Electronic invoicing in Brazil
Modifying an existing IT architecture to maintain compliance with Brazilian fiscal law

Student: Giovanni Martina
Student number: 1129228
Company: Gráfica Soset Indústria e Comércio Ltda.
Date: 08-10-2010
Document: Final Report

Faculty EWI
Delft University of Technology, October 2010
IN3405 Bachelor Project
Foreword

This document gives a complete project description, overview and final results obtained from work performed during the IN3405 Bachelor Project by student Giovanni Martina who is currently enrolled at the Faculty Electrical Engineering, Mathematics and Computer Science of the Delft University of Technology. Readers of this final report are assumed to have a general knowledge of Internet related technologies such as web services and XML at the minimum.

For those readers who are interested in the implementation of a client and server side technology that allows for digital invoicing in accordance with the latest Brazilian fiscal law, I happily refer you to chapter 5 of this report. Readers with an interest in the general architecture designed by the project development group of the Brazilian Digital Invoice should refer to chapter 4. And finally those who are interested in the long-distance cooperation between a Brazilian company with little time to comply with new Brazilian law and a student of the TU Delft should refer to chapter 3.

I would like to give my gratitude firstly to Mr. Joao Carlos Ferreira of Grafica Soset for his support and faith in me tackling a project that would have major legal repercussions should failure arise. Secondly I would like to thank Mr. Robson M. Pavlawski for his invaluable insights into Brazilian fiscal law, this project could not have been completed without his input. Thirdly I would like to thank Mr. Peter van Nieuwenhuizen for his input as supervisor and his understanding regarding delays that have arisen during the project’s implementation phase.
# Table of Contents

1. **Introduction** .......................................................................................................................... 6
2. **User Interaction** ...................................................................................................................... 9
   1. Login ........................................................................................................................................ 9
   2. Invoice ..................................................................................................................................... 11
5. **Project Scope** ....................................................................................................................... 12
   1. NF-e Invoicing Process ............................................................................................................. 12
   2. NF-e Web Services .................................................................................................................. 15
   3. Digital Invoice Authorization ................................................................................................. 18
   4. Testing Framework .................................................................................................................. 18

6. **Conclusion** ............................................................................................................................ 20
7. **Recommendations** .................................................................................................................. 21
8. **References** ............................................................................................................................. 21

---

**Appendix I** – Project Description and Approach ........................................................................ 23
1. **Introduction** ............................................................................................................................ 23
2. **Project Description** ................................................................................................................. 25
3. **Approach** ............................................................................................................................... 25
4. **Quality Assurance** .................................................................................................................. 27
5. **Release Schedule** ................................................................................................................... 27
6. **References** ............................................................................................................................... 27

**Addendum of 10-5-2010** ............................................................................................................ 27

**Appendix II** – Orientation Report............................................................................................... 29
1. **Introduction** ............................................................................................................................ 31
2. **IT Infrastructure & Process** .................................................................................................... 31
3. **Tools & Development** ........................................................................................................... 34
   1. List of resources ....................................................................................................................... 34
4. **References** ............................................................................................................................... 35

**Appendix III** – Requirements ..................................................................................................... 38
1. **Introduction** ............................................................................................................................ 38
2. **Requirements** .......................................................................................................................... 39
   1. Legal Requirements ............................................................................................................... 39
   2. User Requirements ................................................................................................................ 39
   3. Functional Requirements ....................................................................................................... 39
   4. Non-functional Requirements ................................................................................................. 39

**Appendix IV** – Architecture ....................................................................................................... 40
1. **Introduction** ............................................................................................................................ 40
2. **Data Model** ............................................................................................................................. 42
   1. Grails MVC ............................................................................................................................. 42
   2. Customer Model ..................................................................................................................... 43
   3. Product Model ......................................................................................................................... 43
   4. Invoice Model ......................................................................................................................... 45
   5. User Model .............................................................................................................................. 45
3. **Use Cases** .............................................................................................................................. 46
   1. SEFAZ Authorization ................................................................................................................ 46
   2. Backup ................................................................................................................................... 47
4. **User Interaction** ...................................................................................................................... 48
   1. Login ....................................................................................................................................... 48
   2. Invoice Emission ..................................................................................................................... 49
   3. Invoice Viewing ..................................................................................................................... 49

References ........................................................................................................................................... 49
1. Summary

Grafica Soset, a Brazilian producer of printed materials, is a company that needs to transition to the use of digital invoices for sale of goods and services. All Brazilian companies have set deadlines per type of industry, created by the Brazilian government, to transition from the use of paper invoices to the use of invoices that exist only in a digital form called the NF-e.

The NF-e standard as devised by Brazil consists of a set of web services that must be implemented by taxpayers in order to have their internal IT architecture create, digitally sign and securely transmit XML archives containing sales and tax information to the state-level Brazilian Tax Authorities.

The existing ERP system in use at Grafica Soset must be updated in order to comply with the new NF-e regulations. In order to achieve this a set of Open Source tools were chosen to form the basis of an implementation of the NF-e standard as mandated by Brazil. Specifically the foundation of the new implementation is the Grails framework that allows for the creation of Java-based Web Applications. The chosen toolset has led to the implementation of a software package that manages to read production data coming from Grafica Soset’s production database and display this information to users who can subsequently emit digital invoices. The created invoices are transformed into XML by the implementation and digitally signed with a X.509 certificate that has been obtained by Grafica Soset. The use of the digital certificate and the registration with the Brazilian Tax Authority of the state of Sao Paulo has allowed Grafica Soset to become an official emitter of digital invoices and as such the implemented software is able to produce these invoices and have them validated by the Brazilian Tax Authority, which results in their availability in the national Brazilian Digital Invoice repository, open for consult to all companies and private individuals for verification of the legal status of the sale of goods or services.

Due to time constraints it has not been feasible to implement the entire NF-e standard and future work should be focused on releasing new iterations of the delivered software. These iterations should implement the remaining web services as mandated by the Brazilian Tax Authorities, provide increased testing and code coverage, provide client-side testing and provide a means for remote archival of digital invoices.
2. Introduction

Brazil is a country that has in the past had to deal with major bouts of tax evasion. The reasons for which are many and contain among others stringent labor laws, high interest rates and the existence of hefty taxes. Brazilian business owners are quick to point out that the fiscal burden placed by the government has led to international smuggling and rampant tax evasion [1].

One aspect of new Brazilians laws passed in the year 2005 are to better enable the Brazilian government to combat tax evasion by requiring a process of digital invoicing regarding the sale of goods or services. Having set up a work group to decide how this is effectively done we fast forward a couple of years to reach our current point in time where the Digital Invoice Law or “Nota Fiscal Eletronica” is a reality for Brazilian companies large and small.

This Bachelor project aims to provide an implementation of the Digital Invoice specification set forth by the Brazilian government for a company located in the Sao Paulo region of Brazil. The company in question, Grafica Soset Industria e Comercio Ltda., is a mid-sized company specializing in the graphics arts. They produce a variety of printed materials for a huge number of clients scattered throughout the Sao Paulo region called “O Vale do Ribeiro” which is located in the south of the Brazilian state. Having had previous experience working for this company as a Java programmer the desire of having their own in-house Digital Invoice implementation came forth from management with a certain time-critical aspect.

Mr. Robson M. Pavlawski who performs accounting duties for the company has indicated that companies performing services in the graphics area have a hard deadline of October 1st, 2010 in order to transition from the older paper invoices to the digital invoices and failure to comply with this deadline will result in fines being applied per paper invoice emitted until such a transition has taken place. With this in mind there has been a certain pressure on the successful implementation of Digital Invoicing for Grafica Soset.

The transition from paper invoices to digital invoices is an elaborate process that involves the alteration of the existing IT infrastructure to support the Digital Invoice standard as set forth by the Digital Invoice working group in Brazil. This implies the addition of software that is able to connect to servers maintained by the Brazilian ministries of finance. The process of emitting paper invoices is composed of applying for official invoice numbers called AIDF numbers that are necessary for printing fiscal documents. On receipt of these numbers the company is allowed to print invoices on impact printers for delivery to its customers. With the advent of the digital invoice, authorization of emitting invoices through electronic means supersedes this process. What was previously a process not truly verifiable by the Brazilian ministries of finance is now a completely automated process based on open technologies such as web services, digital certificates and XML that in real-time checks the data of the invoice, validates the data against fiscal regulations, checks the status of the company and depending on the results of these checks authorizes the emission of the invoice. The invoice itself exists only in digital form but a human-readable form is available for attachment to shipping products.

Readers that would like to have additional background on the process around digital invoice emission in compliance with the new Nota Fiscal Eletronica law in all Brazilian states are referred to Appendix I. The remainder of this report contains detailed information about the implementation of digital invoices for Grafica Soset.

3. Project Organization

Reaching a consensus around the implementation of digital invoices for Grafica Soset requires the cooperation of several key-people. This chapter provides a detailed description of the organization of this project and the functions of the people involved.
3.1. International Teamwork and Project Stakeholders

I have dedicated my time on the project implementation with supervision and guidance provided by Mr. Joao Carlos Ferreira. The successful implementation of fiscal regulations has been achieved through cooperation with Mr. Robson M. Pavlawski who is an acting accountant with the Augusto Contabilidade Firm that handles accounting duties for Grafica Soset.

Regular contact through both e-mail and via calls over Skype have been performed to keep all parties up to date with the current state of the project and to inform each other about eventual problems that have arisen and the necessary implementation changes that needed to be made.

The guidance provided by Mr. Joao Carlos was focused on ensuring the resulting digital invoice implementation was easy to use by Grafica Soset's employees and correctly handles the fiscal regulations that currently apply to the company. As the various legislations are of complex nature and an implicit assumption of accounting knowledge is contained in the technical manual regarding implementation of the various open standard technologies, this project could never have been successfully embarked on without solid information provided by both Mr. Joao Carlos Ferreira and Mr. Robson M. Pavlawski.

Mr. Robson M. Pavlawski specifically provided his interpretation of the various fiscal regulations contained within the official implementation manual that apply to Grafica Soset and allowed me to correctly apply tax calculations to the products and services sold by Grafica Soset. He also successfully registered the company with the Brazilian ministry of finance for the state of Sao Paulo as an emitting entity of the digital invoice and obtained a digital certificate with one of the official certificate authorities in Brazil, all of which subsequently allowed me to start experimenting with the official test authorization servers maintained by SEFAZ¹.

In light of Mr. Joao Carlos Ferreira's position in the management of the company I have forwarded him an evaluation form available from the Internationalization Office at the Faculty Electrical Engineering, Mathematics and Computer Science of the TU Delft [2].

3.2. Revised Project Planning

Before implementation work started an estimate and planning was made and verified by Bachelor project supervisor for the TU Delft, Mr. Peter van Nieuwenhuizen. Readers interested in the actual planning made during the analysis phase of the project should refer to Appendix I. This planning however was not met due to unforeseen circumstances regarding the existing IT infrastructure’s incompatibilities with the digital invoice standard as mandated by the Brazilian ministries of finance. A revised planning is visible in figure 1.

¹ SEFAZ or Secretaria da Fazenda is a recognized abbreviation for Ministry of Finance. Each Brazilian state has its own assigned SEFAZ to deal with taxing regulations. As such the Ministry of Finance of the state of Sao Paulo could be referred to by SEFAZ-SP.
Figure 1 - Revised Planning for Project Tasks

Unforeseen circumstances included the necessity of altering various aspects of the existing Grafica Soset ERP program, specifically with regard to the registration of services and products (a description of the existing ERP program can be found in Appendix I).

In light of Mr. João Carlos Ferreira’s observation that the available product models do not contain vital information regarding the taxation of these same products, several changes were made to the existing ERP program that were not expected and not planned for. These changes include among others the addition of taxation codes as mandated by SEFAZ, the addition of a new subclass of product types to better comply with product classifications as listed by SEFAZ and the addition of different unit types for quantities of products.

3.3. TU Delft Coordination

Bi-weekly meetings were established between TU Delft project coordinator Mr. Peter van Nieuwenhuizen and myself in order to provide him with project status updates. Through these meetings I was able to demonstrate prototypes of the digital invoice application as it was being developed and receive feedback regarding the various pieces of project documentation that was created. After having received the OK on the following items (all of which are available as appendices to this final report) work on the project implementation was commenced.

