of local search. The idea of hill-climbing is:

1. start at randomly generated state
2. move to the neighbour with the best evaluation value
3. if a strict local-minimum is reached then restart at other randomly generated state.

This procedure repeats till the solution is found. In the algorithm, that we present here, the parameter Max_Flips is used to limit the maximal number of moves between restarts which helps to leave non-strict local-minimum.

```
Algorithm Hill-Climbing
  procedure hill-climbing(Max_Flips)
    restart: s <- random valuation of variables;
    for j:=1 to Max_Flips do
      if eval(s)=0 then return s endif;
      if s is a strict local minimum then
          goto restart
      else
        s <- neighborhood with smallest evaluation value
      endif
    endfor
    goto restart
  end hill-climbing
```

Note, that the hill-climbing algorithm has to explore all neighbours of the current state before choosing the move. This can be time-consuming.

Tabu-Search

Tabu search (TS) is another method to avoid cycling and getting trapped in local minimum. It is based on the notion of tabu list, that is a special short term memory that maintains a selective history, composed of previously encountered configurations or more generally pertinent attributes of such configurations (we use a couple <Variable, Value> as the characterisation of the configuration). A simple TS strategy consist in preventing configurations of tabu list from being recognised for the next k iterations (k, called tabu tenue, is the size of tabu list). Such a strategy prevents Tabu from being trapped in short term cycling and allows the search process to go beyond local optima.

Tabu restrictions may be overridden under certain conditions, called aspiration criteria. Aspiration criteria define rules that govern whether next configuration is considered as a possible move even if it is tabu. One widely used aspiration criterion consists of removing a tabu classification from a move when the move leads to a solution better than that obtained so far.

```
Algorithm Tabu-Search
  procedure tabu-search(Max_Iter)
    s <- random valuation of variables;
```

```
nb_iter <- 0;
initialize randomly the tabu list;
while eval(s)>0 & nb_iter<Max_Iter do
  choose a move <V,v'> with the best performance among the non-tabu moves and the moves satisfying the aspiration criteria;
  introduce <V,v> in the tabu list, where v is the current values of V;
  remove the oldest move from the tabu list;
  assign v' to V;
  nb_iter <- nb_iter+1;
endwhile
return s
end tabu-search
```

Again, the performance of Tabu Search is greatly influenced by the size of tabu list tl. Preliminary studies determined the following feasible range of parameter values 10 <= tl <= 35 [].

Local-Search

All above algorithms are based on common idea known under the notion local search. In local search, an initial configuration (valuation of variables) is generated and the algorithm moves from the current configuration to a neighbourhood configuration until a solution (decision problems) or a good solution (optimisation problems) has been found or the resources available are exhausted. This idea is expressed in the following general local search algorithm that enables implementation of many particular local search algorithms via definitions of specific procedures.

```
Algorithm Local-Search
        procedure local-search(Max_Moves,Max_Iter)
          s <- random valuation of variables;
          for i:=1 to Max_Tries while Gcondition do
            for j:=1 to Max_Iter while Lcondition do
              if eval(s)=0 then
                return s
              endif;
              select n in neighborhood(s);
              if acceptable(n) then
                s <- n
              end if
            endfor
            s <- restartState(s);
          endfor
          return s
        end local-search
```

4.2.5.2 Genetic Algorithm (GA)

Inspired by the analogy of genetic inheritance and evolutionary theory in biology, Gas are special stochastic, generate-and test search methods [Goldberg 1989]. The main inspirations come from the assumption that the nature is very effective in selecting the best species, species adapt to the environment, and pass their features to their offspring via *genes*. The idea is to learn from nature, and develop an "environment" which, like nature, allows near-optimal or optimal solutions to be "*evolved*"? Furthermore, it may be possible for programs to be "*cultivated*", instead of being written. Additionally, such ideas could be developed into robust, efficient and effective tools for problem solving. In 1975, John Holland proposed that at least for some problems, this is possible. Such ideas have been accepted with enthusiasm recently.

The goal is to encode solution candidates as string of building blocks. A string is analogous to a *chromosome* and a building block is analogous to a *gene* in molecular biology. In nature, each gene may take one of several possible values, called *alleles*. In Gas, binary values (0's and 1's) are the most commonly used, though sometimes multiple values are used. Each string is evaluated and assigned a numerical value which is called *fitness* of this string. For a GA to work, the fitness of a string must be dependent solely on the values that its building blocks take. In optimization problem, the fitness is normally the function to be optimized.

The basic algorithm works as follows. A set of strings, which is called *population*, is maintained. The population evolves by allowing pairs of members to combine and transform in order to generate new offspring. This is called *mating* and *reproduction*. The fitter a string is, the more chance it is given to mate and reproduce. Therefore, patterns of building blocks that appear in fitter strings will get better chance of being retained. Patterns of building blocks that appear in weaker string are in greater danger of being eliminated.

In order to understand the algorithms, it is important to examine possible way of combining strings. In nature, chromosomes exchange part of their genes during reproduction. This is mimicked by a simple artificial combination operation, which is called *crossover*: given a pair of parents strings, an arbitrary cutoff point is picked. Then the two parents exchange their building blocks at the cutoff point.

Some genes mutate occasionally. This is represented by changing the values of building blocks occasionally in Gas. This is called *mutation*: a random block position is picked, and a random value is assigned to it.

The basic control strategy of Gas is as follows: To start, an initial population of strings is generated (assuming members of this population are being generated randomly). This population is manipulated in as many iterations as necessary to generate acceptable solutions for the problem or until resources, such as time, have run out.

At each iteration, a *mating pool* is picked from the current population. Members of the current population are picked weighted randomly: the fitter a member, the more chance it is given to enter the mating pool. One may allow the same member of the current population to be picked repeatedly, thus allowing more than one copy of it in the mating pool. Then a new population is generated from the mating pool.

To generate the new population, a pair of strings will be selected from the mating pool at a time to form the *parents*. Parents are crossed over using the method described above to generate new offspring. Mutation is allowed to occur occasionally to the offspring so as to allow new patterns of building block to appear. This process is repeated until the new population contains as many members as the original one.
The different steps are then:

(1) **initialization –** to generate the initial population;
(2) **reproduction –** to create the mating pool from the current population;
(3) **parents selection –** to pick parents from the mating pool;
(4) **crossover –** to generate offspring from parents;
(5) **mutation –** to change values of the building blocks in the offspring.

This process is expected to converge to a population including highly fit individuals, however, nothing prevents the possibility of getting stuck in local optimum. This kind of (unintended) termination is called *premature convergence*.

In our application within PISCES probabilistic mechanisms essential in classical Gas has been combined with heuristics, common in traditional CSP solving methods. In such an approach the uniform random and heuristics-based components could counterbalance each other deficiencies. Namely, the application of heuristics may improve the performance of the blind random mechanism and the random component may help to find new search paths if the heuristic is misleading.

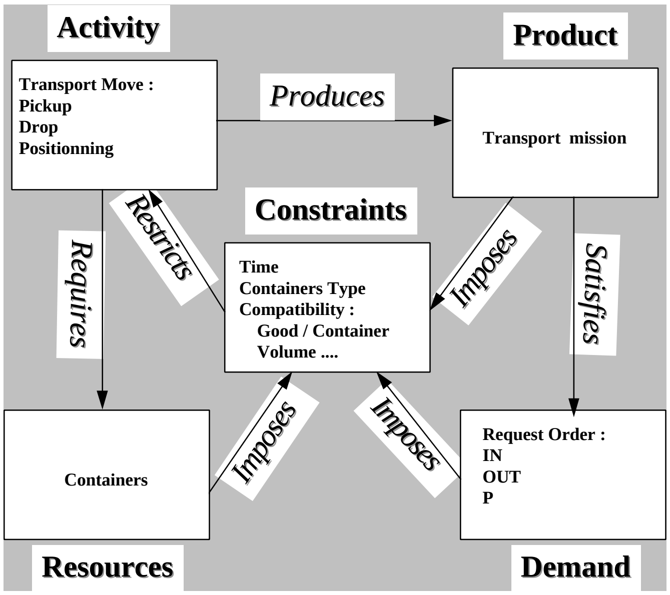
4.2.6 The PISCES Technical Approach

A large variety of VRPs exist in terms of objectives, constraints, size, and other structural characteristics. Available VRP tools suffer from their lack of openness, flexibility, and re-usability. PISCES has developed a comprehensive, CSP and GA library in C-JAVA called Routing Scheduler. This contains both data structures, to implement the description of the VRP instance, and algorithms, to handle this representation and to generate satisfactory solutions. Routing Scheduler is based on the constraint programming paradigm and contains algorithms for constraint satisfaction and optimisation based on genetic algorithm.

The constraint programming approach adopted addresses the efficiency and, flexibility shortcomings of existing VRP tools. Constraint programming enables the formulation and re-formulation of problems in a very efficient way. Constraints are used actively to prune the search space, allowing large VRPs to be addressed. Robust and efficient algorithms based on Genetic Algorithm (GA), has been developed for optimisation. These are designed to provide high quality solutions for large, ill-structured problems (i.e. industrial applications). Implementation of GA algorithms on a generic constraint-programming basis provides benefits across all application domains. It offers fast and easily customised search.

4.2.6.1 Modelling for the Triangulation Problem

The Triangulation problem is described as a resource allocation problem with time constraints, see Figure 3. The problem is modelled as a CSP, which consists of a set of variables, for each variable, a finite set of possible values (its domain), and a set of *constraints* restricting the values that the variables can take simultaneously. A solution to a CSP is an assignment of a value to every variable satisfying all constraints. In PISCES orders, containers, date, products, delivery time and locations are variables. Conceptually an order will have containers, date, products, delivery time, and origin and destination locations that have to be allocated for it, and these are represented as domain variables. A set of constraints represents the container types, products, the mode of transport, the previous contents of a container and so on.



A resource allocation problem can be defined as a process of feasibly synchronising the use of Resources by Activities to satisfy Demands over time. A Demand is an input request for one or more Products, which designate the Goods or Services required. Satisfaction of Demands centres around the execution of Activities. An activity is a process that uses Resources to produce Goods or Services. The use of Resource and the execution of Activities are restricted by a set of Constraints.

Figure 3: Abstraction of the Triangulation Problem

4.2.6.2Constraint-Based Modelling and Solving

The VRP is modelled as a CSP which consists of a set of variables. For each variable, a finite set of possible values (its domain), and a set of *constraints* restricting the values that the variables can take simultaneously. A solution to a CSP is an assignment of a value to every variable satisfying all constraints. The Triangulation Problem is modelled as a CSP.

4.2.6.2.1 Application to the Triangulation Problem

Definition of a Demand is a request for a transport service (Pickup, Drop and Positioning). The set of all Demands determines the current Triangulation Problem to be solved. First, it is important to define the domains of variables:

Containers :
Location :
Request :
Time :
Priority:
Mode_of_Transport :

Now each request is described as a set of variables associated to it and their constraints. Following the above generic description, we apply the approach to the Container Transportation problem and for the different cases that may appear.

i - Request Ri as an ***IN*** *at the terminal with laden containers - (ex : request 1)*

Drop goods to terminal : to describe this request we introduce the following variables, constraints and cost.

Variables :

Release_Date_Ri : Time
Empty_Container_where_from_Ri : Location
Empty_Container_where_to_Ri : Location
How_move_Empty_Container_Ri : Mode_of_Transport
Next_job_what_Ri : Request

Constraints :

/restricts the domain of Empty_Container_where_from_Ri.
*Empty_Container_where_from_Ri : {**Location**(Ri)}*

/restricts the domain of Next_job_what_Ri.
Next_job_what_Ri : {nil, r2, ... , rk} ***/ a subset Ni, of Requests IN to a client location, and/or of Requests P to a depot, with the same shipping line and the same type of containers, the same size of containers, and with compatible volume and commodity.*** *nil* ***meaning the containers remain in the terminal.***

/restricts the domain of Empty_Container_where_to_Ri.
*Empty_Container_where_to_Ri : {**Location**(Ri), c2, ... ,ck}* ***/ a subset of Location that matches Ni.***

/restricts the domain of How_move_Empty_Container_Ri.
How_move_Empty_Container_Ri : {none, t2, ..., tk} ***/a subset of Mode_of_Transport that matches Ni.*** *none* ***meaning the containers remains in the terminal.***

/Constraints on the compability (Next_job_what_Ri, Empty_Container_where_to_Ri).
*(Next_job_what_Ri, Empty_Container_where_to_Ri) : (nil, **Location**(Ri)).*
(Next_job_what_Ri, Empty_Container_where_to_Ri) : (r2, c2).
...
(Next_job_what_Ri, Empty_Container_where_to_Ri) : (rk, ck).

