Lossless Exchange of Data:

Limitations of Localisation Data Exchange Standards and their Effect on Interoperability

A Thesis Submitted for the Degree of
Doctor of Philosophy

By

Ruwan Asanka Wasala
Department of Computer Science and Information Systems,
University of Limerick

Supervisors: Reinhard Schäler, Dr. Chris Exton

Submitted to the University of Limerick, May, 2013
Abstract

Localisation is a relatively new field, one that is constantly changing with the latest technology to offer services that are cheaper, faster and of higher quality. As the complexity of localisation projects increases, interoperability between different localisation tools and technologies becomes ever more important. However, lack of interoperability is a significant problem in localisation. One important aspect of interoperability in localisation is the lossless exchange of data between different technologies and different stages of the overall process.

Standards play a key role in enabling interoperability. Several standards exist that address data-exchange interoperability. The XML Localisation Interchange File Format (XLIFF) has been developed by a Technical Committee of Organization for the Advancement of Structured Information Standards (OASIS) and is an important standard for enabling interoperability in the localisation domain. It aims to enable the lossless exchange of localisation data and metadata between different technologies. With increased adoption, XLIFF is maturing. Solutions to significant issues relating to the current version of the XLIFF standard and its adoption are being proposed in the context of the development of XLIFF version 2. Important matters for a successful adoption of the standard, such as standard-compliance, conformance and interoperability of technologies claiming to support the standard have not yet been adequately addressed. In this research, we aim to fill this gap by focusing on the identification of limitations of the XLIFF standard and the implementations which are leading to interoperability issues.

First, we conducted a pilot study to gain an in-depth understanding of the features of the localisation data-exchange standards. The pilot study involved a systematic comparison of XLIFF and Localisation Content Exchange, Microsoft’s internal localisation data-exchange format. Having gained a better understanding of the features of localisation data-exchange standards, in our main experiment we focused on identifying the limitations of XLIFF and its implementations. For this purpose we have constructed the first large corpus of XLIFF files that has enabled us to perform various statistical
analyses on the real usage of features of the XLIFF specification. From this research, we could not only identify the limitations that are leading to interoperability issues, but also important features of the standard for successful syntactic and semantic interoperability. In parallel to our main experiment, we designed and created a prototype based on service oriented architecture to investigate interoperability issues among distributed localisation tools and technologies used in the localisation process.

As the main contribution of our thesis, we propose a framework called XML - Data Interoperability Usage Evaluation (XML-DIUE), a systematic approach based on our main experiment, to the identification of limitations of data-exchange standards and implementations. The framework accomplishes this by helping the standard developers to empirically identify the most important elements, the less important elements, the problematic areas, usage patterns, associated tools, and so on. The proposed framework provides a practical approach for improving interoperability of data-exchange formats.
Declaration

I hereby declare that this thesis is my own work and that it has not been submitted previously to this for the award of any other degree or academic award at any other institute.

Much of this research has been published as journal articles, conference proceedings and presentations. A full list of publications can be consulted in Appendix I.

Some of the appendices (Appendix B, Appendix C, and Appendix E) could not be made available to the public due to the requirements of confidentiality of content provided by Microsoft.

Signature :_____________________________

Ruwon Asanka Wasala

Date: :______________________________
Acknowledgements

The study was carried out under the supervision of Mr. Reinhard Schäler and Dr. Chris Exton of the University of Limerick, Ireland. I would also like to thank my previous supervisors Dr. Ian O'Keeffe and Dr. Liam Murray for their support. I owe my deepest gratitude to my supervisors for their invaluable guidance, time, advice and kindness. I consider myself very fortunate for being able to work under very supportive eminent scholars like them.

Thank you to Microsoft for the generous scholarship that has allowed me to undertake my Masters research. In particular, many thanks to Mr. Dag Schmidtke for arranging the scholarship, helping me to gain access to Microsoft localisation resources and for guiding me throughout the research. I would also like to take this opportunity to thank the Microsoft staff who took time out of their busy schedules to help me with LCX programming and answering my questions. We would also like to thank the CNGL industrial partner organisations that contributed to the XLIFF corpus. Thanks to Dr. Jim Buckley and Dr. David Filip for all their guidance, suggestions and feedback, especially for our main experiment. I am further indebted to Geraldine Exton, wife of Dr. Chris Exton for proofreading the final version of the dissertation.

Enormous thank you to Mr. Karl Kelly and Ms. Geraldine Harrahill of the Localisation Research Centre for their immeasurable assistance during the course of this work. I would also like to thank all post-doctoral scholars of the LRC for their help and their time. Further thanks should also go to all of my colleagues at the LRC PhD lab.

I would like to extend my thanks to the CNGL staff for allowing me to attend various meetings, training sessions and workshops. The knowledge I gained from those events has immensely assisted my study.

Last but not least, I thank my gorgeous daughter, Siluni, who brought laughter and relieved the stress of my PhD life. I am very much thankful to my parents, my
wonderful wife, and my brother for their cheerful recognition, continuous encouragement and love of all my academic activities.

The PhD research was supported by the Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation (www.cngl.ie) at the University of Limerick, Limerick, Ireland.
Table of Contents

Abstract ........................................................................................................................................... i
Declaration .......................................................................................................................................... iii
Acknowledgements .......................................................................................................................... iv
Table of Contents .............................................................................................................................. vi
List of Tables ...................................................................................................................................... xii
List of Figures .................................................................................................................................. xiii
List of Appendices ............................................................................................................................. xv
List of Abbreviations ....................................................................................................................... xvi
Chapter 1: Introduction ...................................................................................................................... 1
  1.1 Research Introduction .................................................................................................................. 1
    1.1.1 Interoperability .................................................................................................................... 2
    1.1.2 Localisation .......................................................................................................................... 3
    1.1.3 Interoperability in Localisation ............................................................................................ 4
  1.2 Research Questions, Objectives and Scope ................................................................................ 7
  1.3 Methodology .............................................................................................................................. 8
    1.3.1 Selection of a Standard ......................................................................................................... 9
    1.3.2 Pilot Study: Data-Exchange Format Feature Analysis and Comparison ......................... 10
    1.3.3 Development of the XML Data Interoperability Usage Evaluation (XML-DIUE) Framework .......................................................................................................................... 10
    1.3.4 Development of LocConnect: an Interoperability Test-bed ......................................... 11
  1.4 Research Contributions ................................................................................................................ 12
  1.5 Outline of the Thesis .................................................................................................................. 14
Chapter 2: Literature Review ............................................................................................................. 16
  2.1 Introduction ............................................................................................................................... 16
  2.2 Approaches to Solving Interoperability Issues ......................................................................... 18
  2.3 Levels of Interoperability .......................................................................................................... 19
6.2 Conclusions and Recommendations......................................................... 177
   6.2.1 Main Conclusion: The Utility of the XML-DIUE Framework.............. 177
   6.2.2 Other Conclusions and Recommendations........................................ 179
6.3 Research Contributions............................................................................ 185
   6.3.1 Resources Produced by this Research.............................................. 188
6.4 Impact of the Research........................................................................... 189
6.5 Limitations of the Research.................................................................... 191
6.7 Future Work............................................................................................ 192
   6.7.1 Improvements to the XML-DIUE Framework................................. 193
   6.7.2 Other Possibilities........................................................................... 196
6.8 Final Remarks......................................................................................... 197
Bibliography................................................................................................. 198

xi
List of Tables

Table 1. Summary of the XLIFF-LCX File Format Comparison ................................. 84
Table 2. File Sizes and trans-unit Counts of Individual Sources .............................. 110
Table 3. Structure of the Validate Table ................................................................. 111
Table 4. Structure of the Tag Table ......................................................................... 111
Table 5. Structure of the Attribute Table ................................................................. 111
Table 6. Structure of the Children Table ................................................................. 112
Table 7. XML-DIUE Usage Metrics ........................................................................ 114
Table 8. Core Features of the XLIFF Standard ......................................................... 124
Table 9. A Few Example Attributes and their Values .............................................. 125
Table 10. Least Frequently used Elements ............................................................... 129
Table 11. The Least Frequently Used Attributes ...................................................... 132
Table 12. Elements and Attributes Found in the Minimal XLIFF Document .......... 137
Table F1. Attributes and their Typical Values ......................................................... 221
# List of Figures

Figure 1. The Research Scope ................................................................. 16  
Figure 2. The Research Scope – Interoperability ................................... 18  
Figure 3. The Research Scope – Standards ............................................. 28  
Figure 4. The Research Scope – Interoperability and Standards .............. 34  
Figure 5. The Research Scope – Localisation Domain ......................... 45  
Figure 6. The Research Scope – Localisation Standards ....................... 46  
Figure 7. The Research Scope – Localisation, Standards and Interoperability 55  
Figure 8. Segmentation Support in XLIFF .............................................. 76  
Figure 9. Virtual Header and Body Sections of LCX .......................... 77  
Figure 10. Graphical Visualisation of the LCX Format’s `<Item>` Element 80  
Figure 11. A Simple XLIFF-based Localisation Workflow .................... 81  
Figure 12. Basic Localisation Workflow at Microsoft ......................... 82  
Figure 13. LCX-XLIFF as the Intermediate Data Container .................. 90  
Figure 14. Sample UI Resource ............................................................. 92  
Figure 15. Typical Localisation Workflow with LCX-XLIFF Converter ... 94  
Figure 16. The Parser-Repository-Viewer Architecture Proposed for XML-DIUE ... 106  
Figure 17. Commonly Used Features ...................................................... 117  
Figure 18. Plot of log (Relative Usage Frequencies) of XLIFF Elements ... 126  
Figure 19. XLIFF Feature Distribution in Files and Organisations .......... 127  
Figure 20. Validation Results ................................................................. 134  
Figure 21. The Graph of Validation Result vs log(trans-unit Count)........... 135  
Figure 22. Pattern Derived for Eclipse .................................................. 136  
Figure 23. Analysis of XLIFF Editors ...................................................... 140  
Figure 24. Analysis of XLIFF Generators .............................................. 140  
Figure 25. XLIFF-Based Data Container ................................................ 166  
Figure 26. Improved Data Container ...................................................... 175  
Figure D1. Screenshot of the LCX-XLIFF ↔ XLIFF File Converter ....... 212  
Figure D2. XSLT Configuration Dialog ................................................ 213  
Figure G1. Pattern Derived for Eclipse ................................................. 222  
Figure G2. Pattern Derived for Company A .......................................... 223  
Figure G3. Pattern Derived for Company B .......................................... 224  
Figure G4. Pattern Derived for Company C .......................................... 225
# List of Appendices

<table>
<thead>
<tr>
<th>Appendix</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appendix A:</td>
<td>The Localisation Research Centre (LRC) and the Centre for Next Generation Localisation (CNGL)</td>
<td>211</td>
</tr>
<tr>
<td>Appendix D:</td>
<td>File Format Conversion Tool User Guide</td>
<td>212</td>
</tr>
<tr>
<td>Appendix F:</td>
<td>Attributes and their Values</td>
<td>214</td>
</tr>
<tr>
<td>Appendix G:</td>
<td>XLIFF Usage Patterns Derived for Individual Organisations</td>
<td>222</td>
</tr>
<tr>
<td>Appendix H:</td>
<td>LocConnect: System Architecture, Features, API and Installation Guide</td>
<td>226</td>
</tr>
<tr>
<td>Appendix I:</td>
<td>Publications, Presentations, Posters and Achievements</td>
<td>265</td>
</tr>
</tbody>
</table>
## List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>BOM</td>
<td>Byte Order Marker</td>
</tr>
<tr>
<td>BPEL</td>
<td>Business Process Execution Language</td>
</tr>
<tr>
<td>BPML</td>
<td>Business Process Modelling Language</td>
</tr>
<tr>
<td>CAT</td>
<td>Computer Aided Translation</td>
</tr>
<tr>
<td>CDATA</td>
<td>Character Data</td>
</tr>
<tr>
<td>CLDR</td>
<td>Common Locale Data Repository</td>
</tr>
<tr>
<td>CNGL</td>
<td>Centre for Next Generation Localisation</td>
</tr>
<tr>
<td>CSIS</td>
<td>Department of Computer Science and Information Systems</td>
</tr>
<tr>
<td>EDC</td>
<td>European Development Centre</td>
</tr>
<tr>
<td>ETSI</td>
<td>European Telecommunications Standards Institute</td>
</tr>
<tr>
<td>EU</td>
<td>European Union</td>
</tr>
<tr>
<td>GALA</td>
<td>Globalization and Localization Association</td>
</tr>
<tr>
<td>GILT</td>
<td>Globalisation, Internationalisation, Localisation and Translation</td>
</tr>
<tr>
<td>GMX</td>
<td>GILT – Metrics eXchange</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HER</td>
<td>Electronic Health Records</td>
</tr>
<tr>
<td>HTML</td>
<td>HyperText Markup Language</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute of Electrical and Electronics Engineers</td>
</tr>
<tr>
<td>ISO</td>
<td>International Organization for Standardization</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>ITS</td>
<td>Internationalization Tag Set</td>
</tr>
<tr>
<td>L10N</td>
<td>Localisation</td>
</tr>
<tr>
<td>LCX</td>
<td>Localisation Content Exchange</td>
</tr>
</tbody>
</table>
LISA  Localization Industry Standards Association
LRC  Localisation Research Centre
LSP  Localisation Service Provider
MLIF  Multilingual Information Framework
MT  Machine Translation
NLP  Natural Language Processing
OASIS  Organization for the Advancement of Structured Information Standards
OAXAL  Open Architecture for XML Authoring and Localization
ODF  Open Document Format
OLIF  Open Lexicon Interchange Format
OM  Object Model
OOXML  Office Open XML
OSCAR  Open Standards for Container/Content Allowing Re-use
OWL  Web Ontology Language
PO  Portable Object
QoS  Quality of Service
RDF  Resource Description Framework
SOA  Service Oriented Architecture
SOAP  Simple Object Access Protocol
SOLAS  Service Oriented Localisation Architecture Solution
SQL  Structured Query Language
SRX  Segmentation Rules eXchange
SSO  Standard Setting Organisation
TAUS  translation automation user society
TBX  TermBase Exchange
TC  Technical Committee
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIPP</td>
<td>Translation Interoperability Protocol Package</td>
</tr>
<tr>
<td>TM</td>
<td>Translation Memory</td>
</tr>
<tr>
<td>TMS</td>
<td>Translation Management Systems</td>
</tr>
<tr>
<td>TMX</td>
<td>Translate Memory eXchange</td>
</tr>
<tr>
<td>Trans-WS</td>
<td>Translation Web Services</td>
</tr>
<tr>
<td>UDDI</td>
<td>Universal Description Discovery and Integration</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
<tr>
<td>UTF-8</td>
<td>Universal Character Set (UCS) Transformation Format-8</td>
</tr>
<tr>
<td>W3C</td>
<td>World Wide Web Consortium</td>
</tr>
<tr>
<td>WSDL</td>
<td>Web Service Description Language</td>
</tr>
<tr>
<td>WSMO</td>
<td>Web Service Modelling Ontology</td>
</tr>
<tr>
<td>XHTML</td>
<td>Extensible HyperText Markup Language</td>
</tr>
<tr>
<td>XLIFF</td>
<td>XLIFF Localisation Interchange File Format</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
<tr>
<td>xml:tm</td>
<td>XML based Text Memory</td>
</tr>
<tr>
<td>XML-DIUE</td>
<td>XML Data Interoperability Usage Evaluation</td>
</tr>
<tr>
<td>XSL</td>
<td>Extensible Stylesheet Language</td>
</tr>
<tr>
<td>XSLT</td>
<td>Extensible Stylesheet Language Transformations</td>
</tr>
<tr>
<td>YAWL</td>
<td>Yet Another Workflow Language</td>
</tr>
</tbody>
</table>
Chapter 1: Introduction

1.1 Research Introduction

Our research is mainly concentrated in three areas: localisation, interoperability and standards. The research will focus on the interoperability of localisation data. It aims to address significant aspects of data interoperability, mainly by identifying limitations of the existing localisation data-exchange standards and tools and by identifying effective ways of using data-exchange standards to facilitate interoperability across tools. The research also aims to explore successful interoperability frameworks in various domains (including the localisation domain), and to learn important concepts from those frameworks as well as accepted standards to present a practical interoperability framework for localisation-related standard improvement.

At the time of writing this thesis, we have come to a crucial stage where discussions related to localisation standards and interoperability issues are gaining greater attraction. For example, these days there is not a single localisation-related conference that does not talk about the topics of interoperability and localisation standards. Therefore, we believe our research timely in addressing important questions that are extremely relevant not only in the current context, but also in future contexts such as emerging distributed localisation processes.

The research was carried out at the Localisation Research Centre (LRC) of the Department of Computer Science and Information Systems (CSIS) of the University of Limerick (UL) Ireland. The research was supported by Microsoft Corp. (Ireland) and the Centre for Next Generation Localisation (CNGL) project (see Appendix A for a brief introduction to the LRC and the CNGL project).

The rest of the chapter is organised as follows: Sect. 1.1.1-3 provides an introduction to the main areas of research and subsequently, the research problem is framed. In Sect. 1.2, the research questions, research objectives and the scope is presented. Then, the research methodology is described in detail in Sect. 1.3. In Sect. 1.4, the main
contributions of the research are summarised. Finally, the Chapter concludes with an outline of the thesis structure (Sect. 1.5), summarising the individual chapters of the thesis.

1.1.1 Interoperability

“Interoperability” is a term dating back to early 1970s\(^1\). The term interoperability became most often associated with software engineering. While there are many definitions for the term ‘interoperability’, the IEEE definition of interoperability is generally accepted (Ibrahim and Hassan 2010; Naudet et al 2010; Waters et al 2009). This definition is as follows:

**interoperability.** The ability of two or more systems or components to exchange information and to use the information that has been exchanged.

(IEEE 1991)

Lewis et al (2008) defined interoperability slightly differently from the above definition as “The ability of a collection of communicating entities to (a) share specified information and (b) operate on that information according to an agreed operational semantics.” For Naudet et al (2010) interoperability is a problem that can arise when “two or more incompatible systems are put in a relation.” Almeida et al (2010) describes interoperability as the ability of software to co-exist with competing products and connect with other software. Following Powers 2006 (cited in Waters et al 2009) definition, we conceive interoperability as:

**the ability to reuse data from another information system without any intermediate transformation and human intervention.**

Interoperability is becoming increasingly important in heterogeneous environments (Almeida et al 2010). Interoperability facilitates the integration of different entities such as tools, businesses, technologies and processes. Standards play a prominent role in facilitating interoperability by providing agreed notations, rules, guidelines and

recommendations for storage and exchange of data. Tools that conform to the same standard can therefore interoperate with each other. Data-exchange standards that are based on the Extensible Mark-up Language (XML) are becoming ever-more pervasive (Li and Li 2004; Savourel 2001, pp.7-17; Folmer and Verhoosel 2011, p.15) and they can be categorised as either open or proprietary. Examples of popular XML-based open data-exchange standards (also known as open file formats) include: XHTML, DOCBOOK, and Office Open XML (OOXML). However, the definition of such standards is an arduous time-consuming process due to the constantly evolving nature of technologies, businesses, and tools (Ray 2009). That is, standards need to be constantly reviewed and updated to cope with the changing requirements, typically across multiple organisations.

In our research, we mainly focus on the interoperability issues related to localisation data-exchange standards. In the next section, we introduce the localisation domain, highlighting the problem and the significance of the research.

1.1.2 Localisation

The term ‘localisation’ has been defined as the

linguistic and cultural adaptation of digital content to the requirements and locale of a foreign market, and the provision of services and technologies for the management of multilingualism across the digital global information flow.

(Schäler 2009)

As the definition suggests, localisation is a complex process. Localisation is a relatively new field in computer science and it is closely related to fields such as Machine Translation (MT), Natural Language Processing (NLP) and computational linguistics. Localisation is different from translation due to its focus on non-linguistic cultural aspects of products (Lommel 2006). Localisation involves many processes: project management, translation, review, quality assurance (Esselink 2000, pp.3-24; Savourel 2001, p.9). Localisation requires a large effort, as it involves many languages, dealing
with characteristics and challenges unique to different languages such as handling right-to-left languages, collation and locale-specific issues.

Time-frame is another parameter that affects the complexity of localisation processes. Localisation processes require dealing with frequent software updates, short software development life cycles and simultaneous shipments (simship). Van-Genabith (2009) describes three global challenges to localisation: Volume, Access and Personalisation. The volume of content to be localised keeps on growing while the access to content is shifting towards mobile and speech-based innovations. Personalisation is gaining increased attention to further enhance the user experience: “the person is the ultimate locale” (Van-Genabith 2009). Different software is used to support the localisation process ranging from project management software to translation software.

A large number of file formats are encountered during the localisation process. These file formats may consist of both open standard and proprietary file formats. Localisation processes also involve different organisations (e.g. translation providers and localisation providers) and different professions (e.g. translators, reviewers, and linguists). Localisers are confronted by new challenges such as the requirement to localise mobile device content and to integrate with content management systems, especially in the context of rapidly changing web content. In this extremely complex process of localisation, the ultimate goal is to deliver localised content at the adequate level of quality (i.e. quality of translation, functionality, layout of user interfaces) and quantity (i.e. in terms of the volume, the number of languages and locales) while minimising the time and the overall cost. In order to achieve this goal, the successful integration of different entities involved in localisation such as software, platforms, file formats, organisations and professionals is essential. Interoperability is the key to successful integration.

1.1.3 Interoperability in Localisation
As already established, localisation processes are highly complex. They can involve thousands of files in dozens of different and ever evolving formats, cover a hundred
languages, and involve translation and engineering teams from around the world. The increasing time and cost pressure put on localisation teams by their internal and external clients can only be responded to by a whole ecosystem of tools and technologies. Different projects, different languages, different teams require tools and technologies that have been chosen specifically for their particular requirements. Tools and technologies need to be interoperable and need to work with the same data sent by localisers (or their corresponding electronic ‘agents’) along the localisation process represented essentially by connected component technologies.

**Localisation Standards and Interoperability Issues**

According to a recent survey\(^2\) by TAUS (2011a),

Thirty-seven percent of the 111 respondents to our survey think that the lack of interoperability costs their business more than 10% of their total translation budget (or revenue in case of the service providers). Twenty-five percent say it costs them more than 20%. Only nine percent think it costs them less than 5% of their translation budgets. Forty-three percent of the respondents don’t know exactly how much it costs them.

(TAUS 2011a)

Respondents to this survey were predominantly language service providers (41.8%), smaller buyers of translation (19.1%), large buyers of translation (10.9%), translators (8.2%), language technology providers (7.3%) and others (12.7%). According to these respondents, the main cause for lack of interoperability is the *lack of compliance to interchange format standards* (TAUS 2011a). Although the adoption of localisation standards would give benefits such as reusability, accessibility, interoperability, reduced cost to the localisers, software publishers either refrain from the full implementation of standards (Bly 2010) or do not pay attention to conformance testing of their products against standards (Lieske 2011), due to lack of evidence for improved outcomes and associated high costs (Kindrick et al 1996). One of the biggest problems existing today with regards to the software is the pair-wise product drift (Kindrick et al 1996), i.e. the output of one software needs to be transformed in order to compensate for other

software’s non-conforming behaviour. This problem is commonly seen in localisation software too (for example, consider a scenario where one tool implements all nine inline elements defined in the XLIFF\textsuperscript{3} standard specification, whereas another tool only implements a limited number of inline elements: i.e. it implements less than nine inline elements. Then to enable successful interoperability between these tools, mappings have to be created between supporting inline elements of both systems). This problem occurs especially due to the lack of adoption of localisation standards as well as incomplete implementation of standards by the vendors (Bly 2010; Lieske 2011; Vackier 2010).

### Standards and Standard Conformance

Interoperability is not a primary concern when new technologies are launched (Almeida et al 2010). Moreover, conformance testing is an area typically not given much attention by software vendors (Kindrick et al 1996). To exacerbate this situation, most of the key localisation standards only have loosely worded conformance clauses (Filip 2011). In the absence of a clearly defined conformance clause, vendors are unable to develop proper test suites for testing their products for conformance with standards. Some tools have been developed for validating standard-compliance (e.g. for XLIFF, TBX and TMX content), however, no tools have been reported to assess the actual level conformance with these standards, for a given file, except in the EU co-funded IGNITE project (LocFocus 2006; Schäler 2007), which is currently not operational.

Very few academic studies about interoperability issues related to localisation data-exchange formats and standards have been reported in the literature in recent years (Vackier 2010; Bly 2010; Lewis et al 2009). In some cases, partial solutions have also been proposed by both academia and industry (Cruz-Lara et al 2008; Schäler 2007; Schäler 2008; Lewis et al 2009).

The survey by TAUS (2011a) amply confirms the significance of finding solutions to interoperability issues related to localisation data-exchange formats and standards. The primary aim of this PhD research is to fill this gap in the literature, building on the

\textsuperscript{3} The XML Localisation Interchange File Format (XLIFF) standard will be discussed in-depth in Sect. 1.3.1, Sect. 2.8.1 and Sect. 3.3.1.
knowledge of accepted interoperability frameworks, standards and solutions in the localisation domain, as well as other domains, and present a viable, standards-based interoperability solution for the localisation domain.

1.2 Research Questions, Objectives and Scope

This research addresses two questions. Our substantive research question is:

RQ1: What are the limitations of existing localisation data-exchange standards and implementations?

The related secondary research question is:

RQ2: How can these limitations be overcome to adequately address significant aspects of interoperability in localisation?

The overall objective of this research is to improve semantic and syntactic interoperability aspects of prominent localisation data-exchange standards and propose strategies leading to a higher degree of standards-compliance among localisation tools. The first research question addresses the objective of identifying the degree of prescriptiveness and the degree of coverage of a standard by existing implementations. This in turn will allow us to investigate whether a standard is sufficiently detailed and well enough defined to guarantee interoperability at both semantic and syntactic level. The second research question mainly focuses on the identification of optimal, standard-based solutions for facilitating data interoperability among tools.

The scope of this research is limited to the identification of recommendations for standards development, implementation and compliance, leading to improved interoperability. The research will mainly focus on aspects such as standard-compliance, conformance clauses, conformance testing and interoperability testing.
Research in areas such as machine level interoperability, ontologies and ontology definition, workflows, as well as organisational interoperability, are outside the scope of this research.

1.3 Methodology

This section formalises the structure and the overall research methodology used to work towards our research objectives. Our thesis employs a mixed research method (Johnson et al 2007) where different elements of qualitative and quantitative approaches have been used for the purpose of in depth investigation of the research questions.

Mixed methods research is the type of research in which a researcher or team of researchers combines elements of qualitative and quantitative research approaches (e.g., use of qualitative and quantitative viewpoints, data collection, analysis, inference techniques) for the broad purposes of breadth and depth of understanding and corroboration.

(Johnson et al 2007)

We employed strategy triangulation (Oates 2006, p.37), where we combined several research strategies to find answers to our research question.

The use of more than one data generation method to corroborate findings and enhance their validity is called method triangulation ... Strategy triangulation: the study uses two or more research strategies.

(Oates 2006, p.37)

The strategies include a pilot study comparing two localisation data-exchange standards, a survey on XLIFF files and the design and creation of an interoperability test-bed. These strategies are briefly described in Sect. 1.3.2-4.
1.3.1 Selection of a Standard

While there are many different standards involved in the localisation process (see Sect. 2.8.1), we primarily considered the XML Localisation Interchange File Format (XLIFF) standard in this research due to a number of reasons:

The XLIFF standard is specifically aimed at addressing interoperability issues related to localisation data exchange between tools and technologies (XLIFF-TC 2007). One of the primary objectives of the XLIFF is to provide a solution to the proliferating disparate file formats containing localisation resources. The ever growing number of file formats adds to the cost of the localisation process in different ways. For example, localisation tool developers have to implement parsers whenever a new file format appears. On the other hand, upon receiving a new file format, localisers have to learn and use tools that support the new file format. XLIFF addresses these issues by implementing standard strategies to extract localisation data from virtually any file format and make them available to localisers in a standard format. Once the content has been translated, the translations can be inserted back to the original source file. Therefore, the tool developers only have to implement the XLIFF standard, whereas localisers can use tools supporting XLIFF to translate the content. A detailed description of XLIFF including its benefits is given in Sect. 2.8.1 and Sect. 3.3.1.

Our selection is also supported by the previous literature. Filip (2012) defines bitext as “text in two natural languages organized as a succession of ordered and aligned source and target pairs.” and he states that “bitext standardization is the key to localization interoperability.” Arguably, bitext is the most important component of localisation data. XLIFF is a bitext format. Other popular bitext formats include Translation Memory Exchange (TMX) and Portable Object (PO) formats (see Sect. 2.8.1). A comparison of open standards XLIFF and PO suggested that XLIFF is more suitable to be adopted in the localisation process (Frimannsson 2005). TMX is not considered in this research as it is primarily meant for the representation of translation memories, and not to exchange localisation data (see Sect. 2.8.1). Therefore, this leaves “XLIFF” as the only suitable standard for our study. Bly (2010) states “XLIFF is the best (only) hope for interoperability” while Cruz-Lara et al (2008) introduce XLIFF as the “control of the
interoperability between the industrial standards currently used for localisation”. Gracia (2006) also states that “XLIFF may be the next big thing for the localisation industry, as significant as Unicode”. Filip (2012) argues that “XLIFF is the state-of-the-art bitext standard format, superior in many respects (legal and technical) to proprietary and legacy bitext formats.” Having decided on the standard for further analysis and investigation, we used a systematic research methodology for answering research questions presented in Sect. 1.2.

1.3.2 Pilot Study: Data-Exchange Format Feature Analysis and Comparison

First, we carried out a pilot study (details provided in Chapter 3) using qualitative approaches, mainly to gain a thorough understanding of the features of localisation data-exchange standards and also to confirm our selection of the XLIFF standard for further research. The pilot study involved a systematic comparison of the open standard XLIFF and a proprietary file format Localisation Content Exchange (LCX), which is Microsoft’s internal localisation file format. The comparison, as well as the development of a mapping between these two file formats, helped to identify features of localisation file formats as well as their advantages and disadvantages. Having gained a much deeper understanding of the localisation data-exchange standards from the pilot study, our initial intention was to adopt and implement the methodology proposed by Shah and Kesan (2008; 2009) as an experiment to investigate interoperability issues associated with the XLIFF standard (details of Shah and Kesan’s research is presented in Sect. 2.7.2).

1.3.3 Development of the XML Data Interoperability Usage Evaluation (XML-DIUE⁴) Framework

As a part of the Shah and Kesan’s procedure, the most frequently used features of the standard need to be identified. However, their research does not provide a reproducible systematic procedure for determining the most frequently used features of the standard under consideration. Therefore, we decided to design an experiment mainly to identify

⁴ Pronounced as “XML-DEW”
the most frequently used features of the XLIFF standard. This leads to our second and main experiment. In this experiment, we gathered a large number of XLIFF files with the aid of some of the CNGL industrial partners and also by crawling the web for openly available files. Then we constructed a corpus of XLIFF files. The analysis of this corpus revealed many insights about the usage of the XLIFF standard. The initial results of this experiment motivated us to further explore the XLIFF corpus, especially towards finding answers to our first research question, which is the identification of limitations of standards and implementations. We developed a novel empirical framework driven by several sub research questions that provides valuable information about standard usage, for different stakeholders of the standard. This quantitative experiment yielded some significant results in answering our primary research question. Therefore we diverted from our initial goal of implementing Shah and Kesan’s framework to evaluate XLIFF and we concentrated on further refinement of our own framework, leaving implementation of Shah and Kesan’s framework for future work due to constraints and the boundaries of this research. The details of this main experiment can be found in Chapter 4.