- Project Description and Approach (Appendix I)
- Orientation Report (Appendix II)
- Requirements (Appendix III)
- Architecture (Appendix IV)

Project Description and Approach contains in-depth information on the Nota Fiscal Eletronica or Digital Invoice project that needs to be implemented for Grafica Soset and also showcases the current ERP system in use today.

The Orientation Report takes a look at how administrative employees at Grafica Soset currently perform paper invoicing and how this process needs to be updated for digital invoicing. It also provides descriptions of the chosen toolset for the implementation of a new digital invoicing system that can be used next to the already available ERP system.

We take a look at the functional, non-functional, legal and user requirements in the Requirements report. These requirements cover aspects of the implementation such as ease-of-use, compliance with SEFAZ regulations and others.

2 ERP stands for Enterprise Resource Planning. A computer based system with the purpose of facilitating the flow of information between all business functions inside an organization.
Finally the Architecture report gives an overview of the implementation design. Several data models and user interface mockups as well as use cases are provided to give insight into the technical design of the digital invoice implementation.

4. Project Scope

This chapter highlights the overall project scope pertaining to the successful emission of digital invoices in accordance with the digital invoicing standard as devised by the NF-e\textsuperscript{3} work group in Brazil.

The project scope has primarily been bounded by the legal requirements that have come about with regard to the emission of digital invoices and Grafica Soset's own requirements regarding a system that should be easy to use and tightly integrated with their existing ERP system.

4.1. NF-e Invoicing Process

Hard copy invoicing, the previous process of paper invoice emission as well as digital invoicing in compliance with NF-e regulations is described in Appendix I. For the sake of continuance the digital invoice process is outlined here with some additional details.

Digital invoicing is an electronic real-time process. Taxpayers’ computer systems electronically communicate over the public Internet via the exchange of SOAP XML\textsuperscript{4} messages with the computer systems of the designated state authority.

The taxpayer generates an XML document containing the description and taxation of products sold and services performed. This XML document must be in compliance with numerous regulations regarding what type of information and what type of taxation must be present as well as the format for the data within it. The committee that oversees the national NF-e project has publicly made these regulations and methods of operations available. The official Integration Manual, as this document is called, has not been attached to this report due to size constraints but is available for download [3].

The aforementioned XML document must be digitally signed before transmission in order to verify the sender’s identity. The standard chosen for digital signing of XML documents is the W3C Recommendation for Signature Syntax and Processing [4].

Apart from signing the XML document with a digital signature, a secure communication layer must be established during transmission of the XML document to again ensure the sender’s identity as well as the receiver’s identity. In order to achieve this secure communication layer the sender must establish an SSL\textsuperscript{5} connection with SEFAZ and validate the receiver’s digital certificate. The receiver, on connection establishment, should validate the sender’s digital certificate and only when both parties manage to validate each other’s identities is the transmission of the XML document allowed.

The digital certificates available to taxpayers are under the control of ICP-Brasil, which is the national authority on Brazilian Public Key Infrastructure [5]. Grafica Soset’s digital certificate was purchased through a licensed entity of ICP-Brasil by Mr. Antonio Carlos Ferreira, part of Grafica Soset’s management team.

On delivery of the XML document to SEFAZ a data validation is performed by SEFAZ and the XML document, or digital invoice, is verified according to NF-e regulations as stated by law. As such the properties of the documents are ensured to be valid and in compliance with the law. If validation could not be completed or if the status of the taxpayer is irregular, the digital invoice is rejected and not authorized for use. Whenever rejection occurs the invoice cannot be used and the sale of goods or services and its transport cannot proceed.

Three possible results could be returned from SEFAZ. Either the invoice is authorized for use, rejected due to some data entry error, XML Schema validation error or invalid signature or the invoice is denied due to irregularities regarding the taxpayer's fiscal condition. Denial of an invoice is a serious occurrence and indicates SEFAZ has in one way or another discovered that something is wrong with the taxpayer’s fiscal condition. This could occur with companies attempting tax evasion and is exactly the reason for the existence of

\textsuperscript{3} NF-e is a shorthand notation for Nota Fiscal Eletronica or Digital Invoice.  
\textsuperscript{4} SOAP is a protocol specification for exchanging structured information in the implementation of web services. It relies on Extensible Markup Language or XML for its message format.  
\textsuperscript{5} SSL or Secure Sockets Layer is a cryptographic protocol that provides security for communications over the Internet.
the NF-e project in the first place. Discovering fiscal irregularities is beyond the scope of this report and will not be explored any further.

On invoice authorization the release of goods is approved and the taxpayer may proceed with transport. During transport fiscal authorities may at any time verify the legal status of transport of goods and check if the transport was legally authorized through SEFAZ. In order to simplify this procedure a standard for a human readable form of the digital invoice has been established by the NF-e work group. This human readable form is called DANFE\(^6\) and should be printed on A4 paper. The DANFE contains the access code, which will be explained in subsequent chapters, of the digital invoice in CODE-128C barcode form [3]; this allows for the rapid consultation of the digital invoice through barcode scanning and verification of the existence on the national digital invoice repository [6]. A DANFE file as generated by the application is visible in figure 2.

![Generated DANFE form](image)

---

\(^6\) DANFE is an auxiliary human readable form of the digital invoice and should accompany the transport of goods.
The final stage of the invoicing process is enabling the availability of the generated and authorized invoice to the invoice recipients. As specified by NF-e law [3], the recipient of an invoice should have some form of access to the authorized invoice in order to register the invoice into his or her own proprietary systems. Several methods of access exist and the recommended approach by Mr. Joao Carlos Ferreira was the use of e-mail. As such a method had to be devised that would allow for sending of mail containing the authorized invoice for a recipient.

The preceding process description is visualized in Appendix I; please refer to figure 1 of Appendix I for a visualization of the complete NF-e process.

4.2. NF-e Web Services

Communication between a taxpayer’s existing IT programs and SEFAZ NF-e systems takes place through the use of web services adhering to the Basic Profile Version 1.1 as defined by the Web Services Interoperability Organization [7].

As mentioned before the message format as chosen by the NF-e work group is the SOAP protocol version 1.2 [3]. Each SEFAZ has their own server implementation of the NF-e specification and as Grafica Soset is located in the state of Sao Paulo, all communication takes place between Grafica Soset’s system and the NF-e server implementation of SEFAZ-SP.

The descriptions of the SOAP web services offered by SEFAZ-SP are available in [3]. Web services are offered in both a sandbox environment and true production environment in order to allow for testing and to enable a seamless integration with existing IT infrastructure. The project’s scope has been limited to the creation of digital invoices and the processes necessary to request digital invoice authorization from SEFAZ. As such not all web services have been implemented but a description of those that have and haven’t is provided for future reference.

The web services offered by SEFAZ are displayed in table 1.

<table>
<thead>
<tr>
<th>SOAP 1.2 Web Service</th>
<th>SEFAZ Implementation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NfeRecepcao2</td>
<td>Asynchronous</td>
<td>An asynchronous implementation of the NF-e authorization process</td>
</tr>
<tr>
<td>NfeRetRecepcao2</td>
<td>Asynchronous</td>
<td>An asynchronous implementation of the NF-e authorization process. This service is used to obtain the results of a call to NfeRecepcao2</td>
</tr>
<tr>
<td>NfeCancelamento2</td>
<td>Synchronous</td>
<td>A synchronous implementation of the NF-e cancellation process</td>
</tr>
<tr>
<td>NfeInutilizacao2</td>
<td>Synchronous</td>
<td>A synchronous implementation of the NF-e number cancellation process</td>
</tr>
<tr>
<td>NfeConsulta2</td>
<td>Synchronous</td>
<td>A synchronous implementation of the NF-e status request process</td>
</tr>
<tr>
<td>NfeStatusServico2</td>
<td>Synchronous</td>
<td>A synchronous implementation of the NF-e service status request process</td>
</tr>
<tr>
<td>NfeConsultaCadastro2</td>
<td>Synchronous</td>
<td>A synchronous implementation of the NF-e taxpayer information request process</td>
</tr>
</tbody>
</table>

The foremost services in table 1 are NfeRecepcao2 and NfeRetRecepcao2. They are the SOAP web services that need to be called in order to authorize digital invoices. As can be seen from table 1 they are the only services that have an asynchronous implementation. This implies the receipt of a service acknowledgement from SEFAZ that must be used to obtain the result of processing at a future moment in time. These services have been implemented and are the one with the greatest priority.

NfeCancelamento2 is a SOAP web service that needs to be called when previously authorized digital invoices need to be cancelled due to data entry errors or due to customer requests. It is a synchronous service, which implies the receipt of the result of processing after the service call has been completed.

NfeInutilizacao2 is a SOAP web service designated for the cancellation of ranges of invoice numbers. NF-e law mandates the use of a sequence of invoice numbers ranging from 1 to 999,999,999. Whenever a sequence gets out of synch due to communication errors or depending on the implementation of taxpayers’ IT systems, a notice should be sent to SEFAZ to inform them of non-usage of specific invoice numbers or a range of

\[7\] SEFAZ-SP is shorthand notation for the state level ministry of finance in the state of Sao Paulo, Brazil.
invoice numbers. Implementation of this service is a future task not applicable to the current scope of the Bachelor’s project.

NfeConsulta2 is a SOAP web service designated for the consultation of a specific digital invoice. Being a synchronous service it directly returns the result of processing to the calling service. As invoices can be directly consulted through either the sandbox or production environments of each SEFAZ the implementation of this service is not seen as a priority and thus doesn’t apply to the current scope of the project.

NfeStatusServico2 is a SOAP web service designated for the consultation of the status of the NF-e authorization servers. On request this synchronous service directly answers with the current status of servers that are responsible for processing digital invoices, indicating whether they are offline or online and available for service requests. The implementation of this service is part of the project’s scope.

NfeConsultaCadastro2 is a synchronous SOAP web service that allows for information requests regarding taxpayer information such as registered business name, tax number, address etc. Such a service is useful for the verification of Grafica Soset’s existing customer data with data registered at SEFAZ, allowing for less errors on invoice authorization due to mismatching records or data entry errors by employees. This service is not part of the current project’s scope and thus will be implemented at some future date.

5. Design and Testing Implementation

The project architecture that was devised during the analysis phase of this project has been implemented and tested according to the documentation available in Appendix IV of this report. Some changes during implementation however were unavoidable and will be elaborated upon in this chapter. This chapter will also go into further detail on the actual implementation of digital invoicing for Grafica Soset along with the testing procedures used to validate the implementation code.

5.1. Grails Framework

As a departure from the existing ERP system in use at Grafica Soset, which is currently seen as a difficult to maintain system, it has been decided to provide the NF-e invoicing application as a browser-based application that relies on a combination of client side HTML, JavaScript and CSS code in order to provide for the user interface. The server side code is running a combination of Java and Groovy\(^8\) code running in a Jetty 6.1 HTTP Server and Java Servlet Container.

The underlying framework that the application is based on is the open-source Grails framework version 1.3.4. The grails framework is described as a full-stack framework that attempts to solve many pieces of the web development puzzle. It includes open-source packages such as Hibernate, Spring Model-View-Controller, Spring Dependency Injection and many others [8]. Any readers interested in learning more about Grails are referred to [8].

Another aspect of using the Grails framework is the notion of working by “convention”. Grails conventions could be described as best practices in development and when used simplify the operational model of the architecture. During this project Grails conventions were used as much as possible and as such we can list the code packages that have been created in line with Grails conventions in table 2.