/Constraints on the compability (Next_job_what_Ri, How_move_Empty_Container_Ri).
(Next_job_what_Ri, How_move_Empty_Container_Ri) : (nil, none).
(Next_job_what_Ri, How_move_Empty_Container_Ri) : (r2, t2).
...
(Next_job_what_Ri, Empty_Container_where_to_Ri) : (rk, tk).

/Constraints on the Due_Date(Next_job_what_Ri).
Release_Date_Ri **+**
***Time**(**Distance**(Empty_Container_where_from_Ri, Empty_Container_where_to_Ri),*
How_move_Empty_Container_Ri)
***Before** Due_Date(Next_job_what_Ri).*

/the following constraints may be used as pre-processing during the construct of Ni.
*Shipping-line(Ri) **Compatible** Shipping-line(Next_job_what_Ri).*
*Container(Ri) **Compatible** Container(Next_job_what_Ri).*

Volume(Ri) **<=** *Volume(Next_job_what_Ri).*
Commodity(Ri) ***Compatible*** *Commodity(Next_job_what_Ri).*

Cost :
Distance*(Empty_Container_where_from_Ri, Empty_Container_where_to_Ri)*

ii - Request Ri as an ***IN*** *at the terminal with empty containers - (ex : request 2)*
Pickup goods from terminal : to describe this request we introduce the following variables, constraints and cost.

Variables :
Empty_Container_where_from_Ri : Location
Empty_Container_where_to_Ri : Location
Full_Container_where_to_Ri : Location
How_move_Full_Container_Ri : Mode_of_Transport
Next_job_what_Ri : Request
Previous_job_what_Ri : Request

Constraints :
/restricts the domain of Empty_Container_where_to_Ri.
*Empty_Container_where_to_Ri : {****Location****(Ri)}*

/restricts the domain of Next_job_what_Ri.
Next_job_what_Ri : {r1, ... , rk} ***/a subset Ni, of Requests IN at a client location, with the same shipping line, the same type of containers, the same size of containers, and with compatible volume and commodity.***

/restricts the domain of Full_Container_where_to_Ri.
Full_Container_where_to_Ri : { c1, ... ,ck} ***/a subset of Location that matches Ni.***

/restricts the domain of How_move_Full_Container_Ri.
How_move_Full_Container_Ri : {t1, ... , tk} ***/a subset of Mode_of_Transport that matches Ni.***

/Constraints on the compability (Next_job_what_Ri, Full_Container_where_to_Ri).
(Next_job_what_Ri, Full_Container_where_to_Ri) : (r1, c1)
...
(Next_job_what_Ri, Full_Container_where_to_Ri) : (rk, ck)

/Constraints on the compability (Next_job_what_Ri, How_move_Full_Container_Ri).
(Next_job_what_Ri, How_move_Full_Container_Ri) : (r1, t1)
...
(Next_job_what_Ri, How_move_Full_Container_Ri) : (rk, tk)

/restricts the domain of Previous_job_what_Ri.
Previous_job_what_Ri : {p1, ..., pl} ***/a subset Pi, of Requests OUT at a client location, with the same shipping line, the same type of containers, the size of containers, and with compatible commodity.***

/restricts the domain of Empty_Container_where_from_Ri.

Empty_Container_where_from_Ri : {c1, ..., cl} ***/a subset of Location that matches Pi.***

/restricts the domain of How_move_Empty_Container_Ri.
How_move_Empty_Container_Ri : {t1, ..., tl} ***/a subset of Location that matches Pi.***

/Constraints on the compability (Previous_job_what_Ri, Empty_Container_where_from_Ri).
(Previous_job_what_Ri, Empty_Container_where_from_Ri) : (p1,c1)
...
(Previous_job_what_Ri, Empty_Container_where_from_Ri) : (pl,cl)

/ Constraints on the compability on (Previous_job_what_Ri, How_move_Empty_Container_Ri).
(Previous_job_what_Ri,How_move_Empty_Container_Ri) : (p1,t1)
...
(Previous_job_what_Ri,How_move_Empty_Container_Ri) : (pl,tl)

/Constraints on the Due_Date(Ri).
Release_Date(Previous_job_what_Ri) **+**
***Time**(**Distance**(Empty_Container_where_from_Ri, Empty_Container_where_to_Ri), How_move_Empty_Container_Ri)*
***Before** Due_Date(Ri).*

/the following constraints may be used as pre-processing during the construct of Pi.
Shipping-line(Ri) ***Compatible*** *Shipping-line(Previous_job_what_Ri).*
Container(Ri) ***Compatible*** *Container(Previous_job_what_Ri).*
Volume(Ri) **<=** *Volume(Previous_job_what_Ri).*
Commodity(Ri) ***Compatible*** *Commodity(Previoust_job_what_Ri).*

Cost :
***Distance**(Empty_Container_where_from_Ri, Empty_Container_where_to_Ri)*

iii - Request Ri as an **IN** at a client location with laden containers - (ex : request 3)
Drop goods to a client location : to describe this request we introduce the following variables, constraints and cost.

Variables :
Release_Date_Ri : Time
Empty_Container_where_from_Ri : Location
Empty_Container_where_to_Ri : Location
How_move_Empty_Container_Ri : Mode_of_Transport
Next_job_what_Ri : Request

Constraints :
/restricts the domain of Empty_Container_where_from_Ri.
*Empty_Container_where_from_Ri : {**Location**(Ri)}*
/restricts the domain of Next_job_what_Ri.
Next_job_what_Ri : { r1, ... , rk} ***/a subset Ni, of Requests OUT at a client location, and/or of Requests P at a depot, or of Requests IN at a Terminal, with the same shipping line and the same type of containers, the same size of containers, and with compatible volume and commodity.***

/restricts the domain of Empty_Container_where_to_Ri.
*Empty_Container_where_to_Ri : {**Location**(Ri), c2, ... ,ck}* ***/a subset of Location that matches Ni.***

/restricts the domain of How_move_Empty_Container_Ri.
How_move_Empty_Container_Ri : {t1, ..., tk} ***/a subset of Mode_of_Transport that matches Ni.***

/Constraints on the compability (Next_job_what_Ri, Empty_Container_where_to_Ri).
(Next_job_what_Ri, Empty_Container_where_to_Ri) : (r1, c1)
...
(Next_job_what_Ri, Empty_Container_where_to_Ri) : (rk, ck)

/Constraints on the compability (Next_job_what_Ri, How_move_Empty_Container_Ri).
(Next_job_what_Ri, How_move_Empty_Container_Ri) : (r1, t1)
...
(Next_job_what_Ri, Empty_Container_where_to_Ri) : (rk, tk)

/Constraints on the Due_Date(Next_job_what_Ri).
Release_Date_Ri ***+ Time (Distance*** *(Empty_Container_where_from_Ri,*
Empty_Container_where_to_Ri),
How_move_Empty_Container_Ri) ***Before*** *Due_Date(Next_job_what_Ri).*

/the following constraints may be used as pre-processing during the construct of Ni.
Shipping-line(Ri) ***Compatible*** *Shipping-line(Next_job_what_Ri).*
Container(Ri) ***Compatible*** *Container(Next_job_what_Ri).*
Volume(Ri) **<=** *Volume(Next_job_what_Ri).*
Commodity(Ri) ***Compatible*** *Commodity(Next_job_what_Ri).*

Cost :
Distance*(Empty_Container_where_from_Ri, Empty_Container_where_to_Ri)*

*iv - Request Ri as an **OUT** at the terminal with laden containers - (ex : request 8)*
Drop goods to terminal from client location : to describe this request we introduce the following variables, constraints and cost.

Variables :
Release_Date_Ri : Time
Empty_Container_where_from_Ri : Location
Empty_Container_where_to_Ri : Location
How_move_Empty_Container_Ri : Mode_of_Transport
Previous_job_what_Ri : Request

Constraints :
/restricts the domain of Previous_job_what_Ri.
Previous_job_what_Ri : {p1, ..., pl} ***/a subset Pi, of Requests IN laden at a terminal or/and at a client location, with the same shipping line and the same type of containers, the same size of containers, and with compatible volume and commodity.***

/restricts the domain of Empty_Container_where_from_Ri.
Empty_Container_where_from_Ri : {c1, ..., cl} ***/a subset of Location that matches Pi.***

/restricts the domain of How_move_Empty_Container_Ri.
How_move_Empty_Container_Ri : {t1, ..., tl} ***/a subset of Location that matches Pi.***

/Compatibility on (Previous_job_what_Ri, Empty_Container_where_from_Ri).
(Previous_job_what_Ri, Empty_Container_where_from_Ri) : (p1,c1)
...
(Previous_job_what_Ri, Empty_Container_where_from_Ri) : (pl,cl)

/Compability on (Previous_job_what_Ri, How_move_Empty_Container_Ri)
(Previous_job_what_Ri, How_move_Empty_Container_Ri) : (p1,t1)
...
(Previous_job_what_Ri, How_move_Empty_Container_Ri) : (pl,tl)

/Constraints on the Due_Date(Next_job_what_Ri).
Release_Date_Ri **+**
Time *(**Distance** (Empty_Container_where_from_Ri, Empty_Container_where_to_Ri),*
How_move_Empty_Container_Ri)
Before *Due_Date(Next_job_what_Ri).*

/the following constraints may be used as pre-processing during the construct of Ni.
Shipping-line(Ri) ***Compatible*** *Shipping-line(Next_job_what_Ri).*
Container(Ri) ***Compatible*** *Container(Next_job_what_Ri).*
Volume(Ri) **<=** *Volume(Next_job_what_Ri).*
Commodity(Ri) ***Compatible*** *Commodity(Next_job_what_Ri).*

Cost :
Distance*(Empty_Container_where_from_Ri, Empty_Container_where_to_Ri)*

*v - Request Ri as a **P** at a depot/terminal with empty containers - (ex : request 12)*
Position containers at a depot/terminal : to describe this request we introduce the following variables, constraints and cost.

Variables :
Release_Date_Ri : Time
Empty_Container_where_from_Ri : Location
Empty_Container_where_to_Ri : Location
How_move_Empty_Container_Ri : Mode_of_Transport
Previous_job_what_Ri : Request

Constraints :
/restricts the domain of Empty_Container_where_to_Ri.
Empty_Container_where_to_Ri : { ***Location****(Ri)}*

/restricts the domain of Previous_job_what_Ri.

Previous_job_what_Ri : {p1, ..., pl} ***/a subset Pi, of Requests IN at a client or/and of Requests OUT with empty containers at a terminal or/and of Requests IN with laden containers, with the same shipping line and the same type of containers, the same size of containers, and with compatible volume and commodity.***

/restricts the domain of Empty_Container_where_from_Ri.
Empty_Container_where_from_Ri : {c1, ..., cl} ***/a subset of Location that matches Pi.***

/restricts the domain of How_move_Empty_Container_Ri.
How_move_Empty_Container_Ri : {t1, ..., tl} ***/a subset of Location that matches Pi.***

/Compatibility on (Previous_job_what_Ri, Empty_Container_where_from_Ri).
(Previous_job_what_Ri, Empty_Container_where_from_Ri) : (p1,c1)
...
(Previous_job_what_Ri, Empty_Container_where_from_Ri) : (pl,cl)

/Compability on (Previous_job_what_Ri, How_move_Empty_Container_Ri).
(Previous_job_what_Ri, How_move_Empty_Container_Ri) : (p1,t1)
...
(Previous_job_what_Ri, How_move_Empty_Container_Ri) : (pl,tl)

/Constraints on the Due_Date(Next_job_what_Ri).
Release_Date_Ri **+**
Time *(****Distance*** *(Empty_Container_where_from_Ri,*
Empty_Container_where_to_Ri),
How_move_Empty_Container_Ri)
Before *Due_Date(Next_job_what_Ri).*

/the following constraints may be used as pre-processing during the construct of Ni.
Shipping-line(Ri) ***Compatible*** *Shipping-line(Previous_job_what_Ri).*
Container(Ri) ***Compatible*** *Container(Previous_job_what_Ri).*
Volume(Ri) **<=** *Volume(Previous_job_what_Ri).*
Commodity(Ri) ***Compatible*** *Commodity(Previous_job_what_Ri).*

Cost :
Distance*(Empty_Container_where_from_Ri, Empty_Container_where_to_Ri)*

vi - Request Ri as an ***OUT*** *at the terminal with laden containers - (ex : request 45)*
Collect goods from client location to terminal : to describe this request we introduce the following variables, constraints and cost.

Variables :
Release_Date_Ri : Time
Empty_Container_where_from_Ri : Location
Empty_Container_where_to_Ri : Location
How_move_Empty_Container_Ri : Mode_of_Transport
Next_job_what_Ri : Request

Constraints :

/restricts the domain of Empty_Container_where_from_Ri.

Empty_Container_where_from_Ri : {Terminal}

/restricts the domain of Next_job_what_Ri.

