Implementation of an Electronic Invoice Solution using the BizTalk Service Oriented Architecture

Bruno Miguel Brito de Gouveia

Dissertation for the achievement of the degree
Master in Information Systems and Computer Engineering

Examination Committee
Chairman: Prof. José Carlos Martins Delgado
Supervisor: Prof. José Alves Marques
Advisor: Eng. Bruno Ricardo Brás Franco
Advisor: Prof. Miguel Leitão Bignolas Mira da Silva

October 2008
Acknowledgements

No project can achieve success if the author is not open to contributions and criticisms of others.

I would first like to thank Professor Jose Alves Marques for his orientation and his criticism that proved to be an essential contribution to achieving the goals of this dissertation, as well as the opportunity to conduct this project in a business environment at Link Consulting.

I would also like to thank Professor Miguel Mira da Silva and Professor José Delgado for their contribution and their innovative vision of the business world.

A special thanks to Eng. Bruno Franco for the constant monitoring and the total availability during this past year making this project a reality.

To Eng. Mário Duarte for the understanding and the way I was integrated into the business unit, and also to the Link colleagues Raquel Pinto, Luís Vasques and David Paisana for their support and exchange of ideas.

A word of appreciation to Eng. Rui Pereira from Microsoft for his contribution in solving problems related with BizTalk.

To my two major course colleagues, André Coelho and Luís Tarrataca, for the endless nights of work at IST and the constant support and friendship. To them a big thank you.

To my family for all the affection, especially my parents, Paulo and Margarida, for I have never missed anything, for the understanding and by allowing the realization of my goals and ambitions.

Finally to my girlfriend and love of my life, Lígia, for the unconditional support and motivation in the good and bad moments.

Thank you.

Lisbon, October 30th 2008.

Bruno Gouveia
Agradecimentos

Nenhum projecto pode alcançar o sucesso se o seu autor não estiver aberto aos contributos e críticas de terceiros.

Em primeiro lugar gostaria de agradecer ao Professor José Alves Marques pela sua orientação e pelas suas críticas que se revelaram um contributo essencial para alcançar os objectivos desta dissertação, bem como pela oportunidade de realizar este projecto num ambiente empresarial na Link Consulting.

Também gostaria de agradecer aos Professores Miguel Mira da Silva e José Delgado pelos seus contributos e pelas suas visões inovadoras do mundo empresarial.

Um especial obrigado ao Eng. Bruno Franco pelo seu acompanhamento constante e disponibilidade total ao longo deste último ano tornando este projecto numa realidade.

Ao Eng. Mário Duarte pela sua compreensão e pela forma como fui integrado na unidade de negócio, bem como aos colegas da Link Raquel Pinto, Luís Vasques e David Paisana pelo seu apoio e pelas trocas de ideias.

Uma palavra de apreço ao Eng. Rui Pereira da Microsoft pelo seu contributo na resolução de problemas relacionados com a ferramenta BizTalk.

Aos meus dois grandes colegas de curso André Coelho e Luís Tarrataca pelas noites infindáveis de trabalho no IST, pelo apoio constante e camaradagem. A eles um grande obrigado.

À minha família por todo o carinho, em especial aos meus pais, Paulo e Margarida, por nunca me ter faltado nada, pela compreensão e por possibilitarem a concretização dos meus objectivos e ambições.

Finalmente à minha namorada e Amor da minha vida, Lígia, pelo apoio incondicional e pela motivação inesgotável nos bons e nos maus momentos.

Obrigado.

Lisboa, 30 de Outubro de 2008

Bruno Gouveia
Abstract

In 1998, resolution number 115/98 of the Portuguese Minister Council started a national initiative of electronic commerce. It stated that in order to keep up with the evolving globalization of companies and services, the Portuguese information society should adopt a new business model based on the electronic commerce, bringing to the Portuguese economical structure a renewed competitive capability in order to keep up with other European and global business partners.

After the establishment of the equality of the paper invoice and the Electronic Invoice in 1999, the Portuguese government has been developing legislation concerning electronic commerce and electronic invoicing.

The European focus on process optimization led more recently to the 2001/115/EC directive, where each member state is free to adopt its own formats and standards in order to answer to the new requirements on electronic invoicing. With the intent of simplifying the conditions applicable to invoicing in terms of VAT, by 2005 the Portuguese government has started a renewed initiative that had as a primary objective to provide the means necessary to implement electronic invoicing in the public administration and in Portuguese enterprises.

The legislation opened many doors to companies that had significant amount of transactions of commercial documents between business partners. But with the spread of business formats and standards to support legislation, also came ambiguity and various ways of complying with the legal requirements.

This dissertation focus on the Portuguese legislation regarding Electronic Invoices, standards and formats recommended for messages, storage, security and transport mechanisms for exchanged commercial documents. Using Microsoft’s BizTalk Server, we aim to show how this technology can answer almost to every aspect of an electronic invoicing solution, including integration with existing systems, compliance with message formats and security, overall performance and high level of extensibility and flexibility. We also aim to provide end-to-end business process visibility by monitoring transactions between business partners.

Keywords

Electronic Invoice, XML, UBL2.0, BizTalk, AS2, Advanced Electronic Signatures, BAM, SAP
Resumo

Em 1998, a resolução 115/98 do conselho de ministros Português deu inicio à iniciativa nacional para a Factura Electrónica. Esta afirmava que a sociedade de informação Portuguesa deveria adoptar um novo modelo de negócio baseado no comércio electrónico para conseguir acompanhar a crescente globalização das organizações e serviços. Portugal trazia assim para a sua estrutura económica uma capacidade competitiva renovada de forma a conseguir acompanhar os parceiros de negócio a nível europeu e global.

Depois do estabelecimento da equiparação da factura física e da Factura Electrónica em 1999, o governo Português tem vindo a desenvolver legislação relativa ao comércio electrónico e à facturação electrónica.

A preocupação com a optimização de processos levou à directiva europeia 2001/115/EC, onde cada estado membro é responsável por adoptar os seus próprios formatos e standards de forma a responder aos novos requisitos da facturação electrónica. Com o intuito de simplificar as condições aplicáveis à facturação em termos de IVA, em 2005 o governo Português deu inicio a uma iniciativa que tinha como principal objectivo fornecer os meios necessários para implementar a facturação electrónica na administração pública e nas empresas Portuguesas.

A legislação abriu diversas portas às empresas que tinham um volume significativo de transacções de documentos de negócio entre parceiros de negócio. Mas com o aumento dos formatos e dos standards para suportar a legislação, também surgiu a ambiguidade e várias formas de garantir a compliance com os requisitos legais.

Esta dissertação foca na legislação Portuguesa em termos de Factura Electrónica, nomeadamente standards e formatos recomendados para as mensagens, arquivo, segurança e mecanismos de transporte para a troca de documentos comerciais. Ao usar tecnologia Microsoft BizTalk Server, o objectivo é demonstrar como é que esta tecnologia pode responder a quase todas as necessidades de uma solução de Factura Electrónica, incluindo integração com sistemas existentes, compliance com formatos de mensagens e segurança, performance geral e um elevado nível de extensibilidade e flexibilidade. Também se pretende fornecer visibilidade ao processo de facturação através da monitorização das transacções entre parceiros comerciais.

Palavras-Chave

Factura Electrónica, XML, UBL2.0, BizTalk, AS2, Assinaturas Electrónicas Avançadas, BAM, SAP
Contents

1 Introduction
   1.1 Contextualization ................................................. 1
   1.2 Electronic Invoice Concepts ..................................... 2
      1.2.1 Invoices and Electronic Invoices ......................... 2
      1.2.2 Exchanging Electronic Invoices ......................... 3
      1.2.3 Certification Authorities and Public Key Infrastructure ...... 4
   1.3 Past Work ...................................................... 5
   1.4 Document Structure ........................................... 6

2 State of the electronic invoicing systems in Europe and Portugal .............. 7
   2.1 European Electronic Invoicing Directive ...................................... 7
   2.2 Example of a member state Electronic Invoice solution ......................... 8
      2.2.1 Description .................................................. 8
      2.2.2 SELIS Architecture ...................................... 9
      2.2.3 Final Regards ............................................ 11
   2.3 Portuguese electronic invoicing dissemination mechanisms ..................... 12
   2.4 Portuguese fiscal machine ......................................... 14
   2.5 Example of a Portuguese Electronic Invoice solution ........................... 15
      2.5.1 Description .................................................. 15
      2.5.2 Netdocs Architecture ...................................... 16
      2.5.3 Final Regards ............................................ 18
   2.6 Reference Architecture .................................................................. 19
      2.6.1 Description .................................................. 19
      2.6.2 eFact Reference Architecture .................................. 20
   2.7 Risk vs. Benefits analysis ............................................. 24

3 Problem Definition ........................................................................ 26
   3.1 The challenges imposed by B2B interactions .................................... 26
      3.1.1 Scalability ...................................................... 27
      3.1.2 Dependability .................................................. 27
      3.1.3 Autonomy ....................................................... 28
      3.1.4 Heterogeneity ............................................... 28
      3.1.5 Coupling among partners ..................................... 28
7.2 Future Work .................................................. 78

A Requirements Document ........................................ 80
  A.1 Functional Requirements ............................................ 80
    A.1.1 FR1 - Signatures ............................................. 80
    A.1.2 FR2 - Sending and Receiving Electronic Invoices ................. 80
    A.1.3 FR3 - Archiving ............................................. 81
    A.1.4 FR4 - Summary Map ........................................... 81
  A.2 Non-Functional Requirements ......................................... 82
    A.2.1 Performance ................................................ 82
    A.2.2 Scalability ................................................ 82
    A.2.3 Security ................................................ 82
    A.2.4 Persistence ................................................ 82
    A.2.5 Transport ................................................ 82

B Service Architecture ........................................... 83

C UMIC Recommendations ........................................... 93
  C.1 Validation Rules ................................................ 93
  C.2 UBL 2.0 Electronic Invoice Map .................................. 94
  C.3 UBL 2.0 Mappings .............................................. 95

D InfoPath Electronic Invoice Template .............................. 96

E Business Activity Monitoring Portal ................................ 98

F Bottlenecks .................................................. 100
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Manual buying process</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>Integrated buying process</td>
<td>2</td>
</tr>
<tr>
<td>1.3</td>
<td>Electronic Invoice Sending Process</td>
<td>3</td>
</tr>
<tr>
<td>1.4</td>
<td>Electronic Invoice Receiving Process</td>
<td>4</td>
</tr>
<tr>
<td>2.1</td>
<td>SELIS architecture. Source: [Service, 2007b]</td>
<td>9</td>
</tr>
<tr>
<td>2.2</td>
<td>Netdocs solution</td>
<td>16</td>
</tr>
<tr>
<td>2.3</td>
<td>Netdocs architecture</td>
<td>17</td>
</tr>
<tr>
<td>2.4</td>
<td>Point-to-Point approach</td>
<td>19</td>
</tr>
<tr>
<td>2.5</td>
<td>Third-party intermediary approach</td>
<td>19</td>
</tr>
<tr>
<td>2.6</td>
<td>eFact reference architecture</td>
<td>20</td>
</tr>
<tr>
<td>2.7</td>
<td>Audit information flow example</td>
<td>22</td>
</tr>
<tr>
<td>2.8</td>
<td>Reference security architecture</td>
<td>23</td>
</tr>
<tr>
<td>4.1</td>
<td>Three category services</td>
<td>37</td>
</tr>
<tr>
<td>4.2</td>
<td>N to N transformations without internal format</td>
<td>37</td>
</tr>
<tr>
<td>4.3</td>
<td>N to N transformations with internal format</td>
<td>37</td>
</tr>
<tr>
<td>4.4</td>
<td>Windows SharePoint Services Repositories</td>
<td>38</td>
</tr>
<tr>
<td>4.5</td>
<td>Business Activity Monitoring Visibility</td>
<td>39</td>
</tr>
<tr>
<td>4.6</td>
<td>AS2 Messages and MDNs</td>
<td>41</td>
</tr>
<tr>
<td>4.7</td>
<td>Global view of the technology</td>
<td>42</td>
</tr>
<tr>
<td>4.8</td>
<td>BizTalk Runtime Architecture. Source: [Chappell, 2007]</td>
<td>43</td>
</tr>
<tr>
<td>4.9</td>
<td>BizTalk BAM Component Interactions. Source: [Microsoft, 2007]</td>
<td>44</td>
</tr>
<tr>
<td>4.10</td>
<td>Windows SharePoint Services Service Bus</td>
<td>45</td>
</tr>
<tr>
<td>4.11</td>
<td>SAP R3 and BizTalk Server interactions</td>
<td>46</td>
</tr>
<tr>
<td>4.12</td>
<td>SAP Adapter for BizTalk Server</td>
<td>48</td>
</tr>
<tr>
<td>5.1</td>
<td>Orchestration Architecture</td>
<td>49</td>
</tr>
<tr>
<td>5.2</td>
<td>Orchestration Architecture and Service Categories</td>
<td>50</td>
</tr>
<tr>
<td>5.3</td>
<td>Core Entry Point Decision Point</td>
<td>52</td>
</tr>
<tr>
<td>5.4</td>
<td>Mapping Between ERPInvoice Format and UBL 2.0 Invoice Format</td>
<td>53</td>
</tr>
<tr>
<td>5.5</td>
<td>Personal Certificate Store</td>
<td>54</td>
</tr>
<tr>
<td>5.6</td>
<td>Other People Certificate Store</td>
<td>54</td>
</tr>
<tr>
<td>5.7</td>
<td>BizTalk Party Properties</td>
<td>55</td>
</tr>
</tbody>
</table>
List of Tables

5.1  Sent Invoices Activity Items .................................................. 63
5.2  Received Invoices Activity Items ............................................. 63
6.1  Estimate Integration Effort ....................................................... 73
6.2  Solution Performance .............................................................. 73
A.1  Functional Requirements - Signatures ....................................... 80
A.2  Functional Requirements - Sending and Receiving Electronic Invoices ................................................................ 80
A.3  Functional Requirements - Archiving .......................................... 81
A.4  Functional Requirements - Summary Map .................................... 81
A.5  Non-Functional Requirements - Performance .............................. 81
A.6  Non-Functional Requirements - Scalability .................................. 82
A.7  Non-Functional Requirements - Security ..................................... 82
A.8  Non-Functional Requirements - Persistence .................................. 82
A.9  Non-Functional Requirements - Transport .................................... 82
C.1  UBL2.0 Validation Rules According to Portuguese Legislation ........ 93
Code Listings

5.1 Core Entry Point Decision Expression for the ERPInvoice Format .................. 52
5.2 Creation of an InfoPath Processing Instruction .............................................. 57
5.3 Example of a S/MIME signed Electronic Invoice ........................................... 57
5.4 Configuration of the SharePoint repository column values ............................ 59
5.5 Core Entry Point Decision Expression for the SAP IDoc INV_ID01 Format ......... 69
# List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACEP</td>
<td>Portuguese Electronic Commerce Association</td>
</tr>
<tr>
<td>AES</td>
<td>Advanced Electronic Signatures</td>
</tr>
<tr>
<td>ALE</td>
<td>Application Link Enabling</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>AS2</td>
<td>Applicability Statement 2</td>
</tr>
<tr>
<td>B2B</td>
<td>Business to Business</td>
</tr>
<tr>
<td>BAM</td>
<td>Business Activity Monitoring</td>
</tr>
<tr>
<td>BAPI</td>
<td>Business Application Programming Interface</td>
</tr>
<tr>
<td>BASDA</td>
<td>Business Application Software Developers Association</td>
</tr>
<tr>
<td>BI</td>
<td>Business Intelligence</td>
</tr>
<tr>
<td>BPM</td>
<td>Business Process Management</td>
</tr>
<tr>
<td>BTS</td>
<td>BizTalk Server</td>
</tr>
<tr>
<td>CA</td>
<td>Certification Authority</td>
</tr>
<tr>
<td>CFO</td>
<td>Chief Financial Officer</td>
</tr>
<tr>
<td>CIO</td>
<td>Chief Information Officer</td>
</tr>
<tr>
<td>CRL</td>
<td>Certificate Revocation List</td>
</tr>
<tr>
<td>EAI</td>
<td>Enterprise Application Integration</td>
</tr>
<tr>
<td>EC</td>
<td>European Council</td>
</tr>
<tr>
<td>EDI</td>
<td>Electronic Data Interchange</td>
</tr>
<tr>
<td>ERP</td>
<td>Enterprise Resource Planning</td>
</tr>
<tr>
<td>ESB</td>
<td>Enterprise Service Bus</td>
</tr>
<tr>
<td>EU</td>
<td>European Union</td>
</tr>
<tr>
<td>HTTP</td>
<td>HyperText Transfer Protocol</td>
</tr>
<tr>
<td>HTTPS</td>
<td>HyperText Transfer Protocol Secure</td>
</tr>
<tr>
<td>IDoc</td>
<td>Intermediate Document</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IS</td>
<td>Information System</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>KPI</td>
<td>Key Performance Indicator</td>
</tr>
<tr>
<td>MAC</td>
<td>Message Authentication Code</td>
</tr>
<tr>
<td>MD5</td>
<td>Message Digest 5</td>
</tr>
<tr>
<td>MDN</td>
<td>Message Disposition Notification</td>
</tr>
<tr>
<td>MOM</td>
<td>Message Oriented Middleware</td>
</tr>
<tr>
<td>MSMQ</td>
<td>Microsoft Message Queue</td>
</tr>
<tr>
<td>OAGIS</td>
<td>Open Applications Group Integration Specification</td>
</tr>
<tr>
<td>OCSP</td>
<td>Online Certificate Status Protocol</td>
</tr>
<tr>
<td>PKCS</td>
<td>Public Key Cryptographic Standards</td>
</tr>
<tr>
<td>PKI</td>
<td>Public Key Infrastructures</td>
</tr>
<tr>
<td>RFC</td>
<td>Remote Function Call</td>
</tr>
<tr>
<td>SELIS</td>
<td>Secure Electronic Invoicing Service</td>
</tr>
<tr>
<td>SHA-1</td>
<td>Secure Hash Algorithm - 1</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>S/MIME</td>
<td>Secure / Multipurpose Internet Mail Extensions</td>
</tr>
<tr>
<td>SME</td>
<td>Small and Medium Enterprises</td>
</tr>
<tr>
<td>SOA</td>
<td>Service Oriented Architecture</td>
</tr>
<tr>
<td>SOAP</td>
<td>Simple Object Access Protocol</td>
</tr>
<tr>
<td>SSL</td>
<td>Secure Sockets Layer</td>
</tr>
<tr>
<td>TPE</td>
<td>Tracking Profile Editor</td>
</tr>
<tr>
<td>UBL</td>
<td>Universal Business Language</td>
</tr>
<tr>
<td>UDDI</td>
<td>Universal Description, Discovery and Integration</td>
</tr>
<tr>
<td>UMIC</td>
<td>Knowledge and Innovation Mission Unit</td>
</tr>
<tr>
<td>VAN</td>
<td>Value Added Network</td>
</tr>
<tr>
<td>VAT</td>
<td>Value Added Tax</td>
</tr>
<tr>
<td>W3C</td>
<td>The World Wide Web Consortium</td>
</tr>
<tr>
<td>WS</td>
<td>Web Services</td>
</tr>
<tr>
<td>WSDL</td>
<td>Web Service Definition Language</td>
</tr>
<tr>
<td>WSS</td>
<td>Windows SharePoint Services</td>
</tr>
<tr>
<td>XACML</td>
<td>XML Access Control Markup Language</td>
</tr>
<tr>
<td>XAdES</td>
<td>XML Advanced Electronic Signatures</td>
</tr>
<tr>
<td>xCBL</td>
<td>XML Common Business Library</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
<tr>
<td>XSD</td>
<td>XML Schema Definition</td>
</tr>
<tr>
<td>XSL</td>
<td>Extensible Stylesheet Language</td>
</tr>
<tr>
<td>XSLT</td>
<td>XSL Transformations</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Contextualization

The recent globalization of markets, technology and competition has increased business’ needs in terms of flexibility, quality, cost-effectiveness and timeliness. Information technology (IT) has played an important role in this field as a way of answering to these requirements [Burgess, 2002]. With the increased use of the Internet, electronic business is one area of IT that has been dramatically affected in the past decade, and so, electronic commerce has transformed and became one of the most important forces shaping business today.

Many companies still have their invoicing processes semi-automated, as the Figure [1.1] shows, typically the flow begins with the delivery of goods or services, and the creation of an invoice in the ERP system. After this first phase, it is necessary to print an invoice that will be sent to the buyer via post mail or fax.

![Figure 1.1: Manual buying process.](image)

Once the invoice is received, it passes through a phase of manual work where it gets digitalized before entering the buyer’s information system (IS), going eventually through the finance department where an order to pay the seller is emitted.

The activities that start with the creation of the invoice at the seller company, until the manual integration
at the buyer company, produce a major payload in terms of time spent in handling and processing of invoices, costs with paper usage and postage, etc.

With this in mind, companies are turning into process optimization and automation. This invoicing process as special features, allowing us to foresee great possibilities in the near future in terms of process optimization and automation. In part this is possible due to the manual workflow involved, approval processes, and some legal aspects that need to be fulfilled. On the other hand, it is related to technological advances in ERPs and integration engines that ease the compliance with legal matters, and provide better support for automation of processes (as we will see in the following chapters).

The goal of this dissertation is therefore to automate this process having into account existing systems and processes at both the seller and the buyer company (see Figure 1.2).

Figure 1.2: Integrated buying process.

This figure shows the main scope of this work, business integration between trading partners in order to bring the supply chain up to another level, allowing faster and more accurate business information to be exchanged between both parties.

The red shaded area includes the activities that the integrated system can put aside, focusing on the real value added activities that are indispensible to the day to day processes.

### 1.2 Electronic Invoice Concepts

In this section we introduce the concepts that are relevant for the understanding of the process of creation and exchange of Electronic Invoices.

#### 1.2.1 Invoices and Electronic Invoices

An invoice is a commercial document that every good or service (e.g. product) provider must issue and describes the products, quantities and prices agreed. It is an essential element for VAT, in the way it offers the clients of a product, or the receiver of a service, the right to deduction of the tax incorporated in the same product [for the Knowledge Society, 2006].
The Electronic Invoice is a commercial document similar to a simple invoice, but represented in an electronic format. If the authenticity of the origin and the integrity of the payload are guaranteed and the Electronic Invoice contains the necessary information foreseen in the law, it is considered to have the same value as the paper invoice.