<table>
<thead>
<tr>
<th>Grails Convention</th>
<th>Classes</th>
<th>Class Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain Model</td>
<td>net.drayah.poderoso.Accountant.groovy</td>
<td>Model for an accountant object that handles fiscal regulations for customers</td>
</tr>
<tr>
<td></td>
<td>net.drayah.poderoso.CFOP.groovy</td>
<td>Model for a fiscal operation adhering to the Brazilian standard for taxation of products</td>
</tr>
<tr>
<td></td>
<td>net.drayah.poderoso.Contact.groovy</td>
<td>Model for customer contact people</td>
</tr>
<tr>
<td></td>
<td>net.drayah.poderoso.Customer.groovy</td>
<td>Model for a customer or a recipient of a digital invoice</td>
</tr>
</tbody>
</table>

\(^8\) For those unfamiliar with it, Groovy is a dynamically typed object-oriented programming language running on the Java Virtual Machine with features similar to Python, Ruby and Smalltalk.
<table>
<thead>
<tr>
<th>Package Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>net.drayah.poderoso.IBGE.groovy</td>
<td>Model for Brazilian geographical data e.g. city, state and district information. In compliance with governmental standards</td>
</tr>
<tr>
<td>net.drayah.poderoso.Location.groovy</td>
<td>Model for a company location or a sender of a digital invoice</td>
</tr>
<tr>
<td>net.drayah.poderoso.Salesperson.groovy</td>
<td>Model for a salesperson that delivers merchandise to customers</td>
</tr>
<tr>
<td>net.drayah.poderoso.Sequence.groovy</td>
<td>Model for a sequence of invoice numbers ranging from 1 to 999999999</td>
</tr>
<tr>
<td>net.drayah.poderoso.Serie.groovy</td>
<td>Model for a unique series of invoice numbers. Each series contains one sequence ranging from 1 to 999999999</td>
</tr>
<tr>
<td>net.drayah.poderoso.User.groovy</td>
<td>Model for a user that has access to the software</td>
</tr>
<tr>
<td>net.drayah.poderoso.billing.Invoice.groovy</td>
<td>Model for an invoice</td>
</tr>
<tr>
<td>net.drayah.poderoso.billing.InvoiceItem.groovy</td>
<td>Model for item descriptions that belong to an invoice</td>
</tr>
<tr>
<td>net.drayah.poderoso.billing.InvoicePriceInstallment.groovy</td>
<td>Model for payment installments belonging to an invoice</td>
</tr>
<tr>
<td>net.drayah.poderoso.billing.InvoiceProduct.groovy</td>
<td>Model for products that are related to an invoice</td>
</tr>
<tr>
<td>net.drayah.poderoso.products.Category.groovy</td>
<td>Model for a category of products</td>
</tr>
<tr>
<td>net.drayah.poderoso.products.CategoryTypeMembership.groovy</td>
<td>Model for a membership relation between types of products and categories of products</td>
</tr>
<tr>
<td>net.drayah.poderoso.products.PrintProduct.groovy</td>
<td>Model for printed products sold such as leaflets</td>
</tr>
<tr>
<td>net.drayah.poderoso.products.StampProduct.groovy</td>
<td>Model for stamp products sold</td>
</tr>
<tr>
<td>net.drayah.poderoso.products.PrintProductType.groovy</td>
<td>Model for types related to a printed product</td>
</tr>
<tr>
<td>net.drayah.poderoso.products.ProductModel.groovy</td>
<td>Model for different product models</td>
</tr>
<tr>
<td>net.drayah.poderoso.products.ProductType.groovy</td>
<td>Model for different product types</td>
</tr>
<tr>
<td>net.drayah.poderoso.products.Stamp.groovy</td>
<td>Model for stamps</td>
</tr>
<tr>
<td>net.drayah.poderoso.products.StampProductType.groovy</td>
<td>Model for types related to a stamp product</td>
</tr>
<tr>
<td>net.drayah.poderoso.products.Unit.groovy</td>
<td>Model for unit of quantities</td>
</tr>
</tbody>
</table>

It is evident from this table of domain classes and the data model given in Appendix IV that some classes have been added and some have been removed from those classes proposed during the analysis phase.

The addition of classes has taken place due to unforeseen circumstances regarding changes requested by Grafica Soset and the necessity of encoding information in new data models as required by the invoicing process. Such an example is the addition of the IBGE data model, which is a class that deals with Brazilian geographical information on a national level. This turned out to be a required addition to the data model due to the fact that the invoicing process requires standardized city codes as decided by the Brazilian government.

Some classes need not be implemented during this project because their information is not necessary for successful invoice authorization. An example of such class removal is the Ink class that handles properties related to ink and colors for printed products, this technical information is not needed for creation of a digital invoice.

The classes listed in table 2 have been implemented and their data is saved to a PostgreSQL database through Grails. Data models on their own however are not sufficient and further packages, again adhering to Grails conventions, need to be provided to provide for a working implementation. Controllers are classes that handle the specific actions specified by user requests made via the user interface (UI). Table 3 lists the implemented controllers for this project.
Specific logic regarding digital invoice creation and authorization is handled by a set of classes in the services package, which is another Grails convention. Service classes are groups of code that logically define interrelated behavior for achieving some specific purpose. The placement of this code fits neither in the domain model of the application nor in the controller package and thus has its own location specified in the services package. The majority of logic required for achieving successful invoice authorization has been implemented in the InvoiceService class and it, in cooperation with some extra utility classes, is the primary logical unit that deals with SSL authentication, communication over HTTP, sending and receiving of SOAP messages, creating and signing of XML and generating of PDF documents amongst other things. Implemented service classes are specified in table 4.

### Table 3 - Grails Controllers

<table>
<thead>
<tr>
<th>Grails Convention</th>
<th>Classes</th>
<th>Class Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controller</td>
<td>net.drayah.poderoso.controllers.AppController.groovy</td>
<td>General controller handling application wide user actions</td>
</tr>
<tr>
<td></td>
<td>net.drayah.poderoso.controllers.InvoiceController.groovy</td>
<td>Controller handling all user actions regarding the creation and authorization of digital invoices</td>
</tr>
<tr>
<td></td>
<td>net.drayah.poderoso.controllers.UserController.groovy</td>
<td>Controller handling user login and logout requests</td>
</tr>
</tbody>
</table>

### Table 4 - Grails Services

<table>
<thead>
<tr>
<th>Grails Convention</th>
<th>Classes</th>
<th>Class Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Services</td>
<td>net.drayah.poderoso.services.AppService.groovy</td>
<td>Provides business logic for creating and registering customers, products and other domain models</td>
</tr>
<tr>
<td></td>
<td>net.drayah.poderoso.services.InvoiceService.groovy</td>
<td>Provides methods required for proper NF-e invoicing via SEFAZ-SP. Contains business logic for sending and receiving SOAP messages</td>
</tr>
<tr>
<td></td>
<td>net.drayah.poderoso.services.SecurityService.groovy</td>
<td>Provides cryptographic functions</td>
</tr>
</tbody>
</table>

Remaining classes that do not naturally fit in the domain model, controller or service packages are listed in table 5. The class implementation languages vary between Java and Groovy and include utility classes that are able to generate XML messages and other low level functionality such as sending data over secured communication channels using SSL. The majority of these utility classes are used by the InvoiceService class in order for it to successfully send and process authorization requests for digital invoices.

### Table 5 - Utility Classes

<table>
<thead>
<tr>
<th>Class Type</th>
<th>Classes</th>
<th>Class Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag Library</td>
<td>net.drayah.poderoso.taglibs.PoderosoTagLib.groovy</td>
<td>Contains utility methods such as date formatting and currency formatting functions</td>
</tr>
<tr>
<td>Java Utility Code</td>
<td>net.drayah.poderoso.reports.DanfeReporter.java</td>
<td>A Java utility class used for generating the DANFE form in PDF format</td>
</tr>
<tr>
<td></td>
<td>net.drayah.poderoso.ssl.KeyStoreProvider.java</td>
<td>Utility class for dealing with Java Key Stores containing X.509 digital certificates</td>
</tr>
<tr>
<td></td>
<td>net.drayah.poderoso.ssl.SSL.java</td>
<td>Utility class used for maintaining SSL connections to remote hosts</td>
</tr>
<tr>
<td></td>
<td>net.drayah.poderoso.xml.XMLSigner.java</td>
<td>Utility class that is able to digitally sign XML data by use of a certificate</td>
</tr>
<tr>
<td></td>
<td>net.drayah.poderoso.xml.XMLValidator.java</td>
<td>Utility class that is able to validate XML data according to the NF-e Schema Validation available from [3]</td>
</tr>
<tr>
<td>Groovy Utility Code</td>
<td>net.drayah.poderoso.xml.XMLNFe.groovy</td>
<td>XML representation of a digital invoice</td>
</tr>
<tr>
<td></td>
<td>net.drayah.poderoso.xml.XMLNFeItem.groovy</td>
<td>XML representation of a product or service item that belongs to a digital invoice</td>
</tr>
</tbody>
</table>
|                      | net.drayah.poderoso.xml.XMLNFeProc.groovy | XML representation of a digital invoice that has been
5.2. Digital Invoice Authorization

As mentioned earlier, the weight of this project lays on successful digital invoice authorization. The implementation code that allows for authorization of invoices has been primarily delivered with the InvoiceService class that, in conjunction with utility classes, manages to connect to SEFAZ-SP, deliver a signed XML message via SOAP and verify the authorization result at a later moment in time. This sequence of events is shown in figures 3 and 4.

![Figure 3 - Invoice emission and Authorization Request](image-url)
Whenever the user has logged into the system and chooses products to invoice, InvoiceService performs the following sequence of events in conjunction with InvoiceController and several utility classes.

- Pre-validate the passed parameters for invoice creation. This makes sure non-nullable properties are properly set and no inconsistent data is present amongst other things.
- Persist the invoice data to the underlying database.
- Transform the invoice data model to XML format for SEFAZ processing and digitally sign the generated XML with the provided digital signature.
- Validate the transformed XML to the NF-e XML Schemas as provisioned by SEFAZ.
- Create a XML SOAP message and transmit the transformed invoice XML to SEFAZ via a call to the NfeRecepcao2 web service.

SEFAZ does not directly return a result of processing and thus a later call has to be made to the NfeRetRecepcao2 SEFAZ web service in order to verify the status of the authorization request. The following sequence of events takes place.

- SefazJob periodically spawns a background thread to check for existing invoice authorization requests that have been sent. It returns the oldest of these to InvoiceService for further processing.
- InvoiceService creates a XML SOAP message and transmits a request for status of processing to SEFAZ.
- InvoiceService updates the underlying invoice with the result of processing by SEFAZ.

All invoices can have up to one emission state that gets updated depending on the result of processing by SEFAZ. The possible invoice states we can distinguish are given in the code fragment visible in figure 5.

```java
//Possible invoice states
private final static String INVOICE_PENDING = "PENDING"
private final static String INVOICE_AUTHORIZED = "AUTHORIZED"
private final static String INVOICE_REJECTED = "REJECTED"
private final static String INVOICE_DENIED = "DENIED"
private final static String INVOICE_CANCELLED = "CANCELLED"
```

Whenever an invoice gets persisted to the database it will have an initial “PENDING” state that indicates the invoice has not been authorized and will be returned by SefazJob for further processing.

An invoice that has been successfully authorized by SEFAZ will have its state set to “AUTHORIZED” and will not be returned by SefazJob for further processing.

Invoices that have been rejected by SEFAZ due to inconsistencies with regard to the data they contain will be rejected and thus not persisted to the national Brazilian repository of digital invoices. Their state gets set to “REJECTED” to indicate the user of this occurrence.

Invoices may be denied by SEFAZ if the taxpayer’s status is irregular. Invoices that have been denied may not be used and law prohibits the transport of these goods. Invoices have their state set to “DENIED” should the preceding occur.
Finally, an invoice could previously be authorized by SEFAZ but for whatever reason may be cancelled by the taxpayer. Whenever such an occurrence takes place the invoice status should be set to “CANCELLED”.

After successful invoice authorization has taken place the invoice can be viewed in the national Brazilian repository of digital invoices through use of the invoice access code [6]. The form of the access code has been decided by the NF-e work group and consists of the following items.

- State code (2 digits)
- Year and month of emission (4 digits)
- Taxpayer number (14 digits)
- Invoice model (2 digits)
- Invoice series (3 digits)
- Invoice number (9 digits)
- Invoice emission type (1 digit)
- Random number (8 digits)
- Verifier digit (1 digit)

The access code is thus formed of 44 digits, the last of which being a verifier to ensure the correctness of the preceding 43 digits. The following modulo-11 based algorithm performs the calculation of the verifier digit.

\[
Multipliers = [2,3,4,5,6,7,8,9]\\
\text{Set } index, sum = 0\\
\text{For } digit \text{ in } accesscode^{\text{reversed}}\\
\text{If } index > 8 \text{ Then } index = 0\\
\text{sum} += digit \times Multipliers[index]\\
\text{index}++\\
\text{End For}\\
\text{Verifier} = 11 - (\text{sum}\%11) \text{ or } 0 \text{ if } \text{sum}\%11 < 2
\]

The verifier digit calculation has been implemented in InvoiceService as shown in figure 6.

```java
def calculateVerifierDigit(number) {
    //pre: number: the 43 digits of an NFe access code as a String
    //post: calculates and returns the 44th verifier digit that needs to be appended to
    //      the end of the NFe access code
    def multipliers = [2, 3, 4, 5, 6, 7, 8, 9]
    def reversed = number.reverse()
    def position = 0
    def sum = 0
    for (digit in reversed) {
        sum += Integer.parseInt(digit) * multipliers[position]
        //increment or cycle the current position
        position++
        if (position == multipliers.size())
            position = 0
    }
    return (sum % 11) < 2 ? 0 : 11 - (sum % 11)
}
```

Figure 6 - Verifier digit implementation

As was shown earlier, an 8 digit random number must be generated as part of the access code. The implementation of which is shown in figure 7.

```java
def generateRandomCode() {
    //post: returns a randomly generated number of size 8 (between 10000000 and 99999999)
    //      for usage with NFe's
    int multiplier = Math.random() * 360
    def random = new Random((new Date()).hashCode() * multiplier)
    int code = (random.nextDouble() * 89999999) + 10000000
    return code
}
```

Figure 7 - Random code generation
5.3. Testing Framework

The majority of the server side code dealing with invoice authorization has been thoroughly tested through use of the JUnit framework. Readers can refer to Appendix V for a complete listing of test results.