1.3.4 Design and Creation: Development of LocConnect: an Interoperability Test-bed

In parallel to the above experiment, as a part of the strategy triangulation, we investigated how the XLIFF standard can be used to define a data container in a Service Oriented Architecture (SOA) to facilitate data interoperability among distributed localisation components. For this purpose, we used a “design and creation research strategy” (Oats 2006, p.106), where we developed an orchestration engine that uses XLIFF as the main messaging format to communicate between different localisation components thorough an open Application Programming Interface (API). The adoption of this research strategy was mainly motivated by existing research: the IGNITE Localisation Factory (see Sect. 2.8.4 for details) (Schäler 2007); the proposal by LRC on the distributed component-based localisation process (Schäler 2008) which was later described by Lewis et al (2009) for enabling interoperability among distributed localisation components using the XLIFF standard; and the proposal to use open API to enable interoperability between tools and technologies (Savourel 2007). The design and
creation of this interoperability test-bed enabled us to study the interoperability requirements of tools and technologies in SOA environments. The interoperability test-bed was named as “LocConnect” in our research. The development of LocConnect has also helped us to gain a better understanding of the Quality of Service (QoS) attributes that are important in terms of enabling interoperability, but are not covered by the standard (Lewis et al 2008). Further details about the development of the LocConnect interoperability test-bed and related findings are available in Chapter 5.

1.4 Research Contributions

This research yields important findings related to interoperability and localisation data-exchange standards and also addresses questions that are extremely relevant, especially for new and emerging distributed localisation processes. The literature review revealed that gaps exist in the academic research relating to interoperability issues associated with localisation data-exchange standards. The research aims to address these gaps by mainly focusing on the identification of limitations of standards and tools that are leading to interoperability issues. A novel framework is developed as part of the research to identify limitations of standards and their implementations in tools and technologies as well as to identify important features of standards for data interoperability. This framework has been applied to the XLIFF standard and the results are discussed in detail. In addition, an interoperability test-bed has been developed based on an SOA to determine and demonstrate the interoperability requirements of distributed localisation tools and technologies used in the localisation process.

The research mainly contributes to the knowledge through:

- Developing and presenting a novel empirical framework: the XML-Data Interoperability Usage Evaluation (XML-DIUE) framework, including a set of standard usage metrics that provide valuable feedback into the standard development and maintenance process. The proposed XML-DIUE framework helps to identify and analyse important features of a standard and important usage characteristics of a standard for successful data-exchange interoperability.
• Academic peer-reviewed publications and scientific presentations (See Appendix I):
  o (Wasala et al 2012a; Filip et al 2012; Wasala et al 2012b; Wasala et al 2011a; Wasala et al 2011b; Aouad et al 2011; Wasala et al 2010; Wasala and Filip 2013)

• The development of a novel SOA based interoperability test-bed named “LocConnect”. The LocConnect facilitates interoperability among distributed localisation components using open standards. It will serve as a testing platform for the interoperability of distributed localisation components. LocConnect has been filed as an Invention Disclosure at the University of Limerick (Wasala and Schäler 2012) and it is being used as the orchestration engine of the Service Oriented Localisation Architecture Solution (SOLAS)\(^5\), connecting over eight localisation components\(^6\) distributed across three universities in Ireland. It will be released as open-source to the community to foster research and further development activities.

• In addition, an important resource resulting from this research is the first large XLIFF corpus, with over 3000 XLIFF files that we collected as part of our main experiment. This will be a valuable contribution to academia as well as industry (e.g. tools developers, different content developers alike).

The research has influenced the development and maintenance of the XLIFF standard\(^7\) (See (Lieske 2011; Wasala et al 2010)). More importantly, we note that the XML-DIUE framework proposed in our research has been presented at the third XLIFF Symposium (Wasala et al 2012a) and subsequently, the XLIFF Promotion and Liaison (P&L) subcommittee of the XLIFF Technical Committee has decided to implement the framework\(^8\) to support the XLIFF standard development and maintenance process. Moreover, our research (especially the pilot study) has also influenced Microsoft’s decisions on adopting the XLIFF standard.

\(^5\) http://www.localisation.ie/solas/ [accessed 08 Jan 2013]
\(^6\) The localisation components are various components developed by fellow PhD researchers in the CNGL project as part of their PhD work. These components range from translation extraction, pre-processing, machine translation, to review and rating of translations.
\(^7\) Specific takeaways by the XLIFF TC from our research is available at: https://wiki.oasis-open.org/xliff/Consolidated%20Takesaways%20from%20First%20XLIFF%20Symposium [accessed 08 Jan 2013]
\(^8\) See XLIFF P&L Sub Committee Discussion: https://lists.oasis-open.org/archives/xliff-promotion/201211/msg00002.html [accessed 08 Jan 2013]
1.5 Outline of the Thesis

The outline of the dissertation is given in the following.

- **Chapter 1: Introduction.** The thesis begins with an introduction to the research domain outlining important terminology. The research questions are presented and primary objectives are described.

- **Chapter 2: Literature review.** This chapter reviews literature related to the three main areas we focused on in our research: interoperability; standards and the localisation domain. The chapter starts by presenting an overview of different aspects of interoperability. The use of standards, the relationship between standards and interoperability, conformance testing and interoperability testing are discussed next. Various standards used in the localisation domain are then presented; their issues and existing solutions are discussed. The chapter concludes by highlighting the need for a systematic framework to identify and address interoperability issues related to localisation data-exchange standards.

- **Chapter 3: XLIFF and LCX Comparison.** In this chapter, the pilot study carried out to gain a better understanding of the features of the localisation data-exchange standard is presented. The chapter commences with an overview of XLIFF and the LCX file format standard. Then, a systematic comparison of these file formats is presented, followed by a proposed schema mapping between the file formats. The chapter concludes with presenting the results of the comparison, including proposed improvements to the selected standards and highlighting interoperability issues related to XLIFF and recognising the vital importance of further research into data-exchange standards compliance and the interoperability of localisation tools and technologies.

- **Chapter 4: Development of XML-DIUE Framework.** This chapter presents the survey on XLIFF files in detail, including the data collection, construction of the first large XLIFF corpus and the development of the XML-DIUE framework. Various standard usage analysis metrics proposed under XML-DIUE are presented. Analysis of the XLIFF corpus using the developed metrics is presented next. The results of the analysis are presented and discussed in detail to illustrate the utility of the framework for identifying and addressing interoperability issues related to...
standards and implementations. Various limitations of the framework are then discussed. Finally, a set of guidelines on adapting XML-DIUE for evaluating the usage of similar standards is presented.

- **Chapter 5: Development of LocConnect.** The details of the design and creation of the LocConnect interoperability test-bed is presented in this chapter. The chapter describes the system architecture of LocConnect in detail. Then a critical discussion is presented. The chapter concludes highlighting how XLIFF can be used as a data container in an SOA and also presenting possible future work.

- **Chapter 6: Conclusions and Future Work.** This chapter summarises the conclusions drawn from the thesis and highlights the important contributions of this research at different levels. Finally, the possibilities for future research are presented.
Chapter 2: Literature Review

2.1 Introduction

Our research spans mainly three areas in the literature: interoperability, standards and the localisation domain. While all of these are significantly broad areas, the main emphasis and scope of our research lies in the overlapping section defined by all three areas (see Fig. 1). The following chapter summarises the main areas of our research as discussed in the relevant literature.

First, we review literature related to interoperability. In Sect. 1.1.1, we identified several definitions for interoperability and we selected the most suitable definition for our research. In this vast area of interoperability related research, our primary interest is on the identification of interoperability issues and approaches for addressing those issues. In Sect. 2.2, we briefly present causes and solutions for interoperability issues at a very high level. We then identify different levels of interoperability, where we select the most relevant interoperability levels for our study. We restricted our scope only to study syntactic and semantic interoperability levels and the relevant literature is presented in Sect. 2.3. Interoperability problems are widely discussed not only in the localisation domain, but in other domains too. In Sect. 2.4 we cover industry-specific models of
interoperability, with a detailed review of one applied in healthcare. Before moving onto the standard, in Sect. 2.5, we briefly present service-oriented architecture as a development style for distributed, loosely coupled and interoperable components.

We then review relevant literature on standards in Sect. 2.6. In Sect. 2.6.1, we review several definitions of standards. We identify several types of standards in Sect. 2.6.2. In our research, we specifically focus on open standards. Various definitions for open standards and their advantages as well as disadvantages as reported in the literature are presented. The relationship between open standards and open-source software is also briefly reviewed. We then focus on the literature related to both standards and interoperability (i.e. the overlapping sections of interoperability and standards of Fig. 1). We present a detailed review of literature in this area in Sect. 2.6.3. In the next section (Sect. 2.7) we focus on testing strategies to ensure interoperability, including, conformance testing and certification.

Next, we review literature related to the localisation domain, especially focusing on standards and interoperability issues. In Sect. 2.8.1 we give an overview of the prominent open standards in the localisation domain such as XLIFF, PO, TMX, TBX, SRX, GMX and others. Then we look at localisation-specific interoperability matters related to localisation standards and their implementation in Sect. 2.8.2-3. We narrow down our scope to studying interoperability issues specifically related to the XLIFF standard. We present an in-depth evaluation of the XLIFF standard followed by a review of existing solutions to the lossless exchange of data and interoperability in the localisation domain.

Finally, the chapter concludes by emphasising the necessity of a systematic methodology or a framework towards the discovery and resolution of interoperability issues related to localisation data-exchange standards.
2.2 Approaches to Solving Interoperability Issues

In this section, we briefly look at root causes for interoperability issues and solutions to address them (see Fig. 2).

Naudet et al (2010) identify three main causes for interoperability problems among systems: 1) incompatibility; 2) heterogeneity and 3) misalignment.

To solve various interoperability issues associated with different information systems, a number of alternative solutions have been proposed in the literature. When looking at a very high level, at least two suitable approaches can be identified to address interoperability issues among systems: 1) linking systems using a canonical model and 2) aligning systems two by two (cited in Berges et al 2010). Naudet et al (2010) describe the same two approaches in slightly different ways, using the terms “bridging” and “homogenisation”. Where bridging involves inserting intermediate systems to solve interoperability issues between inter-related systems, homogenisation involves performing syntactic or semantic transformations to homogenise systems.
Research on interoperability frameworks and solutions is based on the above approaches. Such frameworks or solutions can be broadly classified into two categories: 1) domain specific; and 2) non-domain specific (Ibrahim and Hassan 2010). While some interoperability frameworks are conceptual and theoretical, others are pragmatic and technical.

Interoperability can be achieved on different levels among different systems. The next section discusses important levels of interoperability.

### 2.3 Levels of Interoperability

Different levels (or layers) of interoperability have been identified in the literature based on the degree of interoperability and data interchangeability between systems.

Key levels are briefly described below:

1. **System specific data** – proprietary data that is not meant to be shared with other systems, hence these are isolated systems (Wang et al 2009);
2. **Machine level interoperability** – enables basic connectivity and data exchange between systems at machine or hardware level (e.g. TCP/IP protocol) (Ambrosio and Widergren 2007; Lewis et al 2008);
3. **Syntactic interoperability** – enables data exchange between heterogeneous systems by proving a common documented syntactic data structure such as Extensible Mark-up Language (XML). Although, at this level, there is no guarantee that the shared data makes sense and is understood by other systems (Ambrosio and Widergren 2007; Lewis et al 2008; Wang et al 2009; Folmer and Verhoosel 2011, pp.33-34);
4. **Semantic interoperability** – enables meaningful data exchange between systems by agreeing and conforming to common semantics for shared data (e.g. by using ontologies) (Ambrosio and Widergren 2007; Lewis et al 2008; Wang et al 2009; Folmer and Verhoosel 2011, pp.33-34);
5) **Organisational interoperability** – interoperability beyond semantics is achieved by defining context-aware “actionable” data that allows systems to perform actions on data that is understood by other systems. At this level, business procedures, objectives and policies are well understood by different systems (Ambrosio and Widergren 2007; Lewis et al 2008; Wang et al 2009; Folmer and Verhoosel 2011, pp.33-34).

The levels described above are ordered in a manner so that interoperability increases as agreements and conditions in each level are satisfied. However, it is not necessary to address all levels of interoperability for all interoperability problems.

In this research, our objective is to improve interoperability between tools and technologies, by identifying limitations of both standards and tools. Therefore, proprietary data that cannot be shared or hardware level interoperability are irrelevant to our research (so, level 1 and 2). The most important interoperability levels in the context of localisation data interoperability research are the syntactic and semantic interoperability levels. They are briefly discussed in Sect. 2.3.1 and Sect. 2.3.2 below.

Research on organisational interoperability is beyond the scope of our research, as our main focus is on identifying and addressing interoperability issues within the standard, and not on identifying interoperability issues arising from fully automated machine processing and exchange of data between organisations.

### 2.3.1 Syntactic Interoperability

This section defines syntactic interoperability and presents techniques used to achieve syntactic interoperability.
Defining Syntactic Interoperability

Park and Ram (2004) define syntactic interoperability as “the application-level interoperability that allows multiple software components to cooperate even though their implementation languages, interfaces, and execution platforms are different.”

Syntactic interoperability is concerned with the information exchange among systems by using a compatible data structure or format. Data structure and formatting is enforced by syntactic rules (Ambrosio and Widergren 2007) that are usually defined through protocols or standards.

As indicated by Li and Li (2004), the key challenges of syntactic interoperability are:

- identifying all the elements in various systems
- establishing rules for structuring elements
- mapping, bridging, creating crosswalks between equivalent elements using schemas etc.
- agreeing on equivalent rules to bridge different cataloguing and registry systems.

Achieving Syntactic Interoperability

Syntactic interoperability can be achieved through the adoption of standards such as XML, Simple Object Access Protocol (SOAP), Universal Description Discovery and Integration (UDDI) and Web Service Description Language (WSDL) (Park and Ram 2004). XML is widely used to encode data and organise data in a structured manner (Li and Li 2004). Hence, XML and XML based standards are widely used to achieve syntactic interoperability (Folmer and Verhoosel 2011, p.15).

For localisation tools to exchange data, the first step is to agree on a data exchange format. Data exchange standards have been defined for this purpose. Therefore, it is important to identify syntactic interoperability issues associated with these standards in our research.
2.3.2 Semantic Interoperability

Although XML facilitates the structuring of data, it does not provide user understanding of the meaning of data (i.e. semantics) (Waters et al 2009; Berges et al 2010; Li and Li 2004). Therefore, to enable successful interoperability among tools and technologies, it is important for tools to agree on underlying semantics of shared data, in addition to agreeing to the syntactic structure of the data.

In this section, we first examine the definitions of “semantic interoperability”. Then, we review semantic interoperability issues and their causes. Finally, we identify common approaches for addressing semantic interoperability issues.

Defining Semantic Interoperability

The term “semantic interoperability” was first introduced by Heiler (1995). He defined it as the “ability to exchange services and data with one another” adding that “semantic interoperability ensures that these exchanges make sense – that the requester and the provider have a common understanding of the ‘meanings’ of the requested services and data.” Berges et al (2010) define semantic interoperability as “the ability of one computer system to receive some information and interpret it in the same sense as intended by the sender system, without prior agreement on the nature of the exchanged data.”

Semantic Interoperability Issues

Semantic interoperability issues generally arise from incompatibilities and disagreements of terms and concepts used by different organisations (Sartipi and Yarmand 2008). Park and Ram (2004) have classified semantic conflicts into two different levels: at the data level and at the schema level. Data level conflicts such as data-value conflicts, data representational conflicts and data precision conflicts can occur due to multiple representation or interpretation of similar data in different domains. Schema level conflicts such as naming conflicts, entity-identifier conflicts and aggregation conflicts can occur due to “differences in logical structures and/or inconsistencies in the metadata (i.e. schemas) of the same application domain.” Ouksel
and Sheth (1999) identify four enablers of semantic interoperability: 1) terminology (and language) transparency; 2) context-sensitive information processing; 3) rules of interaction mechanisms; and 4) semantic correlation.

Achieving Semantic Interoperability

Achieving semantic interoperability involves making the implicit semantics of data and processes explicit by means of metadata (Heiler 1995). As such, achieving semantic interoperability is an arduous task. Often, human interpretation of data and processes is needed (Lewis et al 2008; Heiler 1995; Park and Ram 2004). Heiler (1995) presents various causes that make it difficult to make semantic information explicit: context-dependent nature of semantic information; the changing nature of the meaning of attributes and values; difficulties of documenting semantics of legacy systems and management of metadata among others. Common metadata models, such as metadata repositories, are proposed as a solution to address the above issues (Heiler 1995).

At present, semantic interoperability issues are also addressed by defining domain-specific ontologies (Lewis et al 2008; Heiler 1995; Waters et al 2009). Other methods for achieving successful semantic interoperability include: metadata (especially domain specific metadata) (Ouksel and Sheth 1999; Li and Li 2004); context (Ouksel and Sheth 1999); mapping-based and query-oriented approaches (Li and Li 2004; Park and Ram 2004); as well as hybrid methods combining all or some of the above methods (Park and Ram 2004). Cross-domain ontologies are proposed as a solution for achieving semantic interoperability among information systems of different domains (Waters et al 2009). Machine-readable high-level languages such as Resource Description Framework (RDF), Web Ontology Language (OWL), Web Service Modelling Ontology (WSMO), Business Process Execution Language (BPEL) and Business Process Modelling Language (BPML) play an important role in defining ontologies as well as representing semantics (Lewis et al 2008; Berges et al 2010; Waters et al 2009). Lewis et al (2008) as well as Park and Ram (2004) discuss the difficulties of creating and standardising ontologies. One of the main problems of using ontologies is the need to customise and adapt ontologies to tailor specific requirements of different organisations.
However, these customisations will adversely affect the ontology standardisation process as interoperability problems may arise between such customised ontologies.

**Summary and Conclusions**

In this section, we identified that syntactic interoperability enforces agreements on the structure of shared data among tools and technologies, while semantic interoperability enforces agreements of the underlying meanings of the shared data. We also identified that structural agreements are usually enforced using data-exchange standards such as XML. While these standards may also carry some semantics of underlying data, additional semantic interoperability needs are met by using methods such as metadata repositories and ontologies.

In the next section, we will show that data-exchange interoperability issues are also discussed in other domains and present how syntactic and semantic interoperability issues have been addressed in these domains.

### 2.4 Industry Domain Specific Interoperability Models

Interoperability issues are of course widely discussed not only within the localisation community, but also within other domains (Naudet et al. 2010) such as healthcare, e-governance, networking, military, and construction (Lipman et al. 2011). Lessons can be learnt by studying how similar issues have been addressed in other domains. As an example, we will briefly review some of the most important interoperability issues in the healthcare domain, mainly to illustrate approaches taken to achieve semantic and syntactic interoperability issues. We have chosen the healthcare domain due to the critical nature of the interoperability of health records.

#### 2.4.1 Healthcare

Interoperability is a crucial aspect of successful healthcare information systems. For example, the exchange of health records (such as the medical history of a patient, or test results) among different applications as well as institutions provides invaluable benefits.
At the same time, the correct interpretation of information that is being exchanged is vital to avoid life-threatening mistakes. Interoperability remains a huge problem in healthcare. A study by Ray (2009) reported that imperfect interoperability adds as much as $77.8 billion per year to the cost of healthcare in the USA. The main reasons include extensive use of “paperwork” in administrative procedures (Ray 2009) as well as the use of heterogeneous proprietary models for representing Electronic Health Records (EHR) by different medical institutions (Berges et al 2010). Therefore, interoperability is an active research area in the healthcare domain. Several healthcare information interoperability frameworks that are relevant to our study are summarised below.

In the healthcare domain, interoperability has been achieved by means of healthcare standards such as HL7 (v3): a standard for the exchange, management and integration of healthcare information (Sartipi and Yarmand 2008); OpenEHR (Berges et al 2010): an open standard supporting the specification of better interoperable EHR systems, and SNOMED: a comprehensive clinical terminology system (Sartipi and Yarmand 2008).

**Standard-based Framework for Achieving Data and Service Interoperability in eHealth Systems**

Sartipi and Yarmand (2008) propose a standard-based framework for achieving data and service interoperability in eHealth systems. The main objective of this research was to migrate data and services used in legacy healthcare systems into standards-based interoperable systems. The methodology adopted in this research involved transformation of data, messages and transaction information used within legacy systems into standard based representations by developing a series of mappings. The semantic interoperability between healthcare applications was achieved by mapping their clinical terms into unique terminology systems. The mapping process was carried out with the help of domain experts. While the meanings of shared data were unified by means of mapping individual terminology systems into unique terminologies, the data transmission aspect of interoperability was achieved by using a proprietary SOA based system (known as Oracle’s Healthcare Transaction Base), due to its compatibility with a selected set of standards. The above framework has been evaluated by means of a case study. The case study involved implementation and deployment of the framework in a
real-world environment with the help of associated industrial partners and two other research groups. Sartipi and Yarmand (2008) highlight the characteristics of selected healthcare standards towards facilitating interoperability and discuss the applicability of their methodology in other domains.

**Semantic Interoperability of EHRs**

Kalra et al 2009 (cited in Berges et al 2010) identified three levels of interoperability related to EHRs: level 1 - syntactical interoperability; level 2 – partial semantic interoperability; and level 3 – full semantic interoperability. Berges et al (2010) propose an ontology-driven framework based on semantic web technologies (i.e. OWL2) to achieve full semantic interoperability of EHRs. The core of this methodology is the development of an ontology called EHROnt, which represents definitions of clinical terms in two levels of abstraction: canonical and application. At the canonical level, ontological definitions of EHR statements are represented in a higher application independent manner. This is carried out by the experts in the medical field. At the application level, definitions of EHR statements are represented as they are understood in specific e-health systems. As such, the canonical layer is common, while separate application layers exist for different e-health systems. Each health institution is responsible for creating their own application layer of the ontology. Berges et al (2010) propose a semi-automatic method translating existing proprietary EHR representations into an OWL2 based application layer. Finally, a mapping is constructed between terms found in each application layer with corresponding definitions found in the canonical layer. The proposed methodology is claimed to enable ‘on the fly’ interpretation of exchanged clinical data among different systems regardless of their individual internal clinical data representation schemes (Berges et al 2010). However, evaluation of the proposed methodology was not presented in this study.
Summary and Conclusions

The definition for interoperability by IEEE (see Sect. 1.1.1) focuses on two aspects: “exchanging data” and “using the exchanged data”. Essentially, these two aspects are related to syntactic interoperability and semantic interoperability. Syntactic interoperability ensures successful data exchange while semantic interoperability ensures proper interpretation of exchanged data (i.e. the usage aspect).

It can be seen that existing research focuses on addressing both these aspects. Sartipi and Yarmand use mappings and SOA to achieve syntactic interoperability while centralised terminologies have been used to enable semantic interoperability. Berges et al mainly focus on addressing semantic interoperability issues by means of developing ontologies.

The main lessons that can be learned from the above studies are two-fold:

- importance of representing proprietary data in a standard format;
- importance of addressing both syntactic and semantic interoperability issues.

In the next section, we briefly review SOA as a popular mechanism to achieve data-exchange interoperability.

2.5 Service Oriented Architecture (SOA)

Service Oriented Architecture (SOA) is a mechanism to facilitate interoperability among systems (Sartipi and Yarmand 2008). Ibrahim and Hassan (2010) define SOA as “an architecture or a development style that builds on distributed, loosely coupled, and interoperable components of software agents called services.”

Most of the SOA based interoperability frameworks are domain specific and generic interoperability frameworks are rare. Existence of different types of SOA based
interoperability architectures adversely affects the seamless interoperability among systems (Ibrahim and Hassan 2010). By analysing several existing interoperability frameworks of different domains, Ibrahim and Hassan (2010) propose 16 attributes that are important for achieving seamless interoperability, especially within the SOA context.

Quality of Service (QoS) characteristics play a major role in SOA based architectures. Therefore work on QoS issues is gaining increased attention. However, standardisation of QoS characteristics has been found to be a difficult task (Lewis et al 2008).

Having looked at the interoperability related literature, in the next section we review literature related to standards, the second major area we considered in our literature review.

### 2.6 Standards

*The nice thing about standards is that you have so many to choose from*

*Tanenbaum 1989 (cited in Folmer and Verhoosel 2011, p.16)*

![Figure 3. The Research Scope – Standards](image_url)
In this section, we focus on standards (see Fig. 3). We first present definitions of the standard and then present different types of standards. We then review the characteristics of open standards and also the relationship between open standards and open-source software. Next, we pay our attention to the connection between standards and interoperability. Then, we review issues associated with standards that are leading to interoperability issues. Finally, approaches to achieve interoperability using standards are presented.

2.6.1 Defining Standards

In our literature review, we noted that there is no agreed upon definition for the term “standard”. In order to get different perspectives on standards, in the following section, we present several definitions to the question: “what is a standard?”

Almeida et al (2010) define standard as a “level of quality or attainment, or an item or a specification against which all others may be measured”, stating that in the technical domain, “standard ... is a framework of specifications that has been approved by a recognised organisation, or is generally accepted and widely used throughout by the industry”. Krechmer (2005) stated that standards “represent common agreements that enable communication, directly in the case of Information Technology (IT) standards and indirectly in the case of all other standards.” The ISO definition (ISO/IEC 2004) of “standardization” is: “the activity of establishing, with regard to actual or potential problems, provisions for common and repeated use, aimed at the achievement of the optimum degree of order in a given context.” While the ISO definition is the most widely used definition in the literature (Folmer and Verhoosel 2011, p.16), all these definitions convey the same idea. In a book published by the Asia Pacific Sub-Committee on Standard and Conformance, standardisation has been identified as a simplification activity:

Standardization is simplification. The foundation of standardization is simplifying certain things that would otherwise grow complex if left alone, thereby raising the degree of interoperability among things. Simplification brings about cost efficiencies and increased convenience.

(APEC-SCSC 2010, p.12)
This layman definition of standardisation is more suitable for our research, as our research aims to identify complexities associated with existing localisation data-exchange standards and to simplify the standards to improve the degree of interoperability among tools and technologies.

In the next section, we identify different types of standards.

### 2.6.2 Different Types of Standards

There are several types of standards. Broadly they can be classified into two ways: 1) open standards; and 2) proprietary standards. Standards can also be domain-specific or non-domain specific. In addition, standards can also be categorised into different types based on many different typologies. Folmer and Verhoosel (2011, pp.16-21) summarise these types in their research.

Regardless of whether the standard is open or proprietary, it may be accepted by the majority of the community. Such standards, that are emerged from the community, are known as *de facto* standards (e.g. PDF) (Lewis et al 2008). Arnold (1994) as well as Lommel (2006) discuss disadvantages of *de facto* standards: e.g. standardisation of a product that is incompatible with other standards, standardisation of patented techniques, prevention of migration to competing products, tight dependence with the product associated with the *de facto* standard. More formalised and controlled standards created by standardisation organisations are known as *de jure* or *by law* standards. A problem with *de jure* standards is that they rarely take real-life experience into account. Therefore *de jure* standard implementations become problematic in most cases (Lewis et al 2008). The logical alternative to avoid such drawbacks is the creation of independent vendor-neutral open standards (Lommel 2006).

In the following, we review open standards in detail, covering their advantages and disadvantages and their relationship with open-source software (OSS).
Open Standards

There are several definitions for open standards. By analysing various definitions for open standards in the literature, Almeida et al (2010) identify three major characteristics that need to be satisfied in order to qualify a standard as an open standard:

- Easy accessibility for all readers and users;
- Developed by a process that is open and fairly easy for anyone to participate;
- No control or linked to any specific group or vendor.

Krechmer (2005) proposes another view, consisting of ten rights that enable open standards (open meeting, consensus, due process, open IPR, one world, open change, open documents, open interface, open use, on-going support). These rights have been identified by understanding the different types of rights that may be desired by standard creators, implementers and users. Krechmer (2005) argues that until each Standard Setting Organisation (SSO) clearly indicates which rights of open standard they support and the level of support, “open standards” will just be another marketing slogan.

Tiemann (2006) proposes four levels of open standards:

Open standard level 0: "The standard is documented and can be completely implemented, used, and distributed royalty free." The standard setting process should be transparent and allow independent participation. Furthermore, the standard should be technology and platform independent.

Open standard level 1: "There is specified OSS that can interoperate with the standard." In this scenario, the users can access the source code to gain full control over data. The selection of this standard will invite competition from software developers, while mitigating the risk of vendor lock-in.
Open standard level 2: "There is an OSS software reference implementation of the standard." While open standard level 1 provides guaranteed exist strategy of a standard implementation, open standard level 2 provides the ability to review the actual implementation of the standard in the reference implementation.

Open standard level 3: “The implementation of the standard is an OSS implementation.” This is the most suitable route for determining faults associated with the standard. Furthermore, this enables better understanding of the standard and its continuous improvement.

**Advantages and Disadvantages of Open Standards**

Benefits of open standards include: the elimination of vendor lock-in, the promotion of interoperability by enabling data exchange, offering better protection for content, and easier conversion from and to other file formats (Almeida et al 2010). Anastasiou (2011) highlights three major benefits of open standards in addition to the above: 1) open access to the standard; 2) ‘customisation’ i.e. ability to adapt standards according to needs and preference; and 3) the transparency of the process. Open standards foster innovation, improve transparency, expand and stabilise the market and ensure economic growth (Almeida et al 2010).

Shapiro (2001, p.88) argues that open standards could constrain innovation. For example by imposing limits on the design choices of organisations. This may result in either reduction in variety or new products that do not conform to standards. Anastasiou (2011) points out that often there is lack of awareness about open standards and extreme customisations of the standard hinders interoperability. Choudhury (2012) also mentions several concerns when adopting open standards: an “overhead of supporting unnecessary features, the perception of restricting creative freedom, the dilution of competitive advantage and the divulgence of proprietary information”.
Open Standard vs. Open-Source

Almeida et al (2010) discuss the relationship between open-source and open standard. Open standards are natural resources for open-source developers. Although open standards are closely related to open-source software, open-source software by itself does not guarantee interoperability between different implementations. According to Tiemann (2006), the availability of an open-source implementation provides a highly robust mechanism to achieve the benefits of the underlying standard. The availability of the source codes of open-source software allows users to modify them so that they will conform to standards. Thus, open-source implementations may promote interoperability through transparency (Almeida et al 2010). Moreover, open-source software is useful as a practical reference implementation of standards, thus encouraging the adoption of standards.

Summary and Conclusions

The aforementioned benefits of open standards motivated us to focus our research on open standards in the localisation domain. As highlighted in the previous section, the availability of open-source implementations of open standards brings enormous benefits. Therefore, it is important to identify the open-source implementations of localisation data-exchange formats in this research.

It is important to note that Tiemann’s interoperability definitions are applicable to open file formats too (Tiemann 2006). Hence, these interoperability levels provide a basis to assess the interoperability of the localisation data-exchange standards in relation to the availability of open-source implementations.

In the next section, we review the literature concerning both standards and interoperability (i.e. the overlapping sections of standards and interoperability. See Fig. 4).
2.6.3 Standards and Interoperability

Problems of interoperability can lie in meeting the needs of different standards.

(Currie et al 2002)

Figure 4. The Research Scope – Interoperability and Standards

Relationship between Standards and Interoperability

Almeida et al (2010) assert that “standardization of technologies, tools or processes guarantees interoperability” and mention that a high degree of interoperability can only be achieved by strict adherence to standards and specifications. Consistent with Almeida et al, Kindrick et al (1996) also state that a key benefit of standards is successful interoperability. Choudhury (2012) points out that numerous industries have benefited in terms of “enabling automation and fostering innovation” as a result of interoperability driven by standardisation. He further indicates that “traditionally, industry-wide interoperability has happened through the adoption of standards created by a consortium or standards organisation, or through the de facto adoption of a market leader’s schema.” Folmer and Verhoosel (2011, pp.33) state “Standards are the means to achieve the goal of interoperability.” and they quote Chari and Seshadri 2004: “Adopting standard-based integration solutions is the most promising way to reduce the long-term cost of integration and facilitate a flexible infrastructure”.