1.2.2 Exchanging Electronic Invoices

Electronic invoicing is the sending of invoices by electronic means, i.e. transmission or making available to the recipient and storage using electronic equipment for processing, including digital compression, storage of data, and employing wires, radio transmission, optical technologies or other electromagnetic means. [Council, 2001]

Typically two alternatives are used in the exchange of Electronic Invoices by electronic means: The exchange of data using Advanced Electronic Signatures (AES) and Electronic Data Interchange (EDI). Due to the drawbacks of EDI, on this work, we will focus mainly on AES.

An Electronic Invoice signed with AES identifies univocally the sender as the author of the document, and is created with means that the he can exclusively control. The goal is to have a mechanism that detects any unhallowed change in the document in useful time [Coffey and Saidha, 1996].

AES is based on asymmetric cryptographic system composed by a series of algorithms that are used to generate a pair of asymmetric keys which are exclusive and independent. One is called the private key and the other the public key.

Basically the owner of the private key uses it to declare authenticity over a specified electronic document. The recipient can use the public key to validate the received document (in terms of authenticity of the origin and integrity of the document).

There are typically two processes relevant for the discussion of this work. First, the process of sending an electronic document (e.g. Electronic Invoice) which has four main steps and is described in Figure 1.3.

![Figure 1.3: Electronic Invoice Sending Process.](image)

1. A digital invoice is created in the information system of the seller company.

2. Using a specific public algorithm, a digest of the document is calculated. The key aspect here is that any entity with the prearranged means can reproduce the same digest given the original document.
3. The private key is used to sign the digest of the document that resulted from the previous operation.

4. The signed digest (the AES) and the document are coupled and sent over a predefined medium to the buying company. To the pair (document + signed digest) we call Electronic Invoice.

On the other side we have process of receiving an electronic document (e.g. invoice) (Figure 1.4).

Figure 1.4: Electronic Invoice Receiving Process.

1. The information system of the buying company receives and splits the Electronic Invoice in two parts: the original received document and the received signed digest.

2. On the first part, the information system calculates the digest using the prearranged algorithm.

3. On the second part, the information system uses the public key to obtain the received digest.

4. If the two digests are equal then the document is validated (both origin and payload are valid and authenticated). If not, then there is a possibility of corruption in the received information, or maybe it was not signed by the pairing private key. Bottom-line: the invoice shouldn’t be accepted as authentic.

### 1.2.3 Certification Authorities and Public Key Infrastructure

From a legal perspective the mechanism described in the previous section is acceptable. Nevertheless it requires the existence of at least one certification authority (CA).

A certification authority is an entity that is responsible for the creation and distribution of private and public keys. It is responsibility of the CA that key pairs are unique and that they match to a specific entity, i.e. organization. Another function of a CA is the issue of certificate revocation lists (CRLs). This list is posted to a directory on a periodic basis, determined by the CA. Users wishing to validate a certificate issued by a CA must check to ensure that the certificate has not been placed on the CRL [Kent, 1998].

These certification authorities can organize themselves into Public Key Infrastructures (PKIs), typically on a national level, and communicate on an international level to guarantee the identity of the holder and the unicity of the keys.

The so called digital certificate is an electronic document created by a certification authority that binds the identity of single or collective person to the key used by this entity to create AESs. In turn, these
certificates are authenticated by an electronic signature provided by the certification authority that created them [Kent, 1998].

Therefore the main role of a certification authority is to act as a third party, assuring that the AES in a document and the entity that owns the key used in the same AES are a perfect match.

1.3 Past Work

This dissertation follows the line of the work done in three previous dissertations from colleagues of Instituto Superior Técnico [Duarte, 2007], [Sequeira, 2007] and [de Sousa Paisana, 2008].

These dissertations already focused on the challenges and legal requirements imposed by legislation, and also provided some solutions to the problems identified allowing a good starting point for further study and development.

Duarte’s work consisted in the integration of with an ERP system available at the e-market through the construction of an adapter capable of sending and receiving Electronic Invoices in a transparent way to the ERPs of the involved companies. Duarte presented a mechanism that uses metadata to describe distinct and specific application repositories to ease the integration between the different formats used by the invoice system users as well as new users.

This work provided some insight into the problematic of integration using adapters and was very important because it alerted to problems we might face in the integration with a ERP system using a pre-packaged adapter.

Sequeira’s work focused on the problems that arise from the electronic exchange of invoices, as well as the legal requirements that support its implementation, and presents an integration solution that allows these documents to be exchanged between different information systems of organizations. Among the main challenges to be considered, there is the absence of international standards regarding the formats of the documents to be exchanged and the need to ensure the security and confidentiality of those documents.

One aspect that we want to explore is Sequeira’s assumption that puts the format conversion on the client and not on the invoice broker. Since we want to develop a solution that is simple from the client’s point of view, we will assume that the invoice can be sent and received in any XML format, and that the invoicing solution is responsible for the format conversion.

Finally Paisana’s work solving two different business scenarios. The first one is when a company buys a system to use together with an ERP. The second one is when a company or a person hires an invoice service that can be used or not with an ERP. In this last case the software is installed in a company that offers the service remotely to its subscribers.

The solution presented in the following chapters may have some points in common with Paisana’s work since they were developed at the same time at Link Consulting. Both use Microsoft technologies like BizTalk and take advantage of standards like AS2 and UBL 2.0. All the work was made considering the Portuguese law and the recommendation made by the Portuguese Agency for the Technological Program (UMIC).
1.4 Document Structure

This dissertation is structured in the following way:

- Chapter 2 - State of the Art: The purpose of this state of the art section is to provide an insight into the actual status of electronic invoicing in the European Union and more specifically, in Portugal. By setting this objective, it will be necessary to provide a holistic view of the recommendations provided by the European Directive so that later we can understand how Portugal has complied with those.

Two existing architectures are analyzed and, in order to set down a path and to support future work, a reference architecture is presented for point-to-point B2B transactions. Finally, in order to have an insight of the business value associated with this type of projects on the long run, we conclude with a risk versus benefits analysis of an Electronic Invoice project implementation in Portuguese SMEs.

- Chapter 3 - Problem Definition: This chapter presents and explains the main problem this dissertation addresses, from a business perspective and a technological point of view. This will be used to create a Service Architecture and consequently to set down a series of milestones that have to be achieved in order to answer the various aspects of an Electronic Invoice solution.

- Chapter 4 - Solution Proposal: In this chapter we present the Service Architecture that supports this work and explain how each specific problem is addressed, and what are the pros and cons of each approach. Here we also explore the technology involved in the further implementation of the solution, having special attention on how each component will interact with each other.

- Chapter 5 - Implementation: This chapter explains the implementation process used and the problems encountered at each stage and how these were resolved. Specific implementation artifacts will be detailed and as a conclusion, a short case study of the pilot solution at a fictitious enterprise will be described.

- Chapter 6 - Results: The objective of this chapter is to analyze the final product of this dissertation in terms of estimate integration effort, performance, bottlenecks, ease of integration with existing information systems and expansion to other message formats.

- Chapter 7 - Conclusions: Finally the last chapter concludes the dissertation, with a critical analysis and with a perspective on future work.
Chapter 2

State of the electronic invoicing systems in Europe and Portugal

2.1 European Electronic Invoicing Directive

In order for the EU single market to function correctly, the EU Commission developed the Directive 2001/115/EC to harmonize regulations surrounding invoicing [Foryszewski, 2002]. The key elements of this Directive are:

- Clarity over the obligation to issue an invoice;
- A standard set of data required on invoices;
- A legal basis for allowing invoices to be issued electronically;
- Regulations surrounding the storage of invoices.

Having in account their internal regulation, member states may also choose the best way to comply with these elements. Aspects like:

- Invoice data;
- Invoice language;
- Whether invoices are required for exempt or zero-rated supplies;
- Time limits;
- Regulations surrounding summary invoices and storage;

are key points that must be addressed in the transition of this directive to internal right of each member of the European Union.

1 For a better understanding of how each country complied with these aspects, there is an interesting report from CompTIA in [CompTIA, 2005].
For electronic invoicing to be VAT-compliant, there are two major conditions: first, the customer should be able to decide whether to accept or decline electronic invoicing by his supplier, and second, authenticity of origin and integrity of content must be guaranteed. \cite{Council, 2001}

These two conditions can be achieved using technologies \cite{e Invoicing, 2007} like:

- Signed document interchange over the internet using Advanced Electronic Signatures (e.g. XML document signed using AES, enveloped in AS2, sent over the internet);
- Electronic Data Interchange (EDI);
- Or other means (e.g. security measures, content checks, audit trails, summary statements and other country-specific requirements).

The European Council is flexible in the way it permits the member states to decide which method to adopt, but this also arises several problems. In fact, standardization is one of the main problems regarding electronic invoicing, many countries use different technologies to achieve a same output, and this compels the existence of some adjustments when integrating for example two business partners.

### 2.2 Example of a member state Electronic Invoice solution

#### 2.2.1 Description

Most electronic invoicing implementations are based on EDI, which is one of the recommendations of the European Directive 2001/115/EC on electronic invoicing, however there are some disadvantages like high operating costs and lack of interoperability \cite{Banerjee and Kumar, 2002, Service, 2007a, Weitzel et al., 2000, Yang and Papazoglou, 2000}, which compels a rise of the costs of these systems.

In fact EDI is more a peer to peer solution mostly used by large enterprises. As electronic invoicing requires a partnership between companies, it is crucial to have a generic, open and well accepted format, so that problems that arise from integration with additional suppliers or customers can be mitigated. If on the one hand EDI was effective for achieving some of the objectives of electronic invoicing, on the other hand it is a fact that it is not adapted for SMEs \cite{Polemi, 2007}.

Other solutions use signed documents exchanged over the internet using Advanced Electronic Signatures to achieve a significant level of security, but there are no existing implementations that guarantee a robust interoperable service provision and, at the same time, satisfy the security requirements on exchange and management of Electronic Invoices \cite{Service, 2007a}.

With this in mind, SELIS (Secure Electronic Invoicing Service) developed by the University of Piraeus, Greece, is a cross-border service for the secure exchange of Electronic Invoices offering a highly adaptable solution. It uses Web Services, XML and PKI standards in alternative to EDI, to provide secure and interoperable service provision, while meeting all the requirements imposed by European Union Policies \cite{Service, 2007a}.

Selected in 2003 as a member state example for an Electronic Invoice solution by CEN/ISSS eInvoicing Focus Group, SELIS’s business model is based in two dimensions:
• The use of chambers\[^2\] to cut down barriers and promote the use of affordable, secure and interoperable electronic invoicing services to SMEs.

• And the integration with legacy systems, ERPs and compatibility with pre-existing financial applications.

### 2.2.2 SELIS Architecture

This subsection depicts the SELIS architecture presented in the Figure [2.1](#).

![Figure 2.1: SELIS architecture. Source: Service, 2007b](#)

As we can see, the service architecture is organized in five big blocks [Service, 2007b]:

- **Administration and Orchestration Services**
  The services inside this component are in charge of managing and orchestrate all other services. These coordination services are transparent to all others, and they are part of the application server container where all services are running.

- **Basic Services**
  These services provide a basic set of functions that are used by the system to perform primitive tasks. These primitive tasks include the interface layer, message transformation and forwarding services, publication, query and notification services.

- **Security Mechanisms and Services**
  In the security layer are addressed all the security requirements of the architecture. Using mechanisms like digital signatures, Advanced Electronic Signatures and encryption, the security mechanisms are then encapsulated into distinct components that are in turn embedded into any other component that needs them, based on policies that govern the services.

---

\[^2\]A chamber (e.g. chamber of commerce) is a legislative or judicial body, like a society or association, that protects one specific interest. In Portugal this could be ACEP, Digital Alliance or UMIC (discussed in the next section).
The SELIS architecture supports the following security mechanisms [Service, 2006b]:

– XML Electronic Signatures with embedded timestamps

An electronic signature is an encrypted digest of a document. A digest, which is a cryptographic checksum of the document, is obtained by passing a document through a hash algorithm. This produces a short string of bytes that is a unique representation of that document. This is typically used to check data integrity.

In terms of non-repudiation, it is necessary to combine existing signatures with an electronic time representation. In SELIS, this is achieved by the inclusion and signing of time-stamping in the data received through the use of Time-Stamping Protocol[3].

Resuming this paragraph, the SELIS solution appends time-stamp data to the document containing the invoice information and then signs the whole document. This is a good way to guarantee a basic level of non-repudiation at a low level technical complexity.

– XML Advanced Electronic Signatures

XAdES extends the W3C/IEFT XML Signature[4] into the domain of non-repudiation. SELIS creates and validates AES by utilizing an enhanced version of the open source XAdES library (OpenXAdES[5]).

– OCSP Interaction

SELIS uses the Online Certificate Status Protocol[6] (OCSP) request and response generators and validators included in the Bouncy Castle Library[7]. This functionality is used by SELIS's Keys and Certificates Management modules in order to include specific certificate status information in XAdES signatures.

• Enterprise Services

The enterprise services represent the business functionality that the organization adopting this architecture wants to deploy. It defines an invoicing transaction sequence that takes place each time users interact with the service to send an Electronic Invoice. It also coordinates the interaction with other architecture services at specific points, so that a business process can be concluded with success.

• Infrastructure Support Services

These services manage the intermediate integration layer of the architecture that connects with back-office systems such as databases, wrapper software on top of legacy systems, existing ERPs, etc.

Regarding the interoperability with other ERP systems, the SELIS solution uses three main methods:

– Support for XML schemas that represent invoice data:
  * XML Common Business Library (xCBL);
  * Business Application Software Developers Association (BASDA);
  * Electronic Business Interchange using XML (eBis-XML Suite);

---

SELIS also includes a transformation module that makes use of transformation templates. In order to integrate with an ERP it is necessary to design a new template so that interaction between systems can take place.

SELIS is based in Web Services technology; therefore it is easier to integrate with ERPs that have published Web Services Interfaces. In most ERP cases, a Service Oriented Architecture is the key factor that enables a less complex integration solution.

The technologies and standards used in building up the SELIS architecture can be summarized as follows:

- Interoperability:
  - SOAP;
  - UDDI;
  - WSDL;
  - WSPL, for the definition of Web Service descriptions;
  - xCBL (XML Common Business Library) for the representation of Electronic Invoice documents in XML;
  - XSL and XSLT, for the transformation between different types of formats.

- Security:
  - XML Signatures;
  - AES;
  - Time-stamping, based on RFC 3161 protocol;
  - Access control, based on OASIS XACML (XML Access Control Markup Language);
  - WS-Security extensions to standard SOAP messages by applying signatures and XML encryption in a standardized way.

### 2.2.3 Final Regards

There are a few important aspects stated in a market analysis study promoted by SELIS workgroup and eTAN [Service, 2006a] that we believe that are important to have in mind before developing an Electronic Invoice solution:

- First of all, in some existing solutions, invoices are stored and managed centrally by the companies that provide the invoicing service. These companies act as trusted third parties, providing storage and transformation services from one invoice format to another. It might be interesting to contemplate this scenario as possible architecture layout.
• The Electronic Invoices are created either by a component of a financial application that the invoice issuer operates, by some plug-in to a pre-existing financial package, or even by an autonomous web-based Electronic Invoices facility. So we have to understand that it is necessary to have an environment where data input can come from numerous channels.

• Exchange of Electronic Invoices is carried out either over secured leased lines (e.g. VANs), or over Internet, using Message Authentication Code (MAC) and Secure Sockets Layer (SSL), technologies for ensuring the integrity, confidentiality and authentication during the exchange of invoices. Once again we must have an infrastructure that enables the use of internet as an open source carrier, and at the same time provide connectivity to VANs.

• Some solutions achieve non-repudiation with PKI and XML Signatures. Others provide a capability to access centrally stored private keys and certificates in order to provide a way for digitally signing Electronic Invoices from any access point.

• There are solutions that involve a dispatch of an email notification between the parties that exchange Electronic Invoices, so that a manual retrieval of the invoices can take place. This is a way to reduce the complexity associated with the security infrastructure required in other approaches, by providing access to a secure server where the invoice was created. But on the other hand, this implies some manual work, and a greater effort to integrate with other business partner’s information system.

• A significant portion of the existing solutions complies with the EDI standard or provides translators for it.

• Some other solutions use XAdES (XML Advanced Electronic Signatures) standard so that long-term digitally signed documents can be preserved.

### 2.3 Portuguese electronic invoicing dissemination mechanisms

This subsection shows the main institutions in charged with the promotion of the Electronic Invoice in Portugal. The main activities and objectives of each organization are also enumerated.

• **ACEP**: Portuguese electronic commerce association, is an independent organization without lucrative ends, formed by collective and individual persons, with the intent of study and implementation of diverse ways of electronic commerce. It is an open and independent forum for the debate, promotion, generalization of electronic commerce in Portugal [Association, 2006].

It was established in 1999, right from the beginning of the Portuguese electronic commerce initiative. Nowadays it is formed by financial institutions, e-marketplaces, internet service providers, online press, and telecommunication operators [Association, 2006].

The main objectives of ACEP are:

- Develop high value added knowledge about technological, economical, sociological, cultural, juridical and political issues associated with electronic commerce;
- Use this knowledge in order to provide services and give advice to enterprises, public administration and general consumers;
– And establish a national community of entities, experts and consumers interested by electronic commerce.

The main ways of diffusion consist in:

– Support and execution of national studies regarding the electronic commerce domain;

– Creation of practical guides to help the development of electronic business inside organizations and from a consumer perspective;

– Other ways consist in the realization of seminars and formation actions to empower a better knowledge from all parties involved in the electronic commerce value chain.

• Aliança Digital[^8] Digital Alliance is a similar non lucrative association with the intention to divulge the Electronic Invoice thematic in Portugal. Established in 2005, Digital Alliance was born from the grouping of many public and private entities that already had some experience from past implementations of electronic invoicing, or in some way, were seeking help to adopt such systems.

The strategy also goes through public sessions with the objective of clarify, experience exchange, and gathering of proposals for the definition of a portfolio of standards and best-practices.

The objectives of Digital Alliance can be summarized as follows:

– Identify, study and promote the knowledge regarding all the market players (government, enterprises and consumers) in relation to document dematerialization in general, and in Electronic Invoice in particular;

– Promote the development and adoption of best-practices and standards in the implementation of processes and technologies associated to the Electronic Invoice, always in articulation with the European and world wide trends;

– Work in partnership with the government and the principal entities responsible for the adoption of Electronic Invoice in Portugal and in Europe, contributing for the Portuguese legislation.

– Give guidance and educate the Portuguese entrepreneurs in aspects like: opportunities created by the ITs in general, and by electronic invoicing in particular;

– And develop a certification program for the electronic invoicing solutions in partnership with the government.

• UMIC: UMIC - Agency for the Knowledge Society is a public organism with the mission of coordinating the information society politics and mobilize it through the promotion of activities of divulgation, qualification and investigation [for the Knowledge Society, 2007]. UMIC is one of the most important mechanisms of dissemination of electronic invoicing thematic in Portugal.

It would be redundant to enumerate all the objectives of UMIC since these could be inferred from the objectives of Digital Alliance and ACEP. But in order to emphasize its position, UMIC has created a guide[^9] which is the Portuguese reference for adoption of electronic invoicing systems.

[^8]: All the information related with Digital Alliance was gathered in a personal meeting with general secretary Gonçalo Carvalhinhos on 26th November 2007.
[^9]: Guia da Factura Electrónica - Contributo para a modernização das empresas e do Estado [for the Knowledge Society, 2006].
This document presents and discusses the most relevant aspects regarding the dematerialization process of the invoice from the point of view of public and private entities. It presents the notions associated with Electronic Invoices, like Advanced Electronic Signatures, legislation, and the implications from the VAT standpoint.

2.4 Portuguese fiscal machine

Since the communitarian harmonization and equality of the Electronic Invoice to the paper invoice, that DGCI - Direcção Geral de Impostos, has the right to access via electronic means to data relative to Electronic Invoice usage by national citizens [for the Knowledge Society, 2006].

In this way, citizens must assure operational integrity of the system and the information stored through electronic means. The operational integrity of the system must be guaranteed through:

- Access control to system functions through an authorization management model;
- Existence of integrity control functions and reliability in terms of information created, received, processed and emitted;
- Existence of control functions for direct or anonymous change in the information managed or used in the system;
- Maintenance of the information necessary to reconstitution and verification of the fiscal relevant procedures, supported by the system;
- Inexistence of applications from any source that allow changing directly the information, away from the control procedures registered for the system and without generating any evidence aggregated to the original information.

The information integrity must be guaranteed through:

- Preservation of information in terms of accessibility and legibility in order to allow its usage 24/7 and without restriction;
- Existence of an integrity control over the archived information, preventing information disclosure or destruction;
- The documentation related with the architecture, functional and organic analysis, and exploration of the Electronic Invoice system must be kept accessible during the timeline foreseen in the law for fiscal documents. The same procedure must be maintained for the storage mechanisms, software and algorithms integrated in the system.

The documentation must address the following aspects:

- Functionalities supported by the system;
- Control functionalities and the respective audit mechanisms;
- Physical or logical mechanisms used in the information integrity preservation;
• The informational model which allows the identification of the content of the data structures and respective life cycle.

Typically, an act of audit consists in an analysis of the consistence and validity of the introduced and processed information, as well as the output produced and stored digitally. In this way, it is of the responsibility of DGCI, through its tax inspection services (serviços de inspeção tributária), the evaluation and approval of the obligations imposed by law [for the Knowledge Society, 2006].

To reach this end, DGCI uses audit teams duly certificated that should have full access to the sites used in the Electronic Invoice activity, so that applications, magnetic storage systems, or other documentation can be examined.

The following four areas are typically involved in the audit process:

• Access to applications and magnetic storage;
• Application analysis and respective data processing mechanisms;
• Documentation copy in magnetic storage;
• Confirmation of the analysis’ programmatic, and execution data conservation.

Given that there is no previous notification that an audit session will take place, the DGCI will have the right to prove in-site that all legislation requirements are being fulfilled. And so, these audit sessions can be made in two ways:

• First, through direct access to the information system that supports the electronic invoicing, by using proprietary software and hardware or by the audited entity own means, so that relevant fiscal data can be analyzed.

• Second, through legible data delivery from the audited entity to the DGCI.

As an example, there is a new XML format introduced in 2008 called Standard Audit File for Tax Purposes Portuguese version (SAFT-PT)\(^{10}\), which as the name shows, is used for audit reasons. This file should contain accounting data extracted from the electronic invoicing system, allowing a better legibility and faster verification by the authorities.

### 2.5 Example of a Portuguese Electronic Invoice solution

#### 2.5.1 Description

In this section, the goal is to analyze the Netdocs\(^{11}\) solution that was adopted by several retailers and suppliers in the Portuguese market, and we’ll try to understand what makes it such a versatile solution.