The tests are split between tests on the unit level and integration tests that simulate a complete NF-e authorization sandbox. Tests on the unit level verify data integrity for individual classes whereas integration tests verify more complex interactions taking place within the system.

The sandbox environment maintained by SEFAZ was used to provide a complete working implementation of the NF-e authorization process and the tests created ensured the validity of digital certificates, creation of SSL connections, emissions of XML data via SOAP and correct processing and interactions of methods within the InvoiceService class and various utility classes. This has enabled Grafica Soset to test the invoice emission process in a complete safe environment that would not in any way negatively interfere with existing production data.

After user testing in the sandbox environment the transition to production has taken place and invoices could successfully be emitted and authorized without failure. The use of testing and the sandbox environment has ensured a smooth transition to the NF-e invoicing process for Grafica Soset and the use of the testing framework is surely recommended in any future endeavors.

6. Conclusion

The project goal of having an implementation of the NF-e invoicing process has been successfully accomplished. The system is currently in use at Grafica Soset. Invoices are successfully being emitted, signed and authorized by SEFAZ-SP. DANFE forms are successfully being printed and digital invoices are successfully being e-mailed to Grafica Soset’s clients.

The successful implementation of this project was a necessity as there would be various serious legal repercussions if Grafica Soset could not commence the NF-e invoice authorization process on the first of October 2010. Each specific industry has had a deadline set by the Brazilian government to transition from the use of paper invoices to digital invoices and the deadline established for Grafica Soset was digital invoice emission on October 1st, 2010.

Due to the successful application of the Grails toolset and JUnit testing the transition from paper invoicing to digital invoice testing to digital invoice emission in production was achieved without major technical difficulties and as such the project has been declared a success by the management of Grafica Soset, who are eagerly envisioning the continued work and expansion on the currently available system.

With the main objective attained, we can now take a look at the established requirements in Appendix III in order to provide an evaluation on which requirements have been achieved and which could possibly be achieved in future releases of the system.

Authorized invoices are electronically archived in Grafica Soset’s production database. But 5-year availability of invoices is not guaranteed in the event of calamities such as fire or earthquakes. In order to guarantee 5-year availability a remote backup system has to be implemented. Authorized invoices are electronically sent to customers and DANFE forms accompany product shipments. Invoices are checked against SEFAZ XML Schemas to provide compliance with the NF-e standard. Grafica Soset is able to authorize invoices in the production environment and is officially licensed by SEFAZ to do so.

Major requirements have indeed been achieved during this project with exception of guarantees regarding data availability. After evaluation of user requirements we can conclude that the delivered invoicing process is easy in use due to the small amount of user input that is required and due to the automatic validations taking place. Because the project has been developed upon a stable toolset such as Grails, PostgreSQL and JUnit it is extendable. The system also simplifies the workflow of users in comparison with other tools due to the automatic processes regarding invoice authorization and invoice authorization results processing.

Remaining functional and non-functional requirements have all been achieved with the exception of one or two cases that deal with remote backup of invoices and implementation of other NF-e web services.

7. Recommendations

The complete NF-e standard as explained in [3] has a larger scope than what could be implemented during the duration of the Bachelor’s project. Several enhancements and complete implementation of the NF-e standard as
described in [3] are desirable and as such Grafica Soset has requested the continuing development of the currently released production system.

Remaining web services to be completed in future iterations of the software are the NfeInutilizacao2 web service for invoice number cancellations, NfeCancelamento2 for complete invoice cancellations, NfeConsulta2 for more complete invoice consultation reports and NfeConsultaCadastro2 for allowing customer data to be verified against data known to SEFAZ. Use of NfeConsultaCadastro2 would allow the existing system to alert users on data entry errors pertaining to the internal customer data such as customer tax numbers and address information. The implementation of the other web services is highly desirable and will be worked on even after completion of the Bachelor project.

The desired requirement of 5-year archival through remote backups has not been achieved due to time constraints even after having explored this in the Analysis phase of the project. As such the architecture will have to be revisited in future system iterations in order to devise a suitable solution to the problem. Proposals were made during the Analysis phase of the project towards using either Google or Amazon services but more profound analysis needs to be done on how a successful implementation would work using either of these services. Seeing as the requirement primarily lies on remote archival of XML files, a solution based on a storage based files service such as Amazon S3 [9] would seem to be appropriate. But this has to be properly researched before being chosen as a definite solution.

I myself have other concerns that are not directly visible to the users but are more questions of maintainability and testability of the system. Even though server side code is to a certain extent being covered by unit and integration tests, no code coverage mechanism is being used and thus it is hard to conclude what percentage of implementation code is actually being covered by tests. Use of a code coverage plugin would be advisable in order to, without a doubt, know the amount of code coverage of the tests. This would also allow developers to write tests for parts of the code that aren’t fully covered and would result in less bugs going undiscovered. With that in mind it must also be noted that client side JavaScript and HTML code is not being covered at all and usage of a tool such as Selenium [10] comes to mind that would allow the client side user interface to be tested via functional tests.
References


NFE Project Brazil

Author: Giovanni Martina
Date: 27-04-2010
Modification Date: 04-05-2010
Modification Date: 10-05-2010
Document: Project Description and Approach

Delft University of Technology, May 2010
IN3405 Bachelor
Table of Contents

1. Introduction ........................................................................................................................................... 23
2. Project Description ................................................................................................................................. 23
3. Approach .................................................................................................................................................. 25
4. Quality Assurance ................................................................................................................................. 27
5. Release Schedule .................................................................................................................................. 27
6. References ............................................................................................................................................... 27
Addendum of 10-5-2010 ............................................................................................................................. 27
Appendix 1A – Release Schedule ............................................................................................................... 28
1. Introduction

The ongoing changes in Brazilian fiscal law have brought about some interesting technological developments. Under the new Nota Fiscal Electrónica (NF-e) law the Brazilian government requires companies to implement the use of digital invoices into their daily business operations. Gradually companies large and small will be required to eliminate hard copy invoices and start using the electronic version. This electronic invoice must comply exactly with the specified XML format as mandated by the Brazilian government.

The electronic invoice is deeply integrated into business operations because the law mandates a particular usage scenario regarding the delivery and sale of goods. A typical scenario consists of the release of goods. Originally it would have been possible to release the goods and offer them to the client along with a hard copy invoice issued by the Brazilian Ministry of Finance (SEFAZ). This hard copy paper would indicate the value of the goods for sale and other peculiarities dealing with the taxation of goods. In the new situation a company would have to first transmit a digital invoice to SEFAZ for review and only when approved would that company be allowed to deliver the goods. After approval the digital invoice is transmitted back to the company for archival and by law the archiving of digital invoices must be guaranteed up to at least five years. Once archived the digital invoice is sent on its way to the receiver of the goods for tracking purposes and in order to enable the receiver to become aware of the goods to be acknowledged. Finally the invoice must also be converted into a human readable format called the DANF-e. This format can be printed as a hard copy and must accompany the shipment of goods. At any point during shipment customs officials who can scan the barcode on the DANF-e to validate its existence may hold up the goods to verify that the government has sanctioned the movement of goods.

The NF-e law has ushered Brazil forwards into a leading position within the global electronic invoicing communities, as it is unique in requiring all companies to comply with the new legislation. Such electronic means of controlling the flow of goods is not seen in the EU for example. This development has lead to a situation where a large number of companies need to update their existing Enterprise Resource Planning (ERP) or IT systems, as those have generally not been developed with this level of regulation in mind. So is the case of Grafica Soset Industria e Comercio Ltda. who have been using an in-house developed customized ERP solution that needs to be updated in order to comply with NF-e regulations as mandated by SEFAZ. Besides these required adaptations they have also requested the additions of new functionality to better deal with the changing complexities of their daily business operations.

2. Project Description

Specifically the project entails the addition of new Java code to an existing custom ERP system in order to attain a working implementation of the processes as described by NF-e to maintain the legality of the goods offered by Grafica Soset.

Currently Grafica Soset offers goods solely via the original invoicing method of hard copy invoices. This method of invoicing can be described as follows.

- Print a legal paper document to close a commercial transaction
- Provide information on products offered and value of goods
- Add legal coding to enable tax authorities to apply tax regulations

The working implementation of digital invoicing must implement the following.

- Real-time integration with Brazilian Tax Authority (SEFAZ)
- Generate invoices as XML files to be sent and authorized by SEFAZ
- Ensure proper remote backups of digital invoice after SEFAZ authorization
- In the event of unavailability of SEFAZ make use of contingency services offered by the Brazilian government
- Print DANF-e to accompany goods in transit for validation en route

---

9 Enterprise resource planning (ERP) is an integrated computer-based system used to manage internal and external resources including tangible assets, financial resources, materials, and human resources
In order to implement these procedures the existing ERP system will have to be updated and the following usage scenario will have to take place during commercial transactions.

1) Enable ERP system to generate correct XML structure from internal data objects
2) Add modifications to ensure NF-e is generated prior to release of goods
3) Ensure correct XML structure via checking mandatory attributes visible in XML schema
4) Add real-time integration with SEFAZ systems via Web Services
5) Back-up plan in case SEFAZ systems experience downtime (use contingency services)
6) Apply digital signature to NF-e (obtained from certification authority ICP-Brasil)
7) Handle in real-time the validation result by SEFAZ of produced NF-e before releasing goods
8) Remotely backup validated invoices for future reference
9) In the event of unavailability of SEFAZ make use of contingency services
10) Print DANF-e, which is a human readable format of NF-e. It should be attached to the goods for delivery
11) Government officials can scan DANF-e at any time to check the validity of the commercial transaction. If deemed invalid penalties can be imposed onto the company that has emitted the invoice
12) Buyer receives goods
The following diagram depicts the usage scenario as described.

![Diagram of NF-e Process]

Besides providing the necessary modifications to the existing ERP system in order to comply with NF-e Grafica Soset has requested the creation of new functionality. A **stock** module dealing with information regarding supplies, a **cost** module dealing with post-calculation of manufactured goods and a **proposal** module dealing with pre-calculation of goods to manufacture should be developed and added to the ERP system. However in light of the time availability of the TU Delft Bachelor project it seems highly unlikely these modules could all be developed before the beginning of July, a hard deadline. If time is available however a start could be made in development of this new functionality. The focus of this project however lies primarily on the creation of NF-e functionality as previously described.

3. **Approach**

Several choices have to be made regarding the methods, tools and practices used during development of the modifications as described in section 2.

The existing ERP system in place has been developed in java and in order to attain a similar level of service the required modifications continue to be developed in java and added as extra features to the existing system. The existing Swing user interface needs to be updated to allow users the capability of interacting visually with the underlying NF-e implementation. A cost-benefit analysis could be made to evaluate the usage of an HTML web application that exposes the NF-e implementation but in order to maintain a uniform user experience with regards to invoicing it is deemed more logical to expand upon the current available Swing GUI as pictured in the figure 2.
The case could be made however that for the development of new modules including stock, cost and proposal the development of the user interface as an HTML web application would not be detrimental to the user experience as they form entirely new functionality and the possible decrease in development time could lead to tighter release schedules. A feasibility study could be done to assess these assumptions, as this is currently not the primary focus of the current project we deviate no further into this matter.

The specific software methodology used in this case focuses on agile methods in order to provide early prototypes that get refined by iteration. This allows for rapid evaluation by Grafica Soset of the proposed solution. The benefits in having early prototypes lie in having feedback directing the software process to the wishes of the customer while avoiding extraneous development. Work will include Java and groovy programming, possible database design and user interface design.