Next_job_what_Ri : {nil, r2, ... , rk} ***/ a subset Ni, of Requests IN to a client location, and/or of Requests P to a depot, with the same shipping line and the same type of containers, the same size of containers, and with compatible volume and commodity.*** *nil* ***meaning the containers remain in the terminal.***

/restricts the domain of Empty_Container_where_to_Ri.

Empty_Container_where_to_Ri : {Terminal, c2, ... ,ck} ***/ a subset of Location that matches Ni.***

/restricts the domain of How_move_Empty_Container_Ri.

How_move_Empty_Container_Ri : {none, t2, ..., tk} ***/a subset of Mode_of_Transport that matches Ni.*** *none* ***meaning the containers remains in the terminal.***

/Constraints on the compability (Next_job_what_Ri, Empty_Container_where_to_Ri).

(Next_job_what_Ri, Empty_Container_where_to_Ri) : (nil,Terminal).

(Next_job_what_Ri, Empty_Container_where_to_Ri) : (r2, c2).

...

(Next_job_what_Ri, Empty_Container_where_to_Ri) : (rk, ck).

/Constraints on the compability (Next_job_what_Ri, How_move_Empty_Container_Ri).

(Next_job_what_Ri, How_move_Empty_Container_Ri) : (nil, none).

(Next_job_what_Ri, How_move_Empty_Container_Ri) : (r2, t2).

...

(Next_job_what_Ri, Empty_Container_where_to_Ri) : (rk, tk).

/Constraints on the Due_Date(Next_job_what_Ri).

Release_Date_Ri **+**

Time *(*

  Distance *(Empty_Container_where_from_Ri,Empty_Container_where_to_Ri),*

  How_move_Empty_Container_Ri)

      Before *Due_Date(Next_job_what_Ri).*

/the following constraints may be used as pre-processing during the construct of Ni.

Shipping-line(Ri) ***Compatible*** *Shipping-line(Next_job_what_Ri).*

Container(Ri) ***Compatible*** *Container(Next_job_what_Ri).*

Volume(Ri) **<=** *Volume(Next_job_what_Ri).*

Commodity(Ri) ***Compatible*** *Commodity(Next_job_what_Ri).*

Cost :

Distance*(Empty_Container_where_from_Ri, Empty_Container_where_to_Ri)*

The modelling process is systematic and can be produced automatically from a database of requests. Once the problem has been modelled, the hybrid algorithm is applied. This will yield solutions that respect the constraints and optimise the objective function.

4.2.6.2.2 CSP Algorithm

Apart from the flexibility of formulating a problem as a CSP, another advantage is the large set of algorithms that have been developed to solve a CSP efficiently. Among these algorithms, forward checking (Frost, D. *et al.,* 1994) turns out to give a good performance for a lots of CSPs. This is adapted in PISCES to compute feasible solutions on a subspace of the search space for the problem.

The algorithm adopted in PISCES provides an efficient method of searching for a set of values that respect the constraints placed on the variables. It uses a representation of the availability of vehicle/containers over time. Since a vehicle/container cannot be used for two tasks at the same time, availability is modelled by a single variable representing a resource. That is, when a resource is allocated to an order, the system will check the resource's availability and then mark it as unavailable to other orders for the duration of the transport. When a resource is marked as unavailable for a period it can also be removed from the domain of any order which would require it to be used during that period. Also compatibility between products and the previous contents of a container is examined. Applying the search mechanism to a set of transport requests modelled as a CSP provides us with a set of routes between terminals/distribution centre, depots, and clients, whilst respecting the different constraints.

4.2.6.3 Genetic Algorithm and Maintaining Constraints

A Genetic Algorithm evolves a population of individuals encoded as chromosomes by creating new generations of offspring through an iterative process until some convergence criteria or conditions are met. Such criteria might, for instance, refer to a maximum number of generations or the convergence to a homogeneous population composed of similar individuals. The best chromosome generated or decoded after generation, provide the solution. The underlying reproduction process is mainly aimed at improving the fitness of individuals; a measure of profit, utility or goodness to be maximised while exploring the solution space. We consider as fitness a multi-objective function that calculates the distance, the cost of the transport, and the possibility of intermodal transport.

The creation of a new generation of individuals involves primarily four major steps or phases: representation, selection, recombination, and mutation. The representation of the solution space consists in encoding salient features of a solution as a chromosome (chromosome encoding), defining an individual member of a population. A chromosome is made up of a sequence of genes that capture the basic characteristics of a solution. The evolution of the encoded solutions are driven by the selection, recombination, and mutation phases respectively. The selection phase consists in choosing randomly two parent individuals from the population for mating purposes. The probability to select a population member (parent) is generally proportional to its fitness in order to emphasise genetic quality while maintaining genetic diversity. Biased toward the best chromosomes, selection is aimed at propagating good solution features from one generation to the next. The recombination or reproduction process propagates genes of selected parents to produce offspring's that will form the next generation. It combines characteristics of chromosomes (parent solutions) to potentially create offsprings with better fitness. As for mutation, it consists in randomly modifying gene(s) of a single individual at a time to further explore the solution space and ensure, or preserve, genetic diversity. The occurrence of mutation is generally associated with a low probability. A new generation is created by repeating the selection, reproduction and mutation processes until all chromosomes in the new population replace those from the old one.

GAs are well suited to the quick global exploration of a large search space to optimise an objective function, enabling several solutions of "good quality" to be identified, using this approach alone. Within PISCES, the set of the feasible solutions is complex due the number of constraints which must be maintained. Furthermore, in most cases, an optimisation problem can naturally be divided into two phases: the search for feasible solutions and then the search of the solution with the lowest cost among them. This division is more or less obvious during the search according to the choice of the optimisation method. However, there is no such simple dichotomy among the set of optimisation problems, and in our case the Triangulation problem has a large search space. These two features exclude the direct and naive use of the GA or a CSP approach alone. We take advantage of the two techniques in a hybrid approach:

- a CSP program to compute feasible solutions on a subspace of the search space.
- a GA to explore the space formed by the solutions provided by the CSP, and performs the optimisation.

The feasibility of solutions is then defined as intrinsic to the chosen representation (i.e. a set of routes for the vehicles/containers between terminals/distribution centre, depots, clients, and respecting the different constraints). This is integrated within the creation of the chromosomes in the different steps (*initialisation*, *crossover* and *mutation*) and within the objective function (i.e. an unfeasible solution is given a low fitness). GA proceeds by iterative improvement.

4.2.6.4 Data Interface

The Scheduling System is fed from the PISCES Database to produce possible routing, minimising distances (total travelled distance, empty running) and costs (Transportation cost, by minimisation of distances and providing an Intermodal alternative whenever possible; and minimising the handling costs). The information used comprises:

The Data Tables

These contain background information about the area covered by the containers' transport. It concerns the resources availability, the locations, and the costs.

<u>Transports Requests</u>

JOB_REFERENCE,

CONTAINER_NUMBER,

CONTAINER_OWNER,

CONTAINER_SIZE,

DATE_STAMP,

DATE_LOAD,

DATE_DISCHARGE,
TIME_LOAD,
TIME_DISCHARGE,
PRIORITY,
I/O/P,
POINT_LOAD,
CITY_LOAD,
TYPE_LOAD,
COUNTRY_LOAD,
IDENTIFIER_LOAD,
POINT_DISCHARGE,
CITY_DISCHARGE,
TYPE_DISCHARGE,
COUNTRY_DISCHARGE,
IDENTIFIER_DISCHARGE,
GOODS_DESCRIPTION

Initial Situation

Describe the initial distribution of Containers between depots and Terminal. It may be grouped by location or by Containers type.
The necessary fields are :

Location
Number of Containers
List of Containers Reference Number

Transport's Cost

This table provide the cost of transport according to the distance and the mode of transport. If possible a formula to calculate it, if not, the necessary fields are :

Travelled Distance
Cost for Road
Cost for Train
Cost for Barge

Locations of Depots

Gives the location of Depots that may be used between transports (i.e. to check, clean and tune the previously used containers).
The necessary fields are:

Depot Country
Depot City
Depot Zip-code
Overnight storage possible (Y/N)
Depots / Ports – Cost?

Locations of Head Train / Barge

Gives the location where an alternative transport mode is available.
The necessary fields are:
Head Location Country
Head Location City
Head Location Zip-code
Alternative mode of Transport

Depot's Cost handling

Gives the cost of containers handling for each depots, including Terminals.
The necessary fields are:
Depot Location Country
Depot Location City
Depot Location Zip-code
Handling Cost
Storage Costs – internal / external / differential for externals

4.2.6.5 The Schedule Generation Process

To address the software development bottleneck even further, an accompanying generic user interface model for VRP has been developed.

In use, the scheduler expects the transportation requests database to pass a set of orders, together with information about resources, and their constraints. It then generates and returns a schedule.

***User Intervention*:** One shortcoming of the approach implemented so far is that the system takes responsibility/control away from the human scheduler. The data is fed in and the system produces the best schedule it can. However the notion of “best" involves a trade-off between meeting user preferences and reducing costs. Ultimately quality is something that only the human scheduler can judge, and so the user will be allowed to exert some influence on the schedule generation process. The HCI have been modified to take into account this point. This has been implemented by allowing a restriction of the initial domain of the relevant variable.
The mechanism for this is through partial specification, that is the user may restrict resources (i.e. containers of type or volume, geographical area) considered for a particular order. For example the user may specify that the area for a particular order should be selected for a specified set of orders, or that any order can be selected apart from those specified.

A second mechanism implemented is to modify the availability of a resource. This is used for example to handle planned maintenance of containers. The user may choose to partially specify the resources for any order at the outset. However the more usual case will be where the system generates a schedule that the user examines and may choose to modify. Essentially modification amounts to re-specifying the problem and allowing the system to generate a new solution, however if there are parts of the schedule that the user likes, the generated schedule can be used as the basis of the resource restrictions.

***Reactive Scheduling*:** Once an acceptable schedule has been created it is used to determine what to do the next day. For this reason it is said to be a predictive schedule.
Unfortunately in the real world there are many things that can happen which cause the schedule to become unworkable. These include breakdowns, delays, cancelled orders, and the arrival of new orders. Although a cancelled order does not necessarily make a schedule unworkable, it provides an opportunity for re-optimisation. To deal with such situations a reactive capability is required.

In the current prototype, reactive scheduling is treated as rescheduling, that is the problem is reformulated to take account of journeys that have already been performed and those that are underway, and the system generates a schedule for the new problem. A corollary of this is that the new schedule may vary greatly from the previous schedule. For this reason rescheduling may not be a suitable approach in some types of application where decisions may have been made on the basis of the predictive. This is not generally a problem in the present application, however the partial specification of resources can be used to retain parts of the existing schedule where necessary.

4.2.7 Rescheduling Problem

4.2.7.1 Problem Presentation

The triangulation scheduler is used to collect a set of transport orders from a transportation requests database, together with information about available resources, and their constraints, and then to generate and return a schedule. Once an acceptable schedule has been created it is used to determine what to do the next day. For this reason it is said to be a predictive schedule.

In the real world there are many events that may happen that which could make the schedule unworkable. These include breakdowns, road accident, delays, arrival of new orders with high priority, and cancelled orders. Although a cancelled order does not necessarily make a schedule unworkable, it provides an opportunity for re-optimisation.
To deal with such situations a reactive capability is required. In PISCES this reactive scheduling is managed as *rescheduling*. The problem is reformulated to take into account journeys that have already been performed and those that are underway, and the system generates a schedule, adapting to the new context.

The rescheduling problem is not the same as the initial predictive scheduling. The new schedule must be an adaptation of the original schedule which respects a different set of constraints and for which the optimisation criteria are often quite different. The provision of the new fitness / optimisation is a difficult and currently poorly understood problem subject to considerable variation between different operational and commercial situations. The approach, based on disruption metrics, described here is an attempt to address these difficult issues.

In Figure 3, a small example of a triangulated routing solution is described, composed of three networks {SN1=(0,1,2); SN2=(0,3,4,5); SN3(0,6,5)}. The nodes represent the locations (node 0 is a depot, and the others nodes (1 .. 7) are the customers). Time-windows are given in square brackets, the actual delivery times are in squares, and the travelling times between locations are in circles.

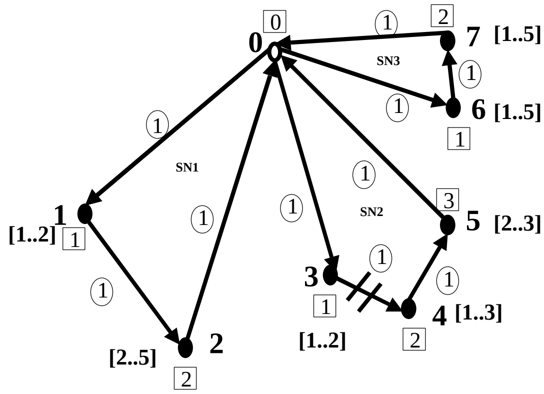


Figure 3: Initial Routing - S0

This illustration is a very simple and uniform situation, intended to review the issues.
We suppose that there is some delay (due to a breakdown for example) during the transport from node 3 to node 4.