The Netdocs system was developed by Softlimits with three objectives in mind:

• To allow the exchange of commercial documents between two business partners;
• To be able to address the needs of the larger and most technologically evolved companies through the use of standard formats like EDIFACT and XML, allowing the direct integration with pre-existing information systems;

• To be able to address the needs of the smallest companies through the design of an easy-to-use solution with no initial investment and low monthly cost.

The Netdocs solution is composed of two major blocks:

• A Web Portal (Portal Netdocs);

• And integration software (Netdocs Integrator).

The Figure 2.2 shows the Netdocs solution with an integration software put in place in an imaginary company.

![Figure 2.2: Netdocs solution.](image)

In this situation, the ERP is in charge of creating and reading the documents exchanged between the business partners. The format of these documents should be agreed between Softlimits and the ABC Company.

The Netdocs Integrator has translation software that makes the conversions between the standard formats (XML and EDIFACT) to flat-file formats typically used internally in the most common ERPs. It also has a communication module responsible for the communication with Netdocs Portal.

The Netdocs Portal manages all necessary storage and digital signing of the Electronic Invoices. It also has receiving and sending capabilities, and in case of companies with ERPs or other information systems, it makes all the necessary conversion to XML or EDIFACT. In case of small companies, the documents will be available online through a webform where it can be accessed, printed or downloaded in PDF format.

### 2.5.2 Netdocs Architecture

The Netdocs Architecture is composed of four main layers as seen in Figure 2.3:

• Connectivity and data entry

  The connectivity services allow the usage of several platforms and link protocols like X400, AS1, AS2,
AS3 and ebXML as well as standard protocols like FTP, POP3, SMTP, HTTP and HTTPS. The data can be sent over in several formats like flat-file, EDIFACT, XML and PDF using Web Services interfaces.

The connectivity services can be divided into two classes:

- **Batch connectivity**: where batch processes focused on the integration with IS/ERPs are used for automatic document transfer.

- **Online documents**: focused on clients without batch connectivity means, this functionality allows data entry through webforms. This kind of solution is most useful for clients with low amount of document traffic.

**Data transformation**

After the input of data in a specific format, this layer provides a set of mechanisms to transform data from an original format to another, always having into account the agreements established between Softlimits and other business partners.

**Data storage**

The data storage services are designed to implement a set of standard rules known as PAIN (Privacy, Authenticity, Integrity and Non-repudiation).

**Web presentation**

The presentation layer is based in webforms allowing search, validations, batch information requisition or direct download. For security reasons, all accesses to the Web interfaces are made over SSL, i.e. all users access HTTPS sites.

This layer also defines three types of users:

- **User**
  
  The User has permissions to do searches, document visualization and printing. According to law, it can also request summary maps ordered by fiscal date or by client. The User also manages document life cycle, and has permissions to request integration information using two formats, XML via AS2 or EDIFACT via VAN. Finally, the User is always notified about new documents in a pro-active way (e.g. email or sms).
– Auditor
  It has search and document visualization permissions. It also has full access to the stored summary maps.

– Administrator
  The Administrator can create users and define access control rules, temporary logins for auditing, etc.

2.5.3 Final Regards

When analyzing this solution, there were also some aspects that we believe that are worth to be mentioned:

- Netdocs solution was developed not only for Electronic Invoices but also for order requests, remittance slips, etc. It is important to understand the business of each integration case, here we can find new ways of optimizing our solution. For instance an electronic invoicing solution can have more value if it is combined with a document management application.

- In the case of the Electronic Invoice, digital signatures are applied and verified only by the portal software (e.g. when a supplier sends an Electronic Invoice through the Netdocs Integrator it isn’t digitally signed. A digital signature is applied only when it arrives at the portal).

- This model has a security gap when the Electronic Invoice is generated by a company and then sent to the Netdocs Portal. In this situation specific security measures must be applied to guarantee that the invoice data is not tampered with.

- To address each property of the PAIN model, our believe is that:
  
  – Privacy
    Privacy tends to be one of the major concerns that clients have about eBusiness [College, 2007]. There is a urge for companies to ensure that private or sensitive information, is kept secure and not used for any purpose other than that agreed to. This goal is usually achieved using encryption.

  – Authenticity
    In this particular case, this is the process by which one entity verifies that a document is valid during a certain period of time. To reach this end we might use certificates (or Advanced Electronic Signatures) and timestamps, so that a document can be tagged as valid in a specific period in time.

  – Integrity
    Integrity is the ability to prevent data from being altered or destroyed in an unauthorized or accidental manner [College, 2007]. The storage layer must ensure that all data and documents cannot be altered without detection. For this, hashing algorithms might be used (e.g. MD5).

  – Non-repudiation
    This mechanism ensures that neither side can back out of a transaction by claiming it never took place. This can be guaranteed through the logging of all operations or by using digital signatures to ensure that a certain entity was involved in the transaction.
Finally, we believe that this is such a versatile solution because it contemplates many business scenarios, and the implementation in layers offers ways to adapt the system functionality to the needs of the clients. A key point is the independence of the system over proprietary technologies allowing the migration of data to another platforms and applications. In fact, the system was developed with the intent of maintaining the format independence; the EDI provider and communication channel independence, as well as use as much open standards as possible.

2.6 Reference Architecture

2.6.1 Description

After the analysis of SELIS and Netdocs solutions and respective architectures, we present a hybrid reference architecture for B2B transactions.

Hybrid because in addiction to the two examples analyzed previously to extract some ideas and concepts, we also focused in the analysis of generic frameworks and architectures of electronic business solutions, and then transposed them into the domain of our problem, i.e. B2B electronic invoicing, so that a reference architecture based on past work with well accepted standards could be developed to support future work.

Besides this, the reference architecture is based the Web Services Framework introduced by Huang and Chung [Huang and Chung, 2003], on the reference system for internet based inter-enterprise electronic commerce by Shina and Leem [Shina and Leem, 2002], and with a mix of our own personal thoughts and concepts gathered from the vast reference architectures proposed in Enterprise Integration - The Essential Guide to Integration Solutions [Gold-Bernstein and Ruh, 2004].

The objective of this reference architecture is to provide the means to develop an independent software component that can be used as a complement to the existing architecture layout, integrating two or more applications in a B2B environment.

Using a point-to-point approach to answer the Electronic Invoice problematic, this Electronic Invoice module is designed to provide the means necessary to guarantee the compliance with the legal requirements in a pre-existing information flow. Figure 2.4 depicts a possible integration scenario.

![Figure 2.4: Point-to-Point approach.](image)

![Figure 2.5: Third-party intermediary approach.](image)

It is important to have in mind that nowadays there are several service providers that guarantee compliance
with these requirements. This reference architecture also provides an alternative approach which is point-to-point connection using a third-party intermediary, Figure 2.5.

### 2.6.2 eFact Reference Architecture

In Figure 2.6 is presented the eFact reference architecture. The outstanding part of this section explains each layer and how each intra layer component interact from a holistic point of view.

![Figure 2.6: eFact reference architecture.](image)

**Interface Layer**

This layer is the main entry point for the architecture. The objective is to have different technology platforms connecting to this layer so it acts like an abstracting coat to all outside users, should they be a user connected to a web browser, a legacy application, or an ERP solution.

It provides the means to:

- Present data in a structured way for users or other applications in a pre-existing architecture to consume;
- Extract data from other existing applications.

Since the domain of our problem consists in reducing the inefficiencies in the manual process, we should look for a solution that offers strong workflow capabilities [Gold-Bernstein and Ruh, 2004].

As an example, here we could have something like a Portal or an enterprise intranet with content management and defined workflows with an interface to display individual work assignments. Or in case of application-to-application integration with automated business processes, we could have adapters that provide means for batch data transfer.

To comply with the Portuguese legislation, there must be adapters that provide the non-repudiation element to transport electronically signed XML invoices between the two parties in the transaction over the internet.

The main goal here is to better adapt to each business need, looking for an enhanced balance between functionality and flexibility.
**Enterprise Service Bus**

Next to the interface layer, an enterprise service bus (ESB) can be used, allowing a more flexible way to spread the functionality along the layers, allowing the establishment of a service infrastructure that supports the use of standards like XML and Web Services, enabling for instance content-based routing.

The ESB is a good solution when Web Services are used as the method of connectivity to other applications [Gold-Bernstein and Ruh, 2004]. Since we also must contemplate this situation as a possible business integration option, we believe this is a good way to provide the infrastructure for the development and management of these services.

Given that on one hand we might want to integrate directly with an application through a Web Service, and on the other hand, have a nice and user friendly interface to do a similar task based also on Web Services, the use of such technology enables an environment where change can come from multiple channels.

The ESB provides connectivity services, including transport protocol and message protocol. Some also provide guaranteed message delivery making use of a light weight message repository for store and forward delivery [Gold-Bernstein and Ruh, 2004].

But the main advantage is the ability to plug other services on a as needed basis, and here we have the means to provide common services such as Business Activity Monitoring, Security mechanisms and Audit features necessary to comply with legislation matters.

**Process Orchestration Layer**

Here resides the main functionality of the solution. On one hand, we must have core services that are responsible for the proper Electronic Invoice certification (Electronic Invoice content validation, electronic signature, etc.), and on the other hand we must have integration business logic services that make use of the well defined core services, in order to address specific business scenarios.

Since in today’s marketplace there are service providers that do almost all the work in terms of core services, i.e. Electronic Invoice certification, storage and legal compliance; it is important to have a mechanism that also contemplates this case. Here we might use only the integration services to define a connection to this service provider, and bypass our own core functionality.

Other main subject that we should address is the message format translation. we believe that in order to have a more flexible and adaptable architecture, we should have into account two concepts:

- The use of a well defined message format for internal computation.
- The use of a repository of translation models.

When we need to include another format, we just have to place the translation map in this repository, and it will become available to the core and integration services.

**Storage Layer**

At a lower level is the storage layer where we propose three types of repositories:

- **Electronic Invoice Archive**
  
  This is the main archive where information regarding Electronic Invoices should be stored. Here the PAIN properties should be addressed with special attention in order to guarantee a high level of security. Mechanisms of encryption, and advanced electronic signature should be applied.
Data format translation models

With a well-defined internal message format, it is possible to develop adapters/translators for in/out message flow, allowing integration with other systems and their internal formats. This is also very important for audit purposes, where typically it is necessary to export the data in a specific format for the audit entities.

To reach this end, we propose a database of data format translation models, so that the archived information could be converted to any type of format necessary. In fact, this is a very flexible way to answer this requirement, as new models only need to be inserted in this database.

Figure 2.7 presents an example of an audit information flow.

Figure 2.7: Audit information flow example.

Configuration

This repository can be seen as a container where all information about configurable components is stored. This includes trading policies, configuration about electronic signatures, encryption algorithms, etc.

Common Services Layer

Strategically vertically placed, the common system services contemplate Business Activity Monitoring (BAM), Audit, and Security at all layers of the reference architecture, providing real-time reporting on the business transactions.

Also very important, here resides the security functionality (Figure 2.8), access control mechanisms, support for audit procedures, etc. The objective is to have these services available at each layer of the reference architecture and plugged into the enterprise service bus, so that changes into the security policies and audit mechanisms can take effect and become accessible at each layer.
The mechanisms to guarantee the PAIN properties should be implemented through this security architecture.

To address the privacy issue we propose the use of XML-Encryption standard. The encryption process involves taking an element from an XML document, encrypting it and its children, and then replacing the original XML content with the generated encrypted XML in such a way as the document remains well formed. Since we have an internal XML format to comply with Portuguese legislation, we should use a mechanism designed specifically to XML.

To address authenticity, integrity and non-repudiation, we propose the use of XAdES. XML Advanced Electronic Signature (XAdES) provides basic authentication and integrity protection and satisfies the legal requirements for Advanced Electronic Signatures as defined in the European Directive [Council, 1999]. In fact XAdES provides the means to implement time-stamping through XAdES-T, and even more interesting for the domain of our problem, there is an extra feature for archiving (XAdES-A), that allows the periodical time-stamping (e.g. each year) of the archived document to prevent compromise caused by weakening signature exposure during long time storage period.

Since a minimum of 10 year storage is the legal requirement for an Electronic Invoice in Portugal, this issue can be addressed using XAdES-A. However there are potential weaknesses resulting from combining signing and encryption operations, and they are related with the two ways these mechanisms are applied [W3C, 2002]:

- **Sign before you encrypt**
  
  Unless proper precautions are taken (such as adding an encrypted random string to the plain text before hashing) the signature may reveal information about the data that now has been encrypted.

- **Encrypt before you sign**
  
  There may be multiple \{data, key\} pairs that result in the same encrypted data, therefore special case must be taken in the selection of the encryption function or in the signature process to mitigate the possibility of signature repudiation (e.g. "I didn’t said this, I signed a different message encrypted under a different key...").

The final idea is to make available an independent set of services that provide functionality to cipher/decipher documents, sign documents, validate signatures, add timestamps, etc.
2.7 Risk vs. Benefits analysis

In this section it is made a brief Risk vs. Benefits analysis of the adoption of Electronic Invoice systems in SMEs. The final idea is to provide a better understanding of the possible business value associated with this kind of systems.

The definition of SME typically varies from author to author. For a more rigorous discussion of the Electronic Invoice thematic, we should define the SME following the 2005 European Commission. It states that SMEs are autonomous, partner or linked enterprises with 10 to 250 employees and a total turnover of less than EUR 50 million or a balance sheet total less than EUR 43 million [Commission, 2005].

Adoption of the Electronic Invoice can be seen as an innovation for SME based on the fact that innovation is something that is new to the adopting organization. Technological innovations present this companies with new means for solving problems and exploiting opportunities [Sundström, 2006].

This Risk vs. Benefits analysis will be done taking into account two kinds of benefits [Sundström, 2006]:

- Direct benefits, which refer to increased internal efficiency of an organization such as cost and time savings;
- And indirect benefits, which in turn refer to the impact of the Electronic Invoice on business processes and relationships.

Some of the direct advantages are listed below [IDC, 2007]:

- Cost savings can be achieved by time savings due to reduced manual paper handling;
- Automation frees staff resources for other tasks;
- Reduced costs with postage and storage;
- Real time knowledge about invoice status;
- Immediate invoice delivery reduces the paying time;
- On-line dispute resolution;
- Better business intelligence.

The indirect benefits are listed below:

- Reduced paper handling is beneficial to the environment;
- From the society point of view, using rational and environment friendly techniques can be very positive;
- Strengthening customer relations often leads to long-term relationships;
- Closer collaboration between supplier and buyer leads to improved customer loyalty;

For a SME with a non IT core business there is a lot of uncertainty and risks involved in the adoption of Electronic Invoice. To prove this point of view we had a discussion with Lismarketing’s steward, and we could understand that part of this risk is due to inadequate resources and limited knowledge about the new technology. He stated that the Electronic Invoice isn’t used because other business partners do not use-it.

---

12Pedro Marques works for Lismarketing and is responsible for the supervision and operational support of the 1st floor warehouse of Mercado da Ribeira, acting as an intermediary between the company and all the suppliers.
and so adopting such technology would oblige Lismarketing to maintain the two systems, one for Electronic Invoice, and another for paper invoice. This would impose a burden that probably a small company like this couldn’t support.

This brings us to the risks associated with the adoption of electronic invoicing. As described in [Sundström, 2006], there are five kinds of risks in this area:

- Social aspects: By optimizing manual intensive processes, some administrative staff members will be relocated or possibly made redundant. This can impose a massive resistance to change.

- Technical obstacles: Integration with the organization’s existing business system is often required. Legacy and sometimes self-made business systems make this process complex and expensive.

- Competence requirements: New electronic invoicing techniques demand new knowledge about these new techniques. Companies also have to spend time and money giving formation in this area.

- High investment and integration costs: the introduction of Electronic Invoice implies an initial cost. This typically varies with the type of organization and with the objectives to reach. This is very important because usually non IT companies have difficulty in measuring the ROI of IT.

- Security issues: the Electronic Invoice system that uses the Electronic Invoice must be developed in a way that ensures operational reliability and minimizes the risks for communication breakdown, unauthorized accessibility and manipulation of information. Once again, organizations with a weak IT culture have problems understanding these kinds of needs, and usually there’s a communication barrier between the IT and business staff. Here legislation compliance also must be addressed.

The solution in which invoices are sent directly over electronic means allows the full potential of an effective electronic workflow. However, this is the most expensive scenario to implement for a private organization [Brun and Lanng, 2006]. It is more attractive to larger enterprises or companies with a large savings potential due to a big amount of invoices being sent to the business partners.

SMEs often do not have an infrastructure to send an Electronic Invoice or the means to extend their infrastructure in this way. Hence, they tend to rely on invoice portals or one scanning agency [Brun and Lanng, 2006].

To conclude this analysis, its our believe that if a SME realizes that the benefits outweigh the risks involved, then it is more likely to adopt the Electronic Invoice in a short period of time.
Chapter 3

Problem Definition

In this section we aim to describe the main requirements for the development of an Electronic Invoice solution. Then in the following subsections we will discuss the implications of the more relevant requirements, specifically those required by law. In terms of further development, it is important to notice that not all requirements will be implemented. For these we will at least describe how they could be implemented and integrated in the final solution.

Process automation is a recurring problem in today’s businesses; companies are seeking to provide solutions to these problems as a means to restructure aging bureaucratized processes in an attempt to achieve the strategic objectives of increased efficiency, reduced costs, improved quality, and greater customer satisfaction [Grover and Kettinger, 1995].

In this case we focus on the automation of the invoicing process between two companies (B2B) on the Portuguese market. This is important because as we saw in the previous sections, a solution of this kind has to comply with country specific law requirements, and at the same time provide the means for flexibility and extensibility for integration with other country specific requirements.

Thus problem we have in hands resumes to a few aspects that need to be addressed, they can be grouped into three distinct domains:

- The challenges imposed by B2B interactions;
- Requirements imposed by legislation;
- Specific Technological Requirements.

3.1 The challenges imposed by B2B interactions

Electronic commerce is not simply about business transactions that run over the Internet, but is fundamentally about the flow of information [Yang and Papazoglou, 2000]. Nowadays the boundaries of organizations are more relaxed than they used to be and this propels the automation of relationships with business partners.

\footnote{The requirements document is presented in Appendix A.}
As a consequence, alliances in which businesses join their applications, databases, and systems to share costs, skills, and resources in offering value-added services, are being formed. The ultimate goal of B2B electronic commerce is therefore to have inter- and intra-enterprise applications evolve independently, yet allowing them to effectively and conveniently use each other’s functionality Medjahed et al., 2003.

One of the most important challenges in B2B electronic commerce is interaction. It is defined as consisting of interoperation and integration with both internal and external enterprise applications Medjahed et al., 2003. Interactions in B2B electronic commerce offer unique challenges because of issues such as scalability, dependability, autonomy, heterogeneity, coupling among partners and security. B2B electronic commerce requires the integration and interoperation of both applications and data. Disparate data representations between partners’ systems must be dealt with. Interaction is also required at a higher level for connecting (i) front-end with back-end systems, (ii) proprietary/legacy data sources, applications, processes, and workflows to the Web, and (iii) trading partners’ systems Medjahed et al., 2003.

In the next subsections we shall briefly describe each challenge imposed by B2B interactions.

3.1.1 Scalability

Scalability refers to the ability of a system to grow in one or more dimensions such as the volume of accessible data, the number of transactions that can be supported in a given unit of time. In this work we want to expand the term scalability to the number of relationships supported, as well as the need to provide mechanisms to allow that new relationships (and possibly new message formats) can be added or removed with a minimal operational impact in the system. Competitive advantage comes from non-stop availability in these changes. Ideally solutions shouldn’t be taken down for updates, performance tuning, system reconfiguration or system failure Lin, 2001.

More importantly, changes in business climate are forcing organizations to merge in order to be effective in the global market. Thus, the cost and effort to support new relationships is an important criterion to consider when evaluating interaction solutions in B2B electronic commerce. Clearly, a low cost establishment of new relationships is desirable. However, in case of long-term relationships, the cost of establishing a new relationship is not of great significance Medjahed et al., 2003.

3.1.2 Dependability

When talking about electronic business between trading partners availability of the systems is one of the most important challenges. High availability can be defined as the implementation of a system design that ensures a high level of operational continuity over a given period of time. “Operational continuity” in this case refers to the level of service that the applications, the services, and the systems as a whole provide to you, your customers, and your employees.

Availability is thus subjective: your availability needs will always center on your individual business needs in a given situation. “High availability” can therefore describe a variety of business goals and technical requirements, from hardware-only targets to mission-critical targets of the service as a whole. The common element to all of these goals is minimizing time that workloads are down or otherwise unavailable.
Whether the task is to avoid complete loss of a service or minimize the degradation of a service’s performance, high-availability solutions rely on the simple principle of supplying redundancy to possible points of failure to maintain service availability. The bottom line is that when referring to availability in an electronic business B2B environment, downtime translates into significant loss of money and consequently, business partners.

3.1.3 Autonomy

Autonomy refers to the degree of compliance of a partner to the global control rules. Partner systems may be autonomous in their design, communication, and execution. Autonomy implies that enterprises should be able to behave as autonomous entities, hiding their internal decisions, activities and processes. Information systems that manage B2B relationships in each enterprise have to be independent [Villarreal et al., 2003]. This means that individual partners select the process and content description and the programming and interaction models with the outside world, etc. In a fully autonomous collaboration, each partner is viewed as a black box, that is able to exchange information (i.e., send and receive messages). Partners interact via well-defined interfaces allowing them to have more local control over implementation and operation of services, and flexibility to change their processes without affecting each other. Usually, a completely autonomous collaboration may be difficult to achieve because it may require sophisticated translation facilities.

3.1.4 Heterogeneity

Heterogeneity refers to the degree of dissimilarity among business partners. The need to access data across multiple types of systems has arisen due to the increased level of connectivity and increased complexity of the data types. Applications use different data structures (e.g., XML, relational databases), standard or propriety semantics (e.g., standardized ontologies). There may also be structural heterogeneity at the business process layer (e.g., use of APIs, document exchange protocols, inter-enterprise workflows). In addition, organizations may, from a semantic point of view, use different strategies for conducting business that depend on business laws and practices [Casati et al., 2001].

3.1.5 Coupling among partners

This dimension refers to the degree of tightness and duration of coupling among business partners. Two partners are tightly coupled if they are strongly dependent on each other. For example, one partner may control the other, or they may control one another. Loosely coupled partners exchange business information on demand. The duration of a B2B relationship may be transient (also called dynamic) or long term [Medjahed et al., 2003]. In transient relationships, businesses may need to form a fast and short term partnership (e.g., for one transaction), and then disband when it is no longer profitable to stay together. Businesses need to dynamically discover partners to team up with to deliver the required service. In long term relationships, businesses assume an a priori defined partnership.
3.1.6 Security

Security is a major concern for inter-enterprise applications. Before B2B electronic commerce reaches its real potential, sophisticated security measures must be in place to boost electronic commerce partners confidence that their transactions are safely handled \cite{Yang and Papazoglou, 2000}.