34
Enabling interoperability involves making agreements on different interoperability levels (Folmer and Verhoosel 2011, pp.33). See Sect. 2.3 for different levels of interoperability. We perceive standards as a mechanism to present these agreements and to make them explicit. Therefore, different standards may enable interoperability at different levels.

In this research, our main focus is on the identification of issues of both standards and implementations leading to interoperability issues. In the next few sub-sections, we review various limitations of standards as reported in literature.

**Achieving Interoperability beyond Syntactic Level with Standards**

While most of the standards help to achieve machine level and syntactic level interoperability, they are not always capable of successfully addressing the requirements of achieving semantic and organisation level interoperability. Lewis et al (2008) present two problems that make it difficult to achieve interoperability in semantic and organisational levels: 1) technical difficulties associated with the development of a language to represent and model a particular domain; and 2) human disagreements on constituents of the domain and their importance.

**Ambiguities in Standards**

Ainsworth (1989) states that “Information Technology (IT) standards generally contain options, 'implementor[sic] defined' and undefined features, and sometimes ambiguities.” This leads to the implementation of particular features in different ways for different products. Krechmer (2005) points out that there is a mismatch between the goals of standard-creators and product-implementers. Product-implementers and standard-creators have different expectations from open standards and this could result in lack of tool support for standards. Berges et al (2010) mention that although standards are useful in solving interoperability problems, however the problems remain until all the standards are merged into a single standard. Ray (2009) mentions standards are often developed without giving attention to testing, along with highlighting the importance of
testing of standard-based solutions to ensure interoperability, as does Garguilo et al (2007).

Adoption and User Involvement in Standardisation

Arnold (1994) discusses factors that affect standard adoption. These include the potential user community of standards, timescales before future revisions and integration with other standards. Arnold observes the need of active involvement of companies (i.e. product manufacturers) in the standardisation process. Companies can closely monitor the standards (e.g. new versions) and influence the development of the standard. Zhao et al 2007 (cited in Folmer and Verhoosel 2011, p.24) identifies three main reasons for participating in the standardisation process:

- adaptation of standards to cater for own business practices and requirements;
- the sooner the standard develops, earlier the benefits are realised;
- companies can get the benefits out of participating in in-depth discussions with peers.

Soderstrom (2003) points out that users of the standards are very important in the standard development process. However, user involvement in standardisation is not widely discussed in the literature (Soderstrom 2003). Soderstrom mentions users are "the ones who ultimately decide the success of a standard - by using it or not using it." Soderstrom summarises the benefits of user contributions in the standard development process. Lelieveldt 2000 (cited in Soderstrom 2003) argues that "every standard requires certain decisions to be made about their use and realisation." As such, lack of user engagement and feedback adversely affect the standardisation process. This lack of engagement may be due to several factors, for example, the high cost associated with obtaining membership of standardisation bodies, or lack of time to be involved actively in the process.
Why Standards may Never be Enough to Guarantee Interoperability?

Lewis et al (2008) argue that standards may never be enough to guarantee interoperability, especially due to the mismatch between expectations and what truly can be achieved by standards. They describe possible reasons as to why standards may never be enough to achieve interoperability:

1) extensions and customisations;
   Some standards are very flexible and they provide extension points. However, this leads to proprietary implementation of standards. Some vendors customise standards in order to compete with similar products. For example, unnecessary use of extensions may result in “accidental ties to particular suppliers” (Arnold 1994).

2) lack of backward compatibility;
   Standard version incompatibilities of different products lead to interoperability issues. Organisations and vendors face various difficulties especially if a new version of a standard is significantly different from the previous versions.

3) existence of bad standards;
   A bad standard exhibits one or more of following characteristics: under specification; over specification (e.g. provision of extreme flexibility in terms of extension points); inconsistencies within the standard; instability (i.e. frequent modifications to the standard); irrelevance (standard does not address legitimate needs).

4) conflicting standards;
   Different standards by competing organisations may result in conflicts. There are three different ways standards can conflict: overlap (several features of multiple standards are common); mutually exclusive standards (e.g. some standards only work with a set of other standards); competing standards (results from groups that are directly in competition and so many features are common e.g. Microsoft’s Office and Open Office formats).

5) inflexible standards;
Standardisation may be destructive especially for emerging domains as it may hinder flexibility.

In the next section, we review proposed solutions to address some of the issues associated with standards.

**Achieving Interoperability using Standards**

Lewis et al (2008) propose a four-step process to achieve interoperability using standards. The main idea here is to identify advantages, limitations and risks associated with standards and devise a careful strategy to mitigate risks. The first step involves assessing the required level of interoperability among systems. The analysis of end-to-end scenarios is recommended for this purpose. The second step involves an understanding of existing standards with respect to identified levels. This includes the assessment of a standard in terms of its adoption by the industry, extensibility, supporters of the standard as well as the standards’ governing body. The third step involves analysing the gaps of standards. The final step involves filling these identified gaps.

In this research, we aim to address the second and third steps above, by assessing and identifying limitations of the existing localisation data-exchange standards. Moreover, Lewis et al’s research gives us indications on possible limitations of the localisation data-exchange standards.

**An Evaluation and Selection Framework for Interoperability Standards**

Mykkänen and Tuomainen (2008) identify the lack of attention paid to the systematic analysis of important aspects of standards in relation to the application of interoperability. They highlight the need for a pragmatic methodology for evaluating important aspects of standards, and propose a conceptual framework particularly for the evaluation of those aspects of standards that are important for application interoperability. This framework comprises the qualitative assessment of a chosen
standard using feature analysis. In this framework, a standard should be examined against 54 considerations proposed. These considerations cover important aspects of the standards such as: 1) scope and domain of the standard; 2) main approaches to interoperability; 3) information and semantics; 4) functionality, behaviour and interactions; 5) architectural considerations; 6) technical considerations; 7) flexibility, accuracy and extensibility; 8) maturity and diffusion; 9) relation to the system’s life cycle; and 10) domain-specific position of the standard. They show that the qualitative assessment of a standard using the above framework helps to identify and clarify important interoperability related aspects of the standard. Moreover, it will also help to compare similar standards with each other in a systematic manner. In the same study, Mykkänen and Tuomainen point out that the modularity of standards promotes scalability, innovation and specialisation, especially due to the fact that different parties can develop different modules.

In this research we aim to further explore aspects 3, 7, and 10 above, in related to localisation data-exchange standards.

In the next section we review various testing strategies and approaches to ensure interoperability.

2.7 Conformance, Interoperability and Certification

Standard based solutions have to be tested to ensure interoperability. The most commonly used testing mechanisms to ensure interoperability are: 1) conformance testing; and 2) interoperability testing (Kindrick et al 1996; Ray 2009; Shah and Kesan 2008).

Kindrick et al (1996) summarise advantages and disadvantages of conformance testing as well as interoperability testing. The ideal scenario is to perform both kinds of tests, however it is impractical in most cases due to associated cost.
The following sub-sections give a detailed overview of conformance testing and interoperability testing.

### 2.7.1 Conformance Testing

*Any consumer knows that just because a product passes a test or conforms to a standard does not mean that the product will satisfy his/her needs.*

Frenkel 1990 (cited in Arnold 1994)

For Arnold (1994) ‘conformance’ means “that an implementation produces effects which the standard permits.” Conformance testing ensures whether a product meets all requirements of a standard and it behaves as expected by a standard (Arnold 1994; APEC-SCSC 2010, p.8). Kindrick et al (1996) give another definition to conformance testing as “the assessment of a product to determine if it behaves as expected while interoperating with a reference system.” We prefer the latter definition as it involves comparison of a product with a reference system in addition to testing against the requirements of the standard.

The history of conformance testing goes back to the early 1970s, for testing computer programming languages (Ainsworth 1989). Ainsworth (1989), Asano (1990) and Kindrick et al (1996) have all stressed the significance of conformance testing for achieving interoperability. Benefits of conformance testing include (Kindrick et al 1996; Arnold 1994):

- early detection of interoperability problems in a product’s life-cycle;
- the use of carefully developed test suites that maximise the coverage while minimising redundant tests;
- the strong connection with standards results in faster problem resolution and feedback to the standard development;
- promotion of confidence and momentum for product development in early stages of the standard’s life cycle;
• provision of widely available stable working references in the middle-to-latter stages of standard’s life cycle;
• provision of recognisable status for products;
• easy recognition of problems of products, due to focus on one implementation at a time;
• need to perform limited number of tests;
• consumers get a known set of functionalities, implemented and tested for conformance;
• make the functionalities independent from the suppliers or products.

Conformance test suites are being used to evaluate implementations and they usually consist of a set of test cases specifying precise objectives, operating conditions, inputs and expected outputs (Kindrick et al 1996; Shah and Kesan 2009). However, conformance test suite development is one of the most challenging tasks as they involve a significant budget, technical expertise and administrative overheads (Ainsworth 1989; Arnold 1994; Kindrick et al 1996; Shah and Kesan 2009). Therefore the development of own conformance test suites might be uneconomical and impractical for individual organisations (Ainsworth 1989). Realising these issues, there are initiatives to pool resources in order to develop conformance testing suites.

Garguilo et al (2007) discuss the importance of accurately defined conformance criteria as well as implementation statements of conformance in standards. A product that shows a successful level of conformance to a published standard implies successful interoperability among other products (Garguilo et al 2007). However, Arnold (1994) presents a contradictory view and mentions conformance testing does not guarantee interoperability. It is likely that non-conforming products will not interoperate effectively (Asano 1990). Shah and Kesan (2009; 2008) assert that “the best method to enhance interoperability is conformance testing.” Unlike interoperability testing, conformance testing is a “one-off” operation for each product (Ainsworth 1989; Asano 1990). Therefore, needless to say, the conformance testing is economical.
Ainsworth (1989) as well as Arnold (1994) highlight that the availability of recognised conformance testing suites have significant positive impact on the conformance level of products of a market. Kindrick et al (1996) mention that the availability of common testing environments and tools results in better products and reduced development costs. Asano (1990) gives examples of successful scenarios that have proven the effectiveness of making available conformance testing suites.

When conformance testing suites are published, vendors can buy access to those test suites and carry out their own tests prior to product certification or launch (Asano 1990; Kindrick et al 1996). Moreover, vendors themselves can apply for establishment of accredited conformance testing laboratories.

2.7.2 Interoperability Testing

Kindrick et al (1996) define interoperability testing as “The assessment of a product to determine if it behaves as expected while interoperating with another product.” Unlike conformance testing, interoperability testing does not attempt to cover all aspects of a standard. Aside from all the benefits of conformance testing, interoperability testing also has its own advantages. As mentioned above, interoperability testing is not confined to the requirements of standards. It goes beyond the standards to assess the implementations of important interoperability requirements of users. Therefore, interoperability testing helps to review the standards as well as to improve the standards by identifying important interoperability requirements that are not covered by standards (Kindrick et al 1996; Anastasiou 2011).

In contrast to conformance testing, interoperability testing takes every possible scenario into consideration and testing has to be repeated among every two products (Shah and Kesan 2008). Therefore the associated testing cost very is high compared to conformance testing (Ainsworth 1989).

Interoperability tests help to identify drawbacks and the effectiveness of standards. Test results can feedback into the further development of standards (Asano 1990). Errors of
the testing methods have to be identified and corrected as otherwise it will hinder the interoperability (Asano 1990).

In the following, we review an empirical methodology proposed to evaluate the interoperability of document formats. It is highly important in the context of this research due to the similar objectives of both. It provides us with valuable insights into a systematic methodology towards evaluating the interoperability of standards in the localisation domain.

**A Framework to Evaluate Interoperability of Document Formats**

The interoperability issues between document format standards such as Office Open XML (OOXML) and Open Document Format (ODF) have been widely discussed. Shah and Kesan (2009; 2008) propose a systematic methodology for assessing the level of interoperability among different software implementations of open document formats, particularly the ODF and OOXML formats. They assert the necessity for improved interoperability testing in order to prevent vendor lock-in. The methodology involves comparing different implementations against a chosen reference implementation. Test suites have been prepared to assess the level of tool support for the most frequently used features. Then, a scoring mechanism has been proposed to assess tools’ ability to: 1) read the documents; 2) read and write; and 3) handle metadata. The research presents significant findings related to interoperability issues and lack of solutions for the selected document format standards.

According to this research, a valid reason for tool developers not to offer full interoperability is the lack of need to support all the features in their tools. However, Shah and Kesan state that users expect 100% interoperability among implementations for various reasons including actual requirements as well as avoiding potential problems. In this research, they only focus on frequently used features of the standard as their goal is to assess whether implementations are "good enough for most users.” These routinely used features have been identified by “examining various instructional
materials for using office productivity software” (Shah and Keser 2008). Unfortunately, the details of their procedure for identifying these features are not given.

### 2.7.3 Certification

Conformance testing provides a basis for product certification (Ainsworth 1989). Product certification by an approved testing organisation will increase the reliability of the product (Asano 1990) and confidence that ‘approved’ products will interoperate (Ainsworth 1989). However, Arnold (1994) warns that “a false sense of security arising from a blind expectation that a certificate of conformance guarantees a particular level of performance is equally dangerous.”

Although certification can be carried out by any organisation, in order to make the certification process meaningful the certifying organisation should possess certain characteristics (Asano 1990; Ainsworth 1989):

1. technical capability to carry out tests and reporting capability;
2. maintain quality of the testing procedures;
3. wide recognition (international recognition);
4. in case of multiple laboratories, the use of the same tools and procedures;
5. trustworthiness and unbiasedness.

ISO/IEC GUIDE 2 (ISO/IEC 2004) defines accreditation of a test laboratory as “the formal recognition that a test laboratory is competent to carry out specific tests or specific types of test.” Accreditation is the process that assures credibility of a testing laboratory. It ensures the above characteristics of a testing laboratory are at a satisfactory level. Usually, accreditation is carried out by the main operating body of a system or a third party with adequate authority (Asano 1990). The main concern of accreditation is on administrative and procedural issues associated with certifying organisations or testing laboratories and to ensure that they are consistent and unbiased with regards to the clients, tools and procedures (Ainsworth 1989). Accreditation is awarded after successful assessment and surveillance of a laboratory (Asano 1990).
Ainsworth (1989) discusses two special considerations that need to be taken into account when certifying products: 1) whether implementation needs to pass all the tests (including other concerns like who decides the number of tests to be passed); and 2) time to carry out retesting (e.g. due to product updates, new releases).

In the next section, we review literature relevant to the localisation domain. This is the third major area considered in our literature review and certainly the most important area of our research.

### 2.8 Localisation Domain

In Sect. 1.1.2-3, we presented an introduction to the localisation domain (see Fig. 5), including definitions, challenges and interoperability issues related to data-exchange standards. In this section, we mainly focus on literature related to localisation standards (i.e. the overlapping sections of the standards and localisation. see Fig. 6).

![Figure 5. The Research Scope – Localisation Domain](image)

In our research, our scope is limited to studying standards used for localisation data exchange. Standards such as Unicode or standards related to locales such as Common
Locale Data Repository (CLDR) are not considered in this research as their primary purpose is not localisation data exchange.

In the next section (Sect. 2.8.1), we review important open standards of the localisation domain mainly to narrow down our scope of the research by identifying potential standards to study further.

### 2.8.1 Open Standards in Localisation

![Figure 6. The Research Scope – Localisation Standards](image)

Standards play an important role in today's localisation industry. Standards have become a prominent discussion in interoperability issues in the localisation industry (Lommel 2006). Apart from the influence of information abstraction and information commodisation requirements, the changing nature of information is another motivational factor for the emergence of localisation standards (Lommel 2006). Standards are important to maintain consistency between linguistic representations of information. Moreover, they provide high-level models for representing technical specifications (especially for translators), abundant, realisable and interoperable information (Cruz-Lara et al 2008). Another motivation behind the creation of
localisation standards is to prevent the language service customers being locked-in to proprietary solutions (Savourel 2007).

Hogan et al (2004) highlight the importance of XML standards in software localisation and internationalisation processes and mention that “the provision of XML standards and the ready availability and adaptability of open-source XML processing tools is one of the more exciting developments for the global-minded SME in recent years.” Savourel (2001, pp.7-429) presents an extensive discussion on using XML and XML-based standards (i.e. document types) in both internationalisation and localisation processes. Some of the prominent XML-based standards used in the localisation industry include: XML Localization Interchange File Format (XLIFF); Translation Memory eXchange (TMX) Format; Term Base eXchange (TBX) Format; and Segmentation Rules eXchange (SRX) Format. These and other major open standards used in the localisation industry are briefly described in the following sections.

**XML Localization Interchange File Format (XLIFF)**

The XML Localization Interchange File Format (XLIFF) is an open standard for the exchange of localisation data and metadata.

The XLIFF standard was first developed in 2001 by a technical committee formed by representatives of a group of companies including Oracle, Novell, IBM/Lotus, Sun, Alchemy Software, Berlitz, Moravia-IT, and ENLASO Corporation (formerly the RWS Group). In 2002, the XLIFF specification was formally published by the Organisation for the Advancement of Structured Information Standards (OASIS) (XLIFF-TC 2008a; XLIFF-TC 2008b; Raya 2004).

The purpose of XLIFF as described by OASIS is to “store localizable data and carry it from one step of the localization process to the other, while allowing interoperability between tools.” By using this standard, localisation data can be exchanged between different companies, organisations, individuals or tools. Various file formats such as plain text, MS Word, DocBook, HTML, and XML can be transformed into XLIFF,
enabling translators to isolate the text to be translated from the layout and formatting of the original file format.

XLIFF was originally developed to address the problems arising from the huge variety and ever-growing number of source formats, and has increasingly been used to exchange localisation data (XLIFF-TC 2007; Lewis et al 2009). In addition to its capability of interfacing with other file formats and standards, XLIFF can be used as a mechanism to store translations (i.e. as a container) with relevant metadata (Gracia 2006; Anastasiou 2010a; Lommel 2006). Due to this ability of associating metadata, it can be used as a “localisation data container” or “localisation memory” (Schäler, 2007; Schäler and Kelly 2007). XLIFF could even replace the functionality of TMX (Filip 2011) by serving as a TM representation format (Morado-Vázquez and Mooney 2010).

One of the core issues faced by translators and localisers is the need to retain and handle formatting information found in source file formats. XLIFF gives a better solution to this problem by isolating formatting information and localisable data (XLIFF-TC 2007; Gracia 2006). Similar standards such as PO and TMX do not provide this capability.

XLIFF has been adopted by major content publishers, localisation vendors and tool developers. Most renowned localisation tools such as Trados, Passolo, MemoQ support the XLIFF standard (Bly 2010). It is also worth mentioning that commercial localisation tools have been developed entirely based on the XLIFF standard9. The open-source community is also interested in XLIFF (Frimannsson and Hogan 2005) and some open-source localisation tools such as Virtaal10 and OmegaT11 already support XLIFF. Large content publishers such as Symantec and Oracle (Leahy 2011; Murphy 2010) internally use XLIFF in their workflows. Companies like Microsoft also show an increased interest in the XLIFF standard (O’Donnell and Stahlschmidt 2012) and by sponsoring the pilot study – MSc, publications such as Wasala et al (2010), Wasala et al (2012b) provide evidence for this. Another indicator for the growing interest in the XLIFF

---

standard is the annually held XLIFF international symposium. The survey by the Translation Automation User Society (TAUS) (2011c) confirms the significance of the XLIFF standard and the growing industrial interest towards improving the standard to enable interoperability (TAUS 2011a).

The XLIFF standard is being continuously developed further by the OASIS XLIFF Technical Committee (TC) (XLIFF-TC 2011). A detailed description about the structure of XLIFF is available in Sect. 3.3.1.

**Portable Object File Format (PO files)**

Localisation of software in open-source projects is usually carried out using a set of tools known as Gettext. Gettext uses a file format known as Portable Object (PO) format (Frimannsson and Hogan 2005), a simple text based file format for storing translations (GNU n.d) in the localisation process. The PO format can be considered as the *de facto* standard as it has been widely adopted by the open-source community to represent localisation data (Frimannsson 2005). A PO file can contain a header section consisting of metadata such as comments, references and so on, followed by translations. Frimannsson (2005) presents a comparison between PO and XLIFF file formats and proposes an XLIFF representation guide for the PO format.

**Translation Memory eXchange (TMX) Format**

The Translation Memory eXchange (TMX), an XML based open standard file format, was developed to exchange translation memory (TM) data between Computer Aided Translation (CAT) and other localisation tools.

The TMX format was developed and maintained by the “Open Standards for Container/Content Allowing Re-use (OSCAR)”, a special interest group of the Localisation Industry Standards Association (LISA)(2004). In March 2011, LISA declared bankruptcy and it submitted its intellectual property to the European
Telecommunications Standards Institute (ETSI) for maintaining the TMX as well as other LISA standards (SRX, TBX, xm:tm and GMX-v).

The objective of the TMX format as described by LISA (2004) is given below:

“The purpose of TMX is to allow easier exchange of translation memory data between tools and/or translation vendors with little or no loss of critical data during the process.”

With the advent of TMX, individuals and organisations can maintain and share TM databases in a vendor-neutral and tool-independent format and reuse their content as much as possible. Reusing existing TM content will significantly reduce costs in translation and localisation projects, according to LISA.

The TMX format is currently at version 1.4b (LISA 2004). LISA was working on TMX 2.0 which was to support inline mark-up before it went out of business. Today, most major CAT and localisation tools support the TMX format and it is considered to be one of the essential standards in the localisation industry (TAUS 2011a).

**Term Base eXchange (TBX) Format**

Glossaries are used with CAT or localisation tools to support translators, to achieve consistent translations and to speed up the translation process. In order to eliminate the difficulties of having glossaries in multiple file formats, Term Base eXchange, an XML-based standard for exchanging structured terminological data, was developed by LISA. The International Organization for Standardization (ISO) has accepted TBX as ISO 0042:2008 (LISA 2008b; ISO 2008). TBX is suitable for creating, maintaining and sharing terminological databases, dictionaries and glossaries. Glossaries represented in TBX can readily be used by many major localisation and CAT tools.

**Segmentation Rules eXchange (SRX) Format**

Different languages have unique segmentation strategies. For some languages, segmentation is a non-trivial task (e.g. in Chinese, Japanese and Korean, the “CJK”
languages) and it might be extremely difficult to implement algorithms to automate the segmentation process.

Having identified various interoperability issues associated with the segmentation of multilingual text, LISA defined a standard called Segmentation Rules eXchange (SRX) describing how tools segment text for processing (LISA 2008a). This vendor-neutral standard allows localisation and CAT tools to share and re-use language-specific segmentation rules and algorithms.

**Globalisation, Internationalisation, Localisation and Translation (GILT) – Metrics eXchange (GMX)**

GMX is a set of standards originally developed by LISA to represent different volumetric information about text in a standard manner (Lommel 2006). Standards developed by LISA including GMX have been taken over by The European Telecommunications Standards Institute (ETSI) since 2011. GMX consists of three parts: GMX-V for volumetrics; GMX-C for representing complexity of localisation tasks; and GMX-Q for representing quality aspects of localisation tasks.

An experiment carried out in 2001 (Lommel 2006) revealed that the differences of word counts by different tools can be as much as 30% depending on the nature of the source text. Word count is a significantly important unit in the localisation industry as most of the localisation costs are estimated based on word counts (Savourel 2007). Therefore, universal agreements between word counts as well as comparison mechanisms were needed. GMX-V attempts to address such issues by specifying rules on volume related metrics and storing (Lommel 2006; Savourel 2007).

**Translation Web Services (Trans-WS)**

Trans-WS is a specification that is being developed by OASIS Trans-WS TC to facilitate and automate communication between different entities involved in a translation or localisation project (e.g. translation vendors, localisation service provides,
clients) via the internet (Trans-WS 2002). According to Trans-WS TC (2002) the objective of Trans-WS is to provide a web-service framework so that “any publisher of content to be translated should be able to automatically connect to and use the services of any translation vendor, over the internet, without any previous direct communication between the two.”

Trans-web mainly focuses on the negotiation of translation and localisation jobs among service providers (Lewis et al 2009) and it aims to address issues related to localisation data and metadata transition, file transmission, job specification and job tracking (Lommel 2006). However, it does not consider language-service integration aspects or workflow automation. Although it is a very important standard in the context of SOA, and web services, the development of the standard is currently on hold and the uptake of this standard by the localisation industry is not evident from the literature.

**Internationalization Tag Set (ITS)**

ITS is a World Wide Web Consortium (W3C) recommendation that defines a set of elements and attributes “to be used with schemas to support the internationalization and localization of schemas and documents” (W3C 2007). With the help of XPath selectors, ITS helps to define and identify various metadata in XML documents (W3C 2007; Savourel 2007) as well as non-XML documents (Filip 2013) that are useful in the localisation process. The metadata categories defined in the ITS 1.0 include: Translate; Localisation Note; Terminology; Directionality; Ruby; Language and Text Elements (Filip 2013). While the current version is 1.0, a new major version (i.e. 2.0) is almost ready to be released (Filip 2013).

**Open Lexicon Interchange Format (OLIF)**

OLIF is an open standard to represent lexical/terminology data (OLIF-Consortium 2008). It is being maintained by the OLIF Consortium. The OLIF standard is primarily intended for the representation and exchange of Natural Language Processing (NLP) data, therefore it is different from TBX. OLIF provides a rich metadata structure suitable for the exchange of complex lexicon entries for tasks such as terminology
management and MT (Filip 2013). Various tools to work with OLIF resources have been provided by the OLIF Consortium (Savourel 2007).

**Multilingual Information Framework (MLIF)**

Having noticed the increasing number of standards related to the localisation industry, Cruz-Lara et al (2008) realised the importance of creating normalised terminological and methodological linguistic resources to improve information exchange and cooperation between standards. Cruz-Lara et al (2008) proposed MLIF, a new standard to normalise multilingual data and products. According to Cruz-Lara et al (2008) “MLIF aims at proposing a high-level abstract specification platform for a computer-oriented representation of multilingual data within a large variety of applications such as translation memories, localisation, computer-aided translation, multimedia, or electronic document management.”

MLIF is currently under development within ISO (AWI 24616) and it is a metamodel (i.e. a high-level generic mechanism for representing content within a specific context following elementary notions: structure, information and methodology) for conceptually representing multilingual content. MLIF attempts to cover all standards related to localisation by specifying a high-level model that represents and integrates a whole set of actors of the translation and localisation domain. The MLIF standard helps to achieve interoperability among localisation tools and to evaluate or compare different multilingual resources tools (Cruz-Lara et al 2008). MLIF is especially focused on representing the exchange of multilingual textual information. However, localisation has to deal not only with textual content, but also non-textual content such as binary files and images.

**XML based Text Memory (xml:tm)**

XML based Text Memory (xml:tm) is an open XML based standard developed by LISA, that allows embedding and representation of translation memories in any XML document (Gracia 2006). xml:tm uses an XML namespace mechanism mainly to embed the following information in XML documents(OAXAL-TC 2009; Gracia 2006):
1) Author Memory - different text segments identified by unique identifiers;

2) Translation Memory - specifies the text that is to be translated.

xml:tm supports interoperability among other localisation standards and it has been designed to work closely with SRX, XLIFF, TMX and GMX standards (Gracia 2006).

**Language Interoperability Portfolio (Linport)**

Language Interoperability Portfolio (Linport) is the latest localisation standard that is being developed within the Linport project\(^\text{12}\). Linport provides an open vendor-neutral format for "packaging translation materials" (Benitez 2011b). The proposed package is a zip container, mainly consisting of source and target text, various administrative data and references (Benitez 2011a). The Linport project is a collaboration between the Globalization and Localization Association (GALA), the European Commission Directorate-General Translation, and the Brigham Young University Translation Research Group. The first Linport symposium\(^\text{13}\) was held in 2011 to bring together interested parties for supporting the emerging standard. It is expected to host Linport under OASIS as a co-standard with ISO (Andrä 2012; Vándor et al 2012).

**Open Architecture for XML Authoring and Localization Reference Model (OAXAL)**

The Open Architecture for XML Authoring and Localization (OAXAL), is being developed under OASIS, defines a reference model (an abstract framework) describing the optimum use of key localisation standards. The stack of standards used in the OAXAL reference model includes: XLIFF, ITS, SRX, TMX, GMX, xml:tm and Unicode TR#29. The OAXAL reference model describes an extensible design architecture based on open standards to automate localisation workflows mainly concerning authoring and localisation of XML-based documents (OAXAL-TC 2009; Zydroń 2008).


2.8.2 Localisation Standards and Interoperability Issues

In this research, our core interest is in interoperability issues related to localisation data-exchange format standards (see Fig. 7). A small number of academic studies have been reported in this area in the literature in recent years. The following section gives an overview of prominent issues associated with major localisation standards.

**Increasing Number of Standards**

As presented in Sect. 2.8.1, a number of localisation-related standards already exist and localisation-related standards continue to increase. Increasing the number of standards is a problem in a particular domain. Lack of attention paid to completeness of interchange formats, renewals, and competition are some causes of succession in standards (Cruz-Lara et al 2008), and in particular, the incompatibilities between standards greatly affect the interoperability.

**Compatibility among Standards**

Although developed to standardise different aspects of localisation, the standards TMX, XLIFF and OLIF have overlapping features (Lommel 2006; Cruz-Lara et al 2008). The
lack of synergy between these standards affects their interoperability. Cruz-Lara et al (2008) point out that most of the translation related standards are non-compatible and mention that normalisation of standards has to be carried out in parallel to the standard development in order to minimise the incompatibilities among standards. The MLIF standard has been proposed to normalise localisation standards. However, neither uptake of this standard by the localisation community nor further development of the standard has been visible.

**De Facto Standards**

Apart from the open standards, some proprietary standards may become *de facto* standards due to their popularity and increased usage. Lommel (2006) highlights the disadvantages of *de facto* standards and points out that to a certain extent, the localisation industry is commonly using a proprietary standard so that it has become a *de facto* standard within the industry.

**Lack of Standard Support in Tools**

Lack of standard support in CAT tools is also a known issue (Anastasiou 2010a; TAUS 2011a). Savourel (2007) identified that data exchange between tools using TMX is not seamless due to implementation differences. Choudhury (2012) exemplified the significance of this problem using a response to their survey by a buyer who mentioned “we saw a leveraging loss of more than 20% when we switched from one CAT tool to another using TMX for data migration.” Limited support of the XLIFF standard and related issues are discussed in greater detail in Sect. 2.8.3.

**Segmentation Issues**

Lommel (2006) and Savourel (2007) discuss how segmentation affects the interoperability of localisation content. Aside from the existence of the SRX standard, segmentation issues have not yet been fully resolved. This greatly affects the interoperability as well as leveraging.
2.8.3 Evaluation of the XLIFF Standard

The core focus of our research is in the XLIFF standard. We are especially interested in XLIFF’s tool support, extensibility features, and complexity. The following section discusses related work in this area in greater detail.

XLIFF Tool Support

A survey conducted by Morado-Vázquez and Filip (2012) reports the status of XLIFF support in computer-aided translation (CAT) tools. This report tracks quarterly changes in XLIFF support in all major CAT tools. The survey was based on a questionnaire designed by the XLIFF Promotion and Liaison Sub-Committee of XLIFF TC and it is aimed at CAT tool producers. The main objective of this survey is to iteratively collect information that is useful for understanding the level of tool support for XLIFF. The survey reports a detailed characterisation of these tools with respect to XLIFF version support, use of custom extensions and XLIFF element and attributes support. In their survey, they tactfully avoid the use of the word “support” due to its ambiguous and prompting nature. Instead, they used the phrase “actively used elements” during the data collection phase. Only “Yes” and “No” answers have been collected. As such, the level of tool support for a certain element or attribute is questionable (e.g. given that a tool “actively uses” the `<file>` element, it does not necessarily imply that it conforms to the XLIFF mandatory requirements for the `<file>` element). Moreover, the credibility of the survey result is slightly doubtful due to the fact that the tool producers may provide biased or misleading information when responding to a survey that their competitors will also respond to, where the results are being made publicly available.