An evaluation has to be made on the underlying software model that implements the requirements. What this exactly entails is not entirely clear at the moment of writing and needs to be further refined in later phases of the project. An example would be evaluating the benefits of using a MVC pattern for the underlying implementation over some other design style.

The tools and languages used during the project are:

- PostgreSQL 8 Database Server running locally in a test environment and remotely in the production environment
- Netbeans code editor
- Java programming language
- Groovy programming language
- Photoshop graphics editor

---

10 Model-View-Controller (MVC) is a software architecture, currently considered as an architectural pattern used in software engineering
In order to acquire the necessary information to successfully embark on the project several contact persons have been assigned by Grafica Soset and other sources of information are available. The primary Grafica Soset contacts are Joao Carlos Ferreira, Solange Bishoff and Antonio Carlos. Further detailed information regarding legislation can be acquired via Robson Pavlawski. The primary source for technical information regarding implementing NF-e is available on the Internet at several websites authored by SEFAZ, the most relevant one located at http://www.fazenda.sp.gov.br/nfe/.

4. Quality Assurance

Quality Assurance during the project is maintained primarily via Unit Testing and beta testing product iterations. The usage of versioning systems is a common practice and originally Subversion (SVN) has been put into place as the primary versioning system. In light of new industry developments regarding distributed version control systems (DVCS) an evaluation will be made regarding the benefits of migrating over to the mercurial distributed version control system.

Users will test early prototypes and iterations; iterations will be done based on their feedback. If deemed necessary iterations are altered to attain user wishes while trying to maintain structural integrity via the already mentioned unit tests.

5. Release Schedule

The release schedule information for the months from May through August can be obtained from Appendix 1A.

6. References

Information regarding NF-e and its correct integration can be obtained from various online sources. The following such sources will be referenced during the project’s lifetime.

Digital Certificate:
http://www.iti.gov.br/twiki/bin/view/Certificacao/CertificadoObterUsar

NF-e:
http://ww2.prefeitura.sp.gov.br/nfe/
http://www.imakenews.com/edi/e_article001600373.cfm
http://nf-eletronica.com/blog/
http://www.fazenda.sp.gov.br/nfe/

Addendum of 10-5-2010

In this Project Description it was previously stated that NF-e functionality should be developed as an additional module to the existing Java desktop client in order to maintain a coherent user interface. However after an evaluation by Grafica Soset it has become evident that due to the nature of desktop migrations, the current Java desktop client is now targeting the Windows 7 and Windows XP operating systems. These two software systems differ in the way they render certain UI elements and this requires the re-architecture of user interface code in the Java desktop client to support both systems.

In light of this development the benefits of adding the NF-e functionality separately as an internal web application running inside the browser become apparent. An application running inside the browser would look uniformly across different operating systems and doesn’t require separate user interface development to target different systems. The web application would be developed to target the existing database the Java desktop client targets in order to provide uniformity. By having access to the same data the Java desktop client has access too, the NF-e functionality can be provided with the current data models unchanged, while designing for the browser mitigates the problems with differing operating system user interfaces. It is therefore proposed to provide the actual NF-e implementation as a set of HTML files, resources including images, possible scripts for user interaction and server side Java code.
Appendix 1A – Release Schedule

---

**Figure i – Schedule Task Outline**

<table>
<thead>
<tr>
<th>Task</th>
<th>Effort</th>
<th>Duration</th>
<th>Completed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Obtain NF-e issuer status</td>
<td>3d</td>
<td>3d</td>
<td>0%</td>
</tr>
<tr>
<td>1.1) Obtain A1 digital certificate from ICP-Brasil</td>
<td>2d</td>
<td>2d</td>
<td>0%</td>
</tr>
<tr>
<td>1.2) Request company NF-e accreditation with SEFAZ</td>
<td>1d</td>
<td>1d</td>
<td>0%</td>
</tr>
<tr>
<td>2) Analyse and write documentation</td>
<td>1w 4d</td>
<td>2w</td>
<td>0%</td>
</tr>
<tr>
<td>2.1) Write orientation report</td>
<td>2d</td>
<td>2d</td>
<td>0%</td>
</tr>
<tr>
<td>2.2) Read and analyse technical manual for NF-e integration</td>
<td>1w</td>
<td>1w</td>
<td>0%</td>
</tr>
<tr>
<td>2.3) Perform requirements analysis and write requirements report</td>
<td>2d</td>
<td>2d</td>
<td>0%</td>
</tr>
<tr>
<td>3) Implementation</td>
<td>9w</td>
<td>11w</td>
<td>0%</td>
</tr>
<tr>
<td>3.1) Start NF-e implementation and tests</td>
<td>5w</td>
<td>7w</td>
<td>0%</td>
</tr>
<tr>
<td>3.2) Deliver Milestone 1 Prototype</td>
<td></td>
<td></td>
<td>0%</td>
</tr>
<tr>
<td>3.3) Evaluate prototype based on user feedback</td>
<td>1w</td>
<td>1w</td>
<td>0%</td>
</tr>
<tr>
<td>3.4) Continue NF-e implementation development</td>
<td>3w</td>
<td>3w</td>
<td>0%</td>
</tr>
<tr>
<td>3.5) Deliver Milestone 2 Prototype</td>
<td></td>
<td></td>
<td>0%</td>
</tr>
<tr>
<td>4) Write end report</td>
<td>1w</td>
<td>1w</td>
<td>0%</td>
</tr>
<tr>
<td>5) Deliver end report</td>
<td></td>
<td></td>
<td>0%</td>
</tr>
</tbody>
</table>

---

**Figure ii – Schedule Gantt Chart**

---

28
NFE Project Brazil

Author: Giovanni Martina
Date: 05-05-2010
Modification Date: 10-05-2010
Document: Orientation Report

Delft University of Technology, May 2010
IN3405 Bachelor
Table of Contents

1. Introduction .......................................................................................................................... 31
2. IT Infrastructure & Process .................................................................................................. 31
3. Tools & Development ......................................................................................................... 34
   List of resources .................................................................................................................. 34
4. References .......................................................................................................................... 35
1. Introduction

In this Orientation Report we take a look at the existing IT infrastructure currently in place at Grafica Soset in order to gain a clear picture as to what parts need to be added or modified. Required NF-e\(^\text{11}\) functionality and the NF-e usage scenario was already described in the Project Description documentation, please refer to it when needing more information on what the NF-e usage scenario entails.

This document firstly describes the current IT infrastructure in place and the current process of invoicing. We subsequently take a look at the parts of the current infrastructure that need to be modified to allow for digital invoicing in compliance with NF-e. Secondly we take a look at the tools used throughout the project, where necessary information is obtained from and how quality assurance is performed during the project lifecycle.

2. IT Infrastructure & Process

The in-house software currently used, a combination of a PostgreSQL RDBMS\(^\text{12}\) and a Java-based desktop client called Poderoso, handles among other things hard copy invoicing for graphics work performed by Grafica Soset. The database software is installed in one central server machine whereas several desktop PCs make use of the Java client in order to perform operations on the database server. Whenever some graphics work has been completed and can be sent on its way to the client a hard copy invoice is created on a government-sanctioned form. This form accompanies the product during shipment to the client.

When an employee chooses to create a hard copy invoice she enters the billing module available within Poderoso and inserts the required information on the fields of the invoice creation form pictured in figure 1.

![Invoice creation](image)

\(^{11}\) Nota Fiscal Eletronica (NF-e) translates roughly to Digital Invoice. It is a new requirement by the Brazilian government for electronic bill of lading

\(^{12}\) Relational Database Management System (RDBMS), a database management system based on the relational model
After entering the required data the employee can subsequently take an official invoice form, place it inside an impact printer and send instructions to the printer to print the data. The end result is a hard-copy invoice that accompanies the merchandise to the client. The invoice data is also saved to the database server. In the event of erroneous information or a change request after the fact, the printed hard-copy should be cancelled by stamping it with a cancellation stamp and a new invoice should be created and printed out, while the original data should be removed by the database by some authorized person.

The NF-e implementation will mainly be needed in this part of the system. As described in the Project Description documentation an employee adhering to NF-e will at the moment of invoicing enter the billing module and again reach a screen where invoice data should be entered. The difference is that after all information has been entered a digitally signed XML invoice is created and a connection is made to the official SEFAZ online systems. From this point on forward there exist several possible courses of action. Whenever a successful connection has been made to SEFAZ, the NF-e request gets processed asynchronously and a receipt of service gets sent back to the client. Now we still don’t know whether the request was authorized or not and in order to figure this out another connection has to be made some time in the future to check the status of our original request, which is to be processed asynchronously by SEFAZ. When receiving the actual authorization we can subsequently print a DANF-e and send the merchandise on its way to the customer. In the event of non-authorization we would need to review the information sent to us from SEFAZ in order to determine the reason of non-authorization in order to submit a modified version of the NF-e that correctly deals with the errors originally encountered.

In the event of non-availability of SEFAZ due to some unforeseen event, a contingency plan has been put into place by the Brazilian government. Various contingency methods are described in the official manual for digital invoice emission in contingency all with varying degrees of difficulty in implementation with some methods requiring the use of official government sanctioned continuous form paper. The various contingency methods available are the following:

- FS\textsuperscript{15} – Use of official continuous form “FS” paper for NF-e in contingency
- FS-DA\textsuperscript{16} – Use of official continuous form “FS-DA” paper for NF-e in contingency
- SCAN\textsuperscript{17} – Alternative official online NF-e systems
- DPEC\textsuperscript{18} – Posterior NF-e transmission

Evaluating these four methods immediately leads us to believe that both option 1 and 2 seem highly impractical, requiring the usage of official continuous form paper not directly available in the Netherlands and requiring testing on impact printers. These two can immediately be disregarded by implementation considerations alone. Out of option 3 and 4 it seems option 3 provides the cleanest implementation. Option 4 requires the creation of software means to maintain a list of NF-e invoices that have not been authorized yet due to unavailability of SEFAZ whereas option 3 allows to transparently use SCAN systems for authorization purposes in the event of unavailability of SEFAZ, a logical choice and a choice that immediately indicates that the implementation shouldn’t care about the location or provider of NF-e authorization.

Another legislative requirement is the necessity to ensure at minimum a five year archival of authorized NF-e documents. This was already mentioned in the Project Description documentation. With this requirement in mind we need to devise a method of performing a backup of authorized documents. A possible backup strategy could be performing backups locally on separate disks or tapes. But this doesn’t ensure data persistence in the event of extreme calamities such as fire. In such a case the better solution would be to use some third party service to perform remote backups of sensitive data and in light of keeping costs down two possible options are proposed here. The first of which is the usage of Amazon S3 services that allows for

\begin{itemize}
  \item FS\textsuperscript{15} – Use of official continuous form “FS” paper for NF-e in contingency
  \item FS-DA\textsuperscript{16} – Use of official continuous form “FS-DA” paper for NF-e in contingency
  \item SCAN\textsuperscript{17} – Alternative official online NF-e systems
  \item DPEC\textsuperscript{18} – Posterior NF-e transmission
\end{itemize}
storage and retrieval of data any time from anywhere. This service provides an API\textsuperscript{19} that allows one to read, write and delete on remote servers to a minimal cost. A second possibility is the usage of Google App Engine as a storage facility. Google App Engine follows the PaaS\textsuperscript{20} model and this indicates the necessity of devising a minimal data storage framework in order to use the model for data storage whereas Amazon S3 is specifically focused on data storage. Google App Engine provides more flexibility whereas Amazon S3 is seemingly easier to implement. Another question with both of these services is cost. Due to the specific nature of the XML data pertaining to NF-e and the amount of graphics work performed by Grafica Soset, both Amazon S3 and Google App Engine fall under very low cost services. Amazon S3 pricing currently states $0.15/GB per month of usage for the first 50TB of data and a cost of $0.01 per 1000 requests.

With these additions in mind the modifications to the current invoicing process in Poderoso can be modeled as given in fig 2.

![Figure 2 - NF-e Invoicing Process](image)

The following list describes the process that should be implemented for the model in fig 2. We make the assumption here that approval will be given by SEFAZ/SCAN.