B2B applications must support mutual authentication, fine grain authentication, communication integrity, confidentiality, non-repudiation, and authorization. These requirements are also addressed by the specific country legislation.

Nevertheless B2B interactions may be based on limited mutual trust, little or no prior knowledge of partners, and transient collaborative agreements. Shared information may include limited capabilities of services.

3.2 Requirements imposed by legislation

In the Portuguese case, there are a set of crucial aspects that every Electronic Invoice solution must address:

- Invoice Format;
- Invoice Integrity;
- Invoice Confidentiality and Privacy;
- Audit;
- Authenticity of origin;
- Non-repudiation of the origin and the destination;
- Invoice Archive;

3.2.1 Invoice Format

The integration needs of electronic business initiatives are driven by enterprise application software implementations and electronic commerce initiatives. Using standards to facilitate the communications between two different systems is the most promising approach to facilitating electronic business integration \cite{Chen, 2003}.

There are several compelling reasons for adopting electronic standards. First, standards prevent IT users from locking in with any specific vendors; therefore, IT users have more bargaining power in acquiring IT products. Second, applications or tools based on standards can be integrated easily. This allows IT implementers to take the "best of breed" approach to build their system architecture in achieving their business and technical objectives \cite{Chen, 2003}.

Since its appearing, XML as been one of the most used standards to represent and structure business information; both inside companies, and with outside partners through the use of well defined message formats. XML is a simple, very flexible text format derived from SGML (ISO 8879) originally designed to meet the
challenges of large-scale electronic publishing. XML is now also playing an increasingly important role in the exchange of a wide variety of data on the Web [W3C, 2008a], allowing its users to define their own elements and the information they represent.

Due to the variety of businesses and the lack of semantics in XML, the normalization committees have been developing various standards for electronic business. This has led to the proliferation of standards, and today there are a hand-full of formats that can be adopted in the invoicing scenario [Medjahed et al., 2003]:

- Commerce XML (cXML)
- Common Business Library (xCBL)
- Electronic Business XML (ebXML)
- GS1 XML
- Universal Business Language (UBL)

This has caused some confusion among companies that didn’t know which standard better suited their needs.

Having this into account, the adoption of a standard for the Electronic Invoice solution is very important. It is crucial to know our business partners, with which standards they operate, and the market we’re in, and only after that we can better judge on the standard we should adopt. This has major implications in terms of interoperability, and conditions the way and the number of transformations we need to perform on messages in order to communicate with our partners.

The requirement we conclude from this section is that we need to adopt a message format that better represents the information we need, and at the same time provides the necessary flexibility to accommodate future changes in the legal required information.

### 3.2.2 Invoice Integrity

Integrity is the protection of information in systems from intentional or accidental unauthorized modification [Bashir et al., 2001]. This is another important issue that must be addressed at two stages in the invoicing process: first in the archiving of documents, and second in the document transmission between two business partners.

In terms of document archiving, information can take various shapes and include complex multipart documents. In a long-term archive these documents may be expected to undergo multiple transformations during their lifetime, including, for example, format changes, modifications to sub-parts and to accompanying metadata. Skeptical users of a digital archive may desire, or in some case may be legally required, to verify the integrity of records that they have retrieved from the archive [Haber and Kamat, 2006].

In document transmission there are also problems regarding attacks (e.g. man in the middle) or network problems that alter the payload at the bit level.

To guarantee the integrity of documents, or to some extent to guarantee that changes in a specific document are detectable in each case, typically one may use cryptographic hash functions as a transformation that compresses input bit-strings of arbitrary length to output bit strings (also called digests, checksums or digital
fingerprints) in such a way that it is computationally infeasible to find two different inputs that produce the same hash value [Haber and Kamat, 2006], and that this digest uniquely identifies the original text. The most used algorithms are MD5 and SHA-1.

3.2.3 Invoice Confidentiality and Privacy

In document transaction confidentiality and Privacy ensures that no one other than the sender and the designated recipients can read the Electronic Invoice.

On the other hand it is also necessary to guarantee confidentiality and privacy in the storage of invoices. A proper access control mechanism can be used to limit the access to information as well as the use of public and private keys to guarantee another level of security.

3.2.4 Audit

The act of audit is typically an evaluation of a person, organization, system, process, project or product. Audits are performed to ascertain the validity and reliability of information, and also provide an assessment of a system’s internal control. In the Electronic Invoice thematic audit authorities may require on demand automatic generation of data reports concerning business data used in a specific window of time.

These reports are called summary maps and require specific data:

- Number and invoice date;
- Date and time of creation of the Electronic Invoice;
- Time of sending;
- Fiscal identification of the sender and the receiver;
- Quantity and name of the traded goods or services;
- The price, net of tax;
- Rates and the amount of tax due;
- Descriptions of the deficiencies detected during each transmission;
- Description and version of the software used.

Besides this, information should be provided in full or by the application of selective criteria of search data. Also, the information contained in the summary map should be accessible on-screen, in electronic format or be made available on paper.

Business partners should have in mind that although confidentiality and privacy are guaranteed to external entities, there is this special case where in fact information must be collected for posterior analysis.
3.2.5 Authenticity of origin

A fundamental issue faced by many companies operating on B2B markets, is how to ensure that you can trust a party that you are dealing. Upon receiving a business document there must be guarantees that in fact the source is who it claims to be.

In our problem, when an Electronic Invoice is received typically the recipient will check its authenticity before moving to the next activity in the invoicing process.

3.2.6 Non-repudiation of the origin and the destination

A party, person or entity having sent or received information cannot deny having taken part in the exchange and repudiate the information content. Depending on whether there is irrefutable proof of origin, receipt or content of the electronic information, there is non-repudiation of origin, non-repudiation of receipt or non-repudiation of content.

Concerning the Portuguese legislation we must focus on the non-repudiation of origin and receipt. This can be achieved collecting data while monitoring business transactions or by using communication protocols that produce information insuring that a transaction really happened and which parties were involved.

Another possibility is to use digital time-stamping. Digital time-stamping scheme is a procedure that solves the following problem: given a digital document x at a specific time t, produce a time-stamp certificate c that anyone can later use (along with x itself) to verify that x existed at time t. Certificates that will pass the verification test should be difficult to forge [Haber and Massias, 2005].

3.2.7 Invoice Archiving

Invoice archiving imposes interesting challenges due to the long term periods of storage that documents are exposed to.

If in the transmission of business documents between partners the challenge of guaranteeing integrity and authenticity are limited to a short period of actual data transmission, in document storage the same requirements are extended to large periods of time (in Portugal, law requires a minimum of 10 years) that documents are archived.

Certificates used to sign these documents are typically issued by a certificate authority that usually limit the certificate’s validity time in order to prevent attacks due to long time key exposure.

This fact imposes a challenge in terms of the model used to legally manage the archive so that integrity and authentication are guaranteed, and at the same time provide the infrastructural means to external audit entities to evaluate the stored information. An important issue here is how these entities can also have access to certificates to gain access to the stored information. Or should the information be sent to these authorities solely based on gentlemen’s trust deal?

Other challenges we face in archiving are related with data loss due to database corruption. Mechanisms of recovery or backup must always be contemplated. Due to the B2B characteristic of this problem, data
replication may occur if both partners have their own archives. These aspects must be described in some sort of contract that states what measures must be taken in these cases.

### 3.3 Specific Technological Requirements

The final domain we must address is related with the specific technological requirements. More often than not, integration projects have to deal with existing IT architectures and existing software.

In this dissertation one of the requirements, in addition to the development of a SOA solution based on BizTalk Server, was the integration with a SAP environment using the BizTalk’s pre-packaged SAP Adapter.

The idea was to prove that it was possible to develop a fully functional Electronic Invoice solution, and then show that the process of integration of this same solution with existing ERPs using adapters was simple and fast enough to compete with, for example, with the development of Web Services.

The main challenge resides in the understanding of BizTalk and SAP interoperability: which mechanisms are available, in which format the messages need to be exchanged and the necessary SAP and BizTalk configurations. We aim to explore the functionalities of BizTalk Server 2006 R2 and SharePoint Services 3.0 and to find the limitations that these two main technologies have and that can affect the developing of an invoicing process. One important aspect is that we want to minimize the effort of the developer by using as much off-the-shelf functionalities as possible and only using APIs when absolutely necessary. When limitations are found, solutions should be presented and implemented to verify their validity.

### 3.4 Dissertation Goals

In order to properly evaluate this work, it is necessary to define a set of goals and measures that can confirm if the objectives were correctly achieved in the end of this dissertation.

The main objective of this solution is to implement an Electronic Invoice Solution that can be used in two main scenarios:

- In a single company scenario, where a company uses this solution to send Electronic Invoices to one or more business partners;
- In an intra-group B2B scenario where companies provide services one to another and there is a central ERP system (e.g. SAP R/3) that manages a set of companies. This solution would allow sending and receiving Electronic Invoices between companies, in and out the SAP System.

We set the objective of achieving autonomy, decoupling and heterogeneity by identifying and developing a set of reusable services using a SOA approach and by using standard XML message formats that can guarantee the interoperability between systems.

In terms of scalability we are aiming to a solution that has great flexibility in the adding and removing of business partners with its own message formats as well as achieving a minimum batch processing capabilities. In the end of this work it is important to have shown how easy is to add a new business partner and how fast we can adopt a new message format. We also want to have a notion of the capacity of batch processing,
and what is the relation between the number of transactions processed in a given moment and the number of errors occurred.

Regarding the compliance with legislation requirements, we want to fully explore BizTalk and SharePoint to see if they are capable of answering these necessities explained earlier. Another important issue is that we want to study if the UMIC recommendations are "doable" and if they fit well with the BizTalk SOA environment.

Another objective is to find out if it is possible to use SharePoint Services as Electronic Invoice archive, and if it is capable of providing the adequate methods of search and security through the built-in access control mechanism having into account the audit necessities and other companies’ user access.

In order to enrich this work, an important objective is to provide a mean to explore business data used on the daily basis. Using Business Activity Monitoring we want data that is stored in the invoice payload to be collected from the message engine and shown in a relevant manner in order to produce information that can be used in business intelligence.

In order to support multiple business partners simultaneously, BizTalk must maintain at least one different certificate per company. Right from the beginning of this work we knew that this would be a challenge because by default BizTalk only supports one encrypting certificate per host instance. As so, and in order to answer the legal aspects of the invoicing process, another important goal is to develop a mechanism that allows each company to have its one certificate to sign Electronic Invoices.

Finally one big objective is related with a specific business case scenario; where we want to integrate with a SAP R/3 system add some extra value besides the capacity of creating Electronic Invoices, like for instance the implementation of AS2 exchange protocol, implementation of Business Activity Monitoring allowing data exploration and the creation of summary maps, and the implementation of the means necessary to search and audit the stored and traded invoices.
Chapter 4

Solution Proposal

After identifying the main problems that arise in the development of an Electronic Invoice solution, we shall now see how each problem can be addressed.

Here we have to provide the answer to two aspects. First we have to resolve the problems from a theoretical point of view, and then we must see how the chosen technology can provide the necessary support to these solutions.

For starters we present (in Section 4.1) the Service Architecture that supports our solution proposal, then we provide a conceptual way of answering the main requirements, and finally we will undertake an overview of the key aspects of the technology, understanding how it could support and ease the task that we have at hands.

4.1 Service Architecture

To better structure our proposal a Service Architecture was developed having into account a set of business events identified. These business events represent the activities that the solution must support. A business event is something that [Gold-Bernstein and Ruh, 2004]:

- Occurs in the business environment;
- Occurs at a given point in time;
- Must be responded to by the system.

The first step of the definition of the Service Architecture consisted in the identification of these same business events. Then for each business event identified, we created a description that explains in what it consists and how it is initialized. We also analyzed what responses were needed to answer to the business event in a correct manner.

The Business Events Table is presented in Appendix B on Figure B.1.

Then the system responses defined in the Business Events Table were used to determine the essential services the system must provide. One important matter, which we will see in the next sections where we explain in
more detail how each main requirement is answered, is that some of these services already exist as part of the technology that we shall use.

So, the next step consisted in the creation of a Service Category Table (presented in Appendix B on Figure B.2) that lists all required responses to business events, and defines whether the functionality already exists in one or more systems, or if it is new functionality the needs to be developed. At this step this table also defines likely services to provide the functionality that are grouped in the main categories:

- Integration
- Transformation
- Core
- Business Activity Monitoring

After this we needed to describe each service at a level sufficient to know which main activities or functions were necessary to accomplish the service’s objective. In this way we constructed a new table (presented in Appendix B on Figure B.3) that groups the main service functions per service category. We also defined how each service could be implemented.

One main aspect of this architecture is that the services we’re identified at a level of granularity that maximizes their reusability.

In the next sections we will understand how each business function will be implemented and how the technology interoperability will help us to achieve our goals.

### 4.2 Invoice Format

Having into account that the use of XML in business documents is crucial in terms of interoperability between business partners, we must adopt a XML internal format that:

- Provides support for the invoicing information;
- Has flexibility in terms of evolution and introduction and/or removal of invoicing information;
- Is suited for exchange over the internet;
- Supports easy transformation between formats;
- Is independent of the context.

Being in the Portuguese market, and as proposed by UMIC, we decided to use UBL 2.0 specification\(^1\) to maintain invoicing information.

In Appendix C is presented the UBL recommendation by UMIC. Here is included all the necessary invoicing information, the validation rules, and the mappings to UBL fields.

The existence of a unique internal format makes it easier to integrate with backend systems, and allows the development of an intermediate layer of services that do all the necessary transformations between received documents and sent documents.

\(^1\)UBL 2.0 Specification is available at http://docs.oasis-open.org/ubl/cs-UBL-2.0/UBL-2.0.html
Figure 4.1: Three category services.

Figure 4.1 shows the intermediate transformation layer that after the arrival of an invoice (at the integration layer), provides the means for a unique core processing format. These transformations will be done using Extensible Stylesheet Language Transformations (XSLT), which are supported by BizTalk Server.

So why is an internal format an advantage? Figures 4.2 and 4.3 exemplify. If we had n input and output formats, we would have at least n(n-1) transformations \[\text{Wüstner et al., 2002}\] to perform (greens arrows only determine the flux of information through the core processing component and not an actual transformation). Adding to this transformation process we would have a highly complex core processing for the Electronic Invoices.

![Figure 4.2: N to N transformations without internal format.](image1)

![Figure 4.3: N to N transformations with internal format.](image2)

On the other hand, if we had an internal format, we would reduce the number of transformations to 2n \[\text{Wüstner et al., 2002}\], but more important than that, we totally simplify the core processing of the Electronic Invoice, and that is the main conclusion we can extract from this section.

This also provides the flexibility so that in the future more formats can be added with a minimum impact in the global solution, the only major impact we should expect is the transformation time that is always needed when converting between two message formats. Nevertheless bottom line here is that we’re aiming at a clean and simple solution.

37
4.3 Electronic Invoice Archive and Process Visibility

In the previous chapter we described the main problems raised by the archival of business documents. One of these problems referred to the necessary mechanisms in terms of data recovery due to corruption of the databases. Although identified, this problem will not be addressed, and will remain as a future work recommendation.

In terms of the proposed solution for archival we will use Windows SharePoint Services (WSS) as the main platform. As we will see further ahead in this chapter, WSS provides an Enterprise Service Bus with all the necessary services for storage and security, allowing a good alignment with the SOA objective.

The final goal is to have a storage layer with specific company repositories (see Figures 4.4 and 4.5) where invoices will be stored in three categories: first, by company specific repositories, then by sent or received transactions with other partners, and finally by a separation of signed invoices and to visualize invoices. This separation allows different visualization methods using InfoPath templates, according to user desire.

Another important matter is that we don’t have to create a domain model from scratch to store information. WSS provides, with the required administrator privileges, an out of the box interface that allows a fully customizable repository, creation of access control policies using groups, users and actions over objects.

It is possible to configure the system so that a specific user only has access to his company repository, or for audit reasons, it is possible to create users that can have total access to the information that passed through the system. These users can have access to all documents, times and dates, items, or other critical data that is needed for the audit process.

Also, WSS will provide the interface and visibility (see Figure 4.5) to all the invoicing process through BAM Portal and integration with BizTalk’s messaging engine. As soon as an digital invoice arrives (by web service invocation, file system, or direct integration using adapters), the core processing component will archive the now Electronic Invoice in the specific company repository and update, through the Business Activity Monitoring activities, the views over business data consumed and produced by the process, as well as business events, real-time business intelligence (BI) and the predefined reports of the summary map and AS2 non-repudiation data.
4.4 Invoice Integrity and Authenticity

The integrity and authenticity of the documents must be guaranteed during the transactions between business partners. For this UMIC recommends the use of S/MIME signatures with the algorithm RSA and with SHA-1 for hashing purposes.

Content integrity guarantees that the receiving trading partner gets the data in its originally sent form and assures that no modifications (i.e., additions, deletions, or changes) have been made to the data when it is in-transit between trading partners.

This is achieved if the sender provides with the data content, a digital signature, which includes an integrity control value. This value can be computed by using an appropriate cryptographic algorithm to "fingerprint" the data content. These cryptographic algorithms are called one-way hash functions or message integrity checks. One-way hash functions don’t require a key, and the hash algorithm typically used is either SHA-1 (Secure Hash Algorithm 1) or MD5 (Message Digest 5).

As a method to determine content integrity, the sending trading partner adds a digital signature to his data content, which includes a one-way hash value of the data content and the MIME content headers. This value is unique and "fingerprints" the transaction. The sending trading partner sends the hash value along with the data. The receiving AS2 trading partner using the same one-way hash function calculates the hash value for the received data content and the MIME content headers. If the received hash value matches the calculated hash value, then the receiving trading partner is assured that the data content has not been tampered with or altered in any way.

S/MIME uses PKCS (Public Key Cryptographic Standards) to provide the mechanism for digital signatures and data encryption. In order to sign and/or encrypt a MIME message, at least one public/private key pair is needed. The public key is provided to users with whom secure communication is desired.

The sender’s private key is used to digitally sign a MIME message. When this message is received by the recipient, he uses the sender’s public key to verify the digital signature. For encryption, the sender uses the recipient’s public key to encrypt the MIME message. When the message is received by the recipient, he uses...
his own private key to decrypt the message.

As long as the private key is protected and is accessible only by the user to which it was assigned, the recipient of a digitally signed message will know from whom the message was sent and the originator of an encrypted message will know that only the intended recipient is able to read it and it has not been tampered with in-transit.

To use S/MIME, digital certificates are needed. UMIC recommends the use of X509 certificates that must be obtained from a certificate authority; if not trading partners should agree and self-certify each other.

The UMIC recommendations fall in perfectly with BizTalk, in the way that they are totally supported by this technology. In fact BizTalk has pipeline components that allow a total automation of the signature process using S/MIME.

### 4.5 Non-Repudiation

An electronic invoicing solution must also guarantee that in a transaction the relations between the sender, the recipient and the invoice are maintained for audit and traceability reasons. Once again UMIC recommends the use of the Applicability Statement 2 (AS2) specification to transport data securely and reliably over the Internet.

The AS2 protocol is based on HTTP and S/MIME. Files are sent as "attachments" in a specially coded S/MIME message (an AS2 message). AS2 messages are always sent using the HTTP or HTTPS protocol, Secure Sockets Layer (SSL) is implied by HTTPS, and usually use the "POST" method. AS2 messages can be signed and/or encrypted, but do not have to be.

AS2 specifies the means to connect, deliver, validate, and reply to (receipt) data in a secure and reliable way. AS2 does not concern itself with the content of the EDI document, only the transport.

To transport data securely and reliably over the internet, AS2 uses the MDN (message disposition notification) mechanism that basically acts like a receipt that proves that the sending and receiving of the invoices really took place, and which parties were involved. The AS2-MDN exists in two varieties: synchronous and asynchronous [IETF, 2005].

- The synchronous AS2-MDN is sent as an HTTP response to an HTTP POST or as an HTTPS response to an HTTPS POST. This form of AS2-MDN is called synchronous because the AS2-MDN is returned to the originator of the POST on the same TCP/IP connection.

- The asynchronous AS2-MDN is sent on a separate HTTP, HTTPS, or SMTP TCP/IP connection. Logically, the asynchronous AS2-MDN is a response to an AS2 message. However, at the transfer-protocol layer, assuming that no HTTP pipelining is utilized, the asynchronous AS2-MDN is delivered on a unique TCP/IP connection, distinct from that used to deliver the original AS2 message.

The advantage of the synchronous MDN is that it can provide the sender of the AS2 Message with a verifiable confirmation of message delivery within a synchronous logic flow. However, if the message is relatively large, the time required to process this message and to return an AS2-MDN to the sender on the same TCP/IP connection may exceed the maximum configured time permitted for an IP connection. But since Electronic Invoices should be relatively small in size, this shouldn’t be a problem.
This applicability statement does not require the use of a certification authority. It is recommended that trading partners self-certify each other if an agreed-upon certification authority is not used. The use of a certification authority is therefore optional. Certificates may be self-signed. In terms of implementation this was the path chosen in order to simplify the development of the solution.

Figure 4.6 exemplifies the process of sending and receiving AS2 messages in a scenario where the BizTalk server is used as integration engine.

Invoice information is extracted from an ERP system and is certified by the invoicing process orchestrated in BizTalk Server. At the end of this process the Electronic Invoice is enveloped in an AS2 message and sent to the trading partner. At the same time BizTalk updates its own non-repudiation database.

Upon the receipt of the message and its successful decryption or signature validation (as necessary) an MDN "success" message will be sent back to the original sender. This MDN is typically signed but never encrypted (unless temporarily encrypted in transit via HTTPS).

Upon the receipt and successful verification of the signature on the MDN, the original sender will "know" that the recipient got their message (this provides the non-repudiation element of AS2) and can thus terminate the transaction on the non-repudiation database.

If there are any problems receiving or interpreting the original AS2 message, a "failed" MDN may be sent back. However, part of the AS2 protocol states that the client must treat a lack of an MDN as a failure as well, so some AS2 receivers will simply not return an MDN in this case.

AS2 is also known as EDIINT or EDI over the internet. AS2 essentially creates a wrapper around EDI or flat files that enables sending them over the Internet, instead of using a connection to a value-added network (VAN). AS2 provides security and encryption around the HTTP packets. It enables information transmitted over public and private global networks to be digitally signed, secured, and non-repudiated.
4.6 Technological Interoperability

This section addresses the technology that will be used in further implementation. The goal here is to have a general view of the main interactions between technological components.

Before going into a more detailed description, Figure 4.7 depicts a view of the platforms and software components that will be used.