Bly (2010) analysed XLIFF support in commonly used tools and presented a matrix containing tools and their level of support for individual XLIFF elements. The level of support is divided into three categories: no support, partial support and full support. Unfortunately, again the author has not defined the terms “support” and “partial-support” precisely. The term `support` can lead to confusion (for example, given the fact that tool $T$ supports element $E$, does this imply that $T$ can process all the attributes of $E$?). Furthermore, the work, as presented, lacks significant details. For example, the methodology used to conduct the study is not revealed. Likewise, the development of
the test suites (e.g. source files) has not been described and the test suites have not been published. This makes the study difficult to replicate and refine.

Nevertheless, Bly (2010) contributes valuable insights: he concludes that tool vendors can conform to standards but still lock in users to their tools. Moreover, he discusses various problems associated with the XLIFF standard, such as their inability to support all the features offered by tools, and their lack of tight definitions. Notwithstanding, he is convinced that “XLIFF is the best (only) hope for interoperability.”

Imhof (2010) also performs an analysis similar to Bly’s (2010). Imhof (2010) identifies three levels of XLIFF support in tools: tools belonging to the first level treat XLIFF files as just XML files; tools belonging to the second level support XLIFF partially (i.e. these tools are capable of interpreting source and target content, but only a limited set of features of the XLIFF standard are implemented); and tools with full XLIFF support are grouped into the third level. As in the case of Bly’s analysis, Imhof also fails to describe the methodology followed in his study.

Frimannsson and Lieske (2010) identify three factors in relation to XLIFF and implementations that adversely affect the data exchange and interoperability. They are:

- existence of only a few XLIFF implementations;
- lack of coverage by implementations (i.e. implementations cover only a subset of XLIFF);
- incorrect handling of XLIFF by implementations.

In another study, Morado-Vázquez and Wolff (2011) present the open-source computer-aided translation (CAT) tool “Virtaal” that claims to support XLIFF. They compare its level of XLIFF support with the matrix presented by Bly. Morado-Vázquez and Wolff conclude that Virtaal is better in terms of XLIFF support than the average XLIFF editing tools checked in Bly’s study. Interestingly, Morado-Vázquez and Wolff point out a weakness in Bly’s methodology. Bly’s analysis does not take into account the
relative importance of different parts of the XLIFF specification. To address this issue Morado-Vázquez and Wolff propose a “weighted sum model” as a possible improvement. Furthermore, they highlight the importance of an element’s attribute and attribute values for tool interoperability. Similar to Bly’s, the exact methodology used to evaluate their tool, the test suites or the use of the term “support” are not included in their publication. Although they mention the use of Bly’s analysis methodology to evaluate Virtaal, the paper does not make explicit that methodology or Bly’s test suites for evaluating Virtaal.

In Anastasiou and Morado-Vázquez (2010) several interoperability tests were performed with three XLIFF compliant CAT tools. Like Bly (2010), they classified selected tools into two categories: XLIFF converters (i.e. generators) and XLIFF editors. Out of the three CAT tools selected, they found that two had the capability to generate XLIFF content and three had the capability to edit XLIFF content, so they were interested in four combinations: for each converter they wanted to see if the other two editing tools could edit the generated content. The researchers’ methodology involved five steps:

1) conversion of a selected HTML file into XLIFF (using the two converters);
2) validation of the converted XLIFF file;
3) manipulation of the XLIFF file using the editors;
4) manual analysis of the XLIFF file;
5) back conversion of the file into HTML and a manual analysis of the converted file.

The results showed that out of the four combinations (i.e. XLIFF generators and editors) considered in this research, only one pair of tools seems to interoperate successfully. The authors recommend “simplicity, better communication between localisation stakeholders and clarification of specification” of the standard and suggest future work on expanding the experiment with more CAT tools as well as different file types. It should also be noted that their experiment only considers the back-conversion of the
XL IFF files using the tool used to generate the XL IF F file. A better analysis could be
carried out if all possible scenarios were taken into account during the back-conversion
process too.

One of the reasons behind lack of tool support of XL IF F standard is the absence of a
proper conformance clause in XL IF F (Filip 2011; Anastasiou and Morado-Vázquez
2010). This also reflects Bly’s ‘lack-of-definition’ finding. Anastasiou (2011) stresses
that “conformance clauses should include criteria about compliance with both
Localisation and Semantic Web standards.”

Complexity of XL IF F

Among the other limitations noted is the complexity of the XL IF F file format. XL IF F is
considered as a very complex standard (Anastasiou 2010a; Anastasiou and Morado-Vázquez
2010). Anastasiou and Morado-Vázquez (2010) observe that the existence of a
large number of elements, attributes and pre-defined attributes in XL IF F contribute
towards its complexity (i.e. XL IF F v.1.2 has 38 elements, 80 attributes, 269 pre-defined
values). Frimannsson and Lieske (2010) describe XL IF F as an “overly complex,
bloated” file format developed under a standard body. They propose another view into
XL IF F’s complexity and they argue that the complexity of XL IF F is mainly due to its
large scope. They go on to list several dimensions that are encompassed in this large
scope. The dimensions include:

- generic representation of inline mark-up
- support for linguistic processing like segmentation
- support for internationalisation (language identifiers, markers for
  non-translatable content, notes for process participants)
- rich meta data on the level of strings (example: length restrictions
  and string type)
- rich meta data to log workflow events
- extensibility features
Finally, they point out that XLIFF covers several domains and layers such as “extraction domain and translation domain”, and this also contributes to XLIFF’s complexity.

In addition to above, Bly (2010) as well as Lieske (2011) point out tools’ ability to achieve the same effect using different tag configurations and semantics, suggesting an ambiguity in the standard. Bly (2010) gives some example scenarios to illustrate this.

**Lack of Awareness**

In another study, ‘lack of awareness’ has been already identified as a drawback of open standards (Anastasiou 2011). Besides XLIFF’s availability from 2002, a recent survey (Anastasiou 2010b) revealed that although 17% of participants in this survey had heard of XLIFF standards, they were not aware of its functionality. As indicated by Reynolds et al (2011), popularity ratio of XLIFF is 30%, however, the source for this statistic is not cited. Despite the lack of awareness about XLIFF, interestingly, the survey conducted by TAUS on XLIFF (Choudhury 2011) shows that the majority of the participants (i.e. 51% out of 434 respondents) in this survey are currently using XLIFF.

**Issues with Extensibility**

While the extensibility of the standard provides certain benefits (e.g. flexibility to adapt and enhance), it may also lead to interoperability problems (e.g. results in alternative methods to standardised methods, non-documented extensions and different flavours of the standard) (Vackier 2010; Filip 2011; Lieske 2011; Anastasiou 2011; Anastasiou and Morado-Vázquez 2010). Bly (2010) asserts that “Extensions are #1 barrier to interoperability” and provide details as to how improper use of extensions affect interoperability. Consistent with Bly, Imhof (2010) states that extensions “lead to incompatibilities as the XLIFF standard does not enforce usage of its own features.” Vackier (2010) presents a detailed study on both the advantages and disadvantages of XLIFF’s extensibility.
**Metadata Support**

Lewis et al (2009) point out that XLIFF is a resource-oriented standard and therefore it is not suitable for metadata aggregation. For example, XLIFF lacks the native support to information such as user information and workflow status. (Lewis et al 2009). While a way around this is to use extensions, Frimannsson and Lieske (2010) propose the use of new approaches such as Resource Description Framework (RDF) to represent additional metadata.

**Issues with Inline Mark-up**

Different tools represent formatting information using different tags. The lack of specification of a proper standardised mechanism of format mark-ups (i.e. inline mark-ups) have been a known issue for both TMX and XLIFF standards (Lommel 2006; Savourel 2007; Cruz-Lara et al 2008; Filip 2011; Imhof 2010).

**Miscellaneous Interoperability Issues**

The other factors that cause interoperability issues as mentioned by Anastasiou and Morado-Vázquez (2010) include lack of converters and version update of the standard. Un-defined processing expectations and poor open-source tool support (Frimannsson and Lieske 2010; Lieske 2011) will also have to be dealt with in relation to XLIFF.

**2.8.4 Existing Solutions**

The following section summarises proposed solutions to address some of the limitations described in the previous two sections (Sect. 2.8.2 and Sect. 2.8.3).
The IGNITE Project

Having identified various issues related to localisation data exchange, standards and tool support, an EU funded project named IGNITE was launched in April 2005 with a number of objectives.

The IGNITE consortium was established bringing together five EU-based organisations representing both industry and academia. The LRC is the main coordinator and contributor to the IGNITE project, and other project partners from industry were Archetypon, Pass GmbH, VeriTest and Vivendi Games. The initial objectives of this two year project were (Schäler and Kelly 2007):

1) to create and enable convenient access to a central repository of European linguistic and localisation resources and to establish a linguistic resources support network consisting of various stakeholders of the localisation industry;

2) to implement a demonstrator capable of reproducing localisation scenarios in a laboratory environment in order to verify localisation standards as well as to demonstrate the best practices in the localisation processes;

3) to establish LRC as an independent operation body that validates localisation standards, verifies implementations for standard-compliance, tests for interoperability and certifies implementations.

However, as the project evolved, more emphasis was placed on the second objective and this resulted in a prototypical implementation called the “IGNITE Localisation Factory”, a configurable and extensible automated localisation workflow execution system that demonstrates the effectiveness of using standards in the localisation processes (Schäler 2007; LocFocus 2006). Among the other important achievements and results of this project include the IGNITE Information Repository, the proposal to use XLIFF as a “localisation memory” and the IGNITE certification programme. These deliverables are briefly described in the following (Schäler 2007; Schäler and Kelly 2007).
The IGNITE Localisation Factory

The IGNITE Localisation Factory is a proof-of-concept of a standard-based end-to-end automated localisation workflow execution system. It uses XLIFF as a data container within its processes. The IGNITE Localisation Factory demonstrates the use of XLIFF to address interoperability between tools and also illustrates how to effectively deal with an increasing number of file formats for localisation.

The IGNITE Information Repository

The IGNITE Information Repository is a web-based infrastructure that enables storage, maintenance and access of a wide variety of resources used in localisation processes. The resources include translation memories, terminology databases, standards and guidelines, and external links. The resources have been made freely available to the public via a web-portal.

XLIFF as a Metadata Container

One of the significant outcomes of the IGNITE project which resulted from the IGNITE Localisation Factory was the development of a “localisation memory” container. A localisation memory container is an XLIFF-based independent extensible container for the representation and exchange of all localisation relevant data and metadata. It is a tool-independent resource container that can be used in end-to-end localisation workflows.

The IGNITE Certification Program

The third objective stated above has been achieved by proposing a proof-of-concept tool compliance testing and certification framework. This framework especially focuses on certifying tools against the XLIFF v. 1.2 standard. The framework involves checking a given XLIFF file against a pre-defined checklist in addition to validating the file against schema. The checklists assess a file for its efficiency, extensible elements in it (i.e. ratio of non-XLIFF elements to total elements) and tool capabilities. The tool capabilities are evaluated against a separate checklist.
Despite the reported success of this project, unfortunately the IGNITE project is no longer actively supported and proceeding.

**A Framework to integrate Globalisation, Internationalisation, Localisation and Translation (GILT) and Semantic Web**

Anastasiou (2011) presents a theoretical framework based on open standards for achieving interoperability between globalisation, internationalisation, localisation and translation (GILT) and semantic web. The methodological approach of the framework consists of following steps (Anastasiou 2011):

i. Internationalisation should be taken into account when standards are developed in the field of Semantic Web and semantics should be considered when GILT standards are created;

ii. Both Localisation and Semantic Web standards should have requirements which should be compatible with each other;

iii. Conformance clauses should include criteria about compliance with both Localisation and Semantic Web standards.

As a prototype implementation of the above framework, an XLIFF to RDF converter has been developed and published. Anastasiou further mentions that any file format that can be converted to XLIFF can be converted to RDF using this converter and therefore the converter can be used to achieve interoperability between other file formats to RDF too.

**Interoperability Now! Initiative**

An initiative named "interoperability-now" was established in 2010 (during the first XLIFF symposium) to create an open platform to collaborate on improving the interoperability of tools and technology within language infrastructures. It is mainly an alliance of LSPs and tool providers (BIOLOOM Group, Kilgray, Medtronic, ONTRAM, Spartan Software, XTM and Welocalize). Their objectives are twofold: first to propose a pragmatic solution to address various interoperability issues encountered

14 [http://interoperability-now.org](http://interoperability-now.org) [accessed 13 Mar 2013]
while sharing localisation project data. Second, to “provide a mechanism to allow lossless exchange of XLIFF data between tools in the localization workflow” (Bly, 2012). In achieving the first objective, they propose the Translation Interoperability Protocol Package (TIPP), a vendor-neutral localisation data container that can be used to share localisation project information. Having realised that there is an overlap of goals with Linport, both groups are now working together to propose a single container for localisation project data (Vándor et al 2012). Their main contribution towards achieving the second objective is the proposal of XLIFF:doc, a reference guide to representing documents such as office documents, FrameMaker, InDesign, XML formats, HTML and XHTML in XLIFF (Vándor et al 2012; Bly 2012). They mention that interoperability is achieved by using a shared namespace (i.e. an XLIFF extension) and be constrained to a specific subset of XLIFF 1.2 vocabulary (Bly 2012). However, the former criterion contradicts the argument over using extensions in XLIFF; i.e. it is widely discussed that XLIFF extensions diminish interoperability (see Sect. 2.8.3).

Interoperability Watchdog Initiative

The Translation Automation User Society (TAUS) is also actively engaged in enabling interoperability through their “Interoperability Watchdog” initiative15. One of the objectives of this initiative is to contribute to the XLIFF standard in order to improve data-exchange interoperability. TAUS performed several activities towards achieving this objective. One of the first activities conducted by TAUS was to conduct a global survey (with the help of XLIFF TC) on XLIFF’s adoption, usage and future requirements. 434 respondents from 37 countries participated in this survey. The survey results were presented at the second XLIFF symposium (Choudhury 2011). One of the highlights of the survey is that the majority of the respondents to this survey believe that XLIFF needs less extensions and less flavours.

Among the other important activities under the above initiative are the proposal of XLIFF core and XLIFF modules (Choudhury 2011) to the XLIFF TC and the

publishing of TAUS Translation API (Waldhör and Choudhury 2012) to address the needs of a common API.

**Next Generation Localisation Factory**

In 2008, Schäler (2008) proposed the concept of a component based distributed localisation process. Schäler described an eco-system of distributed components autonomously interoperating towards enacting localisation workflows depending on available resources and requirements. Later on, Lewis et al (2009) built on these concepts and proposed a conceptual interoperability framework known as a “Next Generation Localisation Factory” based on SOA. The main objective of this study is to integrate various localisation web services using a Business Process Execution Language (BPEL) based orchestration engine to instantiate commercial localisation workflows. XLIFF is being proposed to be used as the primary data exchange format in this framework. Several web services described in this study include, an MT web service which leverages translations from openly available MT systems, a text analytics web service that enriches an XLIFF file by adding various metadata by examining XLIFF content and a crowd-sourcing web service. In this architecture, such web services are orchestrated via BPEL to model localisation workflows.

**XLIFF Symposium**

The first XLIFF Symposium stimulated extensive discussions on interoperability issues related to the XLIFF standard. A summary of the outcomes of the symposium encompassing discussion on prominent issues along with important proposals on improving the XLIFF standard that are under the consideration of XLIFF TC are presented by Lieske (2011). Lieske categorises various feedback received on improving XLIFF into three categories: simplification, clarification and extension. For simplification of XLIFF, modularisation of the standard is proposed where a schema will be defined with minimalistic XLIFF with other optional modules. In terms of clarification, the importance of making explicit the XLIFF processing requirements is highlighted. Lieske presents several possible extensions to XLIFF, such as the provision of a programmable object model, creating a container based representation mechanism (e.g. as in the case of DOCX file format). It is worthy to note that some of those
improvements are direct contributions of our PhD research, as a part of the pilot study and our main experiment (Wasala et al 2010; Wasala et al 2012b; Wasala et al 2012a). The XLIFF Symposium is being held annually following the success of the first XLIFF symposium. The XLIFF TC is currently working on XLIFF 2.0 aiming to solve prominent interoperability issues (Filip 2011; XLIFF-TC 2011).

2.9 Conclusions

In this chapter, we have summarised the discussion in the literature of the main topics concerning our research.

As discussed in Sect. 2.4, interoperability is of course not only a problem in localisation, but in other domains too. Ray (2009) observed that matters around interoperability are, in principle, similar across different industrial domains, hence cost and time can be saved by looking across different domains for solutions, best practices and cost saving tools.

In Sect. 2.8.2 we identified that one of the central problems in localisation is interoperability caused by a lack of and deficits in localisation standards and implementations. Academic research aiming to address interoperability issues is still in its infancy. While most of the researchers report issues with the implementation and the XLIFF standard itself (Anastasiou 2010a; Bly 2010; Imhof 2010; Vackier 2010), a very few propose solutions to address these issues (Anastasiou 2010a; Lieske 2011; Schäler 2007; Frimannsson and Lieske 2010; Lewis et al 2009). Although standard-related interoperability problems have been solved in other domains using various interoperability frameworks (Folmer and Verhoosel 2011, p.38), and although it can reasonably be expected that a similar approach would also help addressing interoperability in localisation, such frameworks have not yet been developed for localisation.

In summary, there is a need for an interoperability framework (or a systematic methodology), similar to those used in other domains, facilitating the discovery of
limitations of standards and implementations that can be used to develop recommendations and guidelines, as well as the preparation of systematically derived test suites improving the data-exchange standards, as well as facilitating standard verification and compliance. Our research will propose such a framework for XLIFF, and investigate its role as the predominant data-exchange standard in localisation today with a strong potential to contribute significantly to interoperability in localisation, and to the problem of lossless exchange of data between localisation technologies.
Chapter 3: Data Exchange Format Feature Analysis and Comparison

3.1 Introduction

In our research, a pilot study was undertaken prior to the main study. This pilot study explored the general characteristics of localisation data-exchange formats, their advantages, limitations, and interoperability issues between those. The scope of the pilot study was limited to comparing two file formats, investigating the coverage and compatibility of the two file formats and implementing a demonstrator converting data between them. XLIFF, the open localisation standard maintained by OASIS and Localisation Content Exchange (LCX), the internal proprietary localisation standard used by Microsoft were chosen for this study.

Microsoft is one of the largest localised software publishers in the world, with their main localised software being Windows and Office. These are being localised into around 100 languages by the end of year 2012 (O’Donnell and Stahlschmidt 2012). Just to give an idea about the complexity involved in this process, Microsoft Office 2003’s help system alone consists of over 700,000 words (Ryan et al 2009). The reason behind Microsoft’s successful localisation strategy is their "proven large scale localisation model" (Ørsted and Schmidtke 2010) developed over 15 years. LCX plays a key role in this process. Therefore, LCX is an ideal candidate for our comparison. On the other hand, Microsoft is an official partner organisation of the CNGL project. Thanks to their interest in our project and support, we had access to their localisation resources throughout this research.

The following section briefly introduces the LCX format. An overview of the document structures of both the LCX and XLIFF formats and their typical usage is discussed next. Then the chapter goes onto details of the design of the pilot study followed by results and conclusions.
3.1.1 Localisation Content Exchange File Format

In order to address the different needs of the Microsoft localisation community, Microsoft uses a series of XML files to store software localisation data and metadata. These files, together, are referred to as the Localisation Content Exchange (LCX) container or Localisation Content Exchange file format. LCX evolved as a replacement for a prior, database-driven, software localisation model and toolset, with the main change being a move to store data in XML. The LCX-based architecture provides a programmable Object Model (OM) to access localisation content encoded in LCX files programmatically. The benefits of the LCX container (Microsoft 2009) include:

- common localisation project and transportation format;
- diff-able text files instead of a binary format;
- published XML schema;
- data transparency through a complete object model.

These LCX files are processed by the Microsoft Localization Studio (LocStudio) suite of tools, and other Microsoft software localisation tools. The LocStudio suite consists of several applications and utilities that provide functions ranging from localisation content extraction to localised product building.

3.2 Pilot Study Design

3.2.1 Data Collection

The dataset used for this pilot study consisted of both existing data as well as data generated specifically for this research. Initially, sample XLIFF files were collected. These comprised of files obtained through the CNGL industrial partners, and openly available XLIFF files downloadable via the Internet. Then, with the aid of several localisation tools, XLIFF files and LCX files were generated from Win32 source files. For further analysis, windows resource (*.res) files were also extracted from some of the above source files.

The study included a brief questionnaire and two main semi-structured interviews. The main objectives of the questionnaire as well as the interviews were to clarify various issues encountered on the LCX format. The questionnaire and the interviews were targeted at the subject matter experts of Microsoft. An interview and observation at Microsoft helped to understand the localisation processes associated with the LCX format and the usage of the LCX’s object model.

3.2.2 Methodology

The first phase of the methodology involved desk research, reviewing official documentations of LCX and XLIFF formats as well as related literature on both file formats. In the second phase, an element-by-element comparison of the LCX and XLIFF schemas was carried out. The third phase involved a manual analysis of sample LCX, XLIFF and resource files that represented localisation data extracted from selected Win32 applications. In the fourth phase, an experiment was designed and carried out to evaluate the interoperability between the XLIFF and LCX formats as well to evaluate the ability of XLIFF to support source file diversity. The experiment involved designing a mapping between the two file formats and the development of a file format converter.

3.3 Document Structures of XLIFF and LCX

The following subsections (Sect. 3.3.1 and Sect. 3.3.2) present the document structures of both XLIFF and LCX file formats.

3.3.1 Document Structure of XLIFF

General Structure

An XLIFF file begins with an XML declaration followed by an XLIFF document enclosed within the <xliff> element. An XLIFF document may consist of one or more <file> elements. A <file> element can only represent a single source and one target language (Savourel 2001, pp.384-385). A <file> element may correspond to a real or virtual file such as part of a database (Lionbridge 2006). A <file> element
has several attributes to identify source and target languages as well as attributes to relate the <file> element to the original document. Each <file> element contains a <header> section for storing various metadata about the project and a <body> section which contains localisable data. The <header> element may comprise of various elements such as <phase-group>, <glossary>, <reference>, <count-group>, <tool>, <prop-group>, <skl> and <note> to enclose various metadata such as contact information, project phases, reference material and information about the skeleton file (Corrigan and Foster 2003). The <body> element contains localisation data arranged in a hierarchical manner. A <body> element can contain textual or binary translation units such as images and icons. Textual data is represented by the <trans-unit> elements whereas binary elements are represented by the <bin-unit> elements. The <trans-unit> elements are organised into <source> and <target> paired elements; <source> elements hold the original data for localisation, whereas <target> elements contain the data that has been localised. Furthermore, <trans-unit> element may contain a number of <alt-trans> elements containing alternative translations. The <trans-units> elements can be grouped together recursively in a hierarchical manner.

**Workflow Metadata**

Workflow metadata can be stored in an XLIFF document during different phases of the localisation process. This metadata can be associated with different translation elements. Metadata specific to different localisation tools and different phases can be stored in the <phase> elements. The information stored in a <phase> element may describe the process or operation carried out on a segment (e.g. translation, editing) and the tool used to perform the process or operation (Lionbridge 2006). The <target> elements can then be associated with phases defined in the <header> section (Savourel 2003; Reynolds and Jewtushenko 2005; Frimannsson 2005). In addition, XLIFF provides a common set of attributes to store metadata specific to certain elements. For example, a few attributes defined for the translation unit include: `maxwidth`, `minwidth`, `size-unit`, `data-type` and `resource-type`. 
**Abstraction of Inline Mark-Up**

Usually, the layout and formatting information such as markers for bold or italics, image references, external links and so on, that are stored within special mark-up elements of the source document, should not be translated. XLIFF provides a mechanism to abstract these mark-up elements and to enclose this information separately within the appropriate inline elements. These inline elements can be represented either by using an encapsulation mechanism, or a placeholder mechanism (Savourel 2003). In the encapsulation mechanism, special mark-up elements are enclosed within the XLIFF inline elements. These inline elements include: `<bpt>` (begin paired-tag), `<ept>` (end paired-tag), `<it>` (isolated tag), and `<ph>` (placeholder tag). Text spanned within the encapsulated code can be delimited by using a `<sub>` element. In the placeholder mechanism, the extracted mark-up elements are stored in the skeleton file and XLIFF placeholder elements are used to track the location of the source document. Placeholder elements include, `<g>`, `<ex>`, `<bx>` and `<x>`. The `<g>` element is the generic placeholder element which is the equivalent of a `<bpt>` element. Its ending tag `</g>` is the equivalent of `<ept>` element. The `<g>` element is generally used to represent text formatting information (e.g. font-face, bold, italic) (Reynolds and Jewtushenko 2005). The `<x/>` element is the placeholder equivalent of the `<ph>` encapsulation element, and it is useful in representing empty mark-up elements such as an HTML image tag `<img>` (Reynolds and Jewtushenko 2005). Overlapping mark-up codes can be handled using `<ex/>` and `<bx/>` elements (Savourel 2003). The isolated `<it>` tag is used to delimit a beginning or an ending sequence of an original mark-up that does not have a corresponding ending or beginning marker within a translation unit (Raya 2004). In addition to the above inline elements, the `<mrk>` element is used to delimit the span of context within text. However, it is a general purpose element (Savourel 2003) and it can also be used to demarcate various properties of text such as semantic grouping and proper names. (Raya 2004).

**Statistics**

Various statistics such as word count, character count and repetitions of certain words that are useful in the localisation processes can be stored within XLIFF. The `<count>`
element is used to store such information. The count-type attribute of the <count> element indicates the type of the information <count> element represents (e.g. total count, number of repetition) and the optional unit attribute indicates the unit of the count (e.g. words, pages, sentences). These statistics can be optionally associated with a <phase> element defined under the <header> section. Furthermore, the <count> elements can be grouped by the <count-groups> element. The statistics can be associated with <header>, <group>, <trans-unit> and <bin-unit> elements (XLIFF-TC 2008b).

Segmentation

Segmentation plays a major role in the translation and the leveraging process. Leveraging can be improved by specifying proper segmentation rules. Having identified the lack of segmentation and alignment support in XLIFF v1.1 (Reynolds and Jewtushenko 2005), several elements (e.g. <seg-source>, <mrk>) have been introduced in the XLIFF v1.2 specification to support segmentation. The textual content within a <source> element can be segmented into several semantically or linguistically distinctive chunks and can be stored in <seg-source> element following the <source> element. After segmenting the source text, individual textual segments are stored within <mrk> elements (with attribute mtype set to the value "seg") surrounded by the <seg-source> element. Segmented text within the <target> element can also be represented using the same manner. Matching segment pairs in the <target> and the <source> elements can be identified by the value of the mid attribute in the <mrk> element. An <alt-trans> element can also have segmented textual content. See Fig. 8 for an example illustrating XLIFF’s segmentation support.

<trans-unit id= "1">
  <source>First sentence. Second sentence.</source>
  <seg-source>
    <mrk mtype="seg" mid="1">First sentence.</mrk>
    <mrk mtype="seg" mid="2">Second sentence.</mrk>
  </seg-source>
  <target>
    <mrk mtype="seg" mid="1">Translated first
Contextual Information

Contextual information related to <source> elements can be specified in <context> elements. Contextual information is useful to select the best translation and to ensure translation consistency. The context-type attribute can be used to specify the context and the type of resource (e.g. database, element, line number, record). Contextual elements can be grouped using the <context-group> element. The <context-group> can be named using the name attribute and can specify its purpose (such as ‘match’, ‘location’) by the purpose attribute (XLIFF-TC 2008b; Frimannsson 2005).

Extensibility

Extensibility is another key feature of XLIFF. Using XML namespaces, an XLIFF document or a part of an XLIFF document can be embedded in another XML document (XLIFF-TC 2007; XLIFF-TC 2008b). Moreover, another XML or a part of it can be embedded within an XLIFF document as well. XLIFF provides a number of extension points for this purpose. The XLIFF elements that can be extended to include customised XML elements include <alt-trans>, <bin-unit>, <group>, <header>, <tool>, <trans-unit> and <xliff>. Furthermore, an XLIFF file can be extended by adding user defined (non-XLIFF) attributes in the following elements: <alt-trans>, <bin-source>, <bin-target>, <bin-unit>, <bpt>, <bx/>, <ept>, <ex/>, <file>, <g>, <group>, <it>, <mrk>, <ph>, <seg-source>, <source>, <target>, <tool>, <trans-unit>, <x/> and <xliff>. It is also possible to add custom attribute values in some pre-defined elements. These attributes include alttranstype, context-type, count-type, ctype, datatype, mtype, purpose, reformat, restype, size-
unit, state, state-qualifier and unit. The XLIFF specification, however, places no restriction on the number of non-XLIFF attributes that can be used, or the order in which to insert those attributes in an XLIFF element. According to the XLIFF specification, customised attribute values have to be specified with an 'x-' prefix. However there is no mechanism to validate customised values (XLIFF-TC 2007). The intentionally extensible nature of XLIFF may result in interoperability issues as discussed in Sect. 2.8.3. Therefore, it is highly advisable to adhere only to standard elements and attributes as much as possible (Frimannsson 2005).

3.3.2 Document Structure of LCX

The LCX format is simpler compared to XLIFF. Unlike XLIFF, LCX does not explicitly define separate header and body sections. However, it can include references to various settings (files), user defined key-value pairs (known as property-bags) and comments on localisation items. Therefore, everything defined prior to actual localisation data can be considered to be included in a virtual header section as depicted in Fig. 9. The virtual header contains references to settings and comments by developers.

![Figure 9. Virtual Header and Body Sections of LCX](image)

77
One of the most distinguishing features of the LCX format over the XLIFF is LCX’s Property Bags feature. A Property Bag can store several key, value pairs. The keys can be defined to hold values in different data types including Numbers (Single and Int32 types), Boolean, Strings, and Dates. The Property Bags are useful to represent various metadata (usually required by Microsoft’s partners) related to localisation items. These values are shown in the custom fields of the LocProject Editor\textsuperscript{17}. Property Bags can be associated with most LCX format elements (Microsoft 2007). Due to all of the above mentioned reasons, the Property Bag feature can be described as the extensibility mechanism unique to LCX. A Property Bag can be defined by the \texttt{<Props>} element. The \texttt{<Props>} element holds different key-value pairs as mentioned above. See the example below:

\begin{verbatim}
<Props>
  <Str Name="Custom1" Val="Office12" />
  <Str Name="$User" Val="Moravia" />
</Props>
\end{verbatim}

In LCX, localisation data is always represented by the \texttt{<Item>} element. Furthermore, the same element is used to group several localisation items together. It also serves the purpose of XLIFF’s \texttt{<group>}, \texttt{<trans-unit>} and \texttt{<bin-unit>} elements.

A typical LCX \texttt{<Item>} element is given below:

\begin{verbatim}
<Item ItemId="11" ItemType="5" PsrId="3" Leaf="false">
  <Str Cat="Dialog Title">
    <Val><![CDATA[Notepad]]></Val>
    <Tgt Cat="Dialog Title" Stat="Loc" Orig="New"
      AutoAppr="N/A">
      <Val><![CDATA[Notepad]]></Val>
    </Tgt>
  </Str>
  <Bin BinId="8">
    <Val><![CDATA[AQAAAAAAAAAAAAAAAAAAAAAA==]]></Val>
  </Bin>
</Item>
\end{verbatim}

\textsuperscript{17}Microsoft® Localization Project Editor (LocProject Editor) is the localisation project management application of the LocStudio suite.
A `<Str>` element enclosed within the `<Item>` element holds the source and target strings. The source and the target strings are always represented in the character data (CDATA) notation. Any binary data associated with a localisation item is included within a `<Bin>` element. In the case of binary resources like accelerators, and bitmaps, the `<Str>` element will not be included in the `<Item>` element. When representing such resources, the `<Tgt>` element will be used inside the `<Bin>` element to store the target binary resource. Furthermore, the `<Item>` element can also store previous source data through the `<Prev>` element enclosed within either `<Bin>` or `<Str>` elements. Although only a single resource can be represented using the `<Prev>` element, several target resources localised into different cultures can be represented by the `<Ref>` element. In addition, each `<Item>` element can also hold a Property Bag (i.e. a `<Props>` element) with different key-value pairs. A special property bag element called `<PsrProps>` is used to store key-value pairs useful for configuring the parser. The LocStudio Editor-specific information related to the rendering of the LCX content is stored in the `<Disp>` element. The `<Modified>` element is used to keep track of the date and time and the person involved in the modification of a certain `<Item>` element.