It is necessary to reschedule the transport routing to produce a new set of routes. The new routing will vary from the initial one, but as any new change in the current routing may lead to some problems or charges for the transport's company, the aim is to obtain a new routing as close as possible to the initial one. The partial specification of resources can be used to retain parts of the existing schedule when necessary.

4.2.7.2 Managing Rescheduling

Following rescheduling, different effects within the chain need to be controlled. There will be delays and variations on the faulty network, SN2 in this case. This may require reducing the propagation of those delays and variations to all the operations of this network, as this may have potential impact on important operational and commercial considerations. Also a reduction in the use of new resources maybe required (drivers, containers, or truck) which are expensive in urgent situations.

Further, it maybe necessary to reduce the effect on other networks (SN1 and SN3 in this example) when trying to exploit their resources to solve the problem that occurs in SN2.
These three requirements are contradictory, and necessitate a trade-off between them to solve the problem.

Firstly, its important to manage the remaining operations of the faulty network, by triangulating them with the operations of other networks. A limitation to this is that the operations of the different networks must be compatible over the constraints. Then, its important to reduce the effect (delay) on the network/operations used to triangulate the operations of the faulty network. Finally, if there is no solution generated using the current resources, then new resources may need to be introduced.

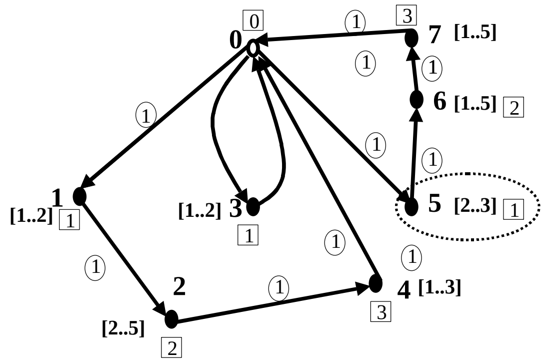


Figure 4: New Routing - S1

Figure 4 shows the new routing S1. The problem is solved using the operations of the two networks SN1 and SN3, the negative impact is important because several resources are shifted from the initial routing, and we arrive too early at node 5.

In new routing S2 (Figure 5), the problem is solved using the operations of the network SN1 only. There are fewer resources shifted from the initial routing, but the negative impact is significant because of the late arrival late at node 5.

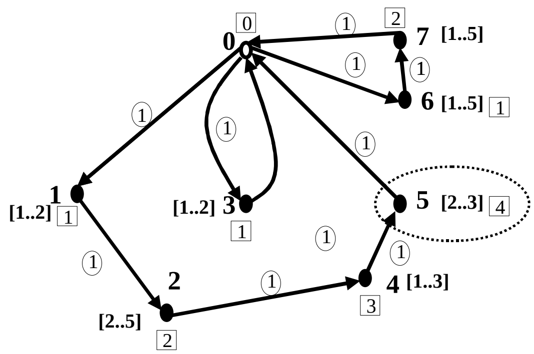


Figure 5: New Routing - S2

In new routing S3 (Figure 6) the operations of the networks SN1 and SN3 are not affected, but the negative effect is significant again, because the problem is solved using new resources and, we arrival at node 5 is too early.

A further problem arises when it maybe necessary to minimise the distance for the new routing (without introducing new resources). This may create routings where the number of networks affected is minimised but the number of operations affected is unacceptable.

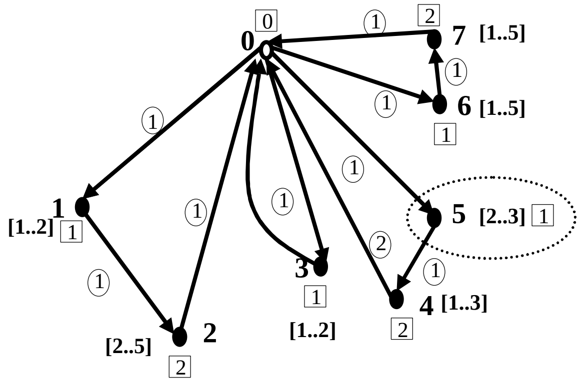


Figure 6: New Routing - S3

In new routing S4 (Figure 7), the same solution (only the operations of the network SN1 are affected) as in Figure 4, and the distance travelled is shorter. But the negative impact is greater than in Figure 4, as arrival at node 5 is too early, and too late at nodes 1 and 4.

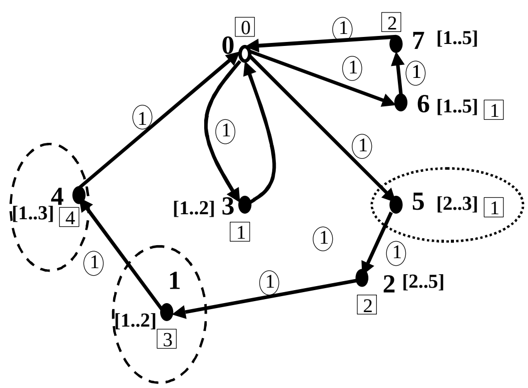


Figure 7: New Routing - S4

From these examples one can see that the choice of the criterion to minimise is not straightforward, and that the simplistic minimisation of a single criterion, without considering the others, may well lead to unacceptable outcomes. An important issue is thus to shape a metric providing a good balance between the different criteria. This balance is estimated from a business/operational point of view, and considers customers (minimise the delay), resources (minimise the introduction of new resources), operations (minimise the number of networks/operations involved in the change), staff (minimise the earliness) and the initial routing (minimise the variations).

4.2.7.3 Disruption Function

Kelleher et al [Kelleher 99], propose an innovative approach to the problem of rescheduling solution within manufacturing systems. They discuss the meaning of rescheduling, using possible metrics based on schedule disruption. They define the notion of schedule disruption, as the difference in allocation of common operations in two schedules, and propose a disruption function. This function is used to estimate the effect of changing an initial schedule to a new one. Disruption is calculated for each operation that is scheduled at a different time and/or resource in the new schedule compared with the initial one. The total disruption between the schedules is the sum of the disruption for each operation. They emphasise the fact that the disruption measured should display a realistic picture of the disruption as seen by the personnel involved in the operational task. This adds substantial complexity to the metrics by discriminating on a micro-level between the individual operation and type of operation-move.

PISCES use a similar technique for the Triangulation problem, within the CSP and GA approach. This function is problem specific and is defined differently.

From a CSP point of view rescheduling introduces an extra set of constraints which need to be addressed. There are related to the need to preserve the initial routing as much as possible. The initial routing represents an investment in planned resources, allocation of transport tasks to containers, trucks and drivers, which should not be disturbed any more than necessary.

From a Genetic algorithm's point of view rescheduling introduces a new fitness function to minimise. This is again a multi-objective function.

Common to both is the need to minimise distance, cost and number of resources. Additionally for rescheduling, its important to minimise the level of disruption. A common problem in optimisation (not just genetic algorithms) is how to define an objective function that accurately, consistently and reasonably captures the effects of multiple objectives.

The solution, adopted, is to weight each of the function against one another or simply define one of the objectives as superior to the other. This is explained in the following section.

4.2.7.4 Disruption Functions Specification

The changes involved in the new routing are said to be disruptive to the system, and we adapt an approach based on [Kelleher 99] to define and then limit the effect of disruption. We define a disruption function that allows the estimation of the effect of changes in the new routing in term of:

- **Variations**: changes in the new routing,
- **Operations**: networks and operations involved in the change,
- **Time**: delays and earliness for each operation,
- **Resources**: number of new resources introduced (truck/driver/containers).

A disruption function is defined for each criteria of changes. Experimentation has seen the use of these metrics on the data provided by Van Ommeren. The results of the tests are presented in terms of charts and discussions asses the accuracy of the metric in expressing the effects of disruption on the different criteria. Finally a global function is proposed, shaped according to the experiments.

4.2.7.4.1 Minimise the Variations

The aim is to minimise the variations between the initial and the new routing. If each routing is considered as a vector of legs, the number of different legs between the initial and the new routing would be of interest.

The disruption function for the variation, Δ_V is given by:

$$\Delta_V = 1-(1/(1+\sum_{i\in V} 1))$$

with

V= Set of Variations produced

The results obtained from the experiments are given in Figure 8.

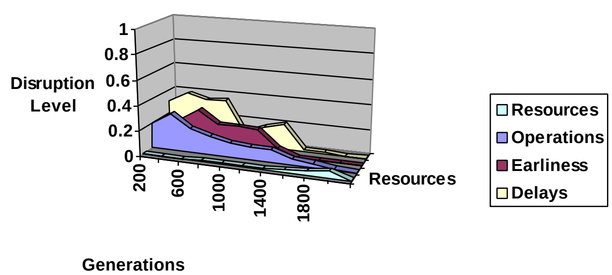


Figure 8: Level of Disruption estimated by Variation Metric

This metric favours the criteria in the following order: Operations / Earliness/ Delays/ Resources.

4.2.7.4.2 Minimise the Number of Networks and Operations

The aim is to minimize the number of networks and operations involved in the changes, (number of network and, transport operations).

In a transport routing we have two levels of operations:

- a sub routine for a given set of resource : network,
- a transport operation that is included in a network : a leg,
- Changes involving more legs for the same number of networks involved, are more disruptive.
- Changes involving more legs for fewer networks involved, are less disruptive.

The disruption function is given by:

$$\Delta_O = 1-(1/(1+\sum_{i\in N} d_i + \sum_{j\in L} d_j))$$

with

O = Set of operations Network/operations

d_i= *(number of networks involved/Total number of network*)

The results obtained from the experiments are given in Figure 9.

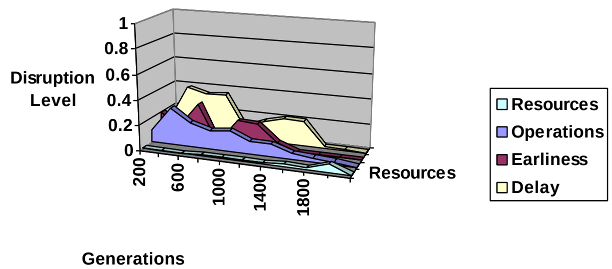


Figure 9: Level of Disruption estimated by Operations Number Metric

This metric favours the criteria in the following order: Earliness/ Operations/ Delays/ Resources.

4.2.7.4.3 Minimise the Number of Delays

The aim is to minimise the number of delays involved in the changes.
The disruption function is given by:

$$\Delta_D = 1 + (1 / (1 + \sum_{i \in D} 1))$$

with
D = Set of delays

The results obtained from the experiments are given in Figure 10.

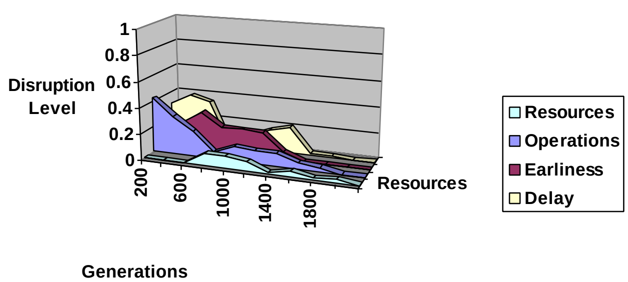


Figure 10: Level of Disruption estimated by Delays' Number Metric

This metric favours the criteria in the following order: Delays/ Earliness/ Resources/ Operations.

4.2.7.4.4 Minimise the Number of Resources

The aim is to minimise the number of new resources involved in the changes. The disruption function is given by:

$$\Delta_R = 1 - (1 / (1 + \sum_{i \in R} 1))$$

with
R = Set of Resources involved in the changes

The results obtained from the experiments are given in Figure 11.

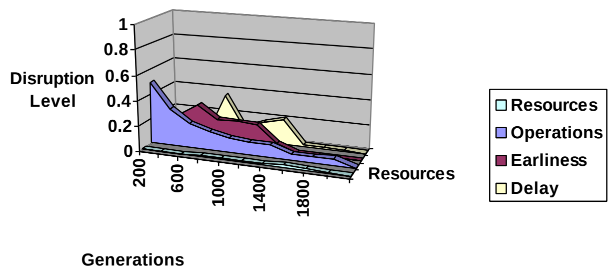


Figure 11: Level of Disruption estimated by Resources Number Metric

This metric favours the criteria in the following order: Delays/ Operations / Resources/ Earliness.

4.2.7.4.5 Global Disruption and Fitness Function

The aim is to find a good trade-off between the different requirements.