![Figure 4.7: Global view of the technology.](image)

This picture shows the main technology used in this work. BizTalk Server message oriented middleware (MOM) will be used as an orchestration engine that will define the necessary activities for Electronic Invoice certification, and secure communication between trading partners.

Microsoft’s BizTalk Server will act as a message broker and will be supported by SQL Server and Windows SharePoint Services for archive purposes. Additional interoperability with the Office System is also supported through InfoPath and Excel. Eventually Portable Document Format (PDF) Electronic Invoices will also be generated by BizTalk Server for email delivery or notification.

BizTalk Server will receive invoices in a digital format and will apply all the necessary transformations, answering all necessary legal requirements before forwarding the (electronic) invoice to another system.

Note the input semantic of digital invoice, i.e., the invoice that doesn’t yet comply with all the necessary legal requirements, and the output semantic of the Electronic Invoice, that has already passed through a process that, due to a set of well defined activities, will certify the invoice in legal terms.

Now, in the following subsections, we will describe each main software component and analyze how they interact with the integration engine. In terms of business support systems we will only focus on SAP R/3, since there was some interest, in terms of research and implementation, from the company where this dissertation was realized.

4.6.1 BizTalk Server 2006 R2

BizTalk Server is a business process management (BPM) server. Through the use of adapters which are tailored to communicate with different software systems, it enables companies to automate and integrate
business processes. In a common scenario, BizTalk enables companies to integrate and manage business processes by exchanging business documents such as purchase orders and invoices between disparate applications, within or across organizational boundaries.

In the next subsection we will analyze the BTS Architecture so that we can observe how the messaging engine can be used to solve our Electronic Invoice problem.

**BTS Architecture**

The BizTalk Server runtime is built on a publish/subscribe architecture in which a message is published into the system, and then received by one or more active subscribers. Different flavors of this architecture exist, but the model implemented in BizTalk Server is often called content-based publish/subscribe [Microsoft, 2006b].

In a content-based publish/subscribe model, subscribers specify the messages they want to receive using a set of criteria about the message. The message is evaluated at the time it is published, and all of the active subscribers with matching subscriptions (indicated by filter expressions), receive the message.

BizTalk Server 2006 is, at its core, a message-handling engine [Microsoft, 2006a]. These messages are usually formatted following a specific XML Schema (also called XSD). XML Schemas express shared vocabularies and allow machines to carry out rules made by people. They provide a means for defining the structure, content and semantics of XML documents [W3C, 2008b].

Picture 4.8 illustrates the runtime architecture of BizTalk. In this simplified view, an incoming message is received through a receive adapter defined in a given receive port. This message is processed by the receive location and then published to the MessageBox (SQL Server) database, which is the main persistence and routing mechanism for BizTalk Server.

![BizTalk Runtime Architecture](chappell.jpg)

**Figure 4.8: BizTalk Runtime Architecture. Source:** Chappell, 2007

In this first processing phase, the message goes through a receive pipeline that can execute specific operations and/or apply a set of transformations to the message using pipeline components. These transformations may include message decryption, signature validation, flat-file to XML conversions, etc.
When the message reaches the MessageBox, the engine evaluates active subscriptions and routes the message to a specific orchestration or to a send port with a matching subscription.

Once an orchestration has processed a message, it typically produces another message destined for some other application. This message is placed in the MessageBox and then picked up by a send port.

A send port can have the same three components as a receive port, and they perform the same functions: mapping the message into its outgoing format, preparing that message for transmission in a send pipeline, then actually transmitting it to its destination using an appropriate send adapter.

All of this process is held together by subscriptions stored in the MessageBox. When a message is processed by a receive port, a message context is created that contains various properties of the message. An orchestration or a send port can subscribe to messages based on the values of these properties. For example, an orchestration might create a subscription that matches all messages of the type "Invoice", or all messages of the type "Invoice" received from the a specific organization, or in case of a broker architecture, all messages of the type "Invoice" received from a specific organization and with a specific destination.

Another important aspect is Business Activity Monitoring (BAM). BAM is a collection of tools that allow the management of aggregations, alerts, and profiles to monitor relevant business metrics (called Key Performance Indicators, or KPIs). It gives end-to-end visibility into business processes, providing accurate information about the status and results of various operations, processes and transactions.

The Figure 4.9 illustrates the interactions of the BAM component for BizTalk Server.

![Figure 4.9: BizTalk BAM Component Interactions. Source: Microsoft, 2007](image)

The information monitored in BizTalk Server engine is stored in tracking databases (SQL Server) and is made available using Web Services, being then accessible through different ways. In this case our interest goes to the BAM Portal that as direct integration with SharePoint Services.

This portal allows information workers to access data and select particular instances of the business processes and then choose specific views into these processes. The main task here is to create the right views for the Electronic Invoice scenario, providing different perspectives such as graphical depictions of per-invoice data, or more high level information like invoices sent by company, by time period, etc.

BAM views rely on one or more activities. An activity represents a specific business process (e.g. Send Invoice, Receive Invoice) with a set defined milestones and business data. This data needs be to be gathered from BizTalk messages inside orchestrations. Typically this is done using a tool (Tracking Profile Editor)
which does the mapping between specific fields in a XML message and the variables defined in the BAM views. But as we will see in the Implementation chapter, this can be done in a different way.

As this section suggests, a complete solution built on BizTalk Server contains various parts (also referred to as artifacts): orchestrations, pipelines, message schemas, and more. As we will describe in the next chapter, the development methodology was based in incremental proof of concepts that at each stage created a specific artifacts that answered the requirements of an Electronic Invoice solution.

4.6.2 Sharepoint Services 3.0

After briefly seeing how BizTalk handles messages since the receivement until the forwarding to a business partner, we can now focus on the technology used for message archivement. Because the typical invoicing process is somewhat manual, we propose the use a technology with strong integration capabilities with a set of productivity tools (e.g. office system) and with the BizTalk message engine.

Since we are focused on building a SOA oriented solution, the SharePoint Services (WSS) folds in perfectly in the task of Electronic Invoice archive. This technology provides two main benefits that are relevant in terms of the final solution. On one hand, having Sharepoint Services acting as a repository provides an out-of-the-box user interface with access control lists, content management, and workflow capabilities based on the stored documents.

This is very important, because we can design processes that when an Electronic Invoice arrives for storage it also triggers an approval workflow that can span different organic units in the organization, or we can design a process that integrate with existing document management infrastructure.

On the other hand, the SharePoint architecture provides a service bus (Figure 4.10) that allows total programmatic control over stored objects.

![Figure 4.10: Windows SharePoint Services Service Bus.](image)

This figure only demonstrates a few of the available Web Services, but it allows us to understand the ease of integration with BizTalk Server. Infact we only have to add a web reference to these services in our BizTalk project and the messages necessary for communication are automatically generated by BizTalk’s message engine. Then the only thing we have to do is orchestrate the calls to the services in order to fulfill our process.

Besides this, having SharePoint as the archive platform allows us to have an archival solution that is also compatible and extendable enough to communicate with other systems besides BizTalk. SharePoint Services also support the interface for BizTalk’s business activity monitoring module. This allows us to feed SharePoint with selected information gathered from the message engine interactions.
4.6.3 SAP R/3

Today, SAP is one of the world’s most popular Enterprise Resource Planning (ERP) products. Nevertheless a large number of SAP customers are also dependent on a variety of complementary applications built on other platforms like Microsoft .NET and J2EE. These organizations also run a variety of other packaged products for specialized tasks. More often than not, these applications need to exchange data between themselves, and with SAP [Nigam and Goyal, 2005].

A SAP transaction often impacts a business process, which then spurs a related transaction in another system. Microsoft BizTalk Server can orchestrate such a business process, allowing transmission of messages to and from the SAP environment. To simplify the process of getting data in and out of SAP, we will use Microsoft BizTalk Adapter v2.0 for mySAP Business Suite, which does not require any custom code development.

Figure 4.11: SAP R3 and BizTalk Server interactions.

Figure 4.11 depicts the interactions between a SAP environment and a BizTalk solution. The SAP environment has three main ways of integration with external applications, or in this case, with BizTalk Server:

- **IDoc** - Intermediate Document
- **BAPI** - Business Application Programming Interface
- **RFC** - Remote Function Call

As we will see further ahead, in terms of implementation we will focus only on IDOCs and RFCs.

**IDoc**

Intermediate Document (IDoc) is a standard data structure for electronic data electronic information exchange between any two SAP systems or between a SAP application and an external program. IDocs serve as the vehicle for data transfer in SAP’s Application Link Enabling (ALE) system. ALE uses IDocs to
One important aspect about IDocs is that they are used for asynchronous transactions: each generated IDoc exists as a self-contained text file that can then be transmitted to the requesting workstation without connecting to the central database. This central database can contain thousands of data tables consuming gigabytes of storage information on customers, vendors, employees, manufacturing status, inventory, shipping, etc. Accordingly, if (for example) a shipping facility only needs to read a limited amount of customer and inventory information in order to print labels for that day’s shipments, it might not be desirable to design and perform the complex remote queries necessary to tap directly into and search the central database.

SAP designed IDocs for exactly this purpose - for encapsulating, moving around, and processing just the sections of data needed for specific purposes.

The IDoc type indicates the SAP format that is to be used to transfer the data for a business transaction between SAP Systems and partner systems. The IDoc types define different categories of data, such as e.g. purchase orders or invoices. An IDoc is the instance of an IDoc type.

### BAPI

Another SAP mechanism, the Business Application Programming Interface (BAPI) is used for synchronous transactions. BAPIs are a set of interfaces to object-oriented programming methods that enable a programmer to integrate third-party software into the proprietary R/3 product from SAP. For specific business tasks such as uploading transactional data, BAPIs are implemented and stored in the R/3 system as remote function calls.

As we said before, BAPIs are used for synchronous transactions.

### RFC

RFC is the SAP variant of the Remote Procedure Call (RPC) [Schneider-Neureither, 2004]. It allows functions to be executed in other programs. There are different remote function call procedures to answer the different requirements of individual applications and their business processes:

- With synchronous RFC (sRFC), function modules in external systems can be called in real-time. With this type of RFC, the client waits until the server has completed processing.

- Long-running processing can be split into different function calls using asynchronous RFCs (aRFC), which are then processed in parallel.

- With transactional RFC (tRFC), data consistency and unique execution is ensured. Transactional RFC is asynchronous and assigns a transaction identifier (TID) to ensure that if data is sent numerous times because of network problems, it is recognized by the server. This helps to avoid data being processed several times, which can lead to erroneous information in the application.

- The queued RFC (qRFC) is an enhancement of the tRFC. With qRFC, the calls are placed in queues. This ensures that function calls are processed in a fixed sequence.

---

2 A logical system is a mechanism used to identify an individual client in a system, for ALE communications between SAP or partner systems.
SAP Gateway and ProgramIDs

All inbound and outbound connections to SAP must first go through SAP Gateway. SAP Gateway is a secure application server. No connections are accepted unless they have been preregistered previously from the SAP presentation Client. A server connection presents itself to the Gateway and exposes a program identifier. If the program identifier is found in the list of registered program ids, the Gateway server then offers a connection to the server, which accepts the connection. This program id is then linked with an RFC destination within SAP, which enables ALE documents (IDocs) to be routed to the destination. The RFC Destination functions as a tag to mask the program id to SAP users.

An RFC server program can be registered with the SAP Gateway and wait for incoming RFC call requests. An RFC server program registers itself under a Program ID at a SAP Gateway and not for a specific SAP system.

In the next chapter we shall see how to configure the program ids and the RFC destinations in SAP (as well as other SAP objects) and how program ids will be used in BizTalk to enable the desired interoperability.

Microsoft BizTalk Adapter v2.0 for mySAP Business Suite

As we saw earlier, BizTalk Server works with messages in XML format. When integrating with Idocs from a SAP system, it is necessary to tell BizTalk how to interpret these messages.

The Microsoft BizTalk Adapter v2.0 for mySAP Business Suite offers design-time and run-time components to support generation of schemas for SAP objects (BAPIs or Idocs), and configuration of routing information such SAP host name, usernames and passwords, etc.

But now let’s focus on the schema generation. This adapter allows the generation of XML Schemas based on IDocs gathered from a SAP System. Figure 4.12 illustrates the procedure.

![Figure 4.12: SAP Adapter for BizTalk Server.](image)

One must first configure the SAP configuration data (address, username, password, etc.) to connect to the remote machine. After this, the adapter fetches all available IDocs. Once the invoice IDoc is selected, the adapter fetches all IDoc metadata and uses it to generate a XML Schema (XSD) that is made available for usage in orchestrations inside BizTalk Server.
Chapter 5

Implementation

This implementation chapter depicts the main development process. To justify the creation of the orchestrations presented in this chapter we first analyze the relation between the Orchestration Architecture and the Service Architecture. After this we can detail the implementation methodology and comprehend why the definition of seven milestones to achieve our goal.

Finally we undertake a journey describing the more important aspects of the implementation of each milestone, the problems encountered and the solutions developed.

5.1 Orchestration Architecture

Before describing each orchestration, it is important to give a general overview of the Orchestration Architecture in the BizTalk solution (see Figure 5.1) and how it relates with the Service Architecture presented in the previous chapter.

![Orchestration Architecture](image)

Figure 5.1: Orchestration Architecture.

Since the objective is to send Invoices from one entity to another, we had the need to split the solution in send side and in receive side logics:

- Send side
  - Set of orchestrations that are responsible for extracting digital (uncertified) invoices from other systems
(through FileSystem, Web Service invocation or direct integration with a SAP system), and for the execution of a series of activities that certify the digital invoice in legal terms (transforming it into an Electronic Invoice) and sending it to the receive side of the solution.

- **Receive side**
  Set of orchestrations that are capable of receiving Electronic Invoices and are intended for integration with other systems. This receive side can be located on a remote machine.

Figure 5.2 shows the Orchestration Architecture composed with the Service Categories.

![Orchestration Architecture and Service Categories](image)

In an analogy with the Service Architecture, we can now identify where each main service category is located:

- **Integration service category** includes the integration logic orchestrations;
- **Transformation service category** includes the core entry and exit points;
- **Core service category** includes the core orchestrations;
- **BAM services** are supported between the two core logics. The idea is to cover the communications between two partners and log all transferred data so that we can then present business information in a more elegant way.

The justification for the logic separation is the creation of a solution that can operate exclusively using the send side logics, extracting information from an existing system and certificating it into a local repository and operating with a third party service provider, or if we need interaction between two different systems, we can use the receive side logics as a receive adapter that can interoperate with the send side. The six main orchestrations will be explained in more detail throughout this chapter.

## 5.2 Methodology

The development methodology was based on two main approaches: incremental development and the use of proof of concepts.
The incremental development is a scheduling and staging strategy allowing portions of the system to be
developed at different times or rates, and integrated as they are completed [Cockburn, 1993].

A proof of concept is a short and/or incomplete realization of a certain method or idea to demonstrate its
feasibility, or a demonstration whose purpose is to verify that some concept or theory is probably capable
of exploitation in a useful manner. It is usually considered a milestone on the way to a fully functioning
prototype.

In this way, the implementation methodology started with the definition of a series of milestones regarding
the requirements that needed to be answered and the Service Categories defined:

- Receive an invoice in a generic format and convert it to an internal format (UBL 2.0);
- Receive an invoice, convert and sign it;
- Store the signed invoice in the archive in two formats (original signed format, and visualization format);
- After storage, send the Electronic Invoice via AS2 to the business partner;
- Receive an Electronic Invoice via AS2, validate its signature and store it in the partner archive;
- Apply business activity monitoring (BAM) to all the solution;
- Integrate with SAP R/3.

After each milestone was reached, all the solution was tested and verified for conceptual errors in terms of
global invoicing process execution.

Then the output of each proof of concept was integrated with the existing solution. This allowed the
development of a system piece by piece and permitted additions to the requirements as well as improvements
in the development process.

At the same time, the milestones provided the necessary guidance, and allowed a division of the Electronic
Invoice problematic in smaller problems, easier to solve, identical to the divide and conquer strategy.

5.3 Milestone 1 - Implementation of the UBL 2.0 Format

In order to complete this first milestone it was necessary to know what kind of information exists in a typical
paper invoice. To reach this end it was necessary to ask UMIC for the legal information requirements and
which UBL 2.0 fields are necessary to hold the invoicing information.

UMIC supplied the information presented in Appendix C. These tables contain the necessary fields and a
brief description on the invoicing information, as well as the validation rules in order to verify the syntax
and semantics of used data. The most essential element of this Appendix is that it provides the mappings to
UBL fields.

In terms of implementation of the UBL format it is important to mention two aspects. First, although there
are optional fields in an Electronic Invoice, all were implemented; and second, the field validation rules were
not implemented because we decided to focus on the invoice as a whole rather than on the contents of the
invoice.
Nevertheless if these rules were to be implemented they could be developed using BizTalk’s business rules component. In this way they could be deployed independently of the other services and thus wouldn’t be closely coupled with the rest of the solution. The idea would be to support more flexibility in future changes in these rules.

As we saw, the orchestration architecture is composed by six main orchestrations that will be explained during this chapter. For now we will concentrate on the "Core Entry Point". This orchestration receives the digital invoice in any XML format and will test the format’s namespace. The namespace will be the key element that will differentiate the various supported XML formats.

Figure 5.21 exemplifies the decision process that starts when a XML formatted message is received.

![Figure 5.3: Core Entry Point Decision Point.](image)

The decision point will test the namespace using the following code:

**Code Listing 5.1 Core Entry Point Decision Expression for the ERPInvoice Format.**

```plaintext
1: ReceivedAnytypeMsg(BTS.MessageType) == 
   "http://eFactCORE.MessageTypes.maindoc.ERPInvoice#ERPInvoice"
```

After detecting the XML format a XLST transformation is applied. This transformation is done using BizTalk’s XML mapper that provides a user interface and generates the XSLT code behind.

The XSLT transformation copies the values from a source XML schema into a destination XML schema (UBL 2.0) (see Figure 5.4). For this task to be accomplished typically some kind of predefined mappings are necessary. If we needed to support other XML formats we would need to get these mappings in order to know how to transform the information into the UBL format.
Figure 5.4: Mapping Between ERPInvoice Format and UBL 2.0 Invoice Format.

For development and test reasons, it was developed a custom XSD called ERPInvoice (to act as a source XML schema) similar to the UBL specification so that we could have a test XML file that could simulate an digital invoice as input and could then be used to test the XSLT transformations into an UBL format.

In terms of extensibility to other formats, this solution simply requires a new decision branch (just like in Figure 5.21) and the creation of a new XLST. As we saw in the previous chapter, this is possible due to the advantages of the use of a well defined internal format.

### 5.4 Milestone 2 - Implementation of Electronic Signatures

UMIC recommended the S/MIME signature type for the Electronic Invoice solution. First of all, it was necessary to get certificates for test purposes. Since it wasn’t easy to contact a certification authority to help us in this issue, we used the Windows Certificate tool to generate a self signed X509 certificate from which we extracted the public and private keys to use in our solution.

The keys must be placed in specific Windows certificate stores (see Figure 5.5 and 5.6). The private keys were stored in the Personal Store and the public key in the Other People Store. This is where BizTalk searches for certificates when defining the administration properties further ahead.
Because the development environment was built in the same local machine this certificate scenario isn’t totally realistic. The private keys for the different companies are stored in the same Personal store and ideally they should be on the different Personal Stores of each company server.

BizTalk provides built-in mechanisms to encode and decode messages using S/MIME. Within BizTalk, the signature process is usually done in the pipeline processing using specific pipeline components. One must create a BizTalk project where a message is sent using a send port with a custom send pipeline. This pipeline must contain an S/MIME encoder that has as a property a certificate thumbprint which is then used to sign the XML message.

This property can be set only in development time, which in this case would obligate to recompile the solution each time a company changes certificate or each time a new company is added, or it can be set as a "per instance" configuration using BizTalk’s administration console which also did not resolve our problem because this configuration would affect all companies at once and not one specific.

Since we had the requirement of signing the Electronic Invoices using different certificates from each company, this situation posed as a serious drawback because we didn’t wanted to design new pipelines or change these properties each time a new company is added to the solution or each time an Electronic Invoice is sent from a different company.

Since BizTalk had this limitation, we had the need to find a way to assign to each company a certificate in order to univocally identify the owner of each Electronic Invoice. After consulting Microsoft Portugal on this issue, we came up with a solution that makes use of the AS2 Party properties, which was excellent since these properties are defined accordingly to each company’s needs and have a specific field where we can specify a certificate per company, (see Figure 5.7, more AS2 implementation details will be depicted further ahead in this chapter, for now we only focus on the certificate properties). The specified company certificate would be accessed through an API inside a custom pipeline with a custom pipeline component (that should be developed).

---

1Method where the send and receive ports properties are defined on BizTalk Administration Console. These configurations can be changed on a per instance basis and will not affect already running instances of services or orchestrations.
In this way we can create a party for each company and specify the signing certificate that each company will be using. Then, instead of hardcoding the certificate on the send pipeline on design time, the pipeline will know dynamically (based on the content of the invoice) which company is sending the Electronic Invoice and consequently, which certificate to use in the signing process.

Figure 5.8 illustrates the logic behind the custom pipeline component.

As the XML message passes through the pipeline component, routing information based on the content of the invoice is extracted and used to determine which party property must be accessed to get the signature certificate. Once the signature certificate is obtained, its thumbprint is extracted and used in the signature process.

In this way it was possible to overcome the limitations in terms of multiple signers and multiple pipelines. We developed a reusable pipeline component that was flexible enough to read in runtime the properties of the sender (based on the content of the invoice) and apply an S/MIME signature.

In terms of the general solution, we wanted to design an information flow where the invoices are only signed
once. This was very important for the implementation process because the signatures can only be applied in the execution of a pipeline. This situation created a problem because we needed to have the signed invoices inside the orchestrations for routing and archival purposes.

To overcome this problem we used a new functionality introduced in the BizTalk Server 2006 R2 which is the execution of send and receive pipelines inside an orchestration. This allowed applying signatures and to replicate invoices to the local archive and then send them to other trading partners. Otherwise it would be necessary to sign separately the invoice each time it was sent to the any destination (archive or trading partner).

This situation also occurs when an Electronic Invoice is received. BizTalk Server usually verifies the signature, certificate revocation list, etc. on the receive pipeline. After this step the message is passed to an orchestration without the appended signature. In this way, the receive pipeline is also executed inside an orchestration, so that all verifications are done in an environment where appropriate measures can be taken in case of a validation failure.

In terms of implementation measures to safeguard this situation were not applied and will remain as future work. Nevertheless in case of a signature validation failure the Electronic Invoice would be temporary stored and a notification sent back to the sending company via Email or other pre-accorded format with the invoice identification, failure details, and eventually invoice payload if necessary. The sender would then resend the Electronic Invoice with a different invoice identification allowing errors to be tracked for a pre-specified period of time.