1. The user creates an invoice for completed graphics work
2. The invoice is sent over HTTP to SEFAZ/SCAN

\textsuperscript{19} Application Programming Interface (API), a software interface to allow different types of software to interact with one another

\textsuperscript{20} Platform as a Service (PaaS), a form of cloud computing where computing resources are provided as an online service
3. SEFAZ/SCAN returns a receipt of acknowledgement
4. The receipt is persisted to the local database server for later consultation
5. A receipt is requested from the local database server
6. A request is sent to SEFAZ/SCAN to obtain current NF-e authorization status
7. Authorization status is returned from SEFAZ/SCAN
8. As soon authorization is confirmed the invoice is backed up on a remote server
9. After authorization the printing of DANF-e is allowed. The document must accompany the shipment of the product
10. The invoice is sent via electronic means to the receiving customer

3. Tools & Development

The NF-e contains information relevant to the taxation of products. This domain specific knowledge is difficult to assess and requires collaboration between the developer and a person trained in the relevant legislation. Grafica Soset has assigned a contact person, who is able to answer questions regarding the actual NF-e implementation. The developer is free to provide the NF-e implementation with the assurance that relevant information can be obtained via conferring with Robson Pavlawski as is mentioned in the Project Description. Other contact persons could also possible answer relevant questions, as has been mention in the Project Description.

The development of the software itself implies the creation of requirements documentation containing technical information regarding the various structures that need to be added to the existing software. The information necessary can be obtained from the official manual for NF-e integration as published by the Brazilian government. This translates to the creation of class diagrams to model the various components that need to be added to the system.

The actual start of the implementation phase will commence after delivery of the requirements by the developer and after delivery of necessary information by Grafica Soset, in particular the availability of the digital certificate and accreditation with SEFAZ in order to access official SEFAZ systems. These requirements have already been communicated with contact person Joao Carlos Ferreira in Brazil.

In previous development iterations the use of SVN21 was prescribed during development. However SVN is a centralized version control system and is limited in use when no connection to the server is possible. In light of this limitation a new distributed version control system will be used throughout the project namely Mercurial. A distributed version control system enables committing to a local repository and synchronizing changes to a remote repository, which allows for greater flexibility.

After delivery of the requirements specification a prototype will be developed and unit tests will be simultaneously written for this prototype to maintain software quality. Grafica Soset will evaluate the prototype and feedback will be taken into account for a second development phase to deliver a second prototype for evaluation. At this moment in time according to the schedule as provided in the Project Documentation the time allotment will be depleted for the Bachelor project but development will continue with prototype-evaluation rounds, an iterative agile development of software, until user satisfaction has been reached and complete accordance to NF-e has been obtained. At this point the software will be put into daily usage.

3.1. List of resources

The following list contains information regarding all relevant tools and resources necessary for successful implementation of the NF-e project.

Database
PostgreSQL 8.4 [1] is the currently running database system in use at Grafica Soset.

Source code management
Mercurial SCM [2] will be used to manage source code for this project.

21 Subversion (SVN), a version control system
User interface and coding methodology
The user interface and all program code will be produced in a combination of Java, Groovy and HTML using the Grails [3] Model-View-Controller framework. The code will be edited in the Netbeans editor by use of the Grails plugin.

User interaction will be provided by means of the jQuery [4] and jQuery-ui javascript toolkits. User interface design will be accomplished by use of Adobe tools such as Adobe Photoshop.

Remote backup
Remote backup will be performed either through the use of Google App Engine [5] or Amazon S3 [6]. At this point in time it is not clear which one of these provides the best benefits and prototypes could possibly be built using both systems.

NF-e manual
The following manuals as provided by the Brazilian government will be consulted.

2. Manual de Contingencia v. 1.01 [8]

Whenever doubts about some pieces of information become evident Robson Pavlawski has been noted as the right person to contact regarding detailed NF-e information.

4. References
Appendix III – Requirements

NFE Project Brazil

Author: Giovanni Martina
Date: 10-05-2010
Document: Requirements

Delft University of Technology, May 2010
IN3405 Bachelor
# Table of Contents

1. Introduction ........................................................................................................................................... 38

2. Requirements ........................................................................................................................................... 38
   - Legal Requirements ............................................................................................................................ 38
   - User Requirements ............................................................................................................................. 38
   - Functional Requirements ..................................................................................................................... 38
   - Non-functional Requirements ............................................................................................................. 39
1. Introduction

This document will describe the requirements for the NF-e project as it pertains to Grafica Soset. Requirements coming forth out of this document will include user and legal requirements divided in functional and non-functional requirements. These requirements fit the overall scope of the NF-e project.

When describing requirements we use a specialized notation to denote the specific type of requirement.

- Legal requirements will be preceded with LR as shorthand notation for “Legal Requirement”
- User requirements will be preceded with UR as shorthand notation for “User Requirement”
- Functional requirements will be preceded with FR as shorthand notation for “Functional Requirement”
- Non-functional requirements will be preceded with NR as shorthand notation for “Non-functional Requirement”

2. Requirements

2.1. Legal Requirements

The following list summarizes legal requirements for NF-e emission as set forth by the Brazilian Ministry of Finance.

**LR1:** Authorized invoices must be electronically archived and available for consultation up to five years after authorization

**LR2:** Authorized invoices must be electronically sent to receiving customers

**LR3:** DANF-e must accompany product shipments

**LR4:** Invoices must comply with standards and law as provided by Brazilian Ministry of Finance (SEFAZ)

**LR5:** In the event of unavailability of authorization systems imply government approved contingency efforts as detailed in the Contingency Manual (please refer to the orientation report regarding NF-e emission in contingency)

**LR6:** Invoices should be digitally signed

**LR7:** The emitting entity (Grafica Soset) should be authorized by SEFAZ to emit digital invoices

2.2. User Requirements

The following list summarizes user requirements for NF-e emission as set forth by the contact persons at Grafica Soset.

**UR1:** NF-e invoicing process should be simple and easily understandable

**UR2:** NF-e invoicing should be extendable as to allow the further addition of a stock/warehousing module

**UR3:** User privileges should be used to limit user functionality based on user roles. That is only administrators should be able to cancel or delete information

**UR4:** Implementations should not add to the complexity of the existing workflow but should strive for reducing existing complexity and transparently handle the NF-e process as mandated by SEFAZ

2.3. Functional Requirements

The following list summarizes functional requirements for NF-e emission.

**FR1:** NF-e invoicing process should create an XML file in compliance with SEFAZ regulations

**FR2:** XML files must be sent over HTTP to pre-defined web service locations for authorization

**FR3:** Receipts of acknowledgements from SEFAZ should be locally persisted for future consultation or reference

**FR4:** Invoices should be locally and remotely persisted. Locally to the existing PostgreSQL database, remotely to either Google App Engine or Amazon S3

**FR5:** Login procedures must prevent unauthorized access to sensitive information
FR6: Users should be able on basis of receipt of acknowledgement request the status of authorization
FR7: Users should be able to cancel authorized invoices
FR8: Users should be able to change authorized invoices
FR9: Users should be only allowed to print DANF-e after authorization
FR10: Users should be able to send authorized invoices to receiving customers any moment after authorization
FR11: Users should be able to request the status and availability of SEFAZ authorization service
FR12: Users should be able to request the status of backed up invoices
FR13: NF-e functionality should not in anyway interfere with existing software and limit the existing functionality

2.4. Non-functional Requirements

The following list summarizes non-functional requirements for NF-e emission.

NR1: NF-e invoicing should be testable via unit tests
NR2: The workflow of invoicing should not allow for data to be created in an inconsistent state. An example would be that remote backups should not be done before NF-e authorization
NR3: The invoicing implementation should be extensible for further addition of stock/warehousing modules
NR4: The invoicing procedures should be performed with fast response times, the system should not appear to be halted to the user. The system should communicate its state to the user
NR5: The invoicing procedures should be reliable, trustable and secure
Appendix IV – Architecture

NFE Project Brazil

<table>
<thead>
<tr>
<th>Author</th>
<th>Giovanni Martina</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>10-05-2010</td>
</tr>
<tr>
<td>Document</td>
<td>Project Architecture</td>
</tr>
</tbody>
</table>

Delft University of Technology, May 2010
IN3405 Bachelor
# Table of Contents

1. **Introduction** .................................................................................................................. 42  
   Grails MVC ........................................................................................................................... 42

2. **Data Model** ..................................................................................................................... 43  
   Customer Model .................................................................................................................... 43  
   Product Model ....................................................................................................................... 43  
   Invoice Model ....................................................................................................................... 45  
   User Model ............................................................................................................................ 45

3. **Flow of Control** ............................................................................................................... 46  
   SEFAZ Authorization ........................................................................................................... 46  
   Backup .................................................................................................................................. 47

4. **User Interaction** .............................................................................................................. 47  
   Login .................................................................................................................................... 47  
   Invoice Emission ................................................................................................................... 47  
   Invoice Viewing .................................................................................................................... 48
1. Introduction

This document contains information on the Project Architecture for the NFE Brazil Project. It has been previously decided (see Project Approach) that, in order to overcome difficulties with user interface testing on differing Microsoft operating systems, a web interface will be built to support NF-e invoicing. In order to allow for rapid deployment, compatibility with the existing infrastructure and future enhancement, the Grails Java framework has been chosen as the underlying toolset to support development of NF-e invoicing.

Grails MVC

Grails (SpringSource, 2010) is an open-source web application framework built on Java that uses the Groovy (SpringSource, 2010) programming language. SpringSource, one of the leading companies in the Java Enterprise software world, is the driving force behind both Groovy and Grails.

Grails has been developed with some goals in mind namely to provide a high-productive, consistent and powerful framework in order to allow for rapid development of web application software. In order to achieve these goals Grails uses proven open-source Java technologies and hides away complexity by adhering to "convention over configuration". This means that software produced using the Grails framework should follow the Grails conventions of web application development, the most important of these being the adoption of the Model-View-Controller (MVC) paradigm.

Models in Grails are formed of Groovy classes containing specific properties. These properties get automatically persisted to an underlying database by the Grails framework without any user configuration. This also implies the omission of getter and setter methods as these types of redundant methods get generated during runtime via dynamic method injection, a feature of the Groovy language. Controllers are Groovy classes that provide the flow of control by directing users of the application to the correct pages and by handing over the model to the view for subsequent rendering. Views themselves are formed of static files that define the User Interface (UI) of the application in the browser. Views are created by a combination of HTML, JavaScript, images, Cascading Style Sheets (CSS) and Groovy Server Pages (GSP).

Some special care has to be taken with the implementation of NF-e functionality using the Grails framework seeing that there already exists a database and data model for various business properties such as Customers and Products. In light of this the NF-e functionality has to be provided in such a way as to not interfere with the existing Java desktop client and database, but as an additional layer on top of the existing database structure to support users in creating invoices. More concretely this demands the creation of Groovy classes for different models and controllers and the creation of various views that allow users to modify or edit the data. This document firstly describes what models need to be implemented in Groovy, how the user flow should be during invoicing and provides mockups of views that showcase the user interaction. In light of agile development no elaborate detailed views will be provided as mockups. User testing should allow for clarification of specific user wishes regarding the views and processes and these will be modified during development accordingly.
2. Data Model

Customer Model

Invoices will contain customer information so a Customer model should be provided in the NF-e layer as a Groovy class. Properties in the model have been obtained from the existing database or from the existing Java source code. Customers may have multiple contact persons; they are linked to at most one accountant and one salesperson. These are modeled as given in figure 1.

Product Model

Products can be divided between those that are printed and those that are non-printed. Non-printed works usually consist of stamps. We model them as given in figure 2. A super class by name of ServiceOrder contains properties shared by both printed products and stamps. Printed products themselves contain various entities that deal with the technicalities regarding printing graphics arts on paper. In figure 2 these entities are depicted as Ink, PaperPrint, Paper, Format and other classes. For the sake of brevity some classes do not have all their properties enumerated in the class diagram. Such is the case for PrintJob. Please refer to the class Java
source for a full list of properties, as it is only necessary to gain insight into the relation among classes, properties have been omitted where possible for the sake of brevity.

figure 2.
Invoice Model

Whenever a product (either printed graphics arts or stamps) gets released for delivery an invoice must be created that pertains to the particular product release. It contains customer and delivery information, product information, sales information and regulatory information. NF-e invoicing should be provided on top of the existing invoicing structure that already has been implemented in the current IT infrastructure. As such the inclusion of a new type of class NFInvoice should ease the migration. This new class should be related to an individual invoice and add all the necessary data in order to comply with the NF-e regulations and procedures as mandated by the Brazilian ministry of finance. The classes that together form the Invoice model are depicted in figure 3.