- **Variations**: number of changes in the new routing,
- **Operations**: number of networks and operations involved in the change,
- **Time**: number of delays and earliness for each operation involved,
- **Resources**: number of new resources introduced (truck/driver/containers). The disruption function is given by:

$$\Delta = 1 - (1/(1 + \sum_{j \in Type(\Delta)} w_j . \sum_{i \in Type(\Delta)} \Delta_i))$$

with
Type(D) = {V, O, D, R}
$W_v = 0.15$
$W_O = 0.15$
$W_R = 0.4$
$W_D = 0.3$

The results obtained from the experiments are given in Figure 12.

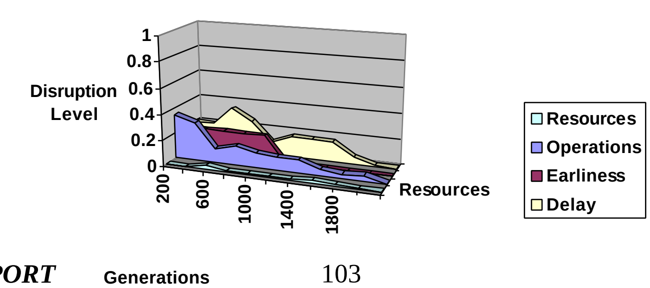


Figure 12: Level of Disruption estimated by Global Disruption Metric

Finally, this global disruption function with the tuned weight, allows us to favour the criteria in the preferred manner: Delays/ Resources / Operations / Earliness.

4.2.8 Status and Results - Scheduler

The main result from the PISCES project is a generic system, which allow rapid configuration and reconfiguration of bespoke decision support systems for optimised resolution of vehicle routing problems, e.g., transport planning systems. The users of the system will typically be consultant companies, IT system vendors, and end user companies with their own IT systems departments.

The PISCES system has been developed to consider qualitative and quantitative improvement within intermodal transport chains, and to solve the Triangulation problem for container transport. Experiments have been carried out to compare the performance of the approach adopted in PISCES, with standard techniques.

4.2.8.1 Experiment on the Pickup and Drop Problem

Please reference the results section of this report.

4.2.8.2 Experiment on VRPTW Problems

The algorithm adopted in PISCES is applied to a standard set of VRP with Time Windows instances, the Solomon's 56 benchmarks [Solomon url], which are single depot vehicle routing problems with a homogeneous fleet of vehicles. The objective is to minimise the total length of the vehicle routes, and the number of vehicles to be used. Each vehicle has a capacity that must not be exceeded, and each customer (of which there are 100 in each problem) must be visited within a particular time window. The travel time between customers is equal to the corresponding Euclidean distance. As these problems vary in fleet size, vehicle capacity, travel time of vehicles, spatial distribution of customers, time window density and width, they are divided into three categories: R-type (uniformly distributed customers), C-type (clustered customers) and RC-type (a mix of R and C types). Two sets of problems are proposed for each of these three categories. The set of benchmarks defines two classes of problem instances, namely, (R1, C1, RC1) and (R2, C2, RC2). The former include problem instances characterised by a narrow scheduling horizon and small vehicle capacity, requiring a large number of vehicles to satisfy customer demands on time. The latter involves problem instances characterised by a large scheduling horizon and large vehicle capacity, that involves fewer routes as a larger number of customers can be serviced by the same vehicle.

Problem Name	**Routes/Total Distance (best)**	**Routes/Total Distance (our result)**	**Cost Excess on the Distance**	**Problem Name**	**Routes/Total Distance (best)**	**Routes/Total Distance (our result)**	**Cost Excess on the Distance**
C101	10/827	10/828	0.12%	R112	10/953	10/1031	8.18%
C102	10/827	10/835	0.96%	R201	4/1281	4/1448	13.03%
C103	10/828	10/840	1.44%	R202	3/1530	4/1248	-18.43%
C104	10/824	10/841	2.06%	R203	3/948	3/1075	13.39%

C105	10/828	10/828	0.00%	R204	2/869	3/821	-5.52%
C106	10/827	10/828	0.12%	R205	3/1063	3/1162	9.31%
C107	10/827	10/828	0.12%	R206	3/833	3/894	7.32%
C108	10/827	10/828	0.12%	R207	3/814	3/800	-1.71%
C109	10/828	10/828	0.00%	R208	2/738	2/761	3.92%
C201	3/591	3/591	0.00%	R209	2/855	3/860	0.58%
C202	3/591	3/591	0.00%	R210	3/967	3/1047	8.27%
C203	3/591	3/600	1.52%	R211	2/949	3/816	-14.01%
C204	3/590	3/601	1.86%	RC101	14/1669	15/1696	1.61%
C205	3/588	3/588	0.00%	RC102	13/1477	13/1480	0.20%
C206	3/588	3/588	0.00%	RC103	11/1110	11/1392	25.40%
C207	3/588	3/588	0.00%	RC104	10/1135	10/1157	1.93%
C208	3/588	3/588	0.00%	RC105	13/1733	14/1652	-4.67%
R101	18/1607	19/1656	3.04%	RC106	12/1384	12/1416	2.31%
R102	17/1434	17/1479	3.13%	RC107	11/1230	11/1303	5.93%
R103	13/1207	14/1310	8.53	RC108	10/1170	10/1281	9.48%
R104	10/982	10/1081	10.08%	RC201	4/1249	4/1616	29.38%
R105	14/1377	14/1404	1.96%	RC202	4/1165	4/1380	18.45%
R106	12/1252	12/1350	7.82%	RC203	3/1079	3/1222	13.25%
R107	10/1159	11/1085	-6.38%	RC204	3/806	3/903	12.03%
R108	9/980	10/965	-1.53%	RC205	4/1333	4/1465	9.09%
R109	11/1235	12/1186	-3.97%	RC206	3/1212	4/1215	0.24%
R110	10/1080	11/1133	4.69%	RC207	3/1085	3/1510	39.17%
R111	10/1129	11/1070	-5.22%	RC208	3/834	3/960	15.10%

Table 2: Solomon's VRPTW instances Results.

The results are summarized in table 3. Benchmarks show instances problem name with the best published solutions [Rochat 1995], compared with PISCES best solutions (number of vehicles and total distance) and the percentage gains made over the VRPTW. White areas indicate all problem instances where PISCES solutions match or perform the best known solutions.

PISCES approach performs well, and results show that computed solutions easily competes with most of the best solutions. Also it shows to be quite satisfactory for clustered problems (C1, C2), as computed solution quality (distance and number of vehicles) nearly match best known results. But the procedure slightly degrades in quality, as total distance may differ up to 10% for R1 and 25% for RC1. This observation becomes more apparent for R2 and RC2 showing some algorithm weaknesses in efficiently combining routes (or vehicles) having a large number of customers. The particular subset of parameters chosen for the genetic operators are believed to insufficiently reduce and confine the search space to good solution neighborhoods for the problems instance R2 and RC2. However, notice that for all standard test problems, the computed number of vehicles mostly corresponds or slightly differs from the best solution reported. On the other hand, the computation of a near-optimal solution is reasonably fast (. < 10 minutes on a Sun Sparc10).

4.2.9 PISCES Scheduler Functionalities

This aspect of the PISCES project enables vehicle routing and scheduling designed to support the dispatcher in planning the routes and schedules for container transport. The objective is the minimisation of distribution costs under complex operational constraints, considering travelled distance, empty-running, and intermodal solution when available.
Advanced optimisation techniques based on constraints satisfaction technology and Genetic algorithm produces operative solution for a wide variety of problems with different operational constraints.

A sophisticated interactive graphic user interface makes the system easy and intuitive to use, even for users with little previous computer skill.

The Scheduler is a modular system, which can be configured in accordance with user needs, maximising the cost/benefits trade-off. It is continuously upgraded, considering user requests. Customisation parameters can be implemented to satisfy special user needs and, to ease user operational practice. The system has been developed under Unix environment in C and Java languages and will also support direct data exchange with the Data Base system.

4.2.9.1 The System Uses

- Periodic (daily, weekly, …) planning and scheduling of pickup, delivery and positioning transport, with on-line connection with Data Base system.
- Dynamic management of last minute orders.
- Dynamic Rescheduling when there is some delay or breakdown on planned transport.
- Design of fixed routes over a period when the customer demand pattern is stable.
- Strategic design of the distribution network: depots location, vehicle fleet dimension, analysis of pricing policies, assignment of distribution areas to the depots, intermodal solution, etc…

4.2.9.2 System Features

The system has been designed to be able to deal with all real-world constraints and requirements, such as:

- Clients;
- Orders;
- Vehicles;
- Road map;
- Trips;
- Operational constraints;
- Tariffs;
- Geographic areas.

Appropriate interfaces support direct data exchange with the Data Base.

4.2.9.3 System Benefits

Benefits which the PISCES scheduling capability offers includes:

- Saving of distribution costs.

- Improved service to customers, (delivery time more consistent with user needs).
- Better control and improved efficiency of the depot operations.
- Optimisation of the trade-off between level of service to customers and distribution costs.
- Better utilisation of company-owned vehicles.
- Better uses of intermodal infrastructure (i.e. road, rail, barge,…), which is environmentally friendly.
- Enhanced facilities to buyers of transport services via better planning information,
- Enhanced track and trace facilities and, more reliable time-slots.

4.2.9.4 Modules of the System

The system is configurable, enabling enhancements to the basic module through the addition of a set of sub-modules that are under the control of user.
These features include:
- Basic Module.
- Operational Constraints.
- Road Map Editing.
- Optimisation.
- Transit Depots.
- Pricing Policies.
- Distance and Time Matrix.
- Barge and Train availability, cost and timetable information
- Geographic Areas.
- User Distances.

Basic Module
Some of its functions are:
- Generation of optimised planning providing triangulated routes for a set of mixed (pickup, delivery, positioning) transport orders;
- Calculation of distances, empty, cost and percentage savings for planning using alternative modes of transport;
- Computation of shortest paths among two or more locations;
- Import/Export routes and, orders with the company information system;
- Evaluation of the cost of imported routes;
- Dynamic Rescheduling of initial planning in case of new orders with a high priority, delay or breakdown of a planned transport.

4.2.9.5 Operational Constraints

All main real-world constraints are managed, for example:
-time window, i.e., a time interval within which the request must be serviced.
-Containers (and/or Vehicles), i.e., they may have different capacities in terms of weight, volume, length, availability etc…
-Some shipments may need special containers (vehicles), i.e., for instance due to dangerous chemicals or special temperature requirements.

-There will be time frames and geographical constraints on tours, including constraints on start and end points.
-drivers working time,
-pickup delivery, positioning, etc.

4.2.9.6 Road Map Editing

The system uses its own road digital network, in order to compute distances and travel times. The user can customise the map, modifying, adding and removing locations and roads.

4.2.9.7 Optimisation

Operational routes are produced by advanced optimisation algorithms, designed to consider real-world constraints and requirements.

4.2.9.8 Transit depots

It is possible to decide whether to directly serve a customer or to make use of transit platforms associated to him. The system computes distribution costs from depots to depots and from depots to customers.

4.2.9.9 Pricing Policies

The cost of a route can be computed according to different pricing policies, based on mileage, weights, etc.

4.2.9.10 Distance and Time Matrix

The user can generate a matrix specifying distances and travel times between every pair of locations from a given list. Such matrix can be exported to the information system.

4.2.9.11 Geographic Areas

The orders of a problem can be partitioned into geographic areas. Each area so obtained defines a sub-problem, which can be solved using only the vehicles associated with it.

4.2.9.12 User Distances

It is possible to specify distances between pairs of locations different from those resulting from the road network.

4.2.10 Scenario of Test

These are going to be screen dumps Loading

Contextual Maps

Loading Database

Editing Database

Editing Constraints

Running the Scheduler

Selection the Scheduler Mode

Generation of a Solution

Running the Rescheduler

Selection of the Rescheduler Mode

Displaying the Results in a Map

Displaying the Results as Data

Displaying an Animation of the Results

Running the Petri Nets System

5. Conclusions and Recommendations

Van Ommeren Agencies have provided data for modelling and testing these problems. The data describe a problem of 94 transport requests for 536 containers to be moved from Rotterdam, for delivery, collection or positioning in a wide area covering Netherlands, Germany, Belgium, Spain and Switzerland. Table 2 shows an extract of the results obtained for the possible opportunities of triangulation and the cost saving to be made (Improved Distance Performance Column).