Finally it is important to mention that the signature process is part of the core orchestrations and so they were implemented in both receive and send side logics as part of the process of verifying and signing Electronic Invoices, respectively.

5.5 Milestone 3 - Implementation of the Archive

The archive component of the solution was implemented in integration with Windows SharePoint Services 3.0 (WSS). This integration was realized through the WSS service bus. The service bus provides the flexibility necessary to guarantee a good decoupling between these two main technologies.

In terms of implementation, the interactions with the archive are part of the core functionality and so they reside in the core orchestrations.

After the execution of the custom send pipeline (that signs the invoice), the Electronic Invoice is archived in the local repository. For this, routing information is extracted from the invoice, namely the sender company name and the invoice identification. Then this information is used to create the necessary storage structure (as described in the solution proposal chapter).

The Electronic Invoices are stored in two formats:

- Unsigned InfoPath XML format (to visualization)
- Signed XML format
The InfoPath XML is created inside the core orchestration by adding a processing instruction (PI) (see Code Listing 5.2) that tells SharePoint that this file should be used with an InfoPath application.

**Code Listing 5.2** Creation of an InfoPath Processing Instruction.

```
1: CertifiedUBLInvoice(WSS.ConfigOfficeIntegration) = "no";
2: CertifiedUBLInvoice(XMLNORM.ProcessingInstructionOption) = 1;
3: CertifiedUBLInvoice(XMLNORM.ProcessingInstruction) = "<mso-infoPathSolution
PIVersion="1.0.0.0" href="http://efactcore:2610" +
"_forms/template.xsn"?>\<\mso-application
progid="InfoPath.Document"?>\<\mso-infoPath-file-attachment-present?>";
```

CertifiedUBLInvoice is the BizTalk message that contains the signed Electronic Invoice and that will be sent to the SharePoint repository. First (on line 1), we turn the context variable ConfigOfficeIntegration off in order to manually insert the InfoPath processing instruction into the form. Then (on line 3), we tell to BizTalk's message engine that we are going to add a new processing instruction and finally (on line 5) we actually create the PI by setting the XMLNORM.ProcessingInstruction context property with the specified value. This string value contains the InfoPath form template location in the SharePoint Repository.

InfoPath allows the creation of custom made forms that are then used as templates in SharePoint Libraries. The templates take as source data the information contained in the XML Electronic Invoices, and then apply an InfoPath form that allows better legibility and understanding of relevant information. With these two steps (adding a pre-instruction and creating a visual template) it is possible to develop various Electronic Invoice views based on specific user preferences.

The figure presented in Appendix D illustrates an XML Electronic Invoice using a visualization template created using InfoPath and associated to a company's SharePoint repository.

It is possible to define these templates according to user (i.e. company) specific needs. This is important in the way that not all companies need to view the same information and this so method allows a full customization with a minimum effort.

The other storage format is S/MIME signed XML format. For interoperability reasons, BizTalk appends the signature to the original message and then encodes automatically all signed messages in Base64 (see Code Listing 5.3).

**Code Listing 5.3** Example of a S/MIME signed Electronic Invoice.

```
1: MIME-Version: 1.0  Content-type: application/pkcs7-mime; smime-type=signed-data;
   name="smime.p7m"  Content-Transfer-Encoding: base64
2: MIME-Version: 1.0  Content-type: application/pkcs7-mime; smime-type=signed-data;
   name="smime.p7m"  Content-Transfer-Encoding: base64
3: MIImaAYJKoZIhvcNAQcCoIImWTCCJUCAQExCzAJBgUrDgMCGgUAMIIj (...)
```

The messages in this format consist in two main parts:

- The header, which contains information about the MIME version, the content type and content transfer encoding (base64 for interoperability reasons), as well as information about the S/MIME signature.
• The payload which contains the invoicing information encoded in base64 and that can also be encrypted but not as a requirement. The signature is appended at the end of the Electronic Invoice and is also encoded in base64.

These messages are created from the same information source as InfoPath XML, and so they’re uploaded simultaneously to the repository (see Figure 5.9) after being sent to the trading partner.

Figure 5.9: Simultaneous Electronic Invoice Storage.

Nevertheless the InfoPath Electronic Invoice does not have any kind of legal value and is intended only for visualization purposes. As so, the only legal and valuable information is that stored in the signed invoices.

The storage is done through the consumption of Web Services in the WSS service bus (all interactions between WSS and BTS are done using this service bus). For an invoice to be properly stored in WSS it is necessary to fill in the columns in the SharePoint repository. These columns are custom made and are intended to show relevant invoicing information extracted from the content of the Electronic Invoices and can be used to start workflows based on rules.

So, to fill up these columns with useful information we need to use specific XML configurations as context properties of messages with its own syntax. Code Listing 5.4 illustrates a message (CertifiedUBLInvoice) used to store an Electronic Invoice and the attribution of a context property that will fill in the column values in SharePoint. If there are any active workflows in the repository when this information is uploaded with the corresponding Electronic Invoice, a trigger will start the workflow and move it to its first state (typically the workflows are started upon the creation of a new object within a SharePoint list).
First we need to configure the namespace alias that will be used in the message (line 1), then we construct the XML string (line 3) that will have the information used to fill in the columns with values. The information is gathered from a message (InvoiceRoutingInformation) that contains data extracted from the content of the invoices. Finally (line 5), we set the WSS.ConfigPropertiesXml context property with the constructed string.

The final result is the storage of the Electronic Invoices (Figure 5.10) and the completion of the columns with data. This data can be easily changed as user needs evolve.

![Figure 5.10: Electronic Invoice Storage in a Sent Folder.](image)

In terms of security, WSS provides its own access control mechanism based on groups, users and actions over objects (see Figure 5.11). Since WSS also provides the interface for the solution, we can have an integrated access control and guarantee the security of the stored information. We can define which user or group can have access to which repository, or define a set of users as part of a company group share a set of permissions among this group. In order to add a user to SharePoint the credentials must be first set on the Local Users and Groups section of the local Server Management Console.

In the top left corner of this figure we can see the three existing main groups: Members, Visitors and Owners.

- Members Group\(^2\) is the group where the companies are included. They can contribute by viewing,
adding, updating and deleting objects on SharePoint and they also have limited access to other specific lists, document libraries, list items, folders or documents when given permission.

- Visitors group is a group created for audit reasons. Here audit entities can have view access permissions. As a complement, they can also have access to specific lists, document libraries, list items, folders or documents when given permission. Permissions must be solicited to the administrators.

- Owners group have full access to the system. They can design, contribute and read all lists, document libraries, list items, folders or documents. This was the group used during the development of this work.

One important security aspect is that when consuming Web Services to access the stored information, one must also provide the necessary credentials; otherwise the information cannot be accessed.

5.6 Milestone 4 - Implementation of AS2 Communications

This milestone consisted in the implementation of the AS2 protocol to answer the non-repudiation requirements. This implementation was realized using the new BizTalk Server capabilities in this area. BizTalk provides the adapter and a non-repudiation database where all AS2 transactions are logged.

To better simulate a connection to a remote machine it was necessary to configure an HTTP receive location on the same machine using the IIS 6.0 bundled in Windows Server 2003. The "remote" machine was listening to AS2 communications on a specific virtual directory.

AS2 communications are also part of the core functionalities of the general solution, and so, they are implemented as part of the core orchestrations, more specifically as the last activity in the Core Logic on the send side and the trigger to the Core Logic in the receive side.

To setup an AS2 connection between two partners using BizTalk it is necessary to configure the AS2 party properties in BizTalk Administration Console. Here we need to specify a party alias to be used for routing as filter expression on send ports and a certificate that is used to uniquely identify each party (trading partner).
This certificate is the one used in the signing of the Electronic Invoices. These properties also define whether or not this AS2 implementation uses the MDN mechanism.

Figure 5.12 illustrates the general interaction between two parties:

![Figure 5.12: AS2 Communication using an HTTP Receive Location.](image)

After all the core processing, the message is sent to the BizTalk's message box where a send port picks it up (based on AS2 alias filter expressions) and passes it through an AS2 send pipeline. The pipeline envelopes the original Electronic Invoice in a S/MIME message, adds to the message header the source and destination for tracking and routing purposes (AS2-From and AS2-To properties) and sends the final message on AS2 over HTTP to the HTTP receive location.

In this process, BTS logs in the non-repudiation database the start of the transaction. When the message arrives at the destination the HTTP receive port receives the message and passes it through the AS2 receive pipeline, the MDN mechanism sends the proof of receipt and the transaction is concluded in the non-repudiation database (MDNs are also stored in the non-repudiation database).

5.7 Milestone 5 - Implementation of the Receive Side

The receive side is like a symmetric copy of the send side orchestrations. In fact, it was developed almost at the same time as all the send functionalities.

Having this into account, there isn’t much new to say about this milestone besides the fact that these orchestrations can be used as an adapter that is always listening to the AS2 communications (at the HTTP location described earlier).

Basically the developed solution is composed of two main components: receive side and send side orchestrations (as Figure 5.13 shows).

As we have been seeing through the last chapters, the send component is responsible for extracting information existing ERPs or legacy systems, certify it and then send it to the receive component of the trading partner. Here the receive component validates the invoice, stores it, and can also integrate with other existing applications (in Figure 5.13 the invoice is stored in an ERP after the being received).
5.8 Milestone 6 - Implementation of BAM

Business Activity Monitoring (BAM) is a collection of tools that allow you to manage aggregations, alerts, and profiles to monitor relevant business metrics (called Key Performance Indicators or KPIs). It gives you end-to-end visibility into business processes, providing accurate information about the status and results of various operations, processes and transactions so that specific problem areas can be addressed and issues resolved.

BAM allows tracking specific business data along the invoicing process. Within BizTalk BAM is composed of activities and views over activities. There are two methods to apply BAM in BizTalk:

- Using Tracking Profiles
- Using the BAM API

Both were implemented and tested. The second one emerged as the most complete and flexible. Although required more manual work, the BAM API method is the one put in practice in the final solution.

Figure 5.14 depicts the two main BAM development processes:
As we can see there are a few steps in common. First we must define activities and views through Bam.xls, an Excel Stylesheet that is used to define the business data we want to monitor. Bam.xls is typically located on `<BizTalk Server Installation Path>\Tracking\Bam.xls` and must be copied to the solution's project folder so that it can be changed as needed.

At this step the activities defined were:

- Sent Invoices
- Received Invoices

Instances of BAM activities contain the data that is used to generate the Key Performance Indicator (KPI) reports. When you define a BAM activity you specify the milestones and other data to collect from the application that is being monitored. These data items are called activity items. After you define the BAM activity and the activity items, you can use the activity to define KPI reports (called "views").

When defining these views, we are also defining cubes that allow excellent data exploration. Views make use of dimensions and measures based on activity items and can also span multiple activities.

The defined views were:

- Sent invoices by company
- Sent invoices by date
- Received invoices by company
- Received invoices by date

A BAM view identifies the BAM activities and activity items that will be used to create KPI reports. A dimension identifies how data will be grouped in reports. You can think of a dimension as a row or column heading in a report. A measure identifies the data that will be computed and displayed in the reports. You can think of a measure as the cells in a report.

For Sent Invoices activity we defined the following activity items:

<table>
<thead>
<tr>
<th>Name</th>
<th>Item Type</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Company Name</td>
<td>Business Data - Text</td>
<td>Text</td>
</tr>
<tr>
<td>Electronic Invoice Amount</td>
<td>Business Data - Decimal</td>
<td>Decimal</td>
</tr>
<tr>
<td>Electronic Invoice Send Date</td>
<td>Business Milestones</td>
<td>Date Time</td>
</tr>
<tr>
<td>Origin Company Name</td>
<td>Business Data - Text</td>
<td>Text</td>
</tr>
</tbody>
</table>

Table 5.1: Sent Invoices Activity Items

For Received Invoices activity we defined the following activity items: In order to view data change in

<table>
<thead>
<tr>
<th>Name</th>
<th>Item Type</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Company Name</td>
<td>Business Data - Text</td>
<td>Text</td>
</tr>
<tr>
<td>Electronic Invoice Amount</td>
<td>Business Data - Decimal</td>
<td>Decimal</td>
</tr>
<tr>
<td>Electronic Invoice Receive Date</td>
<td>Business Milestones</td>
<td>Date Time</td>
</tr>
<tr>
<td>Origin Company Name</td>
<td>Business Data - Text</td>
<td>Text</td>
</tr>
</tbody>
</table>

Table 5.2: Received Invoices Activity Items
real-time we need to choose the real-time aggregations option.

After defining the activities and views in BAM.xls these are exported into a XML file which is then deployed into BAM databases (BAM Primary Import database). BizTalk’s BAM component is supported by a set of databases that are created to store the data we want to collect.

After this step it is necessary to assign the BizTalk source messages (from our orchestrations) with the fields we want to monitor in each activity. This can be done using tracking profiles. They are defined with a tracking profile tool packaged with BizTalk.

The Tracking Profile Editor (TPE) is a graphical user interface to help developers create new tracking profiles or modify existing tracking profiles for the BizTalk engine. Specifically, the TPE tool is used to drag items from orchestrations, messages, or pipelines and map them to the data items in an existing activity. An activity is deployed in the BAM Primary Import database. The TPE can drag items from orchestrations or messages and attach them to an activity (see Figure 5.15). The data about individual activity instances is available for queries in a dynamically created SQL view in the BAM Primary Import database.

![Figure 5.15: BAM Tracking Profile Editor.](image)

On the left side we have activity items from Sent Invoices activity. All we have to do is select an orchestration (e.g. eFactCore.CoreLogicSendProcess) on the right side and select a shape from where we want to collect the message with the business data we need to monitor. Then the message schema (XSD) will appear and we can select an element to map with an activity item. When we’re done, TPE deploys all profile mappings into another BizTalk BAM database.

In this way when the Electronic Invoices are processed, the BizTalk engine know that the specified data needs to be monitored and all information is stored into BAM Primary Import database.

The problem is that this tracking profile method does not support mapping data with repeating fields. This problem occurred when testing with invoices with multiple lines of products/services.
Since this functionality is crucial, a solution was required. After some research we developed a custom pipeline component that makes use of the BAM API. This BAM pipeline component allows to programatically control the data we want to monitor. With this method we could reuse the activities and the views already defined and instead of mapping fields using tracking profiles the data was gathered when it passed through a send or receive pipeline. Figure 5.16 illustrates the data collecting process:

![Figure 5.16: BAM Collecting Process.](image)

This component looks into the Electronic Invoice content and starts a BAM activity, then it collects data, updates the BAM Primary Import database with fresh information and finally closes the activity when no more data is needed. This component allowed to iterate over each invoice line and extract the information we needed.

Finally now that all data could be collected, we needed some way to look into the information in a pleasant way. To extract BAM information, it is best to use a BAM view rather than going directly to the SQL tables in the Primary Import database. The view aggregates information from several tables to surface a complete view of the data. Additionally, the actual table structure may change and affect the created view in future releases.

As so, BAM data can be explored using BAM Portal and WSS, allowing custom made queries to BAM database tables, drill-up and drill-down of data through the defined cubes (see Appendix E, Figure E.1) in real-time using the real-time aggregations.

Finally we used BAM to address the summary map requirement. The elements obliged by legislation were modeled in BAM activities and two summary map views were created. Then relevant business data was monitored when it passed through a receive or send pipeline and finally explored using BAM Portal as shown in Appendix E, Figure E.2

One of the summary map requirement states that information should be provided in full or by the application of selective criteria of search data. This is automatically addressed by BAM Portal because it has a built in query editor based on our own custom data.

The conclusion of the BAM implementation is that although Microsoft says the best way to persist incoming data from a BizTalk process into BAM is to use the Tracking Profile Editor (TPE), we found that the BAM API offers more control and flexibility to the developer.
5.9 Milestone 7 - Integration with a SAP R/3 System

The first step to accomplish the integration with SAP was to establish a connection between the two systems before actually receive any kind of information from SAP. The idea was to configure the two systems so that when an invoice is created in SAP, it automatically triggers a BizTalk orchestration. The orchestration would then undertake all necessary activities for the Electronic Invoice certification. But focusing on the integration we must first configure a SAP RFC destination so that it can interact with BTS.

To configure a RFC destination we must use transaction SM59. Here we must specify ”T” connection type (meaning start an external program via TCP/IP, configure activation type to be ”Registration” and select a name for the program id, for instance ”INVOICESPROGID” (see Figure 5.17).

![Figure 5.17: SAP RFC Destination Configuration.](image)

The next step is to configure a transactional RFC Port. To reach this end we use transaction WE21 that opens the ports in IDoc processing screen. Here we create a new transactional RFC and name it BTSINVPORT. We must also specify the IDoc’s release version, in our case this is version 4.X. (see Figure 5.18).

![Figure 5.18: SAP IDoc Port Configuration.](image)

After this we must define the partner profile configuration by, first, specifying the logical system that is used

---

3 SAP Transactions are menu shortcuts also called T-Codes which, due to the immense complexity of the SAP system, allow a faster and easier access to functionalities.
to identify an individual client in a system allowing ALE communications between SAP systems or external applications (e.g. BizTalk), and then by creating the actual partner profile.

The partner profile contains parameters that define the electronic interchange of data with a partner via IDoc interfaces. Here we specify the partner number that must be the logical system specified earlier, see Figure 5.19.

Now for outbound processing we must specify the parameters like message type (INVOIC), receiver port (BTSINVPORT), output mode (Transfer IDoc) and the basic type for the IDoc type (INV_ID01).

Finally to test the connection, we must configure a BizTalk receive port (and the corresponding receive location). For the test connection to work properly, this receive location must be enabled, otherwise we get a registration error in the SAP Gateway and the connection fails.

With this in mind, we created a one-way port with the SAP transport type. In the SAP transport properties we must specify (between other properties) the program id used by the RFC port in SAP, the Gateway host and service names that will handle the connection (see Figure 5.20).

Going back to SAP, we can now test the connection (with the receive location enabled). BizTalk server is now ready to receive INV_ID01 IDocs from SAP delivered to BTSINVPORT and BTS logical system as defined in the previous procedures.

With the connection established, we now need to configure SAP to generate INV_ID01 IDocs upon which will be delivered to the above logical system at the specified port.

Since our aim was only to guarantee the integration between SAP and BizTalk, we shall not concern with the buying process that must be preconfigured in SAP in order to correctly generate invoices.

As so, we used a test tool for IDoc processing. This tool is part of SAP and it allows to generate IDocs, by specifying their message and base types, it also provides an interface to fill in IDoc fields, and then it allows to send IDocs to the recipient by specifying the partner no. (BIZTALK) and the partner type (LS - Logical System).
Now the next steps will prove the flexibility of the BizTalk SOA and the benefits of the SAP adapter (as proposed in the previous chapter), as well as will show the importance of the architecture design in terms of adoption of a new format (in the core entry point).

The first step to conclude the integration was to use the SAP adapter to generate the SAP IDoc invoice equivalent XML schema (XSD), and to create a receive pipeline that uses a flat-file disassembler to convert from the IDoc flat-file to the IDoc XSD.

Once this first step was accomplished, we used the core entry point design to add a new decision branch for the new XSD format, as we can see in Figure 5.21.

Similarly to the test condition we created to the ERPInvoice format, we must now create a condition to the SAP IDoc format (see Code 5.5).
Core Entry Point Decision Expression for the SAP IDoc INV_ID01 Format.

```
1: ReceivedAnytypeMsg(BTS.MessageType) ==
   "http://eFactCORE.INV_ID01#INV_ID01"
```

Now in this new branch we must create a new map, which will convert from the IDoc XSD to the UBL format. A good way to know which fields we must map from SAP invoice to BizTalk UBL invoice is to use transaction WE63 which given an IDoc type returns all its documentation. Given this we used the necessary invoice fields presented in Appendix C and saw in the SAP documentation the corresponding fields.

Finally we must add the receive location created for connection testing (created in the beginning of this section) to the receive port of the send core functionalities. Here we must also change the receive pipeline to the one we just created earlier that converts the flat-file.

Now when the SAP test tool for IDoc processing is used for outbound purposes, the IDoc is sent to BTS, on its arrival it is converted to the UBL format before it reaches the core orchestration, and then all activities are undertaken oblivious to the original invoice format.

5.10 Case Study

This section aims to portrait a real case study where the solution proposed in this dissertation was implemented in an intra-group B2B environment. For privacy reasons, the companies’ names will not be mentioned although the specific business case scenario is veridical.

This business case started before the implementation and was used as a guideline during the development process as it allowed to instantiate the project in a real environment with real requirements, with existing infrastructure and with a semi-automated invoicing process. This is very important because it allowed to continuously verify the validity of each proof of concept in each incremental milestone.

To simplify, the company group shall be called only “Group” and each company ”Company A, Company B, C”, etc.

5.10.1 Business Scenario

This case study is inserted in an organizational context where companies belonging to a bigger company group provide services one to another in the areas of construction, environment, transportation and real estate.

The business contextualization focused on identifying the reasons that led to the necessity of implementation of an intra-group invoicing system, identifying the actors involved in the intra-group invoicing processes and which were the limitations felt with the existing system.

First of all the Group felt the necessity to automate the invoicing process in intra-group interactions because on most circumstances the invoice had to be printed on Company A and then manually sent to Company B mailbox that on many situations resided in the same building just next to one another (see Figure 5.22).

The Group is supported by a SAP R/3 system where around 30 companies are managed. Group companies not supported by the SAP system are viewed as external to the Group, and so, the invoices emitted to these
are also necessarily printed and manually sent. The supported companies form the group of actors of this case study.

So ideally the group felt the urge to minimize the invoice printing effort and automate the communications between companies weather they were on same physical location, on a remote location and in or out the SAP supported group of companies.

In terms of limitations of the current invoicing process, the Group had the necessity, due to legal requirements, to print the invoices at both the origin and destination. Another important issue was that the group didn’t know how to sign the invoices before sending to the recipient, neither knew how to support an infrastructure where each company has its own certificates.

With this in mind, the flow of the information could be identified between the existing systems could be identified, and is summarized in the following sequence diagram (Figure 5.23).

The information flow starts with an employee logging into the company’s front-end and asking for a Billing Request. The Billing Request is then submitted into the company’s customer management system where it
is verified and a posterior Sales Order is emitted to the SAP system.

The SAP system then sends the digital invoice, without legal certification because there isn’t a implementation of a method to guarantee invoice signing, Electronic Invoice archive and non-repudiation. Simultaneously, another employee has to extract invoice information from SAP, print and mail it to the destination. This implies extra overhead and costs that the Group wants to eliminate.