An `<Item>` element can hold another `<Item>` element. Therefore, the `<Item>` elements are used recursively to build the localisation data hierarchy within the LCX file. This is achieved by grouping related localisation items together (e.g. grouping items in a dialog box). The structure of an `<Item>` element is depicted in Fig. 10.
3.4 XLIFF and LCX Usage

3.4.1 XLIFF Usage

XLIFF eliminates the need for multiple file formats in a localisation workflow. In an XLIFF-based workflow, XLIFF acts as an intermediate file format where all non-XLIFF tool specific file formats will be converted by a pre-processor to XLIFF at a very early stage of the localisation workflow (see Fig. 11). Then the converted XLIFF files will be handed off to the localisation engineers. It is also noteworthy to mention that there are XLIFF compliant tools capable of producing XLIFF files directly, which will eliminate the need for a pre-processor.

---

18 Each rectangle represents an element by its name. Attributes of elements are not included in this figure.
Once tool specific files are converted into XLIFF, XLIFF compliant editors can be used to facilitate the translation process. Having completed the translation process, the XLIFF files will be converted back to native file formats. This workflow simplifies the localisation process by eliminating the need to deal with multiple proprietary file formats throughout the workflow (XLIFF-TC 2008a).

Moreover, XLIFF can also be used as a data container that travels through the workflow (See Chapter 5 and (Wasala et al 2011a; Wasala et al 2011b) for details). The XLIFF file can be populated with various data and metadata during different stages of the workflow. In this scenario, the <alt-trans> feature of XLIFF can be utilised to include translations by these CAT tools in an XLIFF file (XLIFF-TC 2008a).

Viswananda and Scherer (2004) describe two approaches for using XLIFF in a localisation workflow:
1) Permanent XLIFF files
In this approach, XLIFF files will be stored in the source control system. The XLIFF files will be converted to native file formats only when building the target files.

2) Transient XLIFF files
In this approach, localisation content will be stored in the source-control system in native file formats. The native formats will only be converted to XLIFF to exchange with translators.

Out of these two approaches the former approach is recommended by Viswananda and Scherer (2004). However, the selection of the best approach out of the above two will depend on criteria such as the richness of native formats, the performance of supporting tools, including the conversion tools, and the data-loss during the round-trip conversion process (Viswananda and Scherer 2004).

3.4.2 LCX Usage
The LCX based localisation approach consists of seven major steps as depicted in Fig. 12. A variety of software and file format is used throughout the process. The major steps are summarised as follows.

![Basic Localisation Workflow at Microsoft](image)

**Figure 12. Basic Localisation Workflow at Microsoft**
1. Generate LCX source files;

This step is carried out at Microsoft and involves the creation of LCX files by processing localisable files (e.g. EXE, DLL, HTML). This step also involves the incorporation of developer and localisation pilot team comments, instructions and validation rules to the generated LCX files.

2-4. Preparation and handoff;

The main activity carried out in this phase is the auto-translation. Existing glossary files are used for this purpose. Then an XML based virtual archive containing all of the files required for translation is created. This archive is then handed over to the vendors for translation.

5. Translation;

The translation is carried out by vendors using software called LocStudio. This process also involves dialog-resizing. Finally, validations are performed using the same software based on the information contained in the LCX file, per resource.

6. Hand-back and validation;

The archive containing the translated resources is then handed back to Microsoft. The translated resources are extracted and further validations are performed on them prior to the acceptance process.

7. Build target files.

The Final step involves generating localised files (e.g. EXE, DLL, HTML) files from original files and translated LCX files.

3.5 XLIFF and LCX Feature Analysis and Comparison

Table 1 summarises the results of our comparison. A detailed element by element comparison of XLIFF and LCX features is given in Appendix B. In the following subsections we describe the advantages, disadvantages, as well as unique features of XLIFF and LCX formats in detail.

---

19 Appendix B could not be made available to the public due to the requirements of confidentiality of the content provided by Microsoft.
### Advantages and Unique features

- Open and flexible
- Support for variety of resource formats, both binary and textual
- Ability to store supplementary information and metadata
- Inline elements
- Alternative translations
- Ability to associate contextual information
- Ability to specify pointers to external resources
- Ability to specify segmentation of translations

### Disadvantages

- Lack of tool support
- Flexibility of extensions that results in the misuse of extensions
- Extensive use of proprietary data through extensions
- Lack of powerful and accurate mechanism to represent GUI components
- Weaknesses of the `<note>` element
- Ability to represent the same information using different tags or tag configurations
- Lack of support for representing workflow information
- Inability to associate metadata related to external resources

### XLIFF

- Proprietary and controlled
- Binary encoded information representation scheme
- Precise and secure representation of localisation items and GUI components
- Property Bags
- Consistent hierarchy
- Ability to embed target content validation rules
- Ability to associate comments with most of the elements
- Availability of an Object Model

### LCX

- Inability to store alternative translations
- Limited support for non-binary file formats
- Heavy use of CDATA sections
- Lack of specific mechanisms to store contextual information
- Lack of mechanisms to represent segmentation information
- Lack of explicit mechanism to store references to external files

**Table 1. Summary of the XLIFF-LCX File Format Comparison**
3.5.1 Advantages and Unique Features of XLIFF

XLIFF can represent virtually any file format with localisation resources. XLIFF can store both binary resources and textual resources. Therefore, it can be used to represent localisation data from software and program components (e.g. dll, exe) as well as other documentation file formats (e.g. txt, html, xhtml, xml, doc, docx, odf). Moreover, XLIFF can be readily used in web services to facilitate the localisation process (Lewis et al 2009; Mateos 2010). In addition to localisation data, XLIFF can store other supplementary information such as references to glossaries and administrative information such as various metadata related to the localisation workflow. Various metadata and contextual information associated with localisation data can be stored in a self-descriptive hierarchical manner in XLIFF as mentioned in Sect. 3.3.1. In some elements of XLIFF, there is a choice for storing localisation data or related supplementary information (e.g. glossaries, segmentation rules) within the element itself (i.e. internally) or to only store a reference (externally) to the actual resource. This provides a handy mechanism to store only pointers to cumbersome files (e.g. bitmap files, TMX files) reducing the complexity of the XLIFF file itself.

In XLIFF, alternative translations corresponding to a particular localisation item can be stored. The ability to store alternative translations brings significant benefits to the translators. Furthermore, there are nine inline elements defined in XLIFF. These elements are useful to accurately retain the formatting (and other) information attached to localisation data especially found in non-binary file formats and content (e.g. to represent bold text found in an html paragraph element). Also, another noteworthy feature of XLIFF is the ability to specify proper segmentation of translation units.

3.5.2 Disadvantages of XLIFF

A common observation noted with regards to XLIFF is that many features described in the XLIFF specification are either not supported or only partially supported by localisation tools (Bly 2010; Imhof 2010; Lieske 2011). The research revealed that there is no localisation tool capable of correctly exporting Win32 resources into XLIFF v1.2. Though most localisation tools are capable of importing XLIFF files, tools capable of exporting localisation content into XLIFF are rare.
XLIFF provides many extension points through the XML namespace mechanism. Some localisation tools use this mechanism to insert vendor-specific proprietary data into XLIFF files (Vackier 2010). This has inevitably led to the creation of different flavours of XLIFF (Lieske 2011). The extensive use of proprietary data in XLIFF files has adverse effects on interoperability (Vackier 2010). For example, although a pre-defined attribute and attribute values for storing the status of a translation unit in XLIFF exists, in different XLIFF flavours (introduced by different tool providers) different tool-specific attributes and attribute values are used to denote the translation status of a translation unit. The use of proprietary data and extensions in the above manner diminishes the interoperability of XLIFF content among different tools. In the absence of a mechanism to control or manage custom extensions through the XLIFF specification, tools will continue to freely use such extensions.

According to the XLIFF specification (XLIFF-TC 2008b), “The <x/> element is used to replace any code of the original document.” Since ‘any code’ can be replaced by using this element, it can be used to store tool specific information (i.e. proprietary data). Consider the following XLIFF mark-up segment:

```xml
<trans-unit id="1729" translate="no">
  <source>
    <x id="79"/>
    <x id="80"/>
  </source>
</trans-unit>
```

In the sample segment, although the <trans-unit> element has been used, no translatable content is available. However, a generic placeholder is used as a stand-alone element within the <source> element to store tool-specific data or to reference tool specific data. Although an XLIFF file with content similar to the above mark-up can be perfectly valid, the information is not decodable by tools other than the one used to generate it. This use of inline elements significantly affects the interoperability of XLIFF data.
XLIFF does not provide a powerful mechanism to represent Graphical User Interface (GUI) components attached to localisation items. However, such information is very useful for visual editors. A typical scenario is to visualise the localisation resources of a Win32 executable file in a visual XLIFF editor. According to the current XLIFF specification (version 1.2), information needed to visualise GUI components is stored in a textual format using attributes defined mainly in `<group>` and `<trans-unit>` elements. However, this methodology does not provide an accurate, secure and complete mechanism to represent modern GUI components such as Microsoft .NET dialogs, Silverlight components, Adobe Flash Components and wxWidgets.

Another disadvantage of XLIFF is the inability to group or further categorise notes or comments. Moreover, `<note>` elements cannot be extended by including non-XLIFF attributes or elements. In addition, the usage of the `<note>` element is restricted to some elements (e.g. a `<note>` element common to both `<header>` and `<body>` elements cannot be associated with a `<file>` element; `<note>` elements cannot be associated with individual `<source>` elements or `<target>` elements). Due to these weaknesses of the `<note>` element, it is difficult to map similar elements in other localisation file formats to XLIFF’s `<note>` element in a lossless manner.

The same information can be represented in different ways using XLIFF (Lieske 2011; Bly 2010). This makes the implementation of XLIFF in tools complex and difficult. In particular, the grouping of localisable items, segmentation methodology and inline element usage can be different in different tools.

An XLIFF file might travel through different phases and be used by different tools in a localisation workflow. Although XLIFF has a mechanism to describe phases and tools involved, there is no way to track the status of a certain phase in XLIFF, nor is there a mechanism to track the status of a tool (i.e. whether a tool has already performed an action, or is still waiting to perform an action). The only solution to address this

---

20 It could be argued that it would be difficult, if not impossible, for any open exchange file format such as XLIFF to cater for the wide variety of GUI representations available, which makes it not a limitation of XLIFF, but of the approach taken by the XLIFF community.
problem would be to extend the XLIFF schema to include this information. However, that would certainly affect the interoperability.

In XLIFF, references to external files or internal files can be stored. However, these references cannot be further described using metadata. Moreover, the current XLIFF specification does not allow embedding XML content as an internal file reference.

### 3.5.3 Advantages and Unique Features of LCX

The LCX format relies heavily upon binary information, mainly to preserve the formatting related information of localisation items and to accurately represent GUI components. This binary encoded representation scheme provides an accurate, secure and powerful representation mechanism for resources.

The LCX format provides an Object Model, which defines how to programmatically access and manipulate LCX content. Therefore, the development of LCX based tools is easy. LocStudio and other tools that make use of the LCX OM can directly consume and load appropriate external visual editors for binary resource streams embedded in LCX. LCX content is always generated by the LocStudio suite of software, or by other tools making use of the LCX programmable object model. LCX files cannot be generated manually. Therefore, LCX content, the data representation structure and hierarchy are always consistent in the LCX format.

The LCX format has a unique controlled extensibility mechanism through its Property Bags feature (see Sect. 3.3.2). Property Bags are useful in storing various properties of localisation items as well as metadata. Moreover, the LCX format provides a mechanism to embed target content validation rules along with the localisation content. Therefore, the target content validation can be carried out by translators themselves.

---

21 In contrast to the open XLIFF approach, the Microsoft proprietary LCX is fine-tuned to represent Microsoft-specific GUI components.
In LCX it is possible to associate comments with most elements. The comments can be grouped and they can be either enabled or disabled.

### 3.5.4 Disadvantages of LCX

One of the major drawbacks of LCX is its inability to store alternative translations for localisable elements. Alternative translations are very useful for translators, especially in the absence of resources such as translation memories (TM) and glossaries. Moreover, translators can also get an idea about the proper translation for a localisable item by looking at different translations of the same item in other languages.

There are limitations in the support for non-binary file formats such as HTML and XML. The LCX format is designed for storing localisable content from software resource file formats (e.g. Win32 resources, DLL). Although it can represent localisation content of some non-binary file formats, the support for such files is minimal when compared to XLIFF. Moreover, LCX does not have any inline elements. Therefore, it is not as well suited for the handling of complex tagged content, such as the localisable items of an HTML web-page. In Microsoft, the content localisation of HTML, XML and other content file formats is handled by dedicated content localisation tools, not by LocStudio.

Access to contextual information related to a localisation item would be very useful for translators (Sikes 2011). However, in LCX no specific elements are there to store contextual information. Contextual information related to localisation items is usually stored as comments.

LCX does not provide a mechanism to represent a proper segmentation methodology for text. Furthermore, the line breaks included within localisable text are encoded in binary. This is a reflection of the legacy and primary use of LCX for software localisation, where text segmentation is not a primary concern. The LCX representation of an example sentence that includes two line breaks is given in example 1. In example 1, A; represents a line break in LCX format.
Example 1

In LCX, there is no explicit mechanism to store references to external files. Moreover, although the use of the character data (CDATA) sections is not recommended in localisation file formats (XLIFF-TC 2006), LCX relies heavily upon CDATA sections to store localisation data. Therefore, special care has to be taken when programmatically manipulating LCX content.

3.6 XLIFF and LCX Schema Mapping

In order to further investigate unique features of these file formats and to identify incompatibilities as well as interoperability issues between the two file formats, a schema mapping between XLIFF and LCX format was designed.

We identified some LCX specific data that cannot be successfully mapped to XLIFF without extending the XLIFF schema to represent LCX data. Therefore, we propose to extend XLIFF to represent both XLIFF and LCX syntax. We call the new format LCX-XLIFF for the purpose of our research. The LCX-XLIFF format complies with the XLIFF 1.2 transitional schema. We propose to use the LCX-XLIFF format as an intermediate data container between LCX and XLIFF in the conversion process, as it minimises the data-loss during the roundtrip conversion process (see Fig. 13). The research only focuses on the conversion between LCX and LCX-XLIFF formats.

![Figure 13. LCX-XLIFF as the Intermediate Data Container](image-url)
In our research, we propose two different ways of representing LCX content in XLIFF. The two different approaches are categorised as "maximalist" and "minimalist" representation approaches.

The main idea behind the maximalist approach is to decode as much as possible of the information contained within LCX and represent it in XLIFF with few or no extensions. In this approach, information needed to visualise User Interface (UI) resources in localisation tools will be decoded from LCX and represented in XLIFF. For example, binary information such as the coordinates of UI controls in LCX will be decoded and represented in XLIFF in textual format using appropriate tags. Information on accelerators (keyboard shortcuts) will also be decoded and represented in a textual format in XLIFF. Other binary resources included in LCX such as bitmaps and icons will be represented in XLIFF’s binary translation units. Prior to representing textual information in XLIFF, proper segmentation is carried out on the source LCX textual elements. Then, the information with the proper segmentation guidelines is stored in the XLIFF documents. Furthermore, textual information included in the LCX’s CDATA sections will be carefully analysed and represented in XLIFF as entity references, inline elements and Unicode characters. LocStudio specific metadata (e.g. \texttt{<Disp>}, \texttt{<Modified>} elements) will be retained by extending XLIFF using the namespace mechanism. This approach minimises the data loss while maximising the interoperability of converted XLIFF. The maximalist approach is highly recommended for the LCX to XLIFF conversion process. However, the complexity of the conversion process is high in this approach.

In the minimalist approach, binary encoded data in LCX will not be decoded at all. Hence, only the textual localisation data will be included in XLIFF. However, all binary and other LCX specific information will be retained within XLIFF. This is achieved by extending the XLIFF namespace mechanism. Converted XLIFF files using this approach can then be used with different tools as well as with translation memories. Although it will be possible to translate textual content in the converted file, it will not be possible to visualise UI resources and make changes to the UI by using LCX-XLIFF files converted using the minimalist approach.
Figure 14. Sample UI Resource

As an example, consider the LCX content in Example 2 which corresponds to the resource illustrated in Fig. 14.

```xml
<Item ItemId="259" ItemType="130;WIN_DLG_CTRL_" PsrId="3" Leaf="true">
  <Str Cat="Static Text">
    <Val><![CDATA[&Encoding:]]]></Val>
  </Str>
  <Bin BinId="9">
    <Val><![CDATA[
      [AQAAAAAAAAAAAAAAAAAFBEAAEAKAAoAAAAAAAAAAABggAAAA==]]>
    </Val>
  </Bin>
  <Disp Icon="Dlg"/>
</Item>

Example 2

The expected output of the conversion of the above LCX content into XLIFF using the maximalist approach is shown in Example 3:

```xml
<trans-unit id = "206" resname = "259" restype = "static" style = "0x50000000" coord = "68;1;40;40" lcx:ItemId="259" lcx:ItemType="130;WIN_DLG_CTRL_" lcx:PsrId="3" lcx:Leaf="true">
  <source>&amp;Encoding:</source>
  <lcx:Disp Icon="Dlg"/>
</trans-unit>

Example 3

The expected output of the conversion of the same LCX content into XLIFF using the minimalist approach is shown in Example 4:
Example 4

The scope of this research was limited to the minimalist approach. An element by element mapping is proposed to convert between XLIFF and LCX in the minimalist approach. Details of the proposed schema mapping are provided in the Appendix C. 

3.6.1 Prototype Implementation

A prototype was implemented to demonstrate the conversion between the LCX and LCX-XLIFF file formats generated using the minimalistic approach. XSL Transformations (XSLT) were developed for mapping between the LCX and LCX-XLIFF formats. Using an XSLT processor (such as Saxon) or an XML processing tool, the conversion can be carried out. However, to facilitate the conversion, a separate tool was developed. The tool is capable of converting LCX files (known as LCL files) generated by the LocProject Editor into LCX-XLIFF and LCX-XLIFF files into LCX format. A brief user guide of the converter is provided in Appendix D. A general software localisation workflow that involves the aforementioned converter is depicted in Fig. 15.
The workflow in Fig. 15 involves five distinct steps:

Step 1: the LocProject Editor is used to generate a LCL file corresponding to a resource to be localised.

Step 2: the LCL file is converted to LCX-XLIFF by using the converter.

Step 3: the converted LCX-XLIFF file can be processed at this stage using third party XLIFF enabled tools. The translation process can be carried out using the LCX-XLIFF file.

Step 4: the processed LCX-XLIFF file is then converted back to LCL using the converter.

Step 5: the processed LCL file can then be opened for further processing, inspecting, and validation using the LocProject Editor or LocStudio application.
3.6.2 Comparison with an Alternative Schema

In order to verify our proposed schema mapping, we compared it with an alternative schema provided by Microsoft. The common features of both of the schema mappings are given below.

Both mapping mechanisms:

- take advantage of the XLIFF format’s extensibility to include LCX specific data and metadata.
- preserve the entire LCX structure in a converted XLIFF file by extending the schema.
- stress the advantages of converting of as much LCX data as possible into XLIFF.
- result in an XLIFF file compliant with the XLIFF transitional schema (i.e. a file that is not compliant with the XLIFF strict schema).
- identified that LCX specific elements: Property Bags, Comments, Modified and Disp, do not have their equivalent elements in XLIFF. Therefore these elements are represented by extending the XLIFF schema to include LCX namespace.
- identified that, although LCX Comment <Cmts> and XLIFF <note> elements look similar, they cannot be mapped into each other as XLIFF <note> elements cannot be further extended.
- convert data enclosed within CDATA sections of LCX format into HTML/XML entities when representing them in XLIFF.

The differences between the two mechanisms are given Appendix E.

---

24 This alternative schema was only accessible after we developed our schema mapping.
25 Appendix E could not be made available to the public due to the requirements of confidentiality of the content provided by Microsoft.
3.7 Conclusions and Future Work

The pilot study has not only helped us to identify features of localisation data-exchange standards but also some prominent issues associated with them.

The comparison of the XLIFF and LCX file formats revealed that both file formats have advantages and disadvantages unique to each. The unique features of each file format are found to be closely related and are often complementary to each other. However, it is difficult to achieve 100% compatibility due to factors such as extensive use of binary data in LCX format, inclusion of tool specific data in LCX and inline element usage in XLIFF. A file format converter, similar to the one developed for the purpose of this pilot study provides the interoperability between the two file formats.

Furthermore, it was evident that both XLIFF and LCX formats serve much more functionality than just localisation data-exchange. These file formats have been morphed into and are being used as data repositories and storage formats for localisation knowledge. They are now being specifically designed for data maintenance and reuse. Moreover, both file formats have been designed to accommodate efficient workflows.

The evidence for the above claim includes:

- additional features built onto both file formats (e.g. extensibility, metadata);
- files are not discarded at the end of a localisation project, but kept for leveraging and reuse.

The conclusions show that XLIFF and LCX formats seem to be reasonably close in terms of their coverage. While interoperability is limited between these file formats, the development of the ‘LCX-XLIFF’ intermediate format revealed that XLIFF can successfully accommodate Microsoft’s localisation requirements. This was further confirmed by Microsoft’s recent decision on adopting XLIFF (O’Donnell and
Stahlschmidt 2012) in their localisation process. Since LCX is a powerful localisation data-exchange format similar to XLIFF, it can be concluded that the XLIFF standard can adequately address the problem of source file diversity and standard-based data exchange. This also confirms the selection of XLIFF as the better interoperable standard in our research.

Based on the results and analysis, improvements to the XLIFF and LCX formats are proposed.

### 3.7.1 Proposed improvements to XLIFF and LCX formats

#### Proposed Improvements to XLIFF

- A binary representation of user interface (UI) elements (as in LCX format).
- The ability to associate the `<note>` element with any element and to further customise or categorise the `<note>` element (this issue has been identified already and will be implemented in XLIFF version 2.0 (XLIFF-TC 2011).
- A mechanism similar to LCX’s Property Bags in XLIFF to store metadata related to each localisable item. This feature would be useful in defining a localisation memory container using XLIFF.
- A mechanism to embed target validation rules.
- Microsoft uses a special file container named LSPkg (Localization Studio Package) to store several related LCX files (as well as related other files) within a single file. An LSPkg equivalent data container in XLIFF would be useful to store several file elements in a single file. Information common to several file elements (e.g. various settings) could be stored within this container, e.g. a zipped data container (similar to Microsoft DOCX format). This would allow more data to be stored in an efficient manner. This compact data container would be easier to exchange between different systems. It would further serve the purpose of a localisation memory container.
- The ability to further describe internal and external references would be useful (e.g. to include an attribute to describe the type of reference).
Proposed Improvements to LCX

- Inline elements.
- The ability to store references to external files/data sources (e.g. glossaries, translation memories, bitmaps). There should be a mechanism to describe the references with the use of metadata.
- The ability to store alternative translations.
- The ability to store contextual information of a certain localisation item.
- The ability to specify segmentation.
- XML canonicalisation\(^{26}\) (W3C 2001) would improve the interoperability of the LCX file format.

3.7.2 Interoperability Issues Associated with XLIFF

The extensive literature review and our experiment in designing the schema mapping, carried out as a part of the desk research of our pilot study, revealed some interoperability issues associated with the XLIFF standard:

1) Lack of awareness about XLIFF (Anastasiou 2010b);
2) Limitations of the standard;
   
   For example, vaguely defined criteria (Bly 2010) in the XLIFF specification confuse tool developers. In addition, tools providers face various difficulties in the absence of proper metadata storage and usage conventions. Due to the extremely flexible nature of XLIFF, tool providers have come up with different flavours of XLIFF (Lieske 2011; Imhof 2010; Anastasiou 2010a).
3) Improper or incomplete implementation of the standard.

Our research revealed that many XLIFF features are either not supported or only partially supported by tools (Anastasiou 2010a; Bly 2010; Lieske 2011). This seems to be due to two main reasons:

\(^{26}\) XML canonicalisation is a process used to normalise XML documents by enforcing several formatting rules.
1. Business-case requirements

Depending on the requirements of different business cases, different tools have been implemented to support different parts of the XLIFF specification. There are localisation tools that do not support XLIFF at all, which is mainly due to the fact that there is no strong business case demanding XLIFF support from these tool vendors.

2. Complexity and limitations of the standard

Although XLIFF’s formal tool compliance is easy to achieve, complete XLIFF feature implementation in tools is difficult due to the complexity of the standard (Anastasiou and Morado-Vázquez 2010). Moreover, the segmentation and inline element usage is only vaguely defined in the specification. Therefore, tools can represent the same information in different ways in XLIFF. This greatly affects the interoperability.

Therefore, although many tools claim that they are XLIFF compliant, many XLIFF features are missing in those tools. The actual level of XLIFF compliance in localisation tools is still an open question. The XLIFF-TC (2011) is working on introducing an XLIFF conformance clause to address these issues. Some of the observations related to XLIFF support in tools are summarised as follows:

- cannot export localisation content into XLIFF v1.2 format
  - Win32 content in particular
- inability to open XLIFF with multiple file elements
  - e.g. Alchemy Catalyst 8, SDL Passolo 2009
- usage of different source/target language formats
  - e.g. en-US, US
Imhof (2010) and Bly (2010) give further examples:

- support for alternative translation units
  - e.g. MemoQ, Trados 2009 do not implement the `<alt-trans>` element
- the use of custom defined values to denote the status of the translation
  - e.g. Trados 2009, MemoQ
- support for `translate` attribute and `translate state` attributes
- support for the `<note>` element
  - e.g. MemoQ, Trados 2009 do not support the `<note>` element

The lack of tool support for the XLIFF standard, the complex nature of the standard and the internal flaws of the standard greatly affect its ability to deliver on the interoperability promise.

### 3.7.3 Limitations and Future Work

The prototype converter (see Sect. 3.6.1) developed to demonstrate the conversion between XLIFF and LCX formats is based on XSLT. However, due to the complex hierarchy of `<Item>` elements in LCX and the limitations of XSLT processing (e.g. issues associated with CDATA processing, white space preservation) XSLT style sheets were found to be inefficient and inappropriate for the development of an LCX to XLIFF converter. A programmatic approach that utilises the LCX Object Model is recommended for the implementation of a practical converter.

A formal evaluation of the LCX ↔ XLIFF round-trip conversion process should be undertaken as part of a follow-up study by using a set of experimental data as well as real data of the type used in a typical localisation workflow.

The pilot study confirmed the need for further research into standards compliance, conformance and the interoperability of localisation tools and technologies, particularly
to assess the level of implementation of standards in various localisation tools and to evaluate the actual usage of standards in different localisation workflows. These gaps will be addressed in our main experiment, presented in Chapter 4.

A detailed description of the above pilot study as well as the results can be found in (Wasala et al 2010; Wasala et al 2012b).
Chapter 4: Development of the XML Data Interoperability Usage Evaluation (XML-DIUE) Framework

4.1 Introduction

The pilot study presented in the previous chapter (Chapter 3) helped us to gain an in-depth understanding of the features of both XLIFF and LCX file formats. In addition, it also revealed some limitations of the XLIFF standard and implementations. From the pilot study, it was evident that a thorough systematic analysis of the XLIFF standard and implementations is needed to discover limitations leading to interoperability issues. For this purpose, our initial plan was to adapt the methodology used by Shah and Kesan (2009; 2008), to evaluate the interoperability of tools supporting XLIFF (see Sect. 2.7.2 for a brief description of their methodology). In their research, they only focused on the most frequently used features of the standard as their main interest was on determining whether implementations are good enough for a majority of the users. However, when adapting their methodology into our research, we realised that the determination of frequently used features of XLIFF is not trivial. Therefore, for this purpose, we decided to carry out a survey on XLIFF files, mainly to identify the most frequently used features of XLIFF.

although the survey research strategy is often assumed to be based on questionnaires, it can also use other data generation methods such as interviews, documents and observations

(Oates 2006, p.95)

Our plan was to build an authentic XLIFF corpus for the above survey. Having gathered a large number of XLIFF files (see Sect. 4.3.1 for details on the corpus construction process), we analysed the XLIFF corpus to identify the most frequently used features of XLIFF. This analysis provided interesting insights into utilising the gathered XLIFF corpus to finding answers to our main research question (RQ1). We then revisited the literature with the aim of reviewing previous work on using corpus based approaches
for improving standards. We noticed that a very little work has been reported in this area. In the next section we summarise the related work.

4.1.1 Related Work

Analysis of HTML Documents

In 2005, Google carried out work similar to that proposed here. They analysed over a billion HTML/XHTML documents and presented some important findings regarding frequently used HTML elements, attributes, class names, and the average number of unique elements in a web page (Google 2005). For some of these statistics, possible explanations are also given. However, neither important conclusions nor important recommendations have been made regarding improvements to the HTML based on their analysis. They also briefly summarise previous research carried out on HTML, similar to this research.

Microformats

The microformats.org, co-founded by Ryan King and Dan Cederholm, promotes the use of consistent and “structured data on web” by defining and publishing Microformats—“extensions to HTML for marking up people, organizations, events, locations” (Microformats 2012). These Microformats are based on frequently used HTML code snippets (i.e. HTML usage patterns), HTML user practices or conventions. Microformats attach semantics to HTML metadata, hence promoting automation and interoperability between tools and technologies.

Visualising Interoperability using the Aggregation, Rationalisation and Harmonisation (ARH-HA!) Process

In another study, Currie et al (2002) propose a methodology known as Aggregate-Rationalise-Harmonise (ARH-HA!) to visualise the interoperability of metadata and to “harmonise commonly-owned, distributed, heterogeneous metadata collections.” This research aims to address interoperability issues related to metadata records of
governmental departments and agencies. The three major steps of the ARH-HA! process are briefly described below:

- **Aggregation**: this step involves aggregation and analysis of metadata. The collected metadata elements are added to a spreadsheet. Potential issues related to individual metadata elements such as different spellings, alternative element names that can lead to interoperability issues are identified.

- **Rationalisation**: in this step, gathered metadata will be further analysed in order to identify unnecessary variations among them, such as metadata elements where value is represented using different names (and namespaces), elements containing differently formatted values.

- **Harmonisation**: in this final step, data managers of agencies and departments agree on common formats and vocabularies to minimise metadata interoperability issues.

Currie et al illustrate the utility of their methodology by applying it to address interoperability issues related to Victorian government resources, with the help of six participating government departments in Australia. They highlight that one advantage of using spreadsheets is the visualisation of interoperability. For example, the spottiness of the data graphically highlights interoperability issues. In the rationalisation stage, they removed unused metadata items and merged sensible metadata variations. Currie et al state that harmonisation is the most complex stage out of all three stages, as it involves taking decisions such as standardising non-standard metadata elements and implementing mappings between existing metadata variations. In conclusion, they highlight the usefulness of their methodology to address interoperability issues associated with metadata and also suggest the applicability of visualisation of interoperability in the context of other similar standardisation processes.

**Bottom-Up Analysis of XLIFF Constructs**

Interestingly, and to our surprise, we also noticed that bottom up analysis of XLIFF constructs has been already identified as an approach for addressing issues associated with XLIFF (Frimannsson and Lieske 2010). Unfortunately, this proposal lacks details.
For example, a systematic methodology towards the bottom-up analysis of XLIFF files is not given in this study. Therefore, we aim to address this gap in our research.