Triangulated Transport Requests	**Current Total Distance**	**Distance with Triangulation**	**Current Empty Running**	**Empty Running with Triangulation**	**Improved Level of Empty Running**	**Improved Distance performance**
R1-R40	20	20	10	10	50.00%	0.00%
R3-R43	1332	1104	666	438	39.67%	-17.11%
R4-R88	1248	778	624	154	19.79%	-37.66%
R5-R90	612	319	306	13	4.07%	-47.87%
R7-R64	656	529	328	201	37.9%	-19.35%
R10-R62	358	352	179	173	49.14%	-9.21%
R14-R47	854	700	427	273	39.00%	-18.03%
R15-R74	696	400	348	52	13.00%	-42.52%
R16-R26	228	210	112	96	45.71%	-7.89%
R17-R54	704	693	352	341	49.20%	-1.56%
R19-R58	178	132	132	54	40.09%	-25.84%
R21-R81	206	140	140	74	**67%**- 52%	-24.73%
R25-R37	770	522	385	137	26.24%	-32.20%
R27-R63	500	314	250	64	20.38%	-37.20%
R32-R46	482	374	241	133	35.35%	-22.40%
R34-R71	708	458	354	104	22.70%	-35.31%
R35-R70	1162	1061	581	480	45.24%	-8.69%
R36-R38	520	398	260	138	34.67%	-23.46%
R42-R72	184	156	92	64	41.02%	-15.21%
R44-R69	28	14	14	0	0.00%	-50.00%
R50-R79	1404	1276	702	574	44.9%	-9.11%
R52-R86	460	446	230	216	48.43%	-3.04%
R80-R93	148	148	74	74	50.00%	0.00%
Total	**13438**	**10544**	**6807**	**3863**	**36.66%**	**-21.53%**

Table 2: Van-Ommeren 's instances Results.

These results fulfil the benefits as outlined in an earlier section. It sums-up the results and lessons learnt from the project, emphasising the technology applied for the development of a routing scheduler. An integral aspect of the report has involved describing the vehicle routing problem, which is a key to improvements in logistics. A wide variety of VRPs exist across many economical areas, and the system proposed, deals with one of the most difficult instance of this class of problem: the Triangulation problem. It is a Vehicle Routing Problem with Time Window, multi-Port and multi-depot, handling side-constraints and suggesting intermodal transport alternative; and provides a rescheduling facility.

Environmental and economic benefits will result from an improved capability for solving Triangulation problems in industry. Recent technological advances in AI, coupled with general improvements in IT infrastructure in industry and society, might enable the development of high quality decision-support tools for solving the VRP.

PISCES incorporates a generic system that will allow rapid configuration of domain-independent and efficient Routing Planner systems. The underlying technological approach is a combination of constraint satisfaction and genetic algorithm, for combinatorial optimisation. Good results have already been demonstrated, both on standard VRPs benchmarks from the literature, and in data from the industrial partners on the project.

The Questionnaire Results Following The Demonstration Of The System To The Exploitation Board Members

The pilot system as used by Van Ommeren was demonstrated to members of the exploitation board; the response from the UK demonstrations was as follows:

	Exploitation Board Members		
Questions	**Lorival Plastics Ltd**	**Harrison Line Ltd**	**Mersey Dock And Harbour Company**
Do you feel that the system meets your company's information needs?	No, because the need for the level of detail does not currently exist. Transportation is less than 2% of my costs, exports is less than 10% of transport	Possibilities to use parts of the system or partner links. Would imagine that use of the system would grow in time – other partner relationships added as opportunities arise.	In some instances
Do you see the system as being easy to use?	Yes, subject to comments about loading 8MB and speed of interface- also, would need to be possible to customise reports	In time as access from the desktop to the Internet becomes readily available	Yes
What aspects are not clear to you ?	None	More clarification with regard to the security aspects required	The savings outlined Tip/ Load are totally dependent upon the market.
Do you think the system would reduce your costs?	Database – not significantly, scheduling has major possibilities	Certainly there are opportunities to examine possibilities. In particular with use of scheduler module.	Main benefits would be shipping cost and the interchange of equipment
What changes(if any) do you see are needed to the system?	Issue now is one of explosion of capabilities and take up of users, technology appears to be fixed – need to set the target market for set up.	The foundations of the system are in place. Would suggest wider trial of system required to determine this.	
Would you be interested in being involved in future European Commission sponsored projects?	Yes.	Possibly	Yes

6. Annexes

6.1 Presentations of the Project PISCES

6.1.1 Conferences

EMMSEC'99: Publication + Presentation + Demo. Prototype
WESIC'99: Publication + Presentation + Demo. Prototype
ES'99: Publication + Presentation + Demo. Prototype

6.1.2 Workshop

Open Day on Planning and Scheduling Planet 2000: *Presentation + Demo. Prototype*
Intelligent Planning and Scheduling Open Day organised by Group European PLANET (*EU Network of Excellence in Intelligent Planning and Scheduling*) and Centre for Virtual Environments, University of Salford, Salford, United Kingdom Manchester, June 19th, 2000.

6.1.3 Exhibition and Conference

TRL's Surface Transport 2000: Poster Demo. Prototype
Transport Exhibition and Conference organised by EPSRC, June 21 - 22, 2000 at TRL in association with Transport Research Laboratory. Old Wokingham Road Crowthorne, Berkshire.

6.2 Publications

"*Information Management within Intermodal Transport Chain Scheduling*"
F. Arshad, A. El Rhalibi, G. Kelleher. EMMSEC'99, June 22-24 1999, Stockholm - Sweden.

"Pipeline Intermodal System to support Control Expedition and Scheduling System" F. Arshad, A. El Rhalibi and G. Kelleher, 2nd International Workshop on European Scientific and Industrial Collaboration (WESIC'99, Promoting Advances Technologies in Manufacturing). Newport, Wales, UK , 1-3 September, 1999. editors: Prof. Geoff Roberts et al. : the Mechatronics Research Center.

"*Constraints and Genetic Algorithm to solve Transport Scheduling"*
F. Arshad, A. El Rhalibi, G. Kelleher. 19th SGES International Conference on Knowledge Based Systems and Applied Artificial Intelligence (ES'99). Peterhouse College, Cambridge, UK. December 13th-15th 1999. Editors: Prof. Robert Milne et al.: Intelligent Applications Ltd.

6.3 Deliverables

Information Support System in the form of an Online Database : The PISCES Database. Complete and operational.
Decision Support System in the form of a Routing Planner: PISCES Scheduler. Complete and operational.

6.4 Sample of Results for Triangulation

Triangulation for CSAV

==

(Next_job_what_j1_ = j40_)
Date 9/22/97 :
Transport of 2 loaded container(s) type CSAV 40',
for a delivery of kiwis,
from Rotterdam (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 0 km
Date 9/22/97 :
Transport of 2 empty container(s) type CSAV 40',
from Rotterdam (NL) to Dordrecht (NL) by truck, with a low priority
Travelled distance : 10 km
Date 10/1/97 :
Transport of 1 loaded container(s) type CSAV 40',
for a delivery of used clothing,
and of 1 empty container(s) type CSAV 40',
from Dordrecht (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 10 km

Empty running travelled distance with triangulation : 10 km

Total of travelled distance with triangulation : 20 km

Empty running travelled distance without triangulation : 10 km

Total of travelled distance without triangulation : 20 km

===

(Next_job_what_j9_ = j57_)
Date 9/29/97 :
Transport of 3 loaded container(s) type CSAV 40',
for a delivery of kiwis,
from Rotterdam (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 0 km
Date 9/29/97 :
Transport of 3 empty container(s) type CSAV 40',
from Rotterdam (NL) to Rotterdam (NL) by truck, with a high priority
Travelled distance : 0 km

===

(Next_job_what_j10_ = j62_)
Date 9/29/97 :
Transport of 1 loaded container(s) type CSAV 40',
for a delivery of frozen fruit,
from Rotterdam (NL) to Fijnaart (NL) by truck, with a low priority
Travelled distance : 20 km
Date 9/29/97 :
Transport of 1 empty container(s) type CSAV 40',
from Fijnaart (NL) to Wolvega (NL) by truck, with a low priority
Travelled distance : 173 km

Date 10/2/97 :
Transport of 1 loaded container(s) type CSAV 40',
for a delivery of frozen coffee,
and of 0 empty container(s) type CSAV 40',
from Wolvega (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 159 km

Empty running travelled distance with triangulation : 173 km

Total of travelled distance with triangulation : 352 km

Empty running travelled distance without triangulation : 179 km

Total of travelled distance without triangulation : 358 km

==

(Next_job_what_j17_ = j54_)
Date 9/30/97 :
Transport of 1 loaded container(s) type CSAV 40',
for a delivery of raspberries,
from Rotterdam (NL) to Bremerhafen (GER) by truck, with a low priority
Travelled distance : 338 km
Date 9/30/97 :
Transport of 1 empty container(s) type CSAV 40',
from Bremerhafen (GER) to Moerdijk (NL) by truck, with a high priority
Travelled distance : 341 km
Date 10/2/97 :
Transport of 1 loaded container(s) type CSAV 40',
for a delivery of used clothing,
and of 0 empty container(s) type CSAV 40',
from Moerdijk (NL) to Rotterdam (NL) by truck, with a high priority
Travelled distance : 14 km

Empty running travelled distance with triangulation : 341 km

Total of travelled distance with triangulation : 693 km

Empty running travelled distance without triangulation : 352 km

Total of travelled distance without triangulation : 704 km

==

(Next_job_what_j19_ = j58_)
Date 9/30/97 :
Transport of 1 loaded container(s) type CSAV 40',
for a delivery of frozen fruit,
from Rotterdam (NL) to Breuberg (GER) by truck, with a low priority
Travelled distance : 78 km
Date 9/30/97 :
Transport of 1 empty container(s) type CSAV 40',
from Breuberg (GER) to Antwerpen (B) by truck, with a low priority
Travelled distance : 54 km

==

(Next_job_what_j21_ = j81_)
Date 9/30/97 :
Transport of 1 loaded container(s) type CSAV 40',
for a delivery of kiwis,
from Rotterdam (NL) to Bleiswijk (NL) by truck, with a low priority
Travelled distance : 66 km
Date 9/30/97 :
Transport of 1 empty container(s) type CSAV 40',
from Bleiswijk (NL) to Antwerpen (B) by truck, with a high priority
Travelled distance : 74 km

==

(Next_job_what_j25_ = j37_)
Date 9/30/97 :
Transport of 1 loaded container(s) type CSAV 20',
for a delivery of wall/floor tiles,
from Rotterdam (NL) to Meinerzhagen (GER) by barge, with a low priority
Travelled distance : 221 km
Date 9/30/97 :
Transport of 1 empty container(s) type CSAV 20',
from Meinerzhagen (GER) to Hengelo (NL) by truck, with a low priority
Travelled distance : 137 km
Date 10/1/97 :
Transport of 1 loaded container(s) type CSAV 20',
for a delivery of sodiumchloroacetate,
and of 0 empty container(s) type CSAV 20',
from Hengelo (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 164 km

Empty running travelled distance with triangulation : 137 km

Total of travelled distance with triangulation : 522 km

Empty running travelled distance without triangulation : 385 km

Total of travelled distance without triangulation : 770 km

==

(Next_job_what_j38_ = j36_)
Date 10/1/97 :
Transport of 1 loaded container(s) type CSAV 20',
for a delivery of beans,
from Rotterdam (NL) to Amsterdam (NL) by truck, with a high priority
Travelled distance : 76 km
Date 10/1/97 :
Transport of 1 empty container(s) type CSAV 20',
from Amsterdam (NL) to Emmen (NL) by truck, with a high priority
Travelled distance : 138 km
Date 10/1/97 :
Transport of 1 loaded container(s) type CSAV 20',
for a delivery of yarns,
and of 0 empty container(s) type CSAV 20',
from Emmen (NL) to Rotterdam (NL) by truck, with a high priority

Travelled distance : 184 km

Empty running travelled distance with triangulation : 138 km

Total of travelled distance with triangulation : 398 km

Empty running travelled distance without triangulation : 260 km

Total of travelled distance without triangulation : 520 km

==

(Next_job_what_j50_ = j79_)
Date 10/2/97 :
Transport of 1 loaded container(s) type CSAV 20',
for a delivery of stones,
from Rotterdam (NL) to Basel (SW) by rail, with a high priority
Travelled distance : 518 km
Date 10/2/97 :
Transport of 1 empty container(s) type CSAV 20',
from Basel (SW) to Emmen (NL) by truck, with a low priority
Travelled distance : 574 km
Date 10/3/97 :
Transport of 1 loaded container(s) type CSAV 20',
for a delivery of yarns,
and of 0 empty container(s) type CSAV 20',
from Emmen (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 184 km

Empty running travelled distance with triangulation : 574 km

Total of travelled distance with triangulation : 1276 km

Empty running travelled distance without triangulation : 702 km

Total of travelled distance without triangulation : 1404 km

==

(Next_job_what_j52_ = j86_)
Date 10/2/97 :
Transport of 2 loaded container(s) type CSAV 20',
for a delivery of gluten,
from Rotterdam (NL) to Antwerpen (B) by truck, with a low priority
Travelled distance : 46 km
Date 10/2/97 :
Transport of 2 empty container(s) type CSAV 20',
from Antwerpen (B) to Emmen (NL) by truck, with a low priority
Travelled distance : 216 km
Date 10/3/97 :
Transport of 2 loaded container(s) type CSAV 20',
for a delivery of aramide wire,
and of 0 empty container(s) type CSAV 20',
from Emmen (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 184 km