User Model

Lastly we need a user model. Specific privileges need to be present in the NF-e implementation to limit users from performing certain operations such as deleting or cancelling invoices. The existing IT infrastructure allows for separation between users and administrators. This separation of privileges needs to be continued in the NF-e implementation and we therefore model users as given in the model representation presented in figure 4. Depending on a user’s role certain operations such as deletion of invoices is disallowed. Only users are allowed to view or edit data within the system.
3. Use Cases

Flow of control is provided via Grails Controller classes and Filters (refer to the grails documentation regarding more information on Grails artifacts). We can envision the following use cases regarding invoicing which will be described next.

1. The user reaches the application front page
2. The user enters login information
3. Access is granted or denied based on the username/password combination
   a. In case of denial, redirect to front page displaying error message
   b. In case of access, proceed to application
4. The user chooses the NF-e module
5. The user enters a product service code of the form (AA0000) to indicate which product should be invoiced
6. If the product exists and has not yet been invoiced the invoice editor is shown
7. The user enters additional invoicing information if necessary and creates the invoice
8. The user chooses the option for authorizing the invoice via SEFAZ
9. After authorization the invoice gets backed up
10. The user is allowed to print DANFE and is allowed to send the invoice to the receiving customer

Authorization is not a real-time operation. Users are allowed to see a list of recently created invoices and verify the authorization status with SEFAZ. The steps are as follows

1. The user reaches the application front page and performs a login
2. After a successful login the user chooses the NF-e module
3. The user selects an option to view a list of recently created invoices
4. The list of invoices show SEFAZ and backup status of invoices
5. Users can choose an invoice to alter or cancel. Backups are performed automatically via a scheduled job.

Grails Controllers will provide the flow between steps. The first of these becomes immediately evident from the preceding descriptions. A LoginController will be provided to deal with user authentication. An InvoiceController will be provided to deal with flow between views that handle the invoicing process.

SEFAZ Authorization

Another type of Grails artifact is the Service class. The object of which is to encapsulate business logic. A Service is seen as appropriate for handling the authorization process between SEFAZ and the NF-e module. An InvoiceService class will be provided that is able to communicate with SEFAZ via the use of web services. The InvoiceService is depicted in figure 5.
Whenever an authorization attempt is made the InvoiceService class checks for SEFAZ availability and depending on availability transmits a digitally signed XML packet to the designated authorization service. It subsequently changes the NFIInvoice properties depending on whether authorization was given or not.

Backup

Remote backups of invoices should be performed without user intervention and on an automatic basis. The application should therefore take care of remote backups of authorized services. An automatic scheduler will be provided within Grails that performs daily backups of authorized but as of yet non-backed up invoices. To provide for this functionality an InvoiceScheduler job will be created by means of Grails scheduling that will, at the end of each workday, perform backups of invoices.

Performing backups implies the existence of a remote data store and application-programming interface (API). In light of existing usage of Google App Engine (GAE) for several parts of the existing IT infrastructure at Grafica Soset, a remote data store will be built on top of GAE to provide for back-up facility. The design of which will be provided in future documentation.

4. User Interaction

Login

Figure 6 displays a mockup for user interaction pertaining to login.

Upon entering the Grails application the user is first prompted to enter login information. After having performed a successful login the user is presented with available options.

Invoice Emission

Whenever the user chooses to emit an invoice he or she must enter a product service code (a special identifier for customer orders depicted as two letters followed by four numbers as in AA0000). When the appropriate product is found it is displayed with customer information, sales information and product information as is depicted in figure 7a.
The user is then able to create an invoice for the customer order and must enter invoice information as is depicted in figure 7b.

After saving the invoice data the user can proceed with authorization. After authorization with SEFAZ has been successfully performed the user is allowed to print DANFE and to forward the invoice to the receiving customer as is displayed in figure 7c.

Invoice Viewing

Users can request a list of invoices to verify invoice and authorization status. A list of invoices is displayed and should be sortable (on date, value or whatever property deemed interesting). This list should be depicted in a similar way as shown in figure 8.
Invoices are clickable and lead to the invoice editor that was mocked up in figure 7b. Users are able to alter and save the invoice. Users with the correct privileges are able to cancel and delete invoices.

References

Appendix V – Test Reports

Unit and integration tests are divided in packages. The following table lists test packages and gives a description on what part of the implementation is tested by the package. All tests successfully pass as can be seen from the test results table.

<table>
<thead>
<tr>
<th>Package Name</th>
<th>Number of Tests</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>net.drayah.poderoso</td>
<td>17</td>
<td>Tests the domain model classes such as Accountant, Customer, CFOP, Contact, IBGE, Location, Salesperson, Serie and User for proper persistence to the database and for proper logic</td>
</tr>
<tr>
<td>net.drayah.poderoso.billing</td>
<td>10</td>
<td>Tests the creation of Invoices with InvoiceItems and InvoiceProducts</td>
</tr>
<tr>
<td>net.drayah.poderoso.controllers</td>
<td>2</td>
<td>Tests user authentication for the UserController class</td>
</tr>
<tr>
<td>net.drayah.poderoso.products</td>
<td>17</td>
<td>Tests domain model classes such as Category, CategoryTypeMembership, PrintProduct, ProductModel, ProductType, StampProduct and Unit for proper persistence to the database and for proper logic</td>
</tr>
<tr>
<td>net.drayah.poderoso.reports</td>
<td>1</td>
<td>Tests the creation of the DANFE form in PDF format</td>
</tr>
<tr>
<td>net.drayah.poderoso.services</td>
<td>12</td>
<td>Tests application wide logic, proper MD5 hash generation and logic pertaining to invoice authorization</td>
</tr>
<tr>
<td>net.drayah.poderoso.ssl</td>
<td>9</td>
<td>Tests the validity of the supplied digital certificate and for proper SSL connection establishment to SEFAZ</td>
</tr>
<tr>
<td>net.drayah.poderoso.taglibs</td>
<td>12</td>
<td>Tests for proper logic pertaining to the formatting of date and currency values</td>
</tr>
<tr>
<td>net.drayah.poderoso.xml</td>
<td>16</td>
<td>Tests XML related classes for Schema validation errors, correct trimming and digital signing with use of an X.509 certificate</td>
</tr>
</tbody>
</table>