### 4.2 The XML-DIUE Framework

An empirical study is really just a test that compares what we believe to what we observe. Nevertheless, such tests, when wisely constructed and executed and when used to support the scientific method, play a fundamental role in modern science. Specifically, they help us understand how and why things work, and allow us to use this understanding to materially alter our world.

*(Perry et al 2000)*

Our initial experiments with the XLIFF corpus as well as the previous work mentioned above encouraged us to investigate the further analysis that can be performed over the corpus towards finding limitations of the XLIFF standard and implementations. We then diverted from our initial plan of implementing Shah and Kesan’s (2008; 2009) methodology, and focused on our own analysis. As a result, this research has inevitably led to our main experiment and the main contribution of this research.

Building on the previous work, we propose a novel empirical framework: XML Data Interoperability Usage Evaluation (XML-DIUE), which can be used as a tool to evaluate data-exchange standard usage and thus inform on the development, maintenance and evolution of standards. The framework will provide empirical evidence and statistics related to the actual usage of different features of the standard (i.e. XML elements, attributes and attribute values). The empirical results generated by the framework seem useful for identifying important criteria for standard development such as the most frequently used features, the least frequently used features, usage patterns and formats. The findings will also be helpful in identifying compliance issues associated with implementations supporting the standard under study. Furthermore, the results will be helpful for the development of an interoperability test suite. Thus, we

---

27 Pronounced ‘Dew’
believe that this framework will ultimately contribute to improved interoperability among tools and technologies in many verticals.

Our framework is based loosely on the ARH-HA! process proposed by Currie et al (2000). Analogy can be also drawn to the Parser-Repository-Analysis architecture proposed in the ‘90s for reengineering toolkits (van-Zuylen 1993, pp.15-371; Cross-Ii et al 1992). The Parser-Repository-Analysis architecture is appropriate based on the similarity of the concerns. Both involve static analysis (parsing) of structured documents, followed by viewing of interesting information derived from analysis of the parsed repository. The Parser-Repository-Analysis architecture (see Fig. 16) isolates the parsing-the-document components from the view-generation components, by persisting the parsed-analysis in the repository. This allows the addition of new viewing tools against the repository without the requirement of building a new parser. Additionally, if the repository can be made agnostic to a new variant of XML to be analysed, the same suite of viewing tools can be employed without alteration. Only a new parser need be developed.

Figure 16. The Parser-Repository-Viewer Architecture Proposed for XML-DIUE
(Adapted from (Cross-Ii et al 1992))
The overall methodology of our framework resembles the general structure of empirical studies as presented by Perry et al (2000): formulation of question or hypothesis; observe situation; abstraction of observations into data; data analysis and drawing conclusions. The methodology is elaborated in the next section.

4.3 Methodology

This section describes the methodology used for the construction and the use of our empirical framework. The methodology involves the following steps:

- the construction of a corpus of the file type to be analysed;
- the parsing and extraction of data from the collected files;
- the construction of a repository;
- data profiling by means of designing usage-analysis criteria;
- data analysis.

In the following sections, we explain the details of each of the above steps in relation to the analysis of the XLIFF corpus. The core step in this methodology is the “designing usage-analysis criteria” and this will be covered in the greatest detail.

4.3.1 Construction of XLIFF Corpus

The first step in our research was building an authentic reference corpus of XLIFF files. For this purpose, XLIFF files were collected using mainly two methods: 1) through industrial partners of the CNGL project; and 2) by crawling openly available XLIFF files on the world wide web. The CNGL industrial partners were made aware of this experiment and invited to contribute their XLIFF files. Three industrial partner organisations contributed to the XLIFF corpus under condition of anonymity. In a second step, openly available XLIFF files on the web were scraped on two occasions: on the 26th and 29th of August 2011 respectively. The Google search engine was used for this purpose, mainly because of its popularity and widespread use. Using Google’s specific file types search facility (i.e. “filetype:” keyword), all files with the extensions...
.xlf, .xliff and .xml were downloaded. Several additional keywords (e.g. +xliff, +body + “trans-unit”) were used to locate more XLIFF content as well as to filter results. During crawling, it was evident that one open-source software project (Eclipse) uses a significant amount of XLIFF files in its localisation process. Therefore, these files were separately crawled from their project sites.

The next step involves pre-processing the files obtained, as described in the following section.

**Cleaning and Pre-Processing**

The manual analysis of random files revealed various issues associated with some of them. These issues included the use of different encoding methods, duplicated content, inconsistencies. Therefore, prior to the analysis stage, the following pre-processing operations were performed on the files:

1) cleaning of the file names (especially some of the crawled files, which contained file names with special characters). These were automatically renamed using a Python script, from names like “dialog.xlf-spec=svn8-r=8”;
2) removal of non-XLIFF files by analysing content (especially from the crawled files). The content was analysed with the aid of a Python script. Filtered non-XLIFF files were manually validated as inappropriate before they were discarded;
3) removal of duplicated files (by analysing content). A proprietary tool\(^\text{28}\) was used for this purpose;
4) removal of Unicode Byte Order Marker (BOM) and conversion of encoding to UTF-8 without BOM (files were found in various encodings such as UTF-16, UTF-8 with BOM, UTF-8 without BOM);
5) removal of XML directive and DOCTYPE declarations at the beginning of a file (some files included either or both of these directives, while others did not);

6) manual extraction of embedded XLIFF content in downloaded HTML web pages;
7) discarding XLIFF content embedded in other XML documents.

As a result, we constructed what we believe to be the first significant XLIFF corpus available for research, containing 3,179 XLIFF files distributed in the following manner:

- Company A : 38 files;
- Company B : 29 files;
- Company C : 1004 files;
- Crawled : 444 files;
- Crawled from Eclipse project : 1,664 files.

The above files were stored in five separate folders based on their source (i.e. Company A, Company B, Company C, Crawled and Eclipse).

It is worth noting that individual file sizes and distribution of XLIFF features vary greatly in this data-set, based on their different sources (e.g. the total file size of 1,004 files received from company C was 16.8 MB while the total file size of the 38 files received from company A was 8.26 MB). In order to give an indication of the variation of XLIFF content in our XLIFF corpus, the file sizes and their averages of individual sources in kilobytes (KB) and the trans-unit count and the averages are given below in Table 2.
<table>
<thead>
<tr>
<th>Source</th>
<th>File Size (KB)</th>
<th>trans-unit</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>total</td>
<td>average</td>
<td>count</td>
</tr>
<tr>
<td>Company A</td>
<td>8462.40</td>
<td>222.69</td>
<td>19310</td>
</tr>
<tr>
<td>Company B</td>
<td>616.37</td>
<td>21.25</td>
<td>1391</td>
</tr>
<tr>
<td>Company C</td>
<td>17288.10</td>
<td>17.22</td>
<td>20437</td>
</tr>
<tr>
<td>Crawled</td>
<td>40903.56</td>
<td>92.13</td>
<td>124090</td>
</tr>
<tr>
<td>Eclipse</td>
<td>126479.15</td>
<td>76.01</td>
<td>187740</td>
</tr>
<tr>
<td>Overall</td>
<td>193749.58</td>
<td>60.95</td>
<td>353003</td>
</tr>
</tbody>
</table>

Table 2. File Sizes and trans-unit Counts of Individual Sources

4.3.2 Data Exaction and Construction of a Database

The next step involves extracting data from the corpus for analysis. The XLIFF corpus could not be analysed using ordinary (text) corpus analysis tools so proprietary analysis tools were developed to populate the repository. In our research we focus more on the quantitative analysis of features and less on their structures. Therefore, we have chosen to store data in a relational database, rather than a structure-oriented repository such as a native XML database. Python scripts were written to populate the database of XLIFF information. These scripts processed one file at a time iterating through all the files in the corpus. The ultimate objective then was to run various Structured Query Language (SQL) queries on the database for analysing the XLIFF content. In this case the database schema consisted of four tables:

- validate: this table consists of information about the individual files’ well-formedness (i.e. XML parsing status): validation results after validating files against available schema, and reference to errors encountered during the parsing or validation process (see Table 3).
Table 3. Structure of the Validate Table

- tags: this table is used to store data about different tags and their values in individual XLIFF files (see Table 4). The table consists of the following fields like, tag, uri and value. The uri field is used to store information about the namespace for a particular tag.

<table>
<thead>
<tr>
<th>Source</th>
<th>File</th>
<th>Tag</th>
<th>URI</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Company A</td>
<td>2.xlf</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4. Structure of the Tag Table

- attributes: this table is used to store information about attributes and their values (see Table 5).

<table>
<thead>
<tr>
<th>Source</th>
<th>File</th>
<th>Tag</th>
<th>URI</th>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crawled</td>
<td>A.xlf</td>
<td>xllf</td>
<td>urn:oasis:names:tc:xlliff:document:1.2</td>
<td>version</td>
<td>1.2</td>
</tr>
</tbody>
</table>

Table 5. Structure of the Attribute Table

- children: this table is used to store information about tags, their children and the trailing content (if any) of tags (see Table 6).
Table 6. Structure of the Children Table

<table>
<thead>
<tr>
<th>Source</th>
<th>File</th>
<th>Tag</th>
<th>URI_tag</th>
<th>Child</th>
<th>Child_URI</th>
<th>Tail</th>
</tr>
</thead>
</table>

4.3.3 Standard Usage-Analysis

In parallel with the development of the database, we planned our analysis stage where we identified several research questions that drive XLIFF usage-analysis. These research questions are not only helpful in identifying limitations of the standard and tools, but also helpful in identifying important features of a standard for successful data interoperability.

While most of the research questions were designed to gather statistics about XML element or attribute usage, others were designed to recognise various usage patterns, erroneous usages, extreme values of elements and attributes. We then identified several standard usage metrics to gather these statistics. The research questions and standard usage metrics are further discussed in the next section (Sect. 4.4).

4.4 Standard Usage Metrics

Each research question has an associated XML usage evaluation metric, which provides valuable insights into the question. These metrics provide a wealth of empirical evidence useful for different stakeholders of the standard. Although there can be a number of stakeholders involved in the standardisation process (Soderstrom 2003), we mainly considered standard developers, standard users, standard software organisations (i.e. tool vendors) and standard service providers in this research. The following table (Table 7) presents the research questions, associated standard usage metrics and their benefits to the stakeholders.
<table>
<thead>
<tr>
<th>#</th>
<th>Research Question</th>
<th>Example Metric</th>
<th>How it might inform the stakeholders</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>How to identify the syntactic conformance issues of the standard?</td>
<td>Identify validation errors</td>
<td>Gives an estimate of the degree of syntactic conformance to the specification, and also helpful in identifying common validation errors</td>
</tr>
<tr>
<td>2</td>
<td>How to identify the core useful features of the standard?</td>
<td>Identify most commonly used features across organisations</td>
<td>Gives an indication of the core useful features of the standard; these are also the most influential features of the standard, where features that have widest ripples for verticals upon changes (i.e. features that would have widest effects in case of a change)</td>
</tr>
<tr>
<td>3</td>
<td>How to simplify the standard?</td>
<td>Identify least frequently used and never used features</td>
<td>Gives an indication of the features that can be removed from the standard</td>
</tr>
<tr>
<td>4</td>
<td>How to identify the candidate features that can be introduced to the standard?</td>
<td>Identify custom features and frequently added extensions</td>
<td>Identification of the most commonly used custom features and extensions might provide guidance on new features that should be adopted</td>
</tr>
<tr>
<td>#</td>
<td>Research Question</td>
<td>Example Metric</td>
<td>How it might inform the stakeholders</td>
</tr>
<tr>
<td>----</td>
<td>------------------------------------------------------------------------------------</td>
<td>-------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>5</td>
<td>How to identify the best usage practices of the standard?</td>
<td>Identify feature usage patterns</td>
<td>Identify frequently used patterns, where subsequent manual evaluation can provide insight into elegant and consistent solutions to recurring problems in the usage of the standard. This can promote best-usage practices of the standard to improve tools interoperability</td>
</tr>
<tr>
<td>6</td>
<td>How to identify where organisations deviate from the norm with respect to usage of the standard?</td>
<td>Identify frequently used features of individual organisations (not employed by others)</td>
<td>Provides a basis for assessment and comparison of organisations’ individualistic standard usage practices</td>
</tr>
<tr>
<td>7</td>
<td>How to resolve semantic ambiguities of features$^{29}$ of the standard?</td>
<td>Identify different values used for the features under consideration</td>
<td>Helps to identify features and feature-values leading to semantic conflicts</td>
</tr>
</tbody>
</table>

Table 7. XML-DIUE Usage Metrics

Details of the above individual example metrics including their advantages, disadvantages as well as limitations are discussed in the following sections (Sect. 4.4.1-7).

---

$^{29}$ This usage metric may only applicable to a selected subset of features of the standard. See Sect. 4.4.7 for more details.
4.4.1 Validation Errors

Identification of frequent validation errors is especially helpful for standard developers as well as for standard service providers. Common validation errors may indicate errors associated with parsers (e.g. validation tools) or errors associated with schemas. This analysis may also reveal erroneous or unintended usage of the specification. For example, this analysis may help to identify unexpected or unforeseen feature usage patterns that are valid according to the specification, but invalid according to the schema. Such scenarios might occur due to loosely defined criteria in the standard specification, yet correctly implemented in the schema. The validation errors may also be of interest to the tool vendors as they can take measures to prevent such common validation errors.

The degree of syntactic conformance to a standard can be calculated by computing the ratio of number of valid files (of a selected version of the standard) to the total number of files (of the same version) in the entire corpus as a percentage. For example, the degree of syntactic conformance to the XLIFF version 1.2 specifications (regardless of whether transitional schema or strict schema) can be calculated using the following formula (F1):

\[
\frac{\text{Number of valid XLIFF version 1.2 files} \times 100}{\text{Total number of XLIFF version 1.2 files}}
\] (F1)

The higher the above ratio, the better the conformance to the standard in the general usage of the standard. If this figure is low, measures have to be taken to improve the standard conformance; for example, by tightening the standard and by provision of conformance test suites, reference implementations, validation tools.

\[30\] Here, we use the term “specification” to refer to the documentation of a standard that describes the semantics, syntax, conformance criteria, processing requirements in writing, whereas by “schemas” we refer to the syntactic implementation of rules and constraints defined in the standard by means of a schema language such as DTD, Relax-NG and Schematron so that files of the standard can be validated against the schema.
However, two main limitations can be identified in this metric (i.e. the identification of validation errors): 1) this metric does not take into account the relative complexity of individual files. For example, consider two files: a file with one thousand lines, fifty unique elements and which contains a single syntactic error. The other file contains just two lines of code, where it only uses a single (unique) element that has a single syntactic error. Under the above analysis, both files will be counted as invalid, while the first one has more content and it is relatively complex compared to the second one.

The other limitation is that although a file might have several validation errors, most of the validation tools are only capable of detecting one error at a time. In other words, to find the next error, the previous error has to be fixed. In a reasonably sized corpus, the manual correction of errors contained within individual files might not be practical. Hence, the results of this metric might not represent the most important syntactic errors that need to be addressed. Therefore, the results of this metric will depend on the nature of the tool used to validate the files.

4.4.2 Most Commonly Used Features Across Organisations

The most commonly used features are the features implemented by a majority of tools and the features used by the biggest variety of users (see Fig. 17).

Even a slight modification of these features has a great effect on a variety of users and tools. Identification of these features is especially useful for the standard developers. These are the core features that are important in terms of cross-organisational data interoperability. They are also important for providing backward compatibility for new versions of the standard. From the point of view of tool vendors, implementation of these features in their tools may help to maximise the coverage of the standard for minimum effort while maximising the interoperability between other tools and technologies. Similarly, for the standard users, maximised use of these features instead of feasible alternatives will ensure better cross-organisational interoperability. Therefore, these features can be considered as the most influential and the core useful features of the standard.
We propose the calculation of commonality ratio of individual features for the purpose of categorising whether a feature is commonly used or not. The commonality ratio of feature $X$ can be calculated using the following formula (F2):

$$\frac{\text{Number of organisations that use } X \times 100}{\text{Total number of organisations, which contributed to the corpus}} \quad (F2)$$
The higher the commonality ratio of a feature, the more useful is the feature for a majority of organisations. While the features with 100% commonality ratio can be considered as the most useful features of the standard, this threshold can be decided by the researcher depending on the number of organisations which contributed to the corpus.

4.4.3 Least Frequently Used Features

The least frequently used features may give an indication of the features of the standard that are not widely used, that users are unaware of or features that are not implemented by the majority of tools. This information is especially useful for the standard developers to make decisions on removing features of the standard which are never used or used only by a minority of users. By removing redundant features, the standard can be simplified. For the purpose of categorising whether a certain feature is a less frequently used feature or not, we propose to calculate relative usage of individual features (i.e. an attribute or an element) using the following formula. Relative usage of an element or attribute $X$ is given by (F3):

$$\frac{\text{Number of occurrences of } X \text{ in the corpus} \times 100}{\text{Average occurrence of elements or attributes in the corpus}} \quad (F3)$$

Note that if $X$ is an attribute, to compute the denominator, the average occurrence of an attribute has to be computed by counting all occurrences of attributes in the entire corpus. The average occurrences of elements/attributes in the corpus are mainly computed in order to normalise the usage distribution of individual element/attribute with respect to the size of the corpus. This normalisation is especially useful in situations where analysis and comparison of features is necessary on a number of corpora or different versions of the standards. Moreover, the computation of percentages in the above formula provides a basis for a longitudinal study of relative usage of features.
However, it is worth noting that there can be highly important features with a low frequency of usage (for example, the `version` attribute of the `<xliff>` is highly important, but it only appears once in a file). This problem of important features being categorised as less frequently used features, can be addressed to a certain degree by categorising individual features into groups and then identifying the least frequently used features within each category. In the following, some example groups are presented, as identified by Song and Bayerl (2003):

- content features (features used to represent content);
- structural features (feature used to organise or maintain relationship or hierarchy between several related features);
- presentation features (features used to format or present content);
- metadata features (features that carry various metadata).

This metric may also be applicable to individual organisations to determine the least frequently used elements within different organisations. However, to obtain an overall list of least frequently used features of the standard (considering all organisations), a highly representative corpus is needed. Failing this, manual inspection of the infrequently used attributes and elements can be used to assess their importance.

### 4.4.4 Custom Features and Frequently Added Extensions

Commonly used custom features (i.e. attributes and attribute values, elements and element-values) that are not defined in the schema can be manually examined to identify potential features that can be standardised in future. Custom feature identification can be automated, for example by programmatically filtering all features and feature values that are not defined in the schema or specification.

Commonly used custom extensions give subtle hints of sources where new features can be adopted for the standard. Various external schemas used and external features implemented (e.g. implemented using the namespace mechanism) in the files can be analysed under this metric. However, this involves further manual investigation of
related schemas and documentations of the identified extensions to explore specific features that might be adopted. Custom extensions may also give hints about other standards used along with the current standard (e.g. use of ITS extensions within the XLIFF). Therefore, standard developers can implement better interoperable features with such standards.

4.4.5 Feature Usage Patterns

Frequently used tag configuration patterns (e.g. XML snippets) are useful in identifying best usage practices of the standard. Manual or automated (Algergawy et al 2011) pattern identification mechanisms can be employed for this purpose. This metric can be used to identify the frequently used features within the individual organisations along with their structural configurations. Moreover, this metric is also useful for the standard developers to identify syntactically and semantically related feature configurations. The use of semantically identical yet syntactically different feature configurations may increase the complexity of the standard and hinder tools interoperability (Lieske 2011). This metric may give evidence of multiple ways of representing the same information using different feature configurations. Therefore, such scenarios can be identified and addressed by the standard developers. For example, some XML elements are being used to group several related elements together (e.g. the <group> element in XLIFF). Analysis of usage patterns related to such elements is helpful in identifying the scenarios mentioned above. Moreover, identification of best usage practices will be helpful to promote such usage (e.g. by identification of Microformats – see Sect. 4.1.1). Tool vendors may find this information useful to optimise their tools, particularly by analysing and implementing commonly occurring patterns. This will also help to improve the tools’ interoperability.

4.4.6 Most Frequently Used Features of Individual Organisations

Identification of the most frequently used features of individual organisations (not employed by others) gives a basis for comparing features used in different organisations. This metric gives a measure of the widespreadness of a certain feature
in consideration. This information may be useful for standard developers to assess the effect on different organisations of either removal or modification of a certain existing feature. On the other hand, individual companies can identify where they deviate from the norm and also to identify commonly accepted alternatives. Such analysis might improve their efficiency and at the same time contribute towards improving potential data interoperability with other organisations.

While the above metric “Least Frequently Used Features” introduced in Sect. 4.4.3 considers the entire corpus for determining the least frequently used features of a standard, a modified version of the same formula can be used to determine the most frequently used features (and the least frequently used features) of individual organisations. Here the equation is being applied to features derived from XML files obtained from individual organisations. The modified equation for determining the relative frequency of usage with an organisation is given below. The relative usage of element or attribute $X$ in the organisation $Y$ is given by (F4):

$$\frac{\text{Number of occurrences of } X \text{ in the files obtained by } Y \times 100}{\text{Average occurrence of elements or attributes in the XML files obtained by } Y} \quad (F4)$$

Features used by individual organisations can be sorted in the descending order of their relative usage frequencies and listed in a table; where columns contain the sorted lists of features of these organisations. This table can be used to manually compare the difference of individual feature usage within different organisations. The above equation can also be used to determine the least frequently used elements and the most frequently used elements within individual organisations.

### 4.4.7 Features and their Values

For each attribute and element, manual analysis of attribute values and elements values is especially useful for the standard developers to identify semantic issues associated with them (Currie et al 2002). This analysis may provide strong evidence of:
• attributes or elements that are used to represent the same type of information (i.e. ambiguous features);
• the use of different values in attributes or elements to represent the same information (i.e. ambiguous feature values).

The above two criteria can be exemplified by adopting from Abdalla (2003). For example, this analysis may reveal an attribute where its values have been represented in different units (e.g. weight = \{10 Kg, 22 lbs\}) that may lead to semantic interoperability issues. Similarly, features where their values have been represented in different expressions (city = \{Dublin, Dub\}) and precisions (e.g. grade = \{A+, very good\}) that can lead to semantic interoperability issues may also be identified.

In addition, this metric may provide some useful statistics about the usage of pre-defined feature values (defined in the specification). However, manual analysis of values may not be practical for attributes or elements where large or infinite numbers of possibilities exist for their values. Therefore, we recommend the application of this metric for specific and pre-determined sets of features.

### 4.5 Analysis of the XLIFF Corpus

In our research, due to time constraints, a selected set of sub-research questions were identified and investigated using our XML-DIUE framework. These selected sub-research questions contribute towards finding answers to our main research question (RQ1) “what are the limitations of existing localisation data-exchange standards and implementations?” The sub-research questions are listed below:

1) What are the core useful features of the standard?
2) What attributes and attribute values can cause interoperability issues?
3) Is tool support for features appropriately aligned with frequency of feature use?
4) How to simplify the standard?
a. What elements can be removed from the XLIFF specification?
b. What attributes can be removed from the XLIFF specification?

5) What are the possible extensions where new features could be adopted?
6) What tools are involved in processing XLIFF files?
7) On average, what is the degree of syntactic conformance to the standard?
8) What are the general patterns of XLIFF usage of each organisation?

It is important to highlight that the first, fifth and last sub-research questions will not provide us answers leading to the identification of limitations of standards and tools. But they will help us to identify important criteria related to standards and tools in terms of achieving our overall objective (see Sect. 1.2), which is improving the semantic and syntactic interoperability of tools and technologies using the XLIFF standard.

For each of the above sub-research questions, different standard usage analysis metrics were identified and computed out over the database using SQL. It is pertinent to mention that analysis of a single research question may involve several standard usage metrics and several SQL queries too. An example SQL query is given below.

The SQL query for identifying the most frequently used elements by ‘Company A’ (e.g. for the metric “most frequently used features of organisations”):

```
select tag, uri, count (tag) as freq from tags where source="Company A" group by tag, uri order by freq desc
```

4.6 Results

Although the current XLIFF corpus might not represent the true distribution of XLIFF content in production environments (see Sect. 4.8.2 for a discussion of the validity of this research), we are able to share some early and important results that provide answers to our sub-research questions presented in the previous section. The results of various analyses performed in this research on our XLIFF corpus are summarised below.
4.6.1 Core Useful Features of the Standard

We utilised the metric described in Sect. 4.4.2 in order to identify core useful features of the XLIFF standard. This is by identifying the most commonly used XLIFF features across different organisations and sources. First, we calculated the commonality ratio of XLIFF elements and identified the core useful elements by considering elements with over 60% commonality ratio (i.e. features at least used by three organisations/sources). Then, attributes of more than 60% commonality ratio were identified for each element. The core features identified by this analysis are given below in Table 8. In Table 8 below, elements <xliff>, <header>, <body>, <file>, <trans-unit>, <source>, <target> have 100% commonality ratio, meaning they have been used in all five sources.

<table>
<thead>
<tr>
<th>Element</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>xliff</td>
<td>version</td>
</tr>
<tr>
<td>file</td>
<td>original, source-language, target-language, product-name, product-version, build-num, tool</td>
</tr>
<tr>
<td>header</td>
<td>-</td>
</tr>
<tr>
<td>body</td>
<td>-</td>
</tr>
<tr>
<td>trans-unit</td>
<td>id, approved, translate, resname, restype</td>
</tr>
<tr>
<td>source</td>
<td>xml:space</td>
</tr>
<tr>
<td>target</td>
<td>xml:lang, state</td>
</tr>
<tr>
<td>alt-trans</td>
<td></td>
</tr>
<tr>
<td>external-file</td>
<td>href</td>
</tr>
<tr>
<td>note</td>
<td>from</td>
</tr>
<tr>
<td>ph</td>
<td>id</td>
</tr>
<tr>
<td>group</td>
<td>resname, restype</td>
</tr>
</tbody>
</table>

Table 8. Core Features of the XLIFF Standard
4.6.2 Attributes and Attribute Values that can cause Interoperability Issues

In XLIFF, attributes play a major role by carrying important metadata in the localisation process. We manually analysed XLIFF attributes and their values for identifying attributes and attribute values that can cause both semantic and syntactic interoperability issues.

This is done by obtaining unique values used for individual attributes from the database. The values were mainly analysed for the following criteria: formats used to represent values, typical values found, default value usage and the predefined value usage. A few selected attributes and their values found in the corpus are given in Table 9. The full list can be consulted in Appendix F. In Table 9, predefined attribute values in the XLIFF v1.2 specification are marked in bold.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Typical Values/Formats found in the XLIFF Corpus</th>
</tr>
</thead>
<tbody>
<tr>
<td>category</td>
<td>String, Enterprise Software, resourceBundle, Survey Project</td>
</tr>
<tr>
<td>match-quality</td>
<td>100, 100%, 78.46, fuzzy, String, Guaranteed</td>
</tr>
</tbody>
</table>

Table 9. A Few Example Attributes and their Values
4.6.3 Tool Support for XLIFF Elements vs. Frequency of Element Use

Using the formula described in Sect. 4.4.3, relative usage frequencies have been computed for each XLIFF element. The plot of XLIFF elements vs. their log (relative usage frequency) is given below in Fig. 18. Fig. 19 shows the distribution of XLIFF features in individual files and the organisations.

**Figure 18.** Plot of log (Relative Usage Frequencies) of XLIFF Elements

**Figure 19.** Distribution of XLIFF Features in Individual Files and Organisations.
4.6.4 Simplification of the standard

Least frequently used XLIFF elements and attributes have been identified by using the metric presented in Sect. 4.4.3. Due to the significant variation of the XLIFF content in the individual files, for each element, the relative usage frequencies was computed with respect to the total trans-unit count and the total file size.
Least Frequently Used Elements

The total number of occurrences of XLIFF elements in the corpus is 2,275,559. The number of XLIFF elements defined in the XLIFF specification is 38. Hence the average use of an element in the corpus is calculated to be 59883.13. Then, using the formula (F3) described in Sect. 4.4.3, the relative usage of individual elements was calculated for each element. In addition, a slightly modified version of the same formula (F3) was used to calculate the relative usage of individual elements with respect to total trans-unit count (i.e. element count / total trans-unit count) and the total file size (i.e. element count / total file size).

The elements with a relative usage less than 1% are considered as the least frequently used elements in this research. The least frequently used elements are listed below in Table 10 with their relative usage frequencies calculated with respect to the average element usage in the corpus, the total trans-unit count and the total file size.

<table>
<thead>
<tr>
<th>Corpus Average (%)</th>
<th>trans-unit Count (%)</th>
<th>File-Size (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>reference(0.00)</td>
<td>reference(0.00)</td>
<td>reference(0)</td>
</tr>
<tr>
<td>bin-target(0.01)</td>
<td>bin-target(0.00)</td>
<td>bin-target(0)</td>
</tr>
<tr>
<td>ex(0.07)</td>
<td>ex(0.01)</td>
<td>ex(0.02)</td>
</tr>
<tr>
<td>bx(0.07)</td>
<td>bx(0.01)</td>
<td>bx(0.02)</td>
</tr>
<tr>
<td>bin-source(0.09)</td>
<td>bin-source(0.02)</td>
<td>bin-source(0.02)</td>
</tr>
<tr>
<td>bin-unit(0.09)</td>
<td>bin-unit(0.02)</td>
<td>bin-unit(0.03)</td>
</tr>
<tr>
<td>internal-file(0.11)</td>
<td>internal-file(0.02)</td>
<td>internal-file(0.03)</td>
</tr>
<tr>
<td>skl(0.16)</td>
<td>skl(0.03)</td>
<td>skl(0.05)</td>
</tr>
<tr>
<td>external-file(0.18)</td>
<td>external-file(0.03)</td>
<td>external-file(0.06)</td>
</tr>
<tr>
<td>prop-group(0.25)</td>
<td>prop-group(0.04)</td>
<td>prop-group(0.08)</td>
</tr>
</tbody>
</table>
### Table 10. Least Frequently used Elements

<table>
<thead>
<tr>
<th>it (0.28)</th>
<th>it (0.05)</th>
<th>it (0.09)</th>
</tr>
</thead>
<tbody>
<tr>
<td>prop (0.47)</td>
<td>prop (0.08)</td>
<td>prop (0.15)</td>
</tr>
<tr>
<td>sub (0.48)</td>
<td>sub (0.08)</td>
<td>sub (0.15)</td>
</tr>
<tr>
<td>seg-source (0.7)</td>
<td>seg-source (0.12)</td>
<td>seg-source (0.22)</td>
</tr>
<tr>
<td>g (0.93)</td>
<td>g (0.16)</td>
<td>g (0.29)</td>
</tr>
<tr>
<td>-</td>
<td>glossary (0.3)</td>
<td>glossary (0.55)</td>
</tr>
<tr>
<td>-</td>
<td>phase-group (0.3)</td>
<td>phase-group (0.55)</td>
</tr>
<tr>
<td>-</td>
<td>phase (0.31)</td>
<td>phase (0.56)</td>
</tr>
<tr>
<td>-</td>
<td>count-group (0.34)</td>
<td>count-group (0.58)</td>
</tr>
<tr>
<td>-</td>
<td>count (0.34)</td>
<td>count (0.62)</td>
</tr>
<tr>
<td>-</td>
<td>mrk (0.44)</td>
<td>mrk (0.8)</td>
</tr>
<tr>
<td>-</td>
<td>bpt (0.75)</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>tool (0.9)</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>xliiff (0.9)</td>
<td>-</td>
</tr>
</tbody>
</table>

**Least Frequently Used Attributes**

The total number of occurrences of XLIFF attributes in the overall corpus is 1,519,314 and the average number of attribute usage is calculated at 18,528.22. Using these figures, the relative usage of individual attributes was computed using the formula (F3) described in Sect. 4.4.3. Similarly, the relative usage frequencies of individual attributes have been computed with respect to the total trans-unit count and the total file size.