Empty running travelled distance with triangulation : 216 km

Total of travelled distance with triangulation : 446 km

Empty running travelled distance without triangulation : 230 km

Total of travelled distance without triangulation : 460 km

===

(Next_job_what_j61_ = nil)
Date 10/2/97 :
Transport of 2 loaded container(s) type CSAV 40',
for a delivery of frozen chicken parts,
from Rotterdam (NL) to Rotterdam (NL) by rail, with a low priority
Travelled distance : 0 km
Date 10/2/97 :
Transport of 2 empty container(s) type CSAV 40',
from Rotterdam (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 0 km

===

(Next_job_what_j80_ = nil)
Date 10/3/97 :
Transport of 1 loaded container(s) type CSAV 20',
for a delivery of papayas,
from Rotterdam (NL) to De-Meern (NL) by truck, with a low priority
Travelled distance : 74 km
Date 10/3/97 :
Transport of 1 empty container(s) type CSAV 20',
from De-Meern (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 74 km

===

(Next_job_what_j83_ = nil)
Date 10/3/97 :
Transport of 1 loaded container(s) type CSAV 20',
for a delivery of papayas,
from Rotterdam (NL) to De-Meern (NL) by truck, with a low priority
Travelled distance : 74 km
Date 10/3/97 :
Transport of 1 empty container(s) type CSAV 20',
from De-Meern (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 74 km

===

(Next_job_what_j87_ = nil)
Date 10/3/97 :
Transport of 1 loaded container(s) type CSAV 40',
for a delivery of molybdenum,
from Rotterdam (NL) to Deventer (NL) by truck, with a low priority
Travelled distance : 141 km
Date 10/3/97 :

Transport of 1 empty container(s) type CSAV 40',
from Deventer (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 141 km

Triangulation for BOLT

==

(Next_job_what_j3_ = j43_)
Date 9/29/97 :
Transport of 1 loaded container(s) type BOLT Canada 40',
for a delivery of steel coils,
from Rotterdam (NL) to Basel (SW) by rail, with a low priority
Travelled distance : 518 km
Date 9/29/97 :
Transport of 1 empty container(s) type BOLT Canada 40',
from Basel (SW) to Uerdingen (GER) by truck, with a low priority
Travelled distance : 438 km
Date 10/2/97 :
Transport of 1 loaded container(s) type BOLT Canada 40',
for a delivery of chemicals,
and of 0 empty container(s) type BOLT Canada 40',
from Uerdingen (GER) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 148 km

Empty running travelled distance with triangulation : 438 km

Total of travelled distance with triangulation : 1104 km

Empty running travelled distance without triangulation : 666 km

Total of travelled distance without triangulation : 1332 km

==

(Next_job_what_j4_ = j88_)
Date 9/29/97 :
Transport of 2 loaded container(s) type BOLT Canada 40',
for a delivery of wire,
from Rotterdam (NL) to Anroechte (GER) by truck, with a low priority
Travelled distance : 265 km
Date 9/29/97 :
Transport of 2 empty container(s) type BOLT Canada 40',
from Anroechte (GER) to Gronau (NL) by truck, with a low priority
Travelled distance : 154 km
Date 10/4/97 :
Transport of 1 loaded container(s) type BOLT Canada 40',
for a delivery of machinery,
and of 1 empty container(s) type BOLT Canada 40',
from Gronau (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 359 km

Empty running travelled distance with triangulation : 154 km

Total of travelled distance with triangulation : 778 km

Empty running travelled distance without triangulation : 624 km

Total of travelled distance without triangulation : 1248 km

==

(Next_job_what_j5_ = j90_)
Date 9/29/97 :
Transport of 1 loaded container(s) type BOLT Canada 40',
for a delivery of plastic cd holders,
from Rotterdam (NL) to Dusseldorf (GER) by truck, with a low priority
Travelled distance : 158 km
Date 9/29/97 :
Transport of 1 empty container(s) type BOLT Canada 40',
from Dusseldorf (GER) to Uerdingen (GER) by truck, with a low priority
Travelled distance : 13 km
Date 10/4/97 :
Transport of 1 loaded container(s) type BOLT Canada 40',
for a delivery of harmless chemicals,
and of 0 empty container(s) type BOLT Canada 40',
from Uerdingen (GER) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 148 km

Empty running travelled distance with triangulation : 13 km

Total of travelled distance with triangulation : 319 km

Empty running travelled distance without triangulation : 306 km

Total of travelled distance without triangulation : 612 km

==

(Next_job_what_j7_ = j64_)
Date 9/29/97 :
Transport of 2 loaded container(s) type BOLT Canada 40',
for a delivery of steelwire,
from Rotterdam (NL) to Anroechte (GER) by truck, with a low priority
Travelled distance : 265 km
Date 9/29/97 :
Transport of 2 empty container(s) type BOLT Canada 40',
from Anroechte (GER) to s-Hertogenbosch (NL) by truck, with a high priority
Travelled distance : 201 km
Date 10/3/97 :
Transport of 2 loaded container(s) type BOLT Canada 40',
for a delivery of beer,
and of 0 empty container(s) type BOLT Canada 40',
from s-Hertogenbosch (NL) to Rotterdam (NL) by truck, with a high priority
Travelled distance : 63 km

Empty running travelled distance with triangulation : 201 km

Total of travelled distance with triangulation : 529 km

Empty running travelled distance without triangulation : 328 km

Total of travelled distance without triangulation : 656 km

==

(Next_job_what_j14_ = j47_)
Date 9/30/97 :
Transport of 1 loaded container(s) type BOLT Canada 20',
for a delivery of toys,
from Rotterdam (NL) to Bremerhafen (GER) by rail, with a high priority
Travelled distance : 338 km
Date 9/30/97 :
Transport of 1 empty container(s) type BOLT Canada 20',
from Bremerhafen (GER) to Gendt (NL) by truck, with a low priority
Travelled distance : 273 km
Date 10/2/97 :
Transport of 1 loaded container(s) type BOLT Canada 20',
for a delivery of furniture,
and of 0 empty container(s) type BOLT Canada 20',
from Gendt (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 89 km

Empty running travelled distance with triangulation : 273 km

Total of travelled distance with triangulation : 700 km

Empty running travelled distance without triangulation : 427 km

Total of travelled distance without triangulation : 854 km

==

(Next_job_what_j15_ = j74_)
Date 9/30/97 :
Transport of 2 loaded container(s) type BOLT Canada 20'/40',
for a delivery of aerosoles,
from Rotterdam (NL) to Wuppertal (GER) by truck, with a low priority
Travelled distance : 200 km
Date 9/30/97 :
Transport of 2 empty container(s) type BOLT Canada 20'/40',
from Wuppertal (GER) to Uerdingen (GER) by truck, with a high priority
Travelled distance : 52 km
Date 10/3/97 :
Transport of 1 loaded container(s) type BOLT Canada 20'/40',
for a delivery of brede,
and of 1 empty container(s) type BOLT Canada 20'/40',
from Uerdingen (GER) to Rotterdam (NL) by truck, with a high priority
Travelled distance : 148 km

Empty running travelled distance with triangulation : 52 km

Total of travelled distance with triangulation : 400 km

Empty running travelled distance without triangulation : 348 km

Total of travelled distance without triangulation : 696 km

==

(Next_job_what_j16_ = j26_)
Date 9/30/97 :
Transport of 1 loaded container(s) type BOLT Canada 40',
for a delivery of furniture/car,
from Rotterdam (NL) to Honselersdijk (NL) by truck, with a low priority
Travelled distance : 51 km
Date 9/30/97 :
Transport of 1 empty container(s) type BOLT Canada 40',
from Honselersdijk (NL) to s-Hertogenbosch (NL) by truck, with a low priority
Travelled distance : 96 km
Date 10/1/97 :
Transport of 1 loaded container(s) type BOLT Canada 40',
for a delivery of beer,
and of 0 empty container(s) type BOLT Canada 40',
from s-Hertogenbosch (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 63 km

Empty running travelled distance with triangulation : 96 km

Total of travelled distance with triangulation : 210 km

Empty running travelled distance without triangulation : 114 km

Total of travelled distance without triangulation : 228 km

===

(Next_job_what_j27_ = j63_)
Date 10/1/97 :
Transport of 1 loaded container(s) type BOLT Canada 20' dry,
for a delivery of wetsalted hides,
from Rotterdam (NL) to Meppel (NL) by truck, with a low priority
Travelled distance : 157 km
Date 10/1/97 :
Transport of 1 empty container(s) type BOLT Canada 20' dry,
from Meppel (NL) to Barneveld (NL) by truck, with a high priority
Travelled distance : 64 km
Date 10/3/97 :
Transport of 1 loaded container(s) type BOLT Canada 20' dry,
for a delivery of machinery,
and of 0 empty container(s) type BOLT Canada 20' dry,
from Barneveld (NL) to Rotterdam (NL) by truck, with a high priority
Travelled distance : 93 km

Empty running travelled distance with triangulation : 64 km

Total of travelled distance with triangulation : 314 km

Empty running travelled distance without triangulation : 250 km

Total of travelled distance without triangulation : 500 km

===

(Next_job_what_j29_ = nil)
Date 10/1/97 :

Transport of 1 loaded container(s) type BOLT Canada 20',
for a delivery of metalsheeting,
from Rotterdam (NL) to Woerden (NL) by truck, with a low priority
Travelled distance : 57 km
Date 10/1/97 :
Transport of 1 empty container(s) type BOLT Canada 20',
from Woerden (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 57 km

==

(Next_job_what_j31_ = nil)
Date 10/1/97 :
Transport of 1 loaded container(s) type BOLT Canada 20',
for a delivery of harmless all kinds,
from Rotterdam (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 0 km
Date 10/1/97 :
Transport of 1 empty container(s) type BOLT Canada 20',
from Rotterdam (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 0 km

==

(Next_job_what_j32_ = j71_)
Date 10/1/97 :
Transport of 1 loaded container(s) type BOLT Canada 40',
for a delivery of shrinkwristband,
from Rotterdam (NL) to Cruquius (NL) by truck, with a low priority
Travelled distance : 77 km
Date 10/1/97 :
Transport of 1 empty container(s) type BOLT Canada 40',
from Cruquius (NL) to Hengelo (NL) by truck, with a low priority
Travelled distance : 133 km
Date 10/3/97 :
Transport of 1 loaded container(s) type BOLT Canada 40',
for a delivery of sodium chloroacetate,
and of 0 empty container(s) type BOLT Canada 40',
from Hengelo (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 164 km

Empty running travelled distance with triangulation : 133 km

Total of travelled distance with triangulation : 374 km

Empty running travelled distance without triangulation : 241 km

Total of travelled distance without triangulation : 482 km

==

(Next_job_what_j34_ = j46_)
Date 10/1/97 :
Transport of 1 loaded container(s) type BOLT Canada 40',
for a delivery of wire,
from Rotterdam (NL) to Essen (GER) by rail, with a low priority

Travelled distance : 190 km
Date 10/1/97 :
Transport of 1 empty container(s) type BOLT Canada 40',
from Essen (GER) to Hengelo (NL) by truck, with a low priority
Travelled distance : 104 km
Date 10/2/97 :
Transport of 1 loaded container(s) type BOLT Canada 40',
for a delivery of sodium chloroacetate,
and of 0 empty container(s) type BOLT Canada 40',
from Hengelo (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 164 km

Empty running travelled distance with triangulation : 104 km

Total of travelled distance with triangulation : 458 km

Empty running travelled distance without triangulation : 354 km

Total of travelled distance without triangulation : 708 km

==

(Next_job_what_j35_ = j70_)
Date 10/1/97 :
Transport of 1 loaded container(s) type BOLT Canada 20',
for a delivery of bicyclesparts,
from Rotterdam (NL) to Basel (SW) by rail, with a low priority
Travelled distance : 518 km
Date 10/1/97 :
Transport of 1 empty container(s) type BOLT Canada 20',
from Basel (SW) to s-Hertogenbosch (NL) by truck, with a low priority
Travelled distance : 480 km
Date 10/3/97 :
Transport of 1 loaded container(s) type BOLT Canada 20',
for a delivery of beer,
and of 0 empty container(s) type BOLT Canada 20',
from s-Hertogenbosch (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 63 km

Empty running travelled distance with triangulation : 480 km

Total of travelled distance with triangulation : 1061 km

Empty running travelled distance without triangulation : 581 km

Total of travelled distance without triangulation : 1162 km

==

(Next_job_what_j41_ = nil)
Date 10/2/97 :
Transport of 1 loaded container(s) type BOLT Canada 20' dry,
for a delivery of purity phosphine,
from Rotterdam (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 0 km
Date 10/2/97 :