Unit Test Results

<table>
<thead>
<tr>
<th>Class</th>
<th>Name</th>
<th>Status</th>
<th>Time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConfigTests</td>
<td>testNfeEnvironment</td>
<td>Success</td>
<td>0.024</td>
</tr>
<tr>
<td>CustomerTests</td>
<td>testCustomerCreation</td>
<td>Success</td>
<td>0.261</td>
</tr>
<tr>
<td>SerieTests</td>
<td>testSerieAndSequenceRelation</td>
<td>Success</td>
<td>0.212</td>
</tr>
<tr>
<td>InvoiceTests</td>
<td>testInvoiceCreationIntegration</td>
<td>Success</td>
<td>3.092</td>
</tr>
<tr>
<td>UserControllerTests</td>
<td>testLogin</td>
<td>Success</td>
<td>0.301</td>
</tr>
<tr>
<td>UserControllerTests</td>
<td>testLoginFailure</td>
<td>Success</td>
<td>0.035</td>
</tr>
<tr>
<td>CategoryTypeMembershipTests</td>
<td>testCategoryTypeMembershipCreationAndDeletionByCategory</td>
<td>Success</td>
<td>0.080</td>
</tr>
<tr>
<td>CategoryTypeMembershipTests</td>
<td>testCategoryTypeMembershipCreationAndDeletionByProductType</td>
<td>Success</td>
<td>0.041</td>
</tr>
<tr>
<td>PrintProductTests</td>
<td>testProductCreationWithoutType</td>
<td>Success</td>
<td>0.179</td>
</tr>
<tr>
<td>PrintProductTests</td>
<td>testProductCreationWithType</td>
<td>Success</td>
<td>0.073</td>
</tr>
<tr>
<td>ProductTypeTests</td>
<td>testProductTypeModelsRelationship</td>
<td>Success</td>
<td>0.095</td>
</tr>
<tr>
<td>StampProductTests</td>
<td>testStampProductCreation</td>
<td>Success</td>
<td>0.155</td>
</tr>
<tr>
<td>AppServiceTests</td>
<td>testSearchByCustomerName</td>
<td>Success</td>
<td>0.650</td>
</tr>
<tr>
<td>InvoiceServiceTests</td>
<td>testFakeProductionEnvironment</td>
<td>Success</td>
<td>0.021</td>
</tr>
<tr>
<td>InvoiceServiceTests</td>
<td>testTestEnvironment</td>
<td>Success</td>
<td>0.015</td>
</tr>
<tr>
<td>InvoiceServiceTests</td>
<td>testEnvironmentStrings</td>
<td>Success</td>
<td>0.026</td>
</tr>
<tr>
<td>Test Suite</td>
<td>Test Name</td>
<td>Status</td>
<td>Time</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------------------------------</td>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>InvoiceServiceTests</td>
<td>testX509DateValidUntil</td>
<td>Success</td>
<td>0.016</td>
</tr>
<tr>
<td>InvoiceServiceTests</td>
<td>testX509DateValidFrom</td>
<td>Success</td>
<td>0.016</td>
</tr>
<tr>
<td>InvoiceServiceTests</td>
<td>testX509Subject</td>
<td>Success</td>
<td>0.014</td>
</tr>
<tr>
<td>InvoiceServiceTests</td>
<td>testX509Validity</td>
<td>Success</td>
<td>0.012</td>
</tr>
<tr>
<td>InvoiceServiceTests</td>
<td>testRandomCodeGenerationLength</td>
<td>Success</td>
<td>0.064</td>
</tr>
<tr>
<td>InvoiceServiceTests</td>
<td>testVerifierDigitCalculation</td>
<td>Success</td>
<td>0.014</td>
</tr>
<tr>
<td>SSLTests</td>
<td>testCertificateDateUntil</td>
<td>Success</td>
<td>0.019</td>
</tr>
<tr>
<td>SSLTests</td>
<td>testCertificateDateFrom</td>
<td>Success</td>
<td>0.012</td>
</tr>
<tr>
<td>SSLTests</td>
<td>testHasKeyStore</td>
<td>Success</td>
<td>0.011</td>
</tr>
<tr>
<td>SSLTests</td>
<td>testCertificateSubjectName</td>
<td>Success</td>
<td>0.011</td>
</tr>
<tr>
<td>SSLTests</td>
<td>testCertificateValidity</td>
<td>Success</td>
<td>0.010</td>
</tr>
<tr>
<td>SSLTests</td>
<td>testSoapRequestStatusServico</td>
<td>Success</td>
<td>5.205</td>
</tr>
<tr>
<td>AccountantTests</td>
<td>testAccountantCreation</td>
<td>Success</td>
<td>0.792</td>
</tr>
<tr>
<td>AccountantTests</td>
<td>testAccountantConstraints</td>
<td>Success</td>
<td>0.108</td>
</tr>
<tr>
<td>CFOPTests</td>
<td>testCFOPConstraints</td>
<td>Success</td>
<td>0.013</td>
</tr>
<tr>
<td>CFOPTests</td>
<td>testCFOPCreation</td>
<td>Success</td>
<td>0.014</td>
</tr>
<tr>
<td>ContactTests</td>
<td>testContactCreation</td>
<td>Success</td>
<td>0.093</td>
</tr>
<tr>
<td>ContactTests</td>
<td>testContactConstraints</td>
<td>Success</td>
<td>0.042</td>
</tr>
<tr>
<td>IBGETests</td>
<td>testIBGECreation</td>
<td>Success</td>
<td>0.073</td>
</tr>
<tr>
<td>IBGETests</td>
<td>testIBGECreation</td>
<td>Success</td>
<td>0.051</td>
</tr>
<tr>
<td>LocationTests</td>
<td>testLocationConstraints</td>
<td>Success</td>
<td>0.038</td>
</tr>
<tr>
<td>LocationTests</td>
<td>testLocationCreation</td>
<td>Success</td>
<td>0.027</td>
</tr>
<tr>
<td>SalespersonTests</td>
<td>testSalespersonCreation</td>
<td>Success</td>
<td>0.045</td>
</tr>
<tr>
<td>SalespersonTests</td>
<td>testSalespersonConstraints</td>
<td>Success</td>
<td>0.037</td>
</tr>
<tr>
<td>UserTests</td>
<td>testUserConstraints</td>
<td>Success</td>
<td>0.057</td>
</tr>
<tr>
<td>UserTests</td>
<td>testUserCreation</td>
<td>Success</td>
<td>0.032</td>
</tr>
<tr>
<td>InvoiceItemTests</td>
<td>testInvoiceItemConstraints</td>
<td>Success</td>
<td>0.188</td>
</tr>
<tr>
<td>InvoiceItemTests</td>
<td>testInvoiceItemCreation</td>
<td>Success</td>
<td>0.025</td>
</tr>
<tr>
<td>InvoicePriceInstallmentTests</td>
<td>testInvoicePriceInstallmentConstraints</td>
<td>Success</td>
<td>0.027</td>
</tr>
<tr>
<td>InvoicePriceInstallmentTests</td>
<td>testInvoicePriceInstallmentCreation</td>
<td>Success</td>
<td>0.020</td>
</tr>
<tr>
<td>InvoiceProductTests</td>
<td>testInvoiceProductConstraints</td>
<td>Success</td>
<td>0.102</td>
</tr>
<tr>
<td>InvoiceProductTests</td>
<td>testInvoiceProductCreation</td>
<td>Success</td>
<td>0.016</td>
</tr>
<tr>
<td>InvoiceProductTests</td>
<td>testGetProduct</td>
<td>Success</td>
<td>0.005</td>
</tr>
<tr>
<td>Test Suite</td>
<td>Test Name</td>
<td>Status</td>
<td>Duration</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>------------------------------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>InvoiceTests</td>
<td>testInvoiceConstraints</td>
<td>Success</td>
<td>0.128</td>
</tr>
<tr>
<td>InvoiceTests</td>
<td>testInvoiceCreation</td>
<td>Success</td>
<td>0.110</td>
</tr>
<tr>
<td>CategoryTests</td>
<td>testCategoryConstraints</td>
<td>Success</td>
<td>0.102</td>
</tr>
<tr>
<td>CategoryTests</td>
<td>testCategoryCreation</td>
<td>Success</td>
<td>0.016</td>
</tr>
<tr>
<td>CategoryTypeMembershipTests</td>
<td>testCategoryTypeMembershipConstraints</td>
<td>Success</td>
<td>0.023</td>
</tr>
<tr>
<td>PrintProductTests</td>
<td>testPrintProductConstraints</td>
<td>Success</td>
<td>0.072</td>
</tr>
<tr>
<td>PrintProductTests</td>
<td>testPrintProductViasABConstraints</td>
<td>Success</td>
<td>0.053</td>
</tr>
<tr>
<td>ProductModelTests</td>
<td>testProductModelConstraints</td>
<td>Success</td>
<td>0.032</td>
</tr>
<tr>
<td>ProductModelTests</td>
<td>testProductModelCreation</td>
<td>Success</td>
<td>0.022</td>
</tr>
<tr>
<td>ProductTypeTests</td>
<td>testProductTypeConstraints</td>
<td>Success</td>
<td>0.021</td>
</tr>
<tr>
<td>ProductTypeTests</td>
<td>testProductTypeCreation</td>
<td>Success</td>
<td>0.018</td>
</tr>
<tr>
<td>UnitTests</td>
<td>testUnitConstraints</td>
<td>Success</td>
<td>0.029</td>
</tr>
<tr>
<td>UnitTests</td>
<td>testUnitCreation</td>
<td>Success</td>
<td>0.016</td>
</tr>
<tr>
<td>DanfeReporterTests</td>
<td>testDanfePDFGeneration</td>
<td>Success</td>
<td>5.743</td>
</tr>
<tr>
<td>SecurityServiceTests</td>
<td>testMD5</td>
<td>Success</td>
<td>0.062</td>
</tr>
<tr>
<td>KeyStoreProviderTests</td>
<td>testValidKeyStore</td>
<td>Success</td>
<td>0.069</td>
</tr>
<tr>
<td>KeyStoreProviderTests</td>
<td>testInvalidKeyStore</td>
<td>Success</td>
<td>0.003</td>
</tr>
<tr>
<td>KeyStoreProviderTests</td>
<td>testInvalidKeyStorePassword</td>
<td>Success</td>
<td>0.009</td>
</tr>
<tr>
<td>PoderosoTagLibTests</td>
<td>testCurrencyBigDecimal</td>
<td>Success</td>
<td>0.288</td>
</tr>
<tr>
<td>PoderosoTagLibTests</td>
<td>testCurrencyFloat</td>
<td>Success</td>
<td>0.006</td>
</tr>
<tr>
<td>PoderosoTagLibTests</td>
<td>testCurrencyDouble</td>
<td>Success</td>
<td>0.009</td>
</tr>
<tr>
<td>PoderosoTagLibTests</td>
<td>testCurrencyNull</td>
<td>Success</td>
<td>0.007</td>
</tr>
<tr>
<td>PoderosoTagLibTests</td>
<td>testCurrencyNullValue</td>
<td>Success</td>
<td>0.007</td>
</tr>
<tr>
<td>PoderosoTagLibTests</td>
<td>testCurrencyString</td>
<td>Success</td>
<td>0.006</td>
</tr>
<tr>
<td>PoderosoTagLibTests</td>
<td>testCurrencyObject</td>
<td>Success</td>
<td>0.008</td>
</tr>
<tr>
<td>PoderosoTagLibTests</td>
<td>testShortDate</td>
<td>Success</td>
<td>0.031</td>
</tr>
<tr>
<td>PoderosoTagLibTests</td>
<td>testShortDateNullValue</td>
<td>Success</td>
<td>0.007</td>
</tr>
<tr>
<td>PoderosoTagLibTests</td>
<td>testShortDateNull</td>
<td>Success</td>
<td>0.008</td>
</tr>
<tr>
<td>PoderosoTagLibTests</td>
<td>testShortDateObject</td>
<td>Success</td>
<td>0.007</td>
</tr>
<tr>
<td>PoderosoTagLibTests</td>
<td>testShortDateString</td>
<td>Success</td>
<td>0.007</td>
</tr>
<tr>
<td>XMLNFeItemTests</td>
<td>testXMLNFeItemISSQN</td>
<td>Success</td>
<td>0.036</td>
</tr>
<tr>
<td>XMLNFeItemTests</td>
<td>testXMLNFeItemStructure</td>
<td>Success</td>
<td>0.017</td>
</tr>
<tr>
<td>XMLNFeTests</td>
<td>testXMLNFeISSQN</td>
<td>Success</td>
<td>0.034</td>
</tr>
<tr>
<td>Test Suite</td>
<td>Test Description</td>
<td>Result</td>
<td>Time</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------------</td>
<td>----------</td>
<td>-------</td>
</tr>
<tr>
<td>XMLNFeTests</td>
<td>testXMLNFeStructure</td>
<td>Success</td>
<td>0.036</td>
</tr>
<tr>
<td>XMLSignerTests</td>
<td>testXMLDigitalSignature</td>
<td>Success</td>
<td>0.749</td>
</tr>
<tr>
<td>XMLTrimmerTests</td>
<td>testXMLTrimming</td>
<td>Success</td>
<td>0.037</td>
</tr>
<tr>
<td>XMLTrimmerTests</td>
<td>testXMLNormalization</td>
<td>Success</td>
<td>0.030</td>
</tr>
<tr>
<td>XMLValidatorTests</td>
<td>testNFeRetRecepcaoShouldValidate</td>
<td>Success</td>
<td>0.299</td>
</tr>
<tr>
<td>XMLValidatorTests</td>
<td>testNfeRetRecepcaoShouldFailOnReceiptId</td>
<td>Success</td>
<td>0.120</td>
</tr>
<tr>
<td>XMLValidatorTests</td>
<td>testStatusServicoShouldValidate</td>
<td>Success</td>
<td>0.028</td>
</tr>
<tr>
<td>XMLValidatorTests</td>
<td>testStatusServicoShouldFail</td>
<td>Success</td>
<td>0.030</td>
</tr>
<tr>
<td>XMLValidatorTests</td>
<td>testStatusServicoShouldFailOnState</td>
<td>Success</td>
<td>0.027</td>
</tr>
<tr>
<td>XMLValidatorTests</td>
<td>testNFeShouldFail</td>
<td>Success</td>
<td>0.093</td>
</tr>
<tr>
<td>XMLValidatorTests</td>
<td>testNFeShouldValidate</td>
<td>Success</td>
<td>0.131</td>
</tr>
<tr>
<td>XMLValidatorTests</td>
<td>testNFeRecepcaoShouldValidate</td>
<td>Success</td>
<td>0.096</td>
</tr>
<tr>
<td>XMLValidatorTests</td>
<td>testNFeProcShouldValidate</td>
<td>Success</td>
<td>0.101</td>
</tr>
</tbody>
</table>
Appendix VI – Project Evaluation by Grafica Soset

The project evaluation as offered here was downloaded from the TU Delft International Office website and is a form containing questions for international companies that have TU students doing internships. The form has been filled and signed by Mr. Joao Carlos Ferreira, whom as project supervisor, I had the most direct contact with.

It is evident that the produced program is rated as excellent but some improvements could have been made regarding oral and written communication and paying more attention to the social context. Also comments have been made regarding the lack of communication between the university and the company. I would, however, like to point out that some of the experiences could have been negatively affected due to the special nature of this project. Namely the long-distance aspect of it could have adversely affected the company's experience regarding communication between the university and itself and not having the student around is in itself an issue that could create a sense of not knowing the current status of the project.

In light of these comments I would like to point out that if other students in the future have similar requests regarding performing technical work via long-distance, that maybe the university could have procedures in place to have more direct contact with the company in question. The evaluation form follows on the proceeding page.
## Evaluation form for companies offering internships

*Technische Universiteit Delft*  
*EWI onderviskwaliteitsorg*  
*Stage evaluatie J. de Vries*  
*Evaluatie stage bedrijven*  

**Mark as shown:** ☐ ☐ ☐ ☐ Please use a ball-point pen or a thin felt tip. This form will be processed automatically.  
**Correction:** ☐ ☐ ☐ ☐ Please follow the examples shown on the left hand side to help optimize the reading results.

---

### Date:
27/09/2010

### Name of company supervisor:
Joao Carlos Ferreira

### Name of company offering internship:
Grafica Sosset Industria e Comercio Ltda.

### Trainees:

- Period of internship: from **april/2010** to **september/2010**
- BSc programme: ☐ EE

### Assessment of internship by company supervisor

#### Did the internship student meet your expectations regarding academic skills?

<table>
<thead>
<tr>
<th>Level of expertise</th>
<th>Unsatisfactory</th>
<th>Moderate</th>
<th>Good</th>
<th>Excellent</th>
<th>n.e.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Research</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Cooperation</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Oral communication skills</th>
<th>Unsatisfactory</th>
<th>Moderate</th>
<th>Good</th>
<th>Excellent</th>
<th>n.e.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Written communication skills</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Work independently</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Taking social context into account</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

#### Which academic skills did you observe the internship student developing during the internship?

<table>
<thead>
<tr>
<th>Level of expertise</th>
<th>Unsatisfactory</th>
<th>Moderate</th>
<th>Good</th>
<th>Excellent</th>
<th>n.e.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Research</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Cooperation</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Oral communication skills</th>
<th>Unsatisfactory</th>
<th>Moderate</th>
<th>Good</th>
<th>Excellent</th>
<th>n.e.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Written communication skills</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Work independently</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Taking social context into account</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

#### Which factors contributed to this?

- The assignment's content
- The internship student's willingness to learn
- Supervision from the company
- Supervision from the department
Assessment of programme by company supervisor

In view of the student's performance, do you have any suggestions relating to the EEMCS study programme?

Based on the fact that we are a foreign company, many of our demands don't fit on a regular exchange knowledge process, but, we felt along the interchange a lack of contact from the school with us. Basically to know and understand the premises for this project, even knowing that the scope of the project was very well defined on the begging. In our special situation, the suggestion is reduce the gap between the school and the company on future compromises.

What is your opinion about the information provided by the EEMCS department relating to the internship's objectives?

What is your opinion about the information provided by the EEMCS department relating to how the internship is organised?

Would you welcome other internship students from TU Delft/EEMCS?

Comments:
The general evaluation of the job developed was excellent, it met to all our goals and launch the base for a wonderful toll, this tool can be can be improved for many other demands of our company. In point of view of results, it was beyond our expectation; at the very end we have now a product.

Hand in to the Internship Officer EEMCS or mail to J. de Vries
J.deVries@tudelft.nl

Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
International Office (Department Internships)
Mekelweg 4
2628 CD Delft, The Netherlands

Submission date report...27/09/2010

Signature company supervisor...........