### 5.10.2 Operational Scenario

One interesting aspect of the operational scenario is that the Group already uses the BizTalk’s message engine to guarantee message delivery and format conversion. The orchestrations do not implement any business logic. This situation poses a big advantage for the implementation of the proposed solution as costs with licensing would be minimal, and support to the production environment could be provided by in-house personnel.

As we can see, the eFact component can be inserted between the SAP system and the other company’s document management system (see Figure 5.24) acting as an intermediary for the communications and providing the legal compliance necessary to the invoicing process at the same time that it overrides the previous manual work that the Group wanted to eliminate.

![Figure 5.24: Information flow with the eFact component.](image)

The SAP system integrates with BizTalk and automatically starts the orchestrations that provide the invoice certification and that ultimately send the invoice to its destination. At the same time, the eFact component provides visibility as we have seen in the section 5.5 and 5.8.
Chapter 6

Results

6.1 Estimated Integration Effort

Solution cost in an integration project is usually measured by counting the number of integrations needed and the estimated cost of each integration. The number of integrations depends on the specific business case.

Assuming a simple scenario with two companies (CompanyA and CompanyB) exchanging invoices we would have at least two integrations, one of each send side functionality at each company. The cost of these two integrations would depend mainly on the technology involved. For instance, if both companies were using backend systems supported by the Line of Business (LoB) Adapter Pack\(^1\) shipped with BizTalk Server 2006, we would have an almost inexpensive integration effort in time spent on implementation and deployment of the solution. Otherwise we would need the help of an in-house IT expert, with a minimum knowledge of the technology allowing medium effort integration.

On the other hand, we also need to have into account the integrations in the receive functionalities. If the receive side of the solution proposed in this dissertation is used, the integration effort would be out-of-the-box in terms of receiving and storage of Electronic Invoices. Eventually the only integrations needed would be with a company’s backend system if needed. Otherwise (if the receive side is not used) the effort would be more complex in the way it would be necessary to configure the AS2 properties accordingly with the other company’s AS2 adapter.

Table 6.1 summarizes the estimated integration cost (L - Low, M - Medium, H - High) assuming that the send side functionalities are always used by both companies. The receive side can be either used or not because there is the possibility that the company uses a third party infrastructure to receive and store the Electronic Invoices. This integration effort classification is estimated based on the number of necessary integrations, whether or not the technologies involved are supported by adapters or the development has to be custom made.

\(^1\)LoB Adapter Pack is a set of application adapters that provide an interface to Line-of-Business (LOB) systems like SAP, Siebel, Oracle Financials and PeopleSoft.
As we can observe in the most of the cases we can have a low integration cost. The most problematic case is when both companies do not use the receive side of the solution and all the Electronic Invoice signature validation, storage and error handling is done through a third party solution. This case is the most expensive one as we might need a greater number of integrations.

If both send and receive functionalities are used we can guarantee that all legal requirements are fulfilled and no other of integration is needed. If one of the companies do not use the receive functionality then that specific case is classified with a high integration effort because once more we are dealing with third party solutions that might involve a higher number of integrations.

### 6.2 Performance

The performance in this work was a secondary objective. In fact, it wasn’t even modeled as a main requirement. In this way, this section only refers to the observable timings and errors occurred in the processing of single invoices and in batch processing of multiple invoices.

Another limitation is that these tests were conducted in the development machine: an Intel Pentium IV 3.06Ghz single core laptop with 2GB RAM and 512kb cache L2. Being on a single machine meant not having into account the internet transmission delay of Electronic Invoices.

A final and important matter is that the solution was running on a virtual machine only with 1GB RAM to distribute with BizTalk Server, SQL Server and Windows SharePoint Services (besides other running programs and services). Nevertheless, Table 6.2 presents the main performance results observed.

<table>
<thead>
<tr>
<th>Electronic Invoices</th>
<th>FileSystem</th>
<th>SAP R/3</th>
<th>WS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Payload Size (KB)</td>
<td>Time (s)</td>
<td>Errors</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>45</td>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>90</td>
<td>38</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>180</td>
<td>77</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>270</td>
<td>251</td>
<td>1</td>
</tr>
<tr>
<td>40</td>
<td>360</td>
<td>299</td>
<td>0</td>
</tr>
<tr>
<td>50</td>
<td>450</td>
<td>625</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 6.2: Solution Performance.
The results of the FileSystem test show an average 4 seconds processing time for each Electronic Invoice in the 5 to 30 processing interval. Although the transmission time is not taken into account, this seems to be a very good result having in mind the testing machine. From 30 to 40 Electronic Invoices, the timing rises up to a 8 seconds average until a 12 second maximum average with 50 Electronic Invoices delivered at the same time in the receive folder.

In terms of errors observed they were related with deadlocks in the BizTalk message engine. These kinds of faults are totally unpredictable from the developer’s standpoint. One way to overcome these problems is to program the batch processing in groups in a way that a maximum of 20 Electronic Invoices are processed at each time, then after each batch processing a new one is started until no more Electronic Invoices exist.

The SAP adapter showed the more surprising results. It presented the more efficient results without causing any error. This shows that this component already has some capabilities to manage payload and to not overload BizTalk with transactions that cause deadlocks on the message engine. This, allied with the low integration effort, is one of the advantages of using adapters to integrate with existing back-end systems.

The results with the orchestrations exposed as Web Services are relatively similar to the FileSystem. Although less efficient than the former, the Web Service invocation times are not that different in terms of execution time and in errors. In the worse case tested, 50 Electronic Invoices only take one and half minutes more regarding the FileSystem test.

Having into account that the HTTP transport times are missing due to local machine testing, we ought to expect that these numbers should be a little higher and more propitious to errors. Nevertheless we can conclude that the 20 Electronic Invoice mark for group processing is a good start point for the configuration of a batch execution environment.

### 6.3 Bottlenecks

Besides a simple performance analysis, we can also analyze the solution’s bottlenecks to gain better understanding of the activities that consume more time and that should be revisited in future work to improve overall performance.

In Appendix F we present a table containing each activity executed in the invoicing process (for a Web Service invocation), its start and end times, and more important, its total execution time. As we can observe, it is possible to see the separation of the send and receive logics.

Starting with the send logics we see that the more time spending activities are the core entry point, where we test the message xml namespace to detect and perform the transformations to the internal format, and of course, the core processing where we can see that most of the time goes to the application of the signature (Signature timings are implicit on the eFactCORE.CoreLogicSendProcess execution time because the signature pipeline is executed inside this orchestration as explained in section 5.4). The time spent on storage is minimal as it is done through Web Service invocation to SharePoint Services, putting all overhead away from the messaging engine. Finally we can see that other massive time consuming activity is the send and the receive AS2 functionalities that consume around 30% of the total processing time.

Regarding the receive logics we can observe that the core activities consume most of the time as expected, similarly the storage time is minimal and comparable with the times observed in the send side. One interesting
aspect of the core receive side is that it consumes around 3.5 times less time that the core send time. This becomes obvious if we think that the send side has to apply a set of algorithms to sign the Electronic Invoice, and the receive side only has to validate the signature.

Finally the justification for the time of the core exit point is that in this tested scenario the output format chosen was the same as the internal format. As so, there was no time spent on transformation activities.

One last observation to this table makes us believe that integration times can be very variable. For instance, we had a 300ms time for the Web Service invocation and at the end of the process (where we integrate with a SQL Server 2005 database) we had a 3316ms execution time, which makes us conclude that the technologies involved in the integration also have very strong consequences in the overall execution time of the solution.

6.4 Ease of Integration, Expansion and Interoperability

Automating and maintaining business processes in a dynamic and cost-effective manner has proven to be a difficult endeavor for even the most technically sophisticated organizations.

Known as Service Oriented Architecture (SOA), the methodology is based on XML and Web services technologies and has been incorporated into Business Process Management and Enterprise Application Integration (BPM/EAI) platforms. The BizTalk Service Oriented Architecture primes in this field for its capacity of integration with other systems.

The SOA paradigm has redefined the concept of an application. An application is no longer an opaque, procedural implementation mechanism. Instead, it is an orchestrated sequence of messaging, routing, processing, and transformation events capable of processing the exposed declarative properties of rich (XML) documents.

A workflow process, integration scenario, or trading partner interaction are specialized classes of the SOA paradigm distinguished only by the nature of the participants involved, the point of execution, and the participants' individual security requirements. BPM/EAI platforms that incorporate the SOA paradigm are highly compelling because they provide numerous development and operational benefits which include the flexible loose coupling of components on a highly distributed basis, the addition, removal, and reconfiguration of any process activity or component without disrupting the process and the extensibility and reuse of both application components and entire applications.

In this work, these characteristics were explored in integration with a SAP R/3 system but easily systems from other vendors could be incorporated without affecting the general architecture. In fact, the BTS SOA allows the incremental development and the constant adding of new functionalities as services to this solution and from a holistic point of view, the enrichment of the existing service infrastructure as a whole. With well defined core functionality, this solution allows further development to focus on the extensibility in terms of integration with other back-end systems, new message formats and other types of electronic signatures without affecting the global processing.

XML provides the transparency and application independence and uses metatags to "declare" the meaning and function of data. The premise of XML is to convert program-dependent data into self-describing,
program-independent data. This applies not only to content, but also to instructions for processing the content (e.g. InfoPath processing instructions). In addition to XML, XML Schema provides semantic consistency and interoperability.

The Web services protocols, Simple Object Access Protocol (SOAP), and Web Services Definition Language (WSDL) provide the capabilities and messaging facilities required to bind and execute programmatic functionality anywhere, on any platform, without the need for custom code. With the capacity of exposing the developed orchestrations as Web Services the significance of the Web services specifications is that they provide a truly workable architecture for building complex, interoperable computing processes over the Internet.

By using HTTP as its transport mechanism, AS2 messages can go anywhere, use existing SSL facilities for authentication and encryption, and make the most of the infrastructure scale of the World Wide Web.

Although not implemented in this project, Electronic Invoice validation rules could be supported by BizTalk’s business rule composer allowing a rule set incorporated within a BizTalk Server orchestration to be viewed, modified, or completely replaced both at design and run time, without affecting any other process operation or interrupting running instances of the affected process.

The flexibility that an exposed and componentized rule engine provides for modifying business processes is of fundamental significance. In conventional application development, business rule logic is embedded in procedural code and is not accessible for modification without changing the code itself. Because most modifications to a business process life cycle pertain to changes in business rules (as opposed to technology-related modifications), the ability to isolate business rules entirely from procedural code, or any process implementation mechanisms, dramatically improves the efficiencies of managing and adapting business processes throughout their life cycle.
Chapter 7

Conclusions

7.1 Project Conclusions

The main objective of this dissertation was to develop, with the chosen technology, an Electronic Invoicing solution that answers all legal requirements (according to Portuguese Legislation), and at the same time, follows the premises of the BizTalk Service Oriented Architecture. This objective was clearly accomplished and the final result is a solution with still very work to be done in order to improve its capabilities, but yet, already with a set of functionalities that are enough for the deployment of an automated invoicing process between two business partners.

Three main problem groups were identified, the challenges imposed by the B2B model, the requirements imposed by legislation and the specific technological requirements. Although this dissertation focused on the legislation requirements, presenting a set of feasible solutions for each problem, we also addressed specific technological issues with the SAP R/3 integration with BizTalk Server and we took advantage of the SOA benefits to resolve B2B problems such as coupling between partners, allowing autonomy by turning services into black boxes and using a significant level of security provided by the SharePoint Service Bus.

The use of BizTalk technologies in this dissertation proved to be an excellent benefit as this message engine allowed to develop a series of processes in the form of orchestrations that coordinate a set of activities, implemented in a loosely manner just as SOA defends.

To help us design our Orchestration Architecture it was defined a Service Architecture based on a set of identified business events. This Orchestration Architecture allowed us to develop the services with a minimum code writing effort at the same time as functionalities could have their endpoints exposed as Web Services guaranteeing maximum interoperability with other systems.

The main advantage of exposing processes (orchestrations) as Web Services is that we make available public ports as Web methods, and the messages they send and receive are published as types. The benefit here is that you only need to do this generation process if the orchestration ports or schemas change. You can modify the orchestration and redeploy it, and as long as the public interface has not changed you do not need to regenerate the Web service. This decoupling method defined a interface layer and allows that other applications or services that use these (BizTalk defined) services to worry only with WHAT the service does.
and not HOW the service does it.

One of the difficulties encountered during this work was the understanding of the Portuguese legislation. The idea that remains is that anything goes if in general terms the legislation is followed. At the end of this work one must conclude that although there are recommendations from important national entities like UMIC, sometimes these recommendations are not unique and other technologies, messages formats, certificate and signature types might also do the same work in a better or at least an equal way.

After analyzing these recommendations and the BizTalk functionalities we can conclude that both are very well aligned with each other. Just as an example, the signature process recommended by UMIC is the use of X509 certificates with S/MIME signature type, which is directly supported by BizTalk, and the use AS2 communication protocol that is now supported in the new release of BizTalk.

7.2 Future Work

This future work section deals with the potential set of changes, developments and extensions, which can carry on the work already done in order to improve it and expand its scope. There are several work areas that may be expanded and improved, for a better legibility each of these are explained in an independent bullet.

- One important feature that may be explored is how to recover the system from an archive corruption problem. Although there are several studies and solutions to this problem, it might be interesting to implement a recovery process, for example based on periodically backups or database replication, and to study if these processes could be supported by the technology used in this dissertation. Another issue is concerned with recovery of the signature applied in each lost Electronic Invoice. Would the sender need to resign the document? What kind of contracts need to exist between trading partners to safeguard this situation.

- Another important matter is what happens when a certificate used to sign an Electronic Invoice expires before the time foreseen in legislation. What mechanisms could be applied to request new certificates, distribute them and resign the document? If this problem could be automated in a business process what are the security risks and the main advantages.

- Still regarding the certificates and advanced electronic signatures, one possible area of work might be the expansion of the supported signature types. This may be interesting in order to better support other trading partners and other already implemented systems. To achieve this purpose it would be necessary to explore other ways of validation of the signatures since BizTalk only automatically validate and check the certificate revocation lists when using S/MIME signatures with X509 certificates.

- The limitation of the BizTalk implementation of S/MIME signatures and the usage of a single signing certificate per BizTalk Group of applications was circumvented with the development of a custom pipeline component with the help of Microsoft Portugal. We had to adapt the AS2 Party properties of each party (trading partner) in order to support the company certificate. Although this solution worked for this dissertation, one company might have more than one certificate to sign different business documents. In this way, developing a solution for this "new" limitation may be a good work area.
• Although BizTalk already provides some of the functionalities in terms of reliable communications, it would be interesting to implement the AS2 protocol over queues like for instance MSMQ that is already supported by BizTalk.

• In terms of the Portuguese legislation, a new requirement was introduced in the beginning of 2008 that states that all invoicing information must be exported into a well defined XML format for audit reasons. This format is the Standard Audit File for Tax purposes (SAFT-PT).

• In terms of overall performance, runtime execution is conditioned by message size, orchestration, schema and map complexity, pipeline components, tracking data (BAM) and transport adapters. While different business scenarios define the particular transport adapters that are required, different transport adapters have different settings that can be changed to improve performance. One way to improve performance would be to separate orchestration and adapter functionality into separate BizTalk Server hosts to minimize resource contention.

• The solution presented in this work, as well as the technology used, allows a significant level of extensibility. To explore this feature it might interesting to integrate other systems besides the SAP R/3 system. Integration with other systems using the BizTalk’s Line of Business adapter might be a good starting point, but develop reusable ways of integration with other systems using Web Services or other loosely coupled ways respecting the service oriented architecture might be a better objective with a greater value.

• One excellent improvement to the existing solution would be the integration of the back office, currently inexistent because all configurations need to be realized in BizTalk Administration console, with the current front office in the SharePoint Services platform. This would allow a greater security level and a friendlier interface as all runtime interactions with BizTalk would be done through web interfaces. This could be done by developing user controls and using the BizTalk API to set all properties with using the administration console.

• This solution could also be used with document management applications, allowing an expanded scope of action and possibly the integration of the invoicing business process with other business processes.

• One final aspect is related with error handling. Although this was not clearly one of the main focuses this work, it is essential to create log databases and monitor all transactions.

All of these extensions can be developed incrementally, some of them in the form of services, allowing an excellent alignment with the SOA objective, maintaining the solution loosely coupled and adding an extra value.
Appendix A

Requirements Document

A.1 Functional Requirements

A.1.1 FR1 - Signatures

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FR1.1</td>
<td>Each Electronic Invoice should be digitally signed.</td>
</tr>
<tr>
<td>FR1.2</td>
<td>The solution should support different signatures for different entities.</td>
</tr>
<tr>
<td>FR1.3</td>
<td>The solution should use a specific key agreed for each transaction between two entities.</td>
</tr>
<tr>
<td>FR1.4</td>
<td>Electronic invoices must be signed before storage.</td>
</tr>
</tbody>
</table>

Table A.1: Functional Requirements - Signatures

A.1.2 FR2 - Sending and Receiving Electronic Invoices

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FR2.1</td>
<td>The sending and receiving of invoices should be done over the Internet.</td>
</tr>
<tr>
<td>FR2.2</td>
<td>The non-repudiation of the origin and the destination should be guaranteed.</td>
</tr>
<tr>
<td>FR2.3</td>
<td>In case of an error in transmission there must be a mechanism that allows retries in different periods of time.</td>
</tr>
<tr>
<td>FR2.4</td>
<td>Every error detected in transmission or re-transmission should be stored in an error database.</td>
</tr>
<tr>
<td>FR2.5</td>
<td>The receiving entity should be able to request a forwarding of an Electronic Invoice.</td>
</tr>
<tr>
<td>FR2.6</td>
<td>Upon receipt, the Electronic Invoice must be validated in terms of authenticity and integrity.</td>
</tr>
</tbody>
</table>

Table A.2: Functional Requirements - Sending and Receiving Electronic Invoices
A.1.3 FR3 - Archiving

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FR3.1</td>
<td>Electronic Invoices must be stored using mechanisms that guarantee integrity and authenticity over the period specified by law.</td>
</tr>
<tr>
<td>FR3.2</td>
<td>An access control mechanism must be implemented in the archive.</td>
</tr>
<tr>
<td>FR3.3</td>
<td>Regular backups must be done to prevent lost of critical data.</td>
</tr>
<tr>
<td>FR3.4</td>
<td>Electronic invoices must be stored over the period specified in law.</td>
</tr>
</tbody>
</table>

Table A.3: Functional Requirements - Archiving

A.1.4 FR4 - Summary Map

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FR4.1</td>
<td>The solution must support the automatic generation of a summary map.</td>
</tr>
<tr>
<td>FR4.2</td>
<td>The summary map must contain the number and invoice date.</td>
</tr>
<tr>
<td>FR4.3</td>
<td>The summary map must contain the date and time of creation of the Electronic Invoice.</td>
</tr>
<tr>
<td>FR4.4</td>
<td>The summary map must contain the date and time of sending.</td>
</tr>
<tr>
<td>FR4.5</td>
<td>The summary map must contain the fiscal identification of the sender and the receiver.</td>
</tr>
<tr>
<td>FR4.6</td>
<td>The summary map must contain the quantity and the name of the traded goods or services.</td>
</tr>
<tr>
<td>FR4.7</td>
<td>The summary map must contain the price, net of tax.</td>
</tr>
<tr>
<td>FR4.8</td>
<td>The summary map must contain the rates and the amount of tax due.</td>
</tr>
<tr>
<td>FR4.9</td>
<td>The summary map must contain a description of the deficiencies detected during each transmission.</td>
</tr>
<tr>
<td>FR4.10</td>
<td>The summary map must contain a description and version of the software used.</td>
</tr>
<tr>
<td>FR4.11</td>
<td>The summary map should be provided in full or by the application of selective criteria of search data.</td>
</tr>
<tr>
<td>FR4.12</td>
<td>The data contained in the summary should be accessible on-screen, in electronic format or be made</td>
</tr>
</tbody>
</table>

Table A.4: Functional Requirements - Summary Map

A.2 Non-Functional Requirements

A.2.1 Performance

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFR1.1</td>
<td>The system must process at least 10 Electronic Invoices per minute.</td>
</tr>
</tbody>
</table>

Table A.5: Non-Functional Requirements - Performance
A.2.2 Scalability

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFR2.1</td>
<td>The solution must support transactions between multiple entities at the same time.</td>
</tr>
</tbody>
</table>

Table A.6: Non-Functional Requirements - Scalability

A.2.3 Security

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFR3.1</td>
<td>The solution should use an asymmetric cryptographic system.</td>
</tr>
<tr>
<td>NFR3.2</td>
<td>The sending entity should have a digital certificate corresponding to the private key.</td>
</tr>
<tr>
<td>NFR3.3</td>
<td>The receiving entity should have the corresponding public key exported from the sending entity's certificate.</td>
</tr>
<tr>
<td>NFR3.4</td>
<td>The certificates can be emitted by a certification authority recognized internationally.</td>
</tr>
<tr>
<td>NFR3.5</td>
<td>The certificates can be self emitted if the companies agree upon.</td>
</tr>
<tr>
<td>NFR3.6</td>
<td>The distribution of the public keys is from the responsibility of the certificate holder (sender entity).</td>
</tr>
</tbody>
</table>

Table A.7: Non-Functional Requirements - Security

A.2.4 Persistence

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFR4.1</td>
<td>Integrity of the Electronic Invoice must be guaranteed in terms of archiving.</td>
</tr>
</tbody>
</table>

Table A.8: Non-Functional Requirements - Persistence

A.2.5 Transport

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFR5.1</td>
<td>The communication protocol should be HTTP.</td>
</tr>
<tr>
<td>NFR5.1</td>
<td>Integrity of the Electronic Invoice must be guaranteed during transactions.</td>
</tr>
</tbody>
</table>