31 The relative usage frequencies for each element have been rounded to two decimals and are given within the brackets.
The following table (Table 11) lists the least frequently used attributes (again, taking the threshold as 1%), and their relative usages with respect to average number of attribute usage in the entire corpus, total trans-unit count and the total file size.

<table>
<thead>
<tr>
<th>Corpus Average (%)</th>
<th>trans-unit Count (%)</th>
<th>File-Size (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>alttranstype (0.00)</td>
<td>alttranstype (0.00)</td>
<td>alttranstype (0.00)</td>
</tr>
<tr>
<td>annotates(0.00)</td>
<td>annotates(0.00)</td>
<td>annotates(0.00)</td>
</tr>
<tr>
<td>assoc(0.00)</td>
<td>assoc(0.00)</td>
<td>assoc(0.00)</td>
</tr>
<tr>
<td>clone(0.00)</td>
<td>category(0.00)</td>
<td>clone(0.00)</td>
</tr>
<tr>
<td>comment(0.00)</td>
<td>clone(0.00)</td>
<td>comment(0.00)</td>
</tr>
<tr>
<td>extype(0.00)</td>
<td>comment(0.00)</td>
<td>equiv-trans(0.00)</td>
</tr>
<tr>
<td>merged-trans(0.01)</td>
<td>contact-phone(0.00)</td>
<td>merged-trans(0.00)</td>
</tr>
<tr>
<td>uid(0.01)</td>
<td>equiv-text(0.00)</td>
<td>tool-company(0.00)</td>
</tr>
<tr>
<td>equiv-trans(0.02)</td>
<td>equiv-trans(0.00)</td>
<td>tool-version(0.00)</td>
</tr>
<tr>
<td>tool-company(0.03)</td>
<td>extype(0.00)</td>
<td></td>
</tr>
<tr>
<td>tool-version(0.05)</td>
<td>job-id(0.00)</td>
<td>uid(0.00)</td>
</tr>
<tr>
<td>contact-phone(0.07)</td>
<td>merged-trans(0.00)</td>
<td>category(0.01)</td>
</tr>
<tr>
<td>category(0.08)</td>
<td>tool-company(0.00)</td>
<td>contact-phone(0.01)</td>
</tr>
<tr>
<td>equiv-text(0.08)</td>
<td>tool-version(0.00)</td>
<td>equiv-text(0.01)</td>
</tr>
<tr>
<td>job-id(0.08)</td>
<td>uid(0.00)</td>
<td>job-id(0.01)</td>
</tr>
<tr>
<td>charclass(0.17)</td>
<td>charclass(0.01)</td>
<td>charclass(0.02)</td>
</tr>
<tr>
<td>minwidth(0.17)</td>
<td>form(0.01)</td>
<td>form(0.02)</td>
</tr>
<tr>
<td>maxheight(0.19)</td>
<td>maxheight(0.01)</td>
<td>maxheight(0.02)</td>
</tr>
<tr>
<td>Corpus Average (%)</td>
<td>trans-unit Count (%)</td>
<td>File-Size (%)</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>form(0.21)</td>
<td>minwidth(0.01)</td>
<td>minwidth(0.02)</td>
</tr>
<tr>
<td>mime-type(0.32)</td>
<td>help-id(0.02)</td>
<td>mime-type(0.03)</td>
</tr>
<tr>
<td>help-id(0.37)</td>
<td>menu(0.02)</td>
<td>help-id(0.04)</td>
</tr>
<tr>
<td>menu(0.37)</td>
<td>menu-name(0.02)</td>
<td>menu(0.04)</td>
</tr>
<tr>
<td>menu-name(0.37)</td>
<td>menu-option(0.02)</td>
<td>menu-name(0.04)</td>
</tr>
<tr>
<td>menu-option(0.37)</td>
<td>mime-type(0.02)</td>
<td>menu-option(0.04)</td>
</tr>
<tr>
<td>product-name(0.38)</td>
<td>product-name(0.02)</td>
<td>product-name(0.04)</td>
</tr>
<tr>
<td>state-qualifier(0.43)</td>
<td>state-qualifier(0.02)</td>
<td>state-qualifier(0.04)</td>
</tr>
<tr>
<td>product-version(0.52)</td>
<td>product-version(0.03)</td>
<td>product-version(0.05)</td>
</tr>
<tr>
<td>css-style(0.80)</td>
<td>css-style(0.04)</td>
<td>css-style(0.08)</td>
</tr>
<tr>
<td>unit(0.80)</td>
<td>maxbytes(0.04)</td>
<td>exstyle(0.08)</td>
</tr>
<tr>
<td>maxbytes(0.82)</td>
<td>minbytes(0.04)</td>
<td>maxbytes(0.08)</td>
</tr>
<tr>
<td>minbytes(0.82)</td>
<td>minheight(0.04)</td>
<td>minbytes(0.08)</td>
</tr>
<tr>
<td>minheight(0.82)</td>
<td>unit(0.04)</td>
<td>minheight(0.08)</td>
</tr>
<tr>
<td>exstyle(0.86)</td>
<td>exstyle(0.05)</td>
<td>unit(0.08)</td>
</tr>
<tr>
<td>reformat(0.90)</td>
<td>reformat(0.05)</td>
<td>reformat(0.09)</td>
</tr>
<tr>
<td>-</td>
<td>priority(0.06)</td>
<td>priority(0.1)</td>
</tr>
<tr>
<td>-</td>
<td>pos(0.07)</td>
<td>pos(0.13)</td>
</tr>
<tr>
<td>-</td>
<td>prop-type(0.08)</td>
<td>rid(0.14)</td>
</tr>
<tr>
<td>-</td>
<td>rid(0.08)</td>
<td>prop-type(0.15)</td>
</tr>
<tr>
<td>-</td>
<td>size-unit(0.09)</td>
<td>size-unit(0.16)</td>
</tr>
<tr>
<td>-</td>
<td>xid(0.09)</td>
<td>xid(0.16)</td>
</tr>
<tr>
<td>-</td>
<td>maxwidth(0.11)</td>
<td>maxwidth(0.2)</td>
</tr>
<tr>
<td>-</td>
<td>crc(0.15)</td>
<td>crc(0.27)</td>
</tr>
<tr>
<td>-</td>
<td>font(0.24)</td>
<td>font(0.43)</td>
</tr>
<tr>
<td>-</td>
<td>mid(0.24)</td>
<td>mid(0.44)</td>
</tr>
<tr>
<td>-</td>
<td>mtype(0.24)</td>
<td>mtype(0.45)</td>
</tr>
<tr>
<td>-</td>
<td>build-num(0.3)</td>
<td>build-num(0.55)</td>
</tr>
<tr>
<td>Corpus Average (%)</td>
<td>trans-unit Count (%)</td>
<td>File-Size (%)</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------------</td>
<td>---------------</td>
</tr>
<tr>
<td></td>
<td>company-name(0.3)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>contact-email(0.3)</td>
<td>contact-name(0.55)</td>
</tr>
<tr>
<td></td>
<td>contact-name(0.31)</td>
<td>contact-name(0.56)</td>
</tr>
<tr>
<td></td>
<td>process-name(0.31)</td>
<td>process-name(0.56)</td>
</tr>
<tr>
<td></td>
<td>href(0.33)</td>
<td>href(0.6)</td>
</tr>
<tr>
<td></td>
<td>tool(0.33)</td>
<td>tool(0.61)</td>
</tr>
<tr>
<td></td>
<td>count-type(0.34)</td>
<td>count-type(0.62)</td>
</tr>
<tr>
<td></td>
<td>style(0.41)</td>
<td>style(0.74)</td>
</tr>
<tr>
<td></td>
<td>ctype(0.44)</td>
<td>ctype(0.81)</td>
</tr>
<tr>
<td></td>
<td>phase-name(0.53)</td>
<td>phase-name(0.97)</td>
</tr>
<tr>
<td></td>
<td>date(0.59)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>tool-name(0.9)</td>
<td></td>
</tr>
</tbody>
</table>

Table 11. The Least Frequently Used Attributes

It is worth noting that the attributes alttranstype, annotates, assoc, clone and comment have a relative usage of 0.00, meaning those attributes have not been used in the entire XLIFF corpus at all.

4.6.5 Extensions Used

We analysed various extensions as well as namespaces used in the XLIFF files. Different namespaces found in the corpus are listed below:

- http://www.gs4tr.org/schema/xliff-ext
- http://www.idiominc.com/ws/asset
- http://www.w3.org/1999/xhtml
4.6.6 Tools Involved

We analysed various tools associated with the XLIFF files. Mainly the tool attribute of the <file> element and the tool-name attribute of the <tool> element were manually analysed for this purpose. The list of tools is given below:

- Snap-On Ireland, CATFile_Translation UtilityNAIL.LUI 1.6.0.21409
- Idiom WorldServer 9.0.5AgdaVS export tool
- Benten
- Ektron
- IGD-2-XLIFF
- ITS Translate Decorator
- Maxprograms JavaPM
- Okapi.Utilities.Set01
- Pleiades
- Swordfish III
- blancoNLpackGenerator
- genrb
- Idiom WorldServer 9.2.0
- LKR
4.6.7 Validation Results

The XLIFF files have been validated\textsuperscript{32} against the schemas of the current version of the XLIFF (i.e. XLIFF version 1.2) and the previous XLIFF versions (1.1 and 1.0).

![Validation Results](image)

\textbf{Figure 20. Validation Results}\textsuperscript{33}

\textsuperscript{32} After identifying a bug in the Python script we wrote for validating the XLIFF files, the files were manually re-validated using a freely available tool, the XLIFFChecker (version 1.0-2). The XLIFFChecker is a tool specifically designed for validating XLIFF files. The tool is available to download from: [http://www.maxprograms.com/products/xliffchecker.html](http://www.maxprograms.com/products/xliffchecker.html) [accessed 22 Mar 2013].

\textsuperscript{33} The XLIFF files were validated against the version of the XLIFF they stated themselves.
The results show that 1.2 is the predominantly used XLIFF version. Out of the 2,758 XLIFF version 1.2 files, 2,362 files are found to be invalid. Out of the valid XLIFF 1.2 files, 22 files were transitional schema valid and 374 files were strict schema valid. Out of the 232 XLIFF version 1.1 files, 146 files were valid, while out of the 170 XLIFF 1.0 files, only 42 files were valid. While 18 files were found without the required XLIFF version information, 1 XLIFF version 2.0 file was also found. The results are summarised in the above graph (see Fig. 20).

Having realised that 1.2 is the XLIFF version used by a majority, the XLIFF 1.2 files were further analysed for their complexity in terms of the trans-unit count and their validation results. The following graph (see Fig. 21) shows the number of invalid, valid and strict valid files vs. the log (trans-unit count).

![Graph of Validation Result vs log (trans-unit Count)](image)

**Figure 21. The Graph of Validation Result vs log (trans-unit Count)**

### 4.6.8 Feature Usage Patterns

As highlighted in the Sect. 4.4.5, the identification of unique XLIFF patterns and their frequency of use by different organisations are extremely useful in many different ways.
However, XML tree mining is a separate, active and a complex research area. Therefore, the identification of unique XLIFF patterns and their frequency of use could not be computed in this research due to time and resource constraints. Moreover, the use of the relational database in this research limits the ability to retrieve the structural information such as the exact order of tags. Therefore, an alternative approach was used to manually derive general usage patterns of elements for individual organisations. This is by manually listing elements and their child elements used by each organisation. Data in the children table was used for this purpose. Hence, a derived pattern for an organisation represents a generic pattern unique for that organisation. A derived pattern can be considered as a merging of unique patterns used in that organisation.

The pattern derived for Eclipse is given below in Fig. 22. Patterns derived for other organisations are given in Appendix G.

![Diagram: Pattern Derived for Eclipse]

**Figure 22. Pattern Derived for Eclipse**
4.7 Analysis of the Results

Although the current XLIFF corpus is not representative of all XLIFF files used in production environments, conclusions drawn from our analysis are still significant, especially to illustrate the utility of our XML-DIUE framework.

4.7.1 Core Useful Features of the Standard

The analysis of commonly used elements across different sources (see Sect. 4.6.1) leads to the identification of important features in terms of cross-organisational data-exchange interoperability. This result could also support work towards a “modularisation” of the standard, mainly by defining a core module that consists of the core useful features identified by this analysis.

The XLIFF specification (XLIFF-TC 2008b) as well as Frimannsson and Hogan (2005) illustrates a minimal XLIFF document. The elements and attributes presented in the minimal XLIFF document are given in Table 12.

<table>
<thead>
<tr>
<th>Element</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>xliff</td>
<td>version</td>
</tr>
<tr>
<td>file</td>
<td>source-language, datatype, original</td>
</tr>
<tr>
<td>body</td>
<td></td>
</tr>
<tr>
<td>trans-unit</td>
<td>id</td>
</tr>
<tr>
<td>source</td>
<td></td>
</tr>
</tbody>
</table>

Table 12. Elements and Attributes Found in the Minimal XLIFF Document
(Adopted from (XLIFF-TC 2008b))
While most of the core useful features identified using our framework (see Table 8 of Sect. 4.6.1) are presented in the minimal XLIFF document, our research has revealed a few more additional attributes and elements that are in common use. The additional elements we discovered are `<header>`, `<target>`, `<alt-trans>`, `<external-file>`, `<note>`, `<ph>` and `<group>.

Moreover, the core features systematically derived by our research are also consistent with the features proposed by the TAUS as the XLIFF core (TAUS 2011b; Choudhury 2011). They propose the following list of elements to be considered as the XLIFF core: `<xliff>`, `<file>`, `<header>`, `<body>`, `<trans-unit>`, `<source>`, `<target>`, `<alt-trans>`, `<source>`, `<target>`. Our analysis shows that in addition to the above elements, the elements `<external-file>`, `<note>`, `<ph>` and `<group>` are also in common use. It is worth noting that our results are also consistent with the features of the XLIFF Core in XLIFF 2.0.34

4.7.2 Attributes and Attribute Values that can Cause Interoperability Issues

The manual analysis of the attributes and attribute values revealed no use of certain pre-defined attribute values (e.g. the pre-defined value `lisp` for `datatype` attribute is never used in our XLIFF corpus). The pre-defined attribute values that nobody is using increases the complexity of the standard. Therefore standard developers may deprecate those pre-defined attribute values to reduce the complexity of the standard.

Moreover, the analysis revealed semantic interoperability issues associated with some attributes. For example:

- use of incorrect syntax for specifying user defined attributed values (i.e. not using the `x-` prefix) e.g. The use of values such as `text` for the `datatype` attribute;

34 See the working specification of XLIFF 2.0: [https://tools.oasis-open.org/ version-control/browse/wsvn/xliff/trunk/xliff-20/xliff-core.pdf](https://tools.oasis-open.org/version-control/browse/wsvn/xliff/trunk/xliff-20/xliff-core.pdf) [accessed 13 Mar 2013]
• use of custom values regardless of the existence of predefined values (e.g. the use of the `pofile` value instead of the predefined `po` value for the `datatype` attribute).

• use of different expressions to denote the same value (e.g. use of `xml` and `x-text/xml` values for `datatype` attribute), the use of different units to represent values (e.g. use of values such as 100, 100%, 78.46, fuzzy for `match-quality` attribute);

• the use of same values in different attributes (i.e. ambiguous attributes). For example, the use of user defined value `x-tab` in both `context-type` attribute and `ctype` attribute, the use of predefined value `database` in both `datatype` and `context-type` attributes and the use of user defined value `x-button` in `context-type` attribute and `button` in `restype` attribute.

Issues like the ones outlined above can be addressed by:

• standardising most frequently used custom values;

• agreeing on the semantics of values as well as attributes;

• tightening the standard by strict conformance clauses to cover those semantics.

In addition, publishing of conformance test suites may help tool developers to identify and address such issues.

We also noticed the use of extreme values in some features such as the use of extremely lengthy strings for IDs, which may adversely affect the date exchange interoperability. It is worth mentioning, that the lengthiest ID (i.e. value for the ID attribute) found in our corpus is 8,583 characters. Although the specification recommends avoiding spaces within IDs, IDs have been found not only using spaces but also using several line breaks. Although such extreme values may affect interoperability, they are allowed by the specification. Therefore, standard developers have to take measures to tighten the conformance clauses to avoid these kinds of non-conformant behaviours.
Our analysis also revealed attributes that may cause syntactic interoperability issues, mainly the use of incorrect formats to denote values in some attributes. For example, representing language in formats other than formats specified in BCP 47\(^{35}\) (e.g. values such as en_US, ENGLISH, x-dev, unknown, uz-UZ-Cyrl) or representing dates not as specified in ISO 8601\(^{36}\) (e.g. values such as 04/02/2009 23:24:18, 11/06/2008). These kinds of issues may be addressed by developing and publishing conformance test suites.

4.7.3 Tool Support for XLIFF elements vs. Frequency of Element Use

Bly’s (2010) top-down analysis of XLIFF implementations show their level of support for different XLIFF elements. Morado-Vázquez and Wolff (2011) visualise the above information in terms of bar graphs (See Fig. 23 and Fig. 24).

---


\(^{37}\) Number of tools supporting the element vs. element
We present the overall frequency distribution of XLIFF elements in our corpus in Fig. 18 and Fig. 19 (Sect. 4.6.3). An analysis of the graphs reveals a connection between the lack of tools’ support for certain XLIFF elements and the frequency of use of XLIFF elements.

The least frequently used elements identified by our research are presented in Sect. 4.6.4. Out of those elements, except for the <internal-file> element and the <skl> element, the other elements have been only implemented by either one or two tools. This is further confirmed by the survey results presented by Morado-Vázquez and Filip, (2012) showing lack of tool implementation and support of least frequently used elements reported in Sect. 4.6.4.

4.7.4 Simplification of the Standard

In Sect. 4.6.4, the least frequently used features were computed using three different methods: by normalising the individual feature counts with respect to 1) average feature usage in the entire corpus; 2) the total trans-unit count; and 3) the total file-size. The normalisation of feature counts with respect to the average feature usage in the corpus assumes that all the features are equally distributed in the corpus. Although this is not necessarily true due to the organisational structure of features, it provides a generalisable mechanism independent of standards to identify the least frequently used features. In contrast, normalisation of the feature counts with respect to the total trans-unit count is highly specific to XLIFF, and more suitable in the context of our research. Moreover, the file-size is proportional to the trans-unit count. Therefore, the least frequently used features identified by normalising the feature counts with respect to the total file size are a sub set of the least frequently used features identified by normalising feature count with respect to the total trans-unit count. However, the “importance” of a feature depends on both the commonality ratio and the relative usage frequency of that feature. Therefore, to identify the candidate features for deprecation (or removal), both the frequency of usage and the commonality ratio of the feature has to be taken into account. Features with very low relative frequency of usage and extremely low commonality ratio are the ideal candidates for deprecate from the
standard. In our research we only considered the least features identified by normalising the feature count with respect to the total trans-unit count.

The commonality ratios of the least frequently used features identified in Sect. 4.6.4 were analysed. In this research, we propose features with less than a 60% commonality ratio (i.e. used by less than three organisations) out of the least frequently used features are the ideal candidates to deprecate. The analysis of commonality ratios of the least frequently used elements revealed that the <xliff> has a commonality ratio of 100% while the <external-file> element has a 60% commonality ratio, the rest of the elements have less than 60% commonality ratio. As such, the ideal candidates to deprecate from the standard are given below:

```
reference, bin-target, ex, bx, bin-source, bin-unit, internal-file, skl, prop-group, it, prop, sub, seg-source, g, glossary, phase-group, phase, count-group, count, mrk, bpt, tool
```

The attributes alttranstype, annotates, assoc, clone, comment, extype are never used in the corpus. Therefore, these attributes are the ideal candidates for deprecation. The analysis of the commonality ratios of the other least frequently used attributes revealed that product-name, product-version, build-num, date, href and tool are the only attributes with 60% commonality ratio. The rest of the attributes have less than a 60% commonality ratio, therefore they can be deprecated from the standard to reduce the implementation complexity of the standard. The list of candidate attributes for deprecation is given below:

```
alttranstype, annotates, assoc, category, charclass, clone, comment, company-name, contact-email, contact-name, contact-phone, count-type, crc, css-style, ctype, equiv-text, equiv-trans, exstyle, extype, font, form, help-id,
```
job-id, maxbytes, maxheight, maxwidth, menu, menu-name, menu-option, merged-trans, mid, mime-type, minbytes, minheight, minwidth, mtype, phase-name, pos, priority, process-name, prop-type, reformat, rid, size-unit, state-qualifier, style, tool-company, tool-name, tool-version, uid, unit, xid

While our framework helps to identify possible features that can be deprecated to reduce the complexity, the ultimate decision on a deprecation of a certain feature has to be taken by the standard developers (e.g. XLIFF TC), with consensus of all members. Moreover, the standard developers may also decide to isolate such features from the core useful features and construct separate modules instead of deprecating such features.

Further analysis of our results shows that <prop-group> and <prop> elements have been already deprecated in XLIFF version 1.1. Moreover, lack of use of <bin-unit>, <bin-source> and <bin-target> elements have been already identified by the XLIFF TC, and the creation of an optional Binary Data Module has been proposed consisting of those elements in the next version of the standard (i.e. XLIFF 2.0). The lack of use of the <it> inline element as well as issues associated with other inline elements such as <ex>, <bx>, <it> and <g> are being discussed within XLIFF TC meetings, and will be addressed by the inline element sub-committee of XLIFF in XLIFF 2.0. This analysis shows the suitability of our systematic methodology to identify features for deprecation or modularisation.

---

38 See XLIFF TC discussion: [link](http://markmail.org/message/imo755yezrjasmja#query:+page:1+mid:kixvjlbiga3xt2b+state:results) [accessed 17 Jan 2013]

39 For example, see XLIFF TC discussion: [link](http://markmail.org/search/?q=ex+bx+inline+xliiff%query:ex%20bx%20%20inline%20xliiff+page:1+mid_jnw2ljdukwhelbk+state:results) [accessed 17 Jan 2013]
4.7.5 Extensions Used
The identification of extensions used in production environments will lead to a better understanding of these extensions which could then be introduced into the standard itself where appropriate. It will also help to identify other XML based file formats or standards regularly used with XLIFF and could lead to modifications in these standards that could improve their interoperability. In 2012, the XLIFF TC initiated collecting and publishing schemas of the XLIFF extensions. Our research revealed several extensions in common use, in addition to the ones published by the XLIFF TC.

4.7.6 Tools Involved
The identification of tools associated with files along with the identification of invalid, erroneous or problematic XLIFF content will support the standard conformance evaluation of commonly used tools and technologies. It will also help to identify the most widely used tools in the market, either commercial or open-source. The identification of open-source tools is of particular interest as they represent the most accessible sample implementation of the standards. Any problems associated with such tools could then be addressed.

An analysis of the tools landscape demonstrated a lack of open-source tools that can process XLIFF files. The only open-source tool that was used to process files in the current corpus is Okapi (including Rainbow).

4.7.7 Validation Results
The results show that the majority of the XLIFF content is represented in XLIFF version 1.2 although XLIFF 1.1 and XLIFF 1.0 are also still in use. These results suggest a strong requirement of backward compatibility for future versions of XLIFF. Particularly, XLIFF 2.0 should be fully backward compatible with XLIFF 1.2.

---

40 See XLIFF 1.2 Extensions sub-topic of https://wiki.oasis-open.org/xliff [accessed 17 Jan 2013]
According to the formula described in Sect. 4.4.1, the degree of syntactic conformance to the XLIFF 1.2 schema is computed at an alarming 14.36 per cent. This means, the majority of the generated XLIFF 1.2 compatible files are invalid. The lack of conformance to the standard is clearly evident from these results. Undoubtedly, the lack of conformance to the standard can also be attributed to the lack of interoperability.

The plot of validation results of XLIFF 1.2 files vs. trans-unit count (see Fig. 21) shows that there is no clear evidence of a strong correlation between the number of trans-units in a file and the validation results. The majority (i.e. 385) of the invalid files have only two trans-unit elements; whereas the majority (77) of the strict-valid files also have only two trans-unit elements.

Moreover, surprisingly 18 files were found without even the required XLIFF 'version' attribute. Without this attribute, tools might get confused about how to even start processing the content properly.

These results show the strong necessity of thorough conformance testing of tools and technologies supporting XLIFF and taking necessary measures to tightening the standard to ensure stronger tool compliance.

### 4.7.8 Feature Usage Patterns

The derived patterns represent XLIFF footprints unique to each organisation. Each footprint consists of a specific set of elements used in a pattern unique to each. The analysis of XLIFF usage patterns also revealed interesting behaviours of using the `<group>` element. The `<group>` element is mainly intended to be used to maintain the structural hierarchy of several related elements or to represent several elements that have to be processed together. The `<group>` element can be used recursively. The analysis of patterns revealed that it has been used by only two organisations and it has been used to group translation units. Moreover, the `<group>` element has not been used more than two levels of recursion (i.e. the maximum recursion depth of the
<group> element is two). Therefore, it is vital for tools to provide support for the <group> element at least up to two levels of recursion. Also, it is interesting to note the frequent use of <alt-trans> element (which is used to provide an alternative translation for a certain translation) and the use of <note> element (which is used to provide various comments), within the <trans-unit> elements.

4.8 Discussion

In the following, Sect. 4.8.1 discusses how our XML-DIUE framework fills gaps found in literature, whereas Sect. 4.8.2 discusses various limitations of our framework in detail.

4.8.1 Addressing Existing Issues with XML-DIUE

Our framework not only helps to discover deficits in standards and their implementations, but is also helpful in identifying important features of standards that are useful for successful interoperability.

Unlike conceptual frameworks, our framework provides useful information for standard development backed by empirical evidence. Mykkänen and Tuomainen (2008) highlight the need for a systematic approach to identify and document the important aspects of a standard in relation to interoperability. While their framework is suitable for a theoretical self-assessment of a standard, our framework is more pragmatic and provides specific details from the standard usage perspective, for example features that are leading to semantic interoperability issues, identification of extensions and extension points, identification of user defined features. Thus it addresses Mykkänen and Tuomainen’s considerations 3, 4 and 7 (see Sect. 2.6.3). Lewis et al (2008) point out the analysis of gaps in the standard as an important step towards achieving interoperability using standards. They also highlight the importance of backward compatibility of new versions of the standards and the characteristics of bad standards such as over-specification and under-specification (see Sect. 2.6.3). Our framework helps to identify gaps in the standards (especially the limitations of the standards, such as semantic and syntactic interoperability issues associated with features). The XML-
DIUE framework is also helpful in identifying features of standards that are important in terms of backward compatibility (see Sect. 4.4.2). Moreover, this work augments the current literature by assessing how users employ these standards, giving direction on how it may be under-specified or over-specified.

In terms of Shah and Kesan’s work, our framework provides a systematic approach for the identification of most frequently used features. Therefore, our framework can be used as a component of Shah and Kesan’s (2008; 2009) approach. While Shah and Kesan’s methodology is helpful in identifying limitations of implementations with respect to standard support as well as cross-implementation interoperability, our framework not only helps to identify the most frequently used features, but much other important information regarding the usage of a standard, such as the least frequently used features and features that are important for cross organisational interoperability. The use of our framework in combination with Shah and Kesan’s methodology also supports the development of interoperability test suites.

Google presents various statistics by examining over a billion HTML pages; however, they did not propose any recommendation to improve HTML (see Sect. 4.1.1) based on their findings. Our framework goes two steps beyond Google’s work; by making explicit the architecture used to analyse a standard and designing several standard usage analysis metrics that describe the connection between various statistics and their significance for the standard development process. Moreover, our framework can be used to discover Microformats (i.e. frequently used feature usage pattern snippets) in XLIFF as well as other data-exchange formats.

Lack of user feedback in the standardisation process has been identified as a significant problem (Soderstrom 2003). Our framework provides a partial solution to this problem by providing a framework that utilises implicit user feedback for the standard, through a different perspective: by analysing the user generated files of the standard. Therefore, our framework provides an alternative mechanism that represents user’s feedback, and so it is suitable for standards where user interaction in the standard development is minimal in the standard development.
Although XML-DIUE resembles the ARH-HA! process proposed by Currie et al. (2000), our framework is superior to it for several reasons. First, our framework can not only be applied to address interoperability issues related to metadata elements, but it is generally applied to other features of the standards as well. Second, the use of Parser-Repository-Viewer architecture (see Sect. 4.2) yields better scalable data representation mechanism than the use of spreadsheets. The will lead to additional findings as illustrated in our XLIFF corpus analysis.

A survey (Anastasiou 2010b) revealed that although the current XLIFF document structure is adequate, “simplification of inline mark-up, clarification of metadata, stronger compliance requirements, better workflow control” and multilingual content support is important for future versions. Our framework provides empirical evidence on how inline mark-up elements can be simplified by identifying less frequently used inline mark-up elements that can be deprecated. For example, our analysis results (see Sect. 4.6.4 indicate that inline elements such as `<ex>`, `<bx>`, `<it>` and `<g>` are less frequently used, while the `<ph>` inline element is more frequently and commonly used.

Moreover, the empirical data presented by our framework helps to identify features that may lead to semantic conflicts (see examples provided in Sect. 4.7.2). Therefore, it supports the disambiguation of metadata. Lieske (2011), Frimannsson and Lieske (2010), Anastasiou and Morado-Vázquez (2010), as well as Mykkänen and Tuomainen (2008) observed the importance of the modularisation of standards. However, deciding how to modularise an existing standard is a challenging problem. This involves the identification of the core elements of standards and elements that can be treated as individual modules. The Translation User Automation Society (TAUS) is currently in the process of identifying elements that constitute the XLIFF core (Choudhury 2011). Our framework can be used in this process to identify core features of the standard in a systematic manner (see Sect. 4.7.1).
4.8.2 Limitations of the Current Framework

There are several limitations of the current framework. The biggest limitation of our research is the low external validity. This and the other limitations are further discussed in the next two sub-sections.

Validity of the Research

*Validity is a characteristic of an empirical study and is the basis of establishing credible conclusions.*

(Perry et al 2000)

According to Perry et al (2000) external validity means “that the study’s results generalise to settings outside the study.” Therefore, external validity refers to the sample being representative of the community and conditions of study in a real life context. Generalisation refers to the scale of the data: whether it is large enough for statistical techniques to imply that this is representative of a population.

The external validity of our research is low, mainly due to the lack of representativeness of our XLIFF corpus. In other words, the corpus must be “representative” in order to deduce generalisations concerning the XLIFF standard.

Representativeness refers to the extent to which a sample includes the full range of variability in a population.

(Biber 1993)

However, the issue of lack of representativeness in corpora indeed is not peculiar to our corpus and our research, but this is a widely discussed topic in other corpus based studies in the fields of Computational Linguistics and Natural Language Processing (NLP) (Biber 1993; Tognini-Bonelli 2001, pp.47-62; McEnery and Hardie 2011, pp.6-9).
According to Biber (1993), the a) sample size (i.e. the number of texts to be included in the corpus and the number of words per each sample) and b) the range of linguistic distributions in a language are important factors that can be used to determine the representativeness of linguistic corpora. Tognini-Bonelli (2001, pp.54-58) highlights the importance of a) the authenticity of the data and b) the sampling criteria used in the selection, in addition to the representativeness, when compiling linguistic corpora. In terms of our research, the criteria identified by Biber can be identified as a) the number of XLIFF files in the corpus and b) the range of XLIFF feature distribution. We collected over 3000 ‘real life’ XLIFF files, mainly using two methods (with the help of CNGL industrial partners and crawling openly available files in the internet) enhancing its authenticity.

As reported by Biber and Tognini-Bonelli (2001, p.57), defining the amount of data to be collected to build a representative corpus is an extremely difficult if not almost impossible task, which in turns make it difficult to compile a truly representative corpus. Tognini-Bonelli (2001, p.57) points out that “at present we have no means of ensuring [representativeness], or even evaluating it objectively.” McEnery and Hardie (2011, p.10), also mention that balance, representativeness and comparability are important to attain in a corpus however, it is almost impossible. It became evident that this is indeed true when we started collecting XLIFF files for compiling our XLIFF corpus. It should be noted that building a representative XLIFF corpus can be an extremely difficult task, especially due to the fact that XLIFF files can carry confidential information, therefore some companies are not prepared or willing to share their files.

In relation to an XLIFF corpus, we acknowledge that factors such as sampling, representativeness, processing stage and the proportions of the files (including the number of files obtained from different organisations, file sizes, and feature distribution of individual files) of the XLIFF corpus are really important for drawing accurate conclusions.
Biber proposes an approach for constructing representative corpora. Biber suggests that corpus compilation should “proceed in a cyclical fashion” starting from a “pilot corpus” representing “a broad range of variation but also representing depth” (Biber 1993). Then he proposes to continuously conduct empirical research on the pilot corpus to “confirm or modify the various design parameters” of the corpus, as new sample data are being added to the corpus. Accordingly, our corpus can be considered as a pilot corpus.

In conclusion, our XLIFF corpus can be considered as the first step towards constructing a better representative corpus. More importantly, as highlighted in our conclusions (see Chapter 6), our corpus was sufficient to illustrate the core utility of our XML-DIUE framework.

**Other Limitations**

In our research, only a few usage metrics have been selected and analysed. More usage analysis metrics can be introduced (e.g. the analysis of the usage of mandatory attributes). In addition, the inclusion of more fields in the database such as file size, crawl date will provide valuable information for analysis.

Another limitation associated with usage metrics is that they may not provide precise results or the information required for addressing a certain issue or improving certain interoperability related aspects of a standard. They will generate results that provide indications or implications towards improving interoperability related aspects. This makes it difficult to automate most of the analysis. Therefore some analysis tasks may require human intervention to make intelligent decisions towards improving a standard.

It is also important to note that removal of duplicated content in the pre-processing stage may affect the results of some of the metrics.
The current framework mainly focuses on identifying deficits in a file format standard through a bottom-up approach. A top-down analysis, while not covered by our framework, might be helpful in identifying future requirements for standards. In such cases, the use of a top-down approach, such as the framework proposed by Mykkänen and Tuomainen (2008), is recommended. Neither the framework proposed by Mykkänen and Tuomainen (2008) nor our own framework currently consider aspects such as the cost of implementation of standards.

### 4.9 Conclusions

This chapter described the methodology used for the construction of the first large XLIFF corpus for research, and XML-DIUE, our novel empirical framework for the analysis of this corpus, focusing on XLIFF standard usage in production environments and its effect on interoperability. Our research has revealed some important findings related to XLIFF that are important for improving the interoperability of XLIFF-based data.

Our corpus and the framework we developed support research focusing on the role of a data-exchange formats standard in achieving interoperability in production scenarios. The framework does so by helping standard developers to empirically identify the most important elements, the least important elements, problematic areas, usage patterns. To the best of our knowledge, this is the first such framework described in the literature that presents standard usage metrics that can be used in a systematic bottom-up approach for identifying important criteria relating to the standardisation process. One of the significant benefits of the framework is that is can be repeatedly used as the standard evolves to help in the standard refinement and development process.

The framework applied for XLIFF, an open standard file format used in the localisation domain can be applied for similar XML based file formats in other domains for improving important aspects of interoperability. More information on using XML-DIUE for evaluating similar standards is given in Sect. 4.10.
4.10 Using XML-DIUE to Evaluate other XML based Data-Exchange Standards

Based on the experience we gained from this research, we propose some important guidelines that are useful when adapting our framework to evaluate the usage of other similar XML-based data-exchange standards.

4.10.1 Building a Corpus

The first step in our process involves building an authentic reference corpus of standard-based files. These can be from industry or open-source, or both, but their origin should be noted in the XML-DIUE framework protocol. Industry offers the potential of insight into commercial use, but open-source software is becoming increasingly prevalent in many verticals. Thus vertical, availability and declared study focus should ultimately define the choice of corpus.

As illustrated in our XLIFF based study, when analysing the usage of a standard, the main focus is often on the attributes and elements in the files. Thus the size and representative variety of the files in the corpus is often more important for this concern. Measures taken to address both concerns should be reported on for the study, but the corpus size should only be reported after the pre-processing stage (e.g. see Sect. 4.3.1). During the gathering of material good ethical guidelines should also be adhered to. Whenever possible, creators of the XML files should explicitly give their informed consent, and should be made aware of their ability to withdraw at any time. They should also know whether they (and their organisations) are guaranteed anonymity and the conditions under which the data will be held.

4.10.2 Cleaning and Pre-processing of the Corpus

Depending on the origin of the files obtained for the corpus, there may need to be a pre-processing stage, prior to analysis. A number of methods can be used to determine if this is necessary: Often, the expert vertical-knowledge of the researcher will suggest issues like encoding conventions that need to be addressed. Alternatively, or
additionally, a manual sampling of the files can be used to determine other pre-processing issues. Again, the technique used, the pre-processing adopted and the resultant corpus size should be made explicit by the researcher in reporting their work.

4.10.3 Data Extraction and Repository Construction

The next step involves extracting data from the corpus for analysis. In order to persist the data and facilitate analysis (and indeed future, unforeseen analysis) the framework suggests that the data from the XML files be parsed and transferred into a repository. The schema for such a repository would be dependent on the specific data-exchange file format under study and, to a lesser extent, the type of analysis envisaged. But ideally the schema should be as XML-agnostic as possible (see Fig. 16). For example, if the current analysis focuses more on the quantity of certain elements in the files, then a less structure-oriented schema would be appropriate. However, if the schema accommodates detailed and structural information then additional analysis tools can be added later to probe the structural aspects of the XML files without any additional parsing effort. The parsing method and resultant schema should be made explicit and justified.

4.10.4 Designing of the Standard Usage Analysis Metrics

Designing the usage-analysis is at the core of the XML-DIUUE framework. It is here where the researcher identifies the usage characteristics of interest and thus, this analysis activity should be considered in parallel with the development of the repository schema and the parsers. Research questions and associated standard usage metrics should be identified and reported. Various stakeholders of the standard that the researcher considered during the analysis needs to be mentioned. It is also important to highlight both the advantages and disadvantages (e.g. limitations) of these metrics. It is worth noting that most of the standard usage metrics proposed in our study may be directly applicable to evaluating similar standards.

4.10.5 Analysis and Reporting

In this step, the research identifies sub-research questions that will help to investigate important aspects of the standard related to data interoperability among tools and
technologies. The standard usage metrics identified in the last step will be used for finding answers to the sub-research questions identified in this step. Where appropriate, several standard usage metrics may be utilised for finding answers to a single sub-research question. Once the analysis is performed, the research should report the results. Then the last few steps involve discussing the results obtained and deriving and presenting the conclusions.
Chapter 5: Development of LocConnect – an Interoperability Test-bed

5.1 Introduction

From our pilot study, we identified that XLIFF is not just being used to exchange localisation data, but it can be used as a storage format or as a localisation data container in an end-to-end localisation workflow (see Sect. 3.7). XLIFF has many features that make it adaptable and usable for the recycling of localisation data. In particular, the use of XLIFF as a data container brought new insight into our research. We decided to further explore this aspect primarily for finding answers to our second research question, which investigates effective approaches based on data-exchange standards for addressing interoperability issues.

5.1.1 Design and Creation

For this purpose, we used the “design and creation” research strategy (Oates 2006, p.108) and developed an interoperability test-bed for studying the interoperability requirements of tools and technologies in Software Oriented Architecture (SOA) environments.

The design and creation research strategy focuses on developing new IT products, also called artefacts.

(Oates 2006, p.108)

Through this research, our contribution to the knowledge is an instantiation, which is defined as “a working system that demonstrates constructs, methods, ideas, genres or theories can be implemented in a computer-based system” (Oates 2006, p.108). According to Vaishnavi and Kuechler (2004), design and creation is a problem solving approach that involves five steps:
- **Awareness**: review of literature to identify areas for further research;
- **Suggestion**: a tentative proposal on how to address identified research gaps;
- **Development**: implementation of this tentative proposal;
- **Evaluation**: assessment of the developed artefact;
- **Conclusions**: interpretation of the results; identification of their contribution to our knowledge; unexpected results; areas for future research.

In the following, the above steps are described in terms of our research.

### 5.2 Awareness

The main motivations behind choosing the above research strategy include the IGNITE Localisation Factory which uses XLIFF as a localisation memory in the localisation process (see Sect. 2.8.4), the concept of distributed localisation processes by Schäler (2008) as described by Lewis et al (2009) for enabling interoperability among localisation web-services using the XLIFF standard, and the proposal to use an open Application Programming Interface (API) to enable interoperability between tools and technologies (Savourel 2007; Anastasiou 2011).

#### 5.2.1 Open APIs

Currently, software applications are increasingly moving towards distributed models such as cloud based services or software as services. Standards are vital for the interoperability of distributed software applications in these contexts. However, one of the major problems preventing successful interoperability between distributed applications and processes is the lack of (standardised) interfaces between them. There are some notable examples of localisation and translation-centric web services, such as those currently offered by Google, Bing and Yahoo. However, even here we run into interoperability issues as the interfaces provided do not follow any specific standard, and connecting to these services is still very much a manual process requiring the intervention of a skilled computer programmer to set up the call to the service, to validate the data sent in terms of string length, language pairs, and so on, and then to handle the data that is returned. Some localisation Translation Management Systems
(TMS) purport to provide such flexibility, but they tend to be monolithic in their approach, using pre-defined workflows, and requiring dedicated developers to incorporate services from other vendors into these workflows through the development of bespoke APIs. What is needed is a unified approach for integrating components, so that any service can be called in any order in an automated manner. Savourel (2007) highlights the importance of looking into service oriented approaches as well as common interfaces for localisation:

Translation resources access API. As the use of Web Services and web related application grows, it becomes important to develop a way to share translation-related resources across the web, not just through file import and export, but via direct requests and exchange at the segment level or lower. In other words, we need a standard way to query TMs, MT systems and terminology databases. This would allow applications on both sides - the ones needing to query and the ones who have data to provide - to work seamlessly in a heterogeneous environment. Quite a few applications already offer online query system for MT, TM and terminology, but they are using different methods of access. Having a common interface would greatly simplify the development of implementations using such systems and create opportunities for both sides of the transactions.

(Savourel 2007)

Consistent with Savourel, Anastasiou (2011) states “open APIs are important, because they provide a consistent development platform and help sharing content” in the context of localisation and web-services. McConnell (2011) also highlights the importance of a common RESTful API to integrate translation services. He presents suggestions for successful API design principles.

5.2.2 Localisation and Web-Services

Lewis et al (2009) observe that “the localization industry has been slow reap the benefits of modern integration technologies such as web service integration and orchestration.” To address this deficit, building on the idea proposed by Schäler (2008) Lewis et al propose a conceptual framework summarising how web services can be utilised properly in an end-to-end localisation workflow. XLIFF has been recommended to be used within this framework, although details have not been presented. Moreover,
an API has been presented in this study that helps to integrate different web services. Mateos (2010) presents a related research, investigating the requirements of localisation and web services standards to support SOA based localisation scenarios.

5.2.3 Beyond Interoperability Standards
Mykkänen and Tuomainen (2008) argue that “a given standard only specifies some aspects of interoperability” and usually additional standards or project-specific measures are needed to ensure interoperability. Lewis et al (2008) stress the importance of identifying Quality of Service (QoS) requirements, especially when dealing with web-services. This research points out that in addition to the syntactic and semantic components of interoperability, there exists Quality of Service (QoS) component, which describes the run-time characteristic of a system. QoS attributes include: 1) performance; 2) availability; and 3) security. These attributes also depend on various other factors such as a topology of connections between systems and resource sharing models of systems. Many current solutions address QoS characteristics in a rather ad hoc manner and attempts at formalising QoS characteristics are found to be rare (Lewis et al 2008). Moreover, Lewis et al (2008) argue that inconsistencies of workflows may also adversely affect interoperability, even though perfect semantic interoperable models are in place.

5.3 Suggestion
5.3.1 Objectives and Scope of the Research
In terms of Schäler’s (2008) and Lewis et al’s (2009) conceptual framework, and Savourel (2007) as well as Anastasiou’s (2011) proposal on open API, details are lacking on 1) the exact role of the XLIFF standard, 2) the service orchestration aspect and 3) the functions and the functionality of the open API. In terms of Mykkänen and Tuomainen and Lewis et al’s (2008) arguments over project specific parameters important for interoperability and QoS requirements of web-services41, these criteria are not explicitly researched in the localisation domain. Therefore, we aim to address these

41A PhD research is underway on QoS requirements on localisation web-services at the Localisation Research Centre of the University of Limerick.
gaps by the design and creation of a novel software oriented architecture (SOA) based interoperability test-bed, that will enable us to study the role of XLIFF and related QoS requirements in distributed software oriented localisation architecture. This will help us to gain a thorough understanding of using XLIFF as a data container and also to identify the limitations of XLIFF as well as other infrastructure requirements when used in an SOA. Moreover, this will also help to explore the role of XLIFF in the data-exchange aspect of interoperability in relation to its IEEE definition (see Sect. 1.1.1). Therefore, the sub-research question we investigate using the design and creation strategy is: “how can XLIFF standard be used in an SOA to support interoperability in the localisation process?” Our scope is limited to the identification of effective use of the XLIFF standard to address interoperability issues in an SOA. Research on architectural and implementation aspects of SOA are out of the scope.

5.3.2 Distributed Localisation: Ideal Scenario

With advancements in technology, the localisation process of the future can be driven by a successful integration of distributed heterogeneous software components. In this scenario, the components are dynamically integrated and orchestrated depending on the available resources to provide the best possible solution for a given localisation project. However, such an ideal component-based interoperability scenario in localisation is still far from reality. Therefore, in this research, we aim to model this ideal scenario by implementing a series of prototypes. As the initial step, an experimental setup has been designed with the help of fellow CNGL researchers, containing the essential components.

5.3.3 Experimental Setup

The experimental setup includes multiple interacting components. Firstly, a user creates a localisation project by submitting a source file and supplying some parameters through a user interface component. Next, the data captured by this component is sent to a Workflow Recommender component. The Workflow Recommender implements the appropriate business process. By analysing source file content, resource files as well as parameters provided by the user, the Workflow Recommender offers an optimum workflow for this particular localisation project. Then, a Mapper component analyses
this workflow and picks the most suitable components to carry out the tasks specified in the workflow. These components can be web services such as Machine Translation systems, Translation Memory Systems, Post Editing systems. The Mapper will establish links with the selected components. Then a data container will be circulated among the different components according to the workflow established earlier. As this data container moves through different components, the components modify the data. At the end of the project’s life cycle, a Converter component transforms this data container to a translated or localised file which is returned to the user.

**Service Orchestration**

Service Oriented Architecture is a key technology that has been widely adopted for integrating such highly dynamic distributed components (see Sect. 2.5). Our research revealed that the incorporation of an orchestration engine is essential to realise a successful SOA-based solution for coordinating localisation components. Furthermore, the necessity of a common messaging format that will enable the interoperability between components became evident. Thus, in order to manage the processes as well as data interoperability, we incorporated an orchestration engine into the aforementioned experimental setup. This experimental setup along with the orchestration engine provides an ideal framework for the investigation of interoperability issues among localisation components. This framework was later named as the Service Oriented Localisation Architecture Solution (SOLAS) (Aouad et al 2011).

**LocConnect**

At the core of the SOLAS experimental setup are the orchestration engine and the common messaging format, which jointly provide the basis for the exploration of interoperability issues among components. Our contribution to the above experimental setup is the design and creation of the orchestration engine. The orchestration engine was named as LocConnect in our research. The design and creation of LocConnect, which uses XLIFF as the main messaging format, enabled us to accomplish our research objectives mentioned in Sect. 5.3.1. The implementation details and the system architecture of the LocConnect can be consulted in Appendix H.
The following section provides a typical use case for LocConnect in the aforementioned experimental setup.

**Use Case**

A project manager logs into the LocConnect server and creates a LocConnect project (a job) by entering some parameters. Then the project manager uploads a source file. The LocConnect server will then generate an XLIFF file and assign a unique ID to this job. Next, it will store the parameters captured through its interface in the XLIFF file and embed the uploaded file in the same XLIFF file as an internal file reference. The Workflow Recommender will then pick up the job from LocConnect, retrieve the corresponding XLIFF file and analyse it. The Workflow Recommender will generate an optimum workflow to process the XLIFF file. The workflow describes the other components that this XLIFF file has to go through and the sequence of these components. The Workflow Recommender embeds this workflow information in the XLIFF file. Once the workflow information is attached, the file will be returned to the LocConnect server. When LocConnect receives the file from the Workflow Recommender, it decodes the workflow information found in the XLIFF file and initiates the rest of the activities in the workflow. Usually, the next activity will be to send the XLIFF file to a Mapper Component which is responsible for selecting the best web services or components for processing the XLIFF file. LocConnect will establish communication with the other specified components according to the workflow and component descriptions. As such, the workflow will be enacted by the LocConnect workflow engine. Once the XLIFF file is fully processed, XLIFF content can be post-edited online using LocConnect’s built-in post-editing component. During the project’s lifecycle, the project manager can check the status of the components using LocConnect’s live project tracking interface. Finally, the project manager can download the processed XLIFF and the localised files.

**5.4 XLIFF Data Container**

LocConnect uses XLIFF as its messaging format to communicate with components. The core of this architecture is the XLIFF-based data container defined in this research. Maximum effort has been made to abstain from custom extensions in defining this data.
container. Different components will access and make changes to this data container as it travels through different components and different phases of the workflow. The typical structure of the data container is given in Fig. 25.

When a new project is created in LocConnect, it will append parameters captured via the project creation page into the metadata section (see Section 2) of the data container. The metadata is stored as key-value pairs. During the workflow execution process, various components may use, append or change the metadata. The source file uploaded by the user will be stored within the XLIFF data container as an internal file reference (see Section 1). Any resource files uploaded during the project creation will also be stored as external-references as shown in Section 4. The resource files attached to this data container can be identified by their unique IDs and can be retrieved at any stage during the process. Furthermore, the identifier will allow retrieval of the metadata associated with those resources.

After project creation, the data container generated (i.e. the XLIFF file) is sent to the Workflow Recommender component. It analyses the project metadata as well as the original file format to recommend the optimum workflow to process the given source file. If the original file is in a format other than XLIFF, the Workflow Recommender will suggest the data container to be sent to a File Format Converter component. The file format converter will read the original file from the above internal-file reference and convert the source file into XLIFF format. The converted content will be stored in the same data container using the <body> section and the skeleton sections. The data container with the converted file content is then reanalysed by the Workflow Recommender component in order to propose the rest of the workflow. The workflow information will be stored in Section 3 of the data container. When the LocConnect server receives the data container back from the Workflow Recommender component, it will parse the workflow description and execute the rest of the sequence. Once the entire process is completed, the converter can use the data container to build the target file.
In this architecture, a single XLIFF-based data container is being used throughout the process. Different workflow phases and associated tools can be identified by the standard XLIFF elements such as `<phase>` and `<tools>`. Furthermore, tools can include various statistics (e.g. `<count-groups>`) in the same XLIFF file.

The XLIFF data container based architecture resembles the Transmission Control Protocol and the Internet Protocol (TCP/IP) architecture in that the data packet is routed based on its content. However, in this scenario, LocConnect plays several roles, including the role of a router, web server and a file server.
<xliff version="1.2"
xmlns="urn:oasis:names:tc:xliff:document:1.2">
  <file original="hello.txt" source-language="en" target-language="fr" datatype="plaintext" category="medical">
    <header>
      <skl> <internal-file/> </skl>
    </header>
    <reference>
      <internal-file form="base64">
        <original-file fileformat="exe"/>
      </internal-file>
    </reference>
    <reference>
      <internal-file>
        <metadata>
          <meta pname="testProject" pdescription="A test project" startdate="01/04/2011" deadline="10/12/2011" budget="13310" quality-requirement="High" use-mt="yes" use-rating="yes"/>
        </metadata>
        <internal-file/>
      </internal-file>
    </reference>
    <reference>
      <workflow>
        <task tool-id="LMC" order="1" status="pending"/>
      </workflow>
    </reference>
    <reference>
      <external-file href="http://LocConnect/get_resource.php?id=fcb4c5a8f1"/>
    </reference>
  </file>
</xliff>
5.5 Evaluation

We have designed and developed an orchestration engine for localisation services and distributed localisation components, mainly to facilitate interoperability in the localisation processes. LocConnect is the first to use an open standard file format, the XML Localization Interchange File Format (XLIFF), as the messaging format in an SOA to address interoperability issues among different localisation tools.

The proposed data container does not use any proprietary extensions to store additional information required for the service orchestration such as to store workflow information and metadata. Instead, it uses several specific <internal-file> and <external-file> elements to store these data and expects component technologies to interpret data found in these elements. In the current data container, this additional information is being stored in a custom XML format agreed by components. The main advantage of this approach is that it does not interfere with the representation of the core localisation data and metadata. The <body> of the XLIFF content is kept completely intact.

Moreover, the analysis of the derived XLIFF patterns for individual organisations from our XLIFF corpus shows that <internal-file> and <external-file> elements are not in common use (see Sect. 4.6.8). Therefore, these elements are ideal for representing additional information required for the service orchestration as they will minimise the possibility of any interference by components. It can also be seen that
the patterns identified above (see Sect. 4.6.8 and Appendix G) are fully compatible with
the proposed data container and therefore they can be readily represented in the
proposed data container without requiring any transformation.

However, two main limitations can be identified in this approach: first, the use of XML
content within <internal-file> and <external-file> elements invalidates
the XLIFF content. Having identified this issue, in Sect. 5.6.2 we propose an alternative
mechanism to store these data. Second, this approach needs components to agree on
additional semantics, for example the components should be able to identify and
interpret information contained within the embedded XML.

Instead of XLIFF, alternative messaging formats may be derived from the bitext
formats such as PO, TMX, TTX, SDLXLIFF and bilingual RTF. However XLIFF is
superior to all of these formats due to the reasons mentioned in Sect. 1.3.1. For example
PO and TMX are the only open standards; however they lack capability to store
additional metadata. On the other hand, the use of a more generic messaging format like
SOAP\(^{42}\) may be superfluous and may lead to loss of data as the data contained within
XLIFF need to be transformed to such a format. Moreover, the container may become
cumbersome and complex as it travels among components. On the other hand, it may
also impose additional overhead on components to process the redundant wrapper to
extract localisation data. As such, it can be reasonably expected that the use of XLIFF
directly as the messaging format in an SOA simplifies and improves the efficiency of
the localisation data exchange between component technologies.

### 5.6 Conclusions and Future Work

The successful implementation of this LocConnect interoperability test-bed suggests the
suitability of XLIFF as a full project life-cycle data container that can be used to
achieve interoperability in localisation processes. This also proposes a novel use case
for XLIFF, as a messaging format in an SOA environment.

\(^{42}\) [http://www.w3.org/TR/soap/](http://www.w3.org/TR/soap/) [accessed 30 Apr 2013]
Moreover, the framework has revealed various additional metadata and related infrastructure services required for linking distributed localisation tools and services (see Sect. 5.6.1) which will be especially useful when developing a commercial grade orchestration engine.

The design and creation of the LocConnect interoperability test-bed has helped us to achieve our objectives set at the beginning of this study (see Sect. 5.3.1). It has helped us to understand the exact role of XLIFF as the messaging format in an SOA. Thus it complements the conceptual localisation interoperability framework proposed by Lewis et al (2009). In addition, the development of the LocConnect has also revealed detailed syntactical requirements of an open API. The details of the API can be consulted from Appendix H.1 and Appendix H.3. Thus it addresses the gaps in existing proposals (Anastasiou 2011; Savourel 2007) and provides guidance towards developing a comprehensive and complete API. LocConnect also addresses the data exchange aspect of the IEEE definition (see Sect. 1.1.1) where data usage aspect of this definition has been covered by our XML-DIUE framework presented in the previous chapter (see Chapter 4).

LocConnect can not only provide web service and component integration, but can also act as a localisation data repository and a revision control system for localisation data such as a Concurrent Versioning System (CVS). Features of LocConnect are presented in Appendix H.2. To the best of our knowledge, LocConnect is the only orchestration engine reported to use XLIFF as a messaging format to enable data exchange interoperability among distributed localisation components. Similar commercial applications have not been found as yet. It has also been immensely helpful in identifying prominent issues that need to be addressed when developing a commercial application (see Sect. 5.6.1).

It is worth noting that LocConnect has been declared as a University of Limerick invention disclosure under case number 2006164. Originally completely developed by the researcher, the LocConnect has been later taken-over by a dedicated development team attached to the LRC for the further development. Ever since it was developed,
LocConnect has been demonstrated at various national and international events (See Appendix I for details). Details on LocConnect and the SOLAS platform can be sought from Wasala et al (2011a; 2011b) and Aouad et al (2011).

5.6.1 Limitations and Future Work

Whilst the prototype provides a test bed for the exploration of interoperability issues among localisation tools, it has a number of limitations.

Metadata Representation

In the present architecture, metadata is being stored as attribute-value pairs within an internal file reference of the XLIFF data container (see Section 2 of Fig. 25). However, according to the current XLIFF specification (XLIFF-TC 2008b), XML elements cannot be included within an internal file reference. Doing so would result in an invalid XLIFF file. While this could be interpreted as a limitation of the XLIFF standard itself, the current metadata representation mechanism also presents several problems. The metadata is exposed to all the components. Yet there might be situations where metadata should only be exposed to certain components. Therefore, some security and visibility mechanisms have to be implemented for the metadata. Moreover, there may be situations where components need to be granted specific permissions to access metadata, e.g. read or write. These problems can be overcome by separating the metadata from the XLIFF data container. That is, the metadata has to be stored in a separate datastore (as in the case of resource files). Then, specific API functions can be implemented to manipulate metadata (e.g. add, delete, modify, retrieve) by different components. This provides a secure mechanism to manage metadata.

The Resource Description Framework (RDF) is a framework for describing metadata (Anastasiou 2011). Therefore, it is worthwhile exploring the possibility of representing metadata using RDF. For example, API functions could be implemented to return the metadata required by a component in RDF syntax.
Binary vs. Textual Data

The current resource datastore is only capable of storing textual data. Therefore, it could be enhanced to store binary data too. This would enable the storing of various file formats including windows executable files, dll files, video files, images. Once the resource datastore is improved to store binary data, the original file can be stored in the resource datastore and in XLIFF, and a reference to this resource can be included as an external file reference (see Section 1 of Fig. 26).

API Functions

The current API lacks several important functions. Functions should be implemented for deleting projects (and associated XLIFF files), modifying projects, deleting resource files and modifying metadata associated with resource files, and so on. The current API calls set_output and set_status to 'complete' could be merged (i.e. sending the output by a component will automatically set its status to ‘complete’). Furthermore, a mechanism could be implemented for granting proper permissions to components for using the above functions.

User Management

User management is a significant aspect that we did not pay much attention to when developing the initial test bed. User roles could be designed and implemented so that users with different privileges can assign different permissions to components as well as different activities managed through the LocConnect server. This way, data security could be achieved to a certain extent. Furthermore, an API key should be introduced for the validation of components as another security measure. This way, components would have to specify the key whenever they use LocConnect API functions in order to access the LocConnect data.

Component Management

In the present architecture, the information about components has to be manually registered with the LocConnect server using its administrator interface. However, the
architecture should be improved to discover and register ad-hoc components automatically. Moreover, mechanisms have to be implemented to the automatic discovery of component failures. If the LocConnect server could detect communication failures, it could then select substitute components (instead of failed components) to enact a workflow. Architecture similar to internet protocol could be implemented with the help of a Mapper component. For example, whenever the LocConnect server detects a component failure, the data container could be automatically re-routed to another component that can undertake the same task so that the failure of a component will not affect the rest of the workflow.

**Data Security**

The XLIFF data container could contain sensitive data (i.e. source content, translations or metadata) which some components should not be able to access. A mechanism could be implemented to secure the content and to grant permissions to components so that they would only be able to access relevant data from the XLIFF data container. There are three potential solutions to this problem. One would be to let the workflow recommender (or the Mapper) select only secure and reliable components. The second solution could be to encrypt content within the XLIFF data container. The third solution could be to implement API functions to access specific parts of the XLIFF data container. However, the latter mechanism will obviously increase the complexity of the overall communication process due to frequent API calls to the LocConnect server.

Moreover, LocConnect implements REST-based services for communication with external components. Therefore, it is essential to implement our own security measures in the REST-based API. Since there are no security measures implemented in the current LocConnect API, well-established and powerful security measures such as XML encryption, API keys would need to be implemented in the API as well as in the data transmission channel (for example, the use of Secure Socket Layer (SSL) tunnels for REST calls).
Workflow Engine

While the current workflow engine provides essential process management operations, it currently lacks more complex features such as parallel processes and branching. Therefore, incorporation of a fully-fledged workflow engine into the LocConnect server is desirable. Ideally, the workflow engine should support standard workflow description languages such as Business Process Execution Language (BPEL) or Yet Another Workflow Language (YAWL). This would allow the LocConnect server to be easily connected to an existing business process, i.e. localisation could be included as part of an existing workflow. In the current system, the workflow data is included as an internal file reference in the XLIFF data container (see Section 3 of Fig. 25) which invalidates the XLIFF file due to the use of XML elements inside the internal file reference. In future versions, this problem can be easily addressed by simply storing the generated workflow as a separate resource file (e.g. using BPEL) and providing the link to the resource file in the XLIFF data container as an external file reference.

PUSH vs. PULL Architecture

Currently, the LocConnect server implements a ‘PULL’ based architecture where components have to initiate the data communication process. For example, components must keep checking for new jobs in the LocConnect server and fetch jobs from the server. The implementation of both ‘PUSH’ and ‘PULL’ based architectures would very likely yield more benefits. Such architecture would help to minimise communication overhead as well as resource consumption (for example, the LocConnect server can push a job whenever a job is available for a component, rather than a component continuously checking the LocConnect server for jobs). The implementation of both ‘PUSH’ and ‘PULL’ based architectures would also help to establish the availability of the components prior to assigning a job, and help the LocConnect server to detect component failures.

Performance Evaluation

Because the XLIFF standard was originally defined as a localisation data-exchange format, it has, so far, not been thoroughly assessed with regard to its suitability as a
localisation data storage format or as a data container. A systematic evaluation has to be performed on the use of XLIFF as a data container in the context of a full localisation project life cycle, as facilitated by our prototype. For example, during the traversal, an XLIFF-based data container could become cumbersome, causing performance difficulties. Different approaches to addressing likely performance issues could be explored, such as data container compression, support for parallel processing, or the use of multiple XLIFF-based data containers transmitted in a single compressed container. The implications of such strategies would have to be evaluated, such as the need to equip the components with a module to extract and compress the data container.

5.6.2 Proposed Improvements to the XLIFF based Data Container and New Architecture

By addressing the issues related to the above XLIFF-based data container, a fully XLIFF compliant data container could be developed to evaluate its effect on improvements in interoperability. A sample XLIFF data container is introduced in Fig. 26.

This data container differs from the current data container (see Fig. 25) in the following aspects. The new container:

- does not represent additional metadata (i.e. metadata other than that defined in the XLIFF specification) within the data container itself. Instead, this metadata will be stored in a separate metadata store that can be accessed via corresponding API functions;
- does not represent workflow metadata as an internal file reference. Instead, the workflow metadata will be stored separately in the resource datastore. A link to this workflow will then be included in the XLIFF data container as an external file reference (see Section 2 of Fig. 26);
- does not store the original file as an internal file reference. It will also be stored separately in the resource datastore. An external file reference will be included in the XLIFF file as shown in Section 1 of Fig. 26.
The new data container does not use any extensions to store additional metadata or data, nor does it use XML syntax within internal-file elements. Thus, the above architecture would provide a fully XLIFF compliant (i.e. XLIFF strict schema compatible) interoperability architecture