Transport of 1 empty container(s) type BOLT Canada 20' dry,
from Rotterdam (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 0 km

===

(Next_job_what_j42_ = j72_)
Date 10/2/97 :
Transport of 1 loaded container(s) type BOLT Canada 40' dry,
for a delivery of oilfield equipment,
from Rotterdam (NL) to Moerdijk (NL) by truck, with a low priority
Travelled distance : 14 km
Date 10/2/97 :
Transport of 1 empty container(s) type BOLT Canada 40' dry,
from Moerdijk (NL) to Rips (NL) by truck, with a low priority
Travelled distance : 64 km
Date 10/3/97 :
Transport of 1 loaded container(s) type BOLT Canada 40' dry,
for a delivery of furniture,
and of 0 empty container(s) type BOLT Canada 40' dry,
from Rips (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 78 km

Empty running travelled distance with triangulation : 64 km

Total of travelled distance with triangulation : 156 km

Empty running travelled distance without triangulation : 92 km

Total of travelled distance without triangulation : 184 km

===

(Next_job_what_j44_ = j69_)
Date 10/2/97 :
Transport of 2 loaded container(s) type BOLT Canada 20',
for a delivery of corrosive liq.,
from Rotterdam (NL) to Spijkenisse (NL) by truck, with a high priority
Travelled distance : 7 km
Date 10/2/97 :
Transport of 2 empty container(s) type BOLT Canada 20',
from Spijkenisse (NL) to Spijkenisse (NL) by truck, with a low priority
Travelled distance : 0 km
Date 10/3/97 :
Transport of 1 loaded container(s) type BOLT Canada 20',
for a delivery of ships spares,
and of 1 empty container(s) type BOLT Canada 20',
from Spijkenisse (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 7 km

Empty running travelled distance with triangulation : 0 km

Total of travelled distance with triangulation : 14 km

Empty running travelled distance without triangulation : 14 km

Total of travelled distance without triangulation : 28 km

==

(Next_job_what_j65_ = nil)
Date 10/3/97 :
Transport of 1 loaded container(s) type BOLT Canada 20' dry,
for a delivery of metal sheeting,
from Rotterdam (NL) to Woerden (NL) by truck, with a high priority
Travelled distance : 57 km
Date 10/3/97 :
Transport of 1 empty container(s) type BOLT Canada 20' dry,
from Woerden (NL) to Rotterdam (NL) by truck, with a low priority
Travelled distance : 57 km

==

(Next_job_what_j66_ = j89_)
Date 10/3/97 :
Transport of 1 loaded container(s) type BOLT Canada 40' dry,
for a delivery of foodstuffs,
from Rotterdam (NL) to Amsterdam (NL) by truck, with a high priority

7. References

[Arshad 1999] F. Arshad, A. El Rhalibi, G. Kelleher.
"*Information Management within Intermodal Transport Chain Scheduling*", EMMSEC'99, June 22-24 1999, Stockholm - Sweden.

[Arshad 1999] F. Arshad, A. El Rhalibi, G. Kelleher.
"*Constraints and Genetic Algorithm to solve Tansport Scheduling*". 19th SGES International Conference on Knowledge Based Systems and Applied Artificial Intelligence (ES'99). Peterhouse College, Cambridge, UK. December 13th-15th 1999.

[Assad 1988] Assad, Arjang A.
"*Modelling and Implementation Issues in Vehicle Routing*". Vehicle Routing: Methods and Studies. In B.L. Golden and A.A. Assad, Elsevier Science Publishers B.V. (North Holland), 1988.

[Bean 1991] Bean, J.C., J.R. Birge, J. Mittenthal, C.E. Noon,
"*Matchup Scheduling with Multiple Resources, Release Dates and Disruptions*", Operations Research 39(3):470-483, 1991.

[Bertsimas 1993] D. Bertsimas and Van Ryzin.
"Stochastic and Dynamic Vehicle Routing in the Euclidean Plane with Multiple Capacited Vehicles". Operations Research 41(1):60-76, 1993. Special Issue on Stochastic and Dynamic Models in Transportation.

[Blanton 1993] Blanton, J.L. and Wainwright, R.L., "*Multiple Vehicle Routing with Time and Capacity Constraints using Genetic Algorithms*", Proceeding of the 5th International Conference on Genetic Algorithms, Champain, IL, 1993, pp. 452-459.

[Burke 1989] Peter Burke, Patrick Rosser.
"*A Distributed Asynchronous System for Predictive and Reactive Scheduling*".
Technical Report AISL-42, Department of Computer Science, University of Strathclyde, Glasgow, UK. 1989.

[Carnegie Group 1995] Carnegie Group, Inc.
"*Commitment-Constrained Regeneration: An approach to reactive Replanning, and Rescheduling*". Pittsburg, PA, 1995, working paper.

[Davis 1990] Davis, L., "*The Handbook of Genetic Algorithms*", Van Nostrand Reinhold, 1990.

[De Kleer 1986] De Kleer, J.
"*An Assumption-Based Truth Maintenance System*". Artificial Intelligence 28:127-162, 1986.

[Doyle 1979] John Doyle.
"*A Truth Maintenance System*". Artificial Intelligence 12(3):231-272, 1979.

[Dumas 1989] Y. Dumas, J. Desrosiers and F. Soumis.
"Large-Scale Multi-Vehicle Dial-A-Ride Problems". Technical Report, Cahier du GERAD G-89-30, Ecoles des HEC, Montreal, Canada, 1989.

[Elleby 1988] P. Elleby, H.E. Fargher and T.R. Addis.
"Reactive Constraint-Based Job-shop Scheduling". Expert Systems and Intelligent Manufacturing. In Oliff, M.D., Elsevier Science Publishing Co., 1988, pages 1-10.

[Frost 1994] D. Frost, and R. Dechter.
"*In search of the best constraint satisfaction search"*. In Proceedings of the 12th National Conference of AAAI-94, Seattle, WA, pp. 301-306.

[Goldberg 1989] D. E. Goldberg :
"*Genetic Algorithms in Search, Optimization, and Machine Learning*". Addison–Wesley, Reading, Massachusetts, 1989.
[Golden 1987] Golden, B.L. and Assad, A.
"Perspectives on Vehicle Routing: Exciting New Developments". Operations Research 34. 1987.
[Haouari 1990] Haouari, M., P. Dejax and M. Desrochers
"*Modeling and Solving Complex Vehicle Routing Problems Using Column Generation*". Technical Report, Working Paper, Ecole Centrale de Paris, LEIS, 1990.
[Jaw 1986] J. Jaw, A. Odoni, H. Psaraftis and N. Wilson.
"A Heuristic Algorithm for the Multi-Vehicle Advance Request Dial-a-Ride Problem with Time Windows". Transportation Research 20B: 243-257, 1986
[Kelleher et al. 1994] G. Kelleher and J. Spragg.
"Time-Critical Rescheduling Using Truth Maintenance". Technical Report Truth.D43, commission of the European Communities ESPRIT III Program, Truth Consortium – Project 6463, 1994.
[Kelleher et al. 1994] G. Kelleher and P. Retif.
"Controlling Constraint-Based Scheduling Using Focussed RMS". In Proceeding of the IJCAI-93 Workshop on knowledge-Based Production Planning, Scheduling, and Control, pages 195-204. Chambery, France, August, 1993.
[Kelleher 1996] G. Kelleher and J. E. Spragg.
"A Discipline for Reactive Rescheduling". 3rd International Conference on Artificial Intelligence Planning Systems (AIPS-96), AAAI Press, Edinburgh, Scotland, 1996.
[Kelleher 1999] G. Kelleher, P. Cavichiollo.
"*Intelligent support of the Rescheduling of complex Manufacturing domains - an Example Application*". Proceeding of the 2nd International workshop on Intelligent Manufacturing Systems. Leuven, Belgium. September 22-24, 1999. pp257-267.
[Laarhoven 1992] Van Laarhoven, P.J. Aarts, E.H.L., and J.K. Lenstra.
"Job Shop Scheduling by Simulated Annealing". Operations Research 40(1):113-125, 1992.
[Lenstra et al. 1981] J. lenstra and Rinnooy Kan.
"Complexity of Vehicle Routing and Scheduling Problems". Networks 11:221-228, 1981
[Minton 1981] Steven Minton, Mark D.Johnston, Andrew B. Phillips, and Philip Laird.
"*Solving Large-scale constraint satisfaction and scheduling problems using a heuristic repair method*", AAAI-90 (Boston, MA), 1990, pp. 17-24.
[Miyashita 1994] Kazuo Miyashita and Katia Sycara.
"CABINS: A Framework of Knowledge Acquisition and Iterative Revision for Schedule Improvement and Reactive Repair". Technical Report CMU-R-TR-94-34, The Robotics Institute, Carnegie Mellon University, Pittsburgh. 1994.
[Nakakuki 1994] Nakakuki, Yoichiro, and Norman Sadeh.
"Increasing the efficiency of Simulated Annealing Search by Learning to Recognize (un)Promising Runs". In Proceeding of the Twelfth National Conference on Artificial Intelligence, 1994.
[Osman 1992] I.H. Osman

"Meta-Strategy Simulated Annealing and Tabu Search Algorithms for the Vehicle Routing Problem". Technical Report, Institute of Mathematics and Statistics, University of Kent at Canterbury, Kent CT2 7NF, K, 1992
[Potvin 1996] Potvin, J-Y. and Bengio, S.
"*The Vehicle Routing Problem with Time Windows Part II: Genetic Search*", Informis Journal on Computing, Vol. 8, No.2, Spring 1996.
[Psaraftis 1988] Psaraftis, Harilaos N.
"Dynamic Vehicle Routing Problems". Vehicle Routing: Methods and Studies. In B.L. Golden and A.A. Assad, Elsevier Science Publishers B.V. (North Holland), 1988.
[Psaraftis 1980] Psaraftis, H.N.
"*A Dynamic Programming Solution to the Single Vehicle, Many-to-Many, Immediate Request Dial-A-Ride Problem*". Transportation Science 14:130-154, August 1980.
[Rochat 1995] Rochat , Y. and Taillard, E.D.
"Probabilistic diversification and intensification in local search for vehicle routing". Journal of Heuristics, Vol. 1, No. 1, 1995, pp.147-167.
[Savelsbergh 1995] M. Savelsbergh , M. Sol:
"*The General Pickup and Delivery Problem"*. Transportation Science 29, 1995.
[Sadeh 1994] Norman M. Sadeh.
"*Micro-Boss: Toward a New Generation of Manufacturing Scheduling Shells*". In Proceedings of the ARPA/Rome Laboratory Knowledge-Based Planning and Scheduling Initiative. Tucson AZ. 1994.
[Smith 1992] Douglas R. Smith.
"Transformational Approach to Scheduling". Technical Report KES.U.92.2, Kestrel Institute, Palo Alto, CA 94304, 1992.
[Smith 1995] Smith, S.F. and O. Lassila.
"Flexible Coordination in Resource-Constrained Domains". Technical Report CMU-RI-TR-95-02, The Robotics Institute, Carnegie Mellon University, Pittsburgh.
[Solomon 1987] M. M. Solomon.
"*Algorithms for the vehicle routing and scheduling problem with time window constraints*". *Operations Research*, 35:254-265, 1987.
[Solomon et al. 1988] Solomon, M.M. and J. Desrosiers.
"Survey Paper: Time-Window Constrained Routing and Scheduling Problems". Transportation Science 22(1), 1988.
[Solomon url] M.M. Solomon, http://www.cba.neu.edu/~msolomon/problems.htm. "*VRPTW Benchmark Problems"*
[Taillard 1995] E. Taillard, P. Badeau, M. Gendreau, F. Guertain, and J.-Y. Potvin.
"*A new neighbourhood structure for the vehicle routing problem with time windows". Technical Report CRT-95-66*, Centre de Recherche sur les Transports, University of Montreal, 1995.
[Thangiah 1993] Thangiah, S.R. and al.
"Vehicle Routing Problems with time Deadlines using Genetic and Local Algorithms", 5th International Conference on Genetic Algorithms, University of Illinois at Urbana-Champaign, 1993, pp.506-513.
[Wilson 1977] Wilson and N. Colvin.
"Computer Control of the Rochester Dial-a-Ride System". Technical Report, Report R77-31, Dept of Civil Engineering, MIT, Cambridge, MA, 1977.

[Whitley 1991] Whitley, D., T. Starkweather and D. Shaner.
"The Traveling Salesman and Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination". Technical Report CS-91-111, Dept. of Computer Science, Colorado State University, 1991.
[Zweben 1991] Monte Zweben, Eugene Davis, and Michael Deale.
"Iterative Repair for Scheduling and Rescheduling". Technical Report NASA Ames Research Center, MS 244-17, Moffett Field, CA 94035, 1991.