Table A.9: Non-Functional Requirements - Transport
Appendix B

Service Architecture
<table>
<thead>
<tr>
<th>Event Number</th>
<th>Business Event</th>
<th>Event Description</th>
<th>Response</th>
</tr>
</thead>
</table>
| E1           | Receive an Invoice from the FileSystem. | This event is kicked off when a company submits an invoice to a specified folder. It ensures that an Invoice is properly received. | R1.1. Parse XML data.  
R1.2. Delete Invoice from folder.  
R1.3. Return received Invoice. |
| E2           | Receive an Invoice from a SAP R/3 System. | This event is kicked off when a company submits an Invoice through a SAP R/3 System. It ensures that an Invoice is properly received. | R2.1. Parse flat-file data.  
R2.2 Convert flat-file IDoc into a XSD equivalent.  
R2.3 Send confirmation back to SAP R/3 System.  
R2.4 Return received Invoice. |
| E3           | Receive an Invoice through WebService Invocation. | This event is kicked off when a company submits an Invoice through a WebService invocation. It ensures that an Invoice is properly received. | R3.1 Parse XML data.  
R3.2 Return received Invoice. |
| E4           | Receive an Electronic Invoice through AS2 over HTTP. | This event is kicked off when a company submits an Invoice through AS2 over HTTP. It ensures that an Invoice is properly received. | R4.1 Process AS2 headers.  
R4.2 Parse XML data.  
R4.3 Return received Invoice. |
| E5           | Convert an Invoice to the internal format. | This event is kicked off when an Invoice is received from any source. It ensures that an Invoice is converted to an internal format so that it can be processed. | R5.1 Determine original XML format.  
R5.2 Gather XSD information.  
R5.3 Gather XSLT information.  
R5.4 Apply transformation.  
R5.5 Return transformed Invoice. |
| E6           | Convert an Invoice to an output format. | This event is kicked off when an Invoice is processed. It ensures that an Invoice is converted to an output format so that it can be integrated with other systems. | R6.1 Gather destination XSD information.  
R6.2 Gather XSLT information.  
R6.3 Apply transformation.  
R6.4 Return transformed Invoice. |
| E7           | Sign an Invoice. | This event is kicked off when an Invoice needs to be signed during its processing. It ensures that an Invoice is properly signed. | R7.1 Gather sender company information.  
R7.2 Obtain certificate from certificate store.  
R7.3 Apply electronic signature.  
R7.4 Return signed Electronic Invoice. |
| E8  | Validate an Electronic Invoice Signature. | This event is kicked off when an Invoice is received and its signature needs to be verified prior to its processing. It ensures that an invoice signature is verified. | R8.1. Gather sender company information.  
R8.2. Obtain certificate from certificate store.  
R8.3. Verify electronic signature.  
R8.4. Return results from the validation. |
| E9  | Generate a visualization format for an Electronic Invoice | This event is kicked off during invoice processing. It ensures that a visualization format is created for storage. | R9.1. Generate a visual equivalent of the Electronic Invoice.  
R9.2. Return a visualization format of the Electronic Invoice. |
| E10 | Store an Electronic Invoice. | This event is kicked off during invoice processing. It ensures that the Electronic Invoice is stored. | R10.1. Determine type of Invoice for storage (signed or visualization format).  
R10.2. Gather repository information.  
R10.3. Store Electronic Invoice. |
| E11 | Visualize an Electronic Invoice. | This event is kicked off during when a company wishes to visualize an Electronic Invoice. It ensures that all Electronic Invoice data is properly displayed to the company. | R11.1. Determine if the user has rights to visualize the Electronic Invoice.  
R11.2. Open Electronic Invoice for Visualization. |
| E12 | Send an Electronic Invoice via A52 over HTTP. | This event is kicked off after Electronic Invoice processing. It ensures that the Electronic Invoice is sent through A52 protocol to its recipient. | R12.1. Gather sender information.  
R12.2. Gather receiver information.  
R12.3. Gather A52 definitions.  
R12.4. Construct A52 headers.  
R12.5. Send Electronic Invoice through A52 over HTTP. |
| E13 | Visualize business activity data. | This event is kicked off when a company wishes to visualize business activity data. It ensures that information is displayed in various formats. | R13.1. Determine type of information to visualize.  
R13.2. Gather information.  
R13.3. Display information. |
| E14 | Visualize sent Electronic Invoices. | This event is kicked off when a company wants to visualize sent Electronic Invoices. It ensures that information is displayed in various formats. | R14.1. Gather company information.  
R14.2. Gather Electronic Invoice information.  
R14.3. Display information. |
<table>
<thead>
<tr>
<th>E15</th>
<th>Visualize received Electronic Invoices.</th>
<th>This event is kicked off when a company wants to visualize received Electronic Invoices. It ensures that information is displayed in various formats.</th>
<th>R15.1. Gather company information. R15.2. Gather Electronic Invoice information. R15.3. Display information.</th>
</tr>
</thead>
<tbody>
<tr>
<td>E16</td>
<td>Visualize the Summary Map.</td>
<td>This event is kicked off when an audit authority or a company wants to visualize the summary map. It ensures that information is displayed in various formats.</td>
<td>R16.1. Gather company information. R16.2. Gather Electronic Invoice information. R16.3. Display information.</td>
</tr>
<tr>
<td>E17</td>
<td>Find an Electronic Invoice.</td>
<td>This event is kicked off when a company wants to find a specific Electronic Invoice. It ensures that an Electronic Invoice is found.</td>
<td>R17.1. Gather Electronic Invoice information. R17.2. Display information.</td>
</tr>
<tr>
<td>E18</td>
<td>Create a new company.</td>
<td>This event is kicked off when the administrator of the system wants to configure a new company. It ensures that a new user is created for the company.</td>
<td>R18.1. Gather company information. R18.2. Create user for new company. R18.3. Create configurations for new company.</td>
</tr>
<tr>
<td>E19</td>
<td>Delete a company.</td>
<td>This event is kicked off when the administrator wants to delete an existing company user. It ensures that the user is deleted from the system.</td>
<td>R19.1. Gather company information. R19.2. Delete company user. R19.3. Delete company configurations.</td>
</tr>
</tbody>
</table>

Figure B.1: Business Events Table.
<table>
<thead>
<tr>
<th>Response</th>
<th>Description</th>
<th>Service Category</th>
<th>Existing/New Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1.2. Delete Invoice from folder.</td>
<td>Deleted invoice file from the specified folder.</td>
<td>Integration</td>
<td>New – BizTalk Receive Port</td>
</tr>
<tr>
<td>R1.3. Return received Invoice.</td>
<td>Returns the received Invoice.</td>
<td>Integration</td>
<td>New – BizTalk Send Port</td>
</tr>
<tr>
<td>R2.1. Parse flat-file data.</td>
<td>Verifies flat-file syntax.</td>
<td>Integration</td>
<td>New – BizTalk Receive Port</td>
</tr>
<tr>
<td>R2.2 Convert flat-file IDoc into a XSD equivalent.</td>
<td>Converts the IDoc Invoice into a XML equivalent.</td>
<td>Transformation</td>
<td>New – BizTalk Service (SAP Adapter)</td>
</tr>
<tr>
<td>R2.3. Send confirmation back to SAP R/3 System.</td>
<td>Send acknowledge back to SAP R/3 System.</td>
<td>Integration</td>
<td>New – BizTalk Service (SAP Adapter)</td>
</tr>
<tr>
<td>R2.4. Return received Invoice.</td>
<td>Returns the received Invoice.</td>
<td>Integration</td>
<td>New – BizTalk Send Port</td>
</tr>
<tr>
<td>R3.2. Return received Invoice.</td>
<td>Returns the received Invoice.</td>
<td>Integration</td>
<td>New – BizTalk Send Port</td>
</tr>
<tr>
<td>R4.1. Process AS2 headers.</td>
<td>Processes the information contained in the AS2 headers.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R4.3. Return received Invoice.</td>
<td>Returns the received Invoice.</td>
<td>Integration</td>
<td>New – BizTalk Receive Port</td>
</tr>
<tr>
<td>R5.1. Determine original XML format.</td>
<td>Resolve XML namespace.</td>
<td>Transformation</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R5.2. Gather XSD information.</td>
<td>Assembles XSD specification.</td>
<td>Transformation</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R5.4. Apply transformation.</td>
<td>Executes Map transformation.</td>
<td>Transformation</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R5.3. Return transformed Invoice.</td>
<td>Returns the transformed Invoice.</td>
<td>Transformation</td>
<td>New – BizTalk Send Port</td>
</tr>
<tr>
<td>R6.3. Apply transformation.</td>
<td>Executes Map transformation.</td>
<td>Transformation</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R6.4. Return transformed Invoice.</td>
<td>Returns the transformed Invoice.</td>
<td>Transformation</td>
<td>New – BizTalk Send Port</td>
</tr>
<tr>
<td>R7.1. Gather sender company information.</td>
<td>Collects sender company information from the Electronic Invoice.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R7.2. Obtain certificate from certificate store.</td>
<td>Collects certificate information from certificate store.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R7.3. Apply electronic signature.</td>
<td>Applies electronic signature algorithms to the Invoice.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>Requirement</td>
<td>Description</td>
<td>Core</td>
<td>Additional Services</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>------</td>
<td>---------------------</td>
</tr>
<tr>
<td>R7.4. Return signed Electronic Invoice.</td>
<td>Returns the signed Electronic Invoice.</td>
<td>Core</td>
<td>New – BizTalk Send Port</td>
</tr>
<tr>
<td>R8.1. Gather sender company information.</td>
<td>Collects sender company information from the Electronic Invoice.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R8.2. Obtain certificate from certificate store.</td>
<td>Collects certificate information from certificate store.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R8.3. Verify electronic signature.</td>
<td>Verifies the electronic signature applied to the Electronic Invoice.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R8.4. Return results from the validation.</td>
<td>Returns a value indicating whether or not the electronic signature is valid.</td>
<td>Core</td>
<td>New – BizTalk Send Port</td>
</tr>
<tr>
<td>R9.1. Generate a visual equivalent of the Electronic Invoice.</td>
<td>Creates the source code that allows visualizing an Electronic Invoice.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R9.2. Return a visualization format of the Electronic Invoice.</td>
<td>Returns the source code that allows visualizing an Electronic Invoice.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R10.1. Determine type of Invoice for storage (signed or visualization format).</td>
<td>Resolves the type of Invoice that is going to be stored.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R10.2. Gather repository information.</td>
<td>Resolves where the Invoice will be stored.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R10.3. Store Electronic Invoice.</td>
<td>Stores the Electronic Invoice.</td>
<td>Core</td>
<td>SharePoint Services</td>
</tr>
<tr>
<td>R11.1. Determine if the user has rights to visualize the Electronic Invoice.</td>
<td>Verifies if the user has rights to visualize a specified Electronic Invoice.</td>
<td>Core</td>
<td>SharePoint Services</td>
</tr>
<tr>
<td>R11.2. Open Electronic Invoice for Visualization.</td>
<td>Opens an Electronic Invoice for Visualization.</td>
<td>Core</td>
<td>SharePoint Services</td>
</tr>
<tr>
<td>R12.1. Gather sender information.</td>
<td>Collects sender company information from the Electronic Invoice.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R12.2. Gather receiver information.</td>
<td>Collects receiver company information from the Electronic Invoice.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R12.3. Gather AS2 definitions.</td>
<td>Collects AS2 definitions for the sender and receiver.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R12.4. Construct AS2 headers.</td>
<td>Constructs the AS2 headers that will be used in the AS2 protocol.</td>
<td>Core</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R12.5. Send Electronic Invoice through AS2 over HTTP.</td>
<td>Sends an Electronic Invoice through AS2 protocol.</td>
<td>Core</td>
<td>New – BizTalk Send Port (AS2 Adapter)</td>
</tr>
<tr>
<td>R13.1. Determine type of information to visualize.</td>
<td>Determines which information as to be fetched for visualization.</td>
<td>BAM</td>
<td>New – BizTalk Service</td>
</tr>
<tr>
<td>R13.2. Gather information.</td>
<td>Gathers Invoice and Company Information.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R13.3. Display information.</td>
<td>Displays Invoice and Company Information.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R14.1. Gather company information.</td>
<td>Collects company information from the BAM database.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R14.2. Gather Electronic Invoice information.</td>
<td>Collects Electronic Invoice information from the BAM database.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R14.3. Display information.</td>
<td>Displays Invoice and Company Information.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R15.1. Gather company information.</td>
<td>Collects company information from the BAM database.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R15.2. Gather Electronic Invoice information.</td>
<td>Collects Electronic Invoice information from the BAM database.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R15.3. Display information.</td>
<td>Displays Invoice and Company Information.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R16.1. Gather company information.</td>
<td>Collects company information from the BAM database.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R16.2. Gather Electronic Invoice information.</td>
<td>Collects Electronic Invoice information from the BAM database.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R16.3. Display information.</td>
<td>Displays Invoice and Company Information.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R17.1. Gather Electronic Invoice information.</td>
<td>Collects Electronic Invoice information from the BAM database.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R17.2. Display information.</td>
<td>Displays Invoice Information.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R18.1. Gather company information.</td>
<td>Collects company information.</td>
<td>BAM</td>
<td>BAM Portal</td>
</tr>
<tr>
<td>R18.2. Create user for new company.</td>
<td>Creates a new user for the new company.</td>
<td>Core</td>
<td>SharePoint Services</td>
</tr>
<tr>
<td>R18.3. Create configurations for new company.</td>
<td>Creates configurations for the new company.</td>
<td>Core</td>
<td>BizTalk Server</td>
</tr>
<tr>
<td>R19.2. Delete company user.</td>
<td>Deletes the company user.</td>
<td>Core</td>
<td>BizTalk Server</td>
</tr>
<tr>
<td>R19.3. Delete company configurations.</td>
<td>Deletes the company configurations.</td>
<td>Core</td>
<td>BizTalk Server</td>
</tr>
</tbody>
</table>

Figure B.2: Service Category Table.
<table>
<thead>
<tr>
<th>Services</th>
<th>Functions</th>
<th>Descriptions</th>
<th>Existing / New Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integration</td>
<td>• Receive Invoices</td>
<td>Manages the Integration between the solution and other systems.</td>
<td>New – BizTalk Orchestration exposed as WebServices.</td>
</tr>
<tr>
<td></td>
<td>• Send Invoices</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Return Invoices</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Parse XML</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Delete Invoices from Receive Locations</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Send acknowledges back to end systems</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transformation</td>
<td>• Resolve XML namespaces</td>
<td>Manages all aspects related with transformations to the internal format and external formats.</td>
<td>New – BizTalk Orchestration</td>
</tr>
<tr>
<td></td>
<td>• Assemble XSD specifications</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Collect transformation map information</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Execute Map transformation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Core</td>
<td>• Process information contained in the AS2 headers.</td>
<td>Manages all activities that are needed to guarantee Electronic Invoice certification.</td>
<td>New – BizTalk Orchestration</td>
</tr>
<tr>
<td></td>
<td>• Construct AS2 headers that will be used in the AS2 protocol.</td>
<td></td>
<td>SharePoint Functionalities</td>
</tr>
<tr>
<td></td>
<td>• Send an Electronic Invoice through AS2 protocol.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Collect AS2 definitions for the sender and receiver.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Collect sender company information from the Electronic</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
• Apply electronic signature algorithms to the Invoice.

• Return the signed Electronic Invoice.

• Collects certificate information from certificate store.

• Verifies the electronic signature applied to the Electronic Invoice.

• Returns a value indicating whether or not the electronic signature is valid.

• Creates the source code that allows visualizing an Electronic Invoice.

• Returns the source code that allows visualizing an Electronic Invoice.

• Resolves the type of invoice that is going to be stored.

• Resolves where the Invoice will be stored.

• Stores the Electronic Invoice.

• Verifies if the user has rights to visualize a specified Electronic Invoice.

• Opens an Electronic Invoice for
<table>
<thead>
<tr>
<th>Service Definition Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply electronic signature algorithms to the Invoice.</td>
</tr>
<tr>
<td>Return the signed Electronic Invoice.</td>
</tr>
<tr>
<td>Collects certificate information from certificate store.</td>
</tr>
<tr>
<td>Verifies the electronic signature applied to the Electronic Invoice.</td>
</tr>
<tr>
<td>Returns a value indicating whether or not the electronic signature is valid.</td>
</tr>
<tr>
<td>Creates the source code that allows visualizing an Electronic Invoice.</td>
</tr>
<tr>
<td>Returns the source code that allows visualizing an Electronic Invoice.</td>
</tr>
<tr>
<td>Resolves the type of Invoice that is going to be stored.</td>
</tr>
<tr>
<td>Resolves where the Invoice will be stored.</td>
</tr>
<tr>
<td>Stores the Electronic Invoice.</td>
</tr>
<tr>
<td>Verifies if the user has rights to visualize a specified Electronic Invoice.</td>
</tr>
<tr>
<td>Opens an Electronic Invoice.</td>
</tr>
</tbody>
</table>

Figure B.3: Service Definition Table.
Appendix C

UMIC Recommendations

C.1 Validation Rules

<table>
<thead>
<tr>
<th>Regra</th>
<th>Descrição</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>Este campo só pode ter um dos seguintes valores: FC (corresponde à Factura Cliente)</td>
</tr>
<tr>
<td></td>
<td>FF (corresponde à Factura Fornecedor)</td>
</tr>
<tr>
<td></td>
<td>DC (corresponde à nota de Débito do Cliente)</td>
</tr>
<tr>
<td></td>
<td>DF (corresponde à nota de Débito do Fornecedor)</td>
</tr>
<tr>
<td></td>
<td>CC (corresponde à nota de Crédito do Cliente)</td>
</tr>
<tr>
<td></td>
<td>CF (corresponde à nota de Crédito do Fornecedor)</td>
</tr>
<tr>
<td></td>
<td>RC (corresponde à Requisição do Cliente)</td>
</tr>
<tr>
<td></td>
<td>RF (corresponde à Requisição do Fornecedor)</td>
</tr>
<tr>
<td>R2</td>
<td>Este campo só pode conter um dos seguintes valores: true, false</td>
</tr>
<tr>
<td>R3</td>
<td>Os campos alfanuméricos terão que estar todos preenchidos (O sistema SIC guardará apenas maiúsculas)</td>
</tr>
<tr>
<td>R4</td>
<td>Data no formato: YYYY-MM-DD</td>
</tr>
<tr>
<td>R5</td>
<td>Número fiscal de contribuinte prefixado com o código do país</td>
</tr>
<tr>
<td>R6</td>
<td>Valores pré-definidos pelo schema UBL</td>
</tr>
<tr>
<td>R7</td>
<td>Numérico entre 0 e 100, 3.2</td>
</tr>
<tr>
<td>R8</td>
<td>Valores pré-definidos pelo schema UBL</td>
</tr>
<tr>
<td>R9</td>
<td>O atributo currencyID é obrigatório, com um dos valores pré-definidos pelo schema UBL</td>
</tr>
</tbody>
</table>

Table C.1: UBL2.0 Validation Rules According to Portuguese Legislation
### C.2 UBL 2.0 Electronic Invoice Map

<table>
<thead>
<tr>
<th>N.º Referência</th>
<th>N.º Inclusão</th>
<th>Designação dos Campos do Factura Electrónica</th>
<th>Breve Descrição</th>
<th>Tipo</th>
<th>Formato</th>
<th>Dimensione</th>
<th>Renda de Venda</th>
<th>Imposto</th>
<th>Item UBL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Código de Classificação de Serviço (Referência)</td>
<td>Código de um determinado produto/mercadoria</td>
<td>Opcional</td>
<td>Alfanumérico</td>
<td>2</td>
<td>B1</td>
<td>L1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>Número de Linha</td>
<td>Indica a linha da tabela referente ao código de produto anterior</td>
<td>Optional</td>
<td>Alfanumérico</td>
<td>6</td>
<td>L19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>Descrição do serviço ou produtos</td>
<td>Descrição do produto</td>
<td>Opcional</td>
<td>Alfanumérico</td>
<td>7</td>
<td>L20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>Valor Total (sem IVA)</td>
<td>Valor total de todos os produtos</td>
<td>Opcional</td>
<td>Alfanumérico</td>
<td>12</td>
<td>L21</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>Valor IVA Massa</td>
<td>Valor IVA</td>
<td>Opcional</td>
<td>Alfanumérico</td>
<td>15.2</td>
<td>L22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>Base de Cálculo</td>
<td>Base de cálculo</td>
<td>Opcional</td>
<td>Alfanumérico</td>
<td>15.3</td>
<td>L23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>Imposto</td>
<td>Imposto</td>
<td>Opcional</td>
<td>Alfanumérico</td>
<td>15.4</td>
<td>L24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>Taxa</td>
<td>Taxa</td>
<td>Opcional</td>
<td>Alfanumérico</td>
<td>15.5</td>
<td>L25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>Nota Fiscal</td>
<td>Nota Fiscal</td>
<td>Opcional</td>
<td>Alfanumérico</td>
<td>15.6</td>
<td>L26</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>Data da Emissão</td>
<td>Data da emissão</td>
<td>Opcional</td>
<td>Alfanumérico</td>
<td>15.7</td>
<td>L27</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure C.1: UBL 2.0 Electronic Invoice Map**
### C.3 UBL 2.0 Mappings

<table>
<thead>
<tr>
<th>N. Itens</th>
<th>Mapeamento para UBL</th>
<th>Observações</th>
<th>UBL 2.0 ca</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Invoices:bc:Invoice:TypeCode</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Invoices:bc:Invoice:Identifier</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Invoices:bc:Invoice:IssueDate</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Invoices:bc:Delivery:ActualDeliveryDate</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>Invoices:bc:Invoice:Line:cbc:AllowanceTotal:cbc:TotalAmount</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>Invoices:bc:Invoice:Line:cbc:AllowanceTotal:cbc:TotalAmount</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>Invoices:bc:Invoice:Line:cbc:AllowanceTotal:cbc:TotalAmount</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>Invoices:bc:Invoice:Line:cbc:AllowanceTotal:cbc:TotalAmount</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>Invoices:bc:Invoice:Line:cbc:AllowanceTotal:cbc:TotalAmount</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>Invoices:bc:Invoice:Line:cbc:AllowanceTotal:cbc:TotalAmount</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>Invoices:bc:Invoice:Line:cbc:AllowanceTotal:cbc:TotalAmount</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>Invoices:bc:Invoice:Line:cbc:AllowanceTotal:cbc:TotalAmount</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>Invoices:bc:Invoice:Line:cbc:AllowanceTotal:cbc:TotalAmount</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>54</td>
<td>Invoices:bc:Invoice:Line:cbc:AllowanceTotal:cbc:TotalAmount</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>57</td>
<td>Invoices:bc:Invoice:Line:cbc:AllowanceTotal:cbc:TotalAmount</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>Invoices:bc:Invoice:Line:cbc:AllowanceTotal:cbc:TotalAmount</td>
<td>ok</td>
<td></td>
</tr>
</tbody>
</table>

Figure C.2: UBL 2.0 Mappings
Appendix D

InfoPath Electronic Invoice Template
Figure D.1: InfoPath Electronic Invoice Visual Template.
Appendix E

Business Activity Monitoring Portal

Figure E.1: BAM Portal Cube Example.
Figure E.2: Summary Map shown in BAM Portal.
Appendix F

Bottlenecks

<table>
<thead>
<tr>
<th>ServiceName</th>
<th>Service/Type</th>
<th>ServiceInstance/State</th>
<th>StartTime</th>
<th>EndTime</th>
<th>ServiceInstance/Duration</th>
</tr>
</thead>
</table>

Figure F.1: Bottlenecks in WebService invocation.
References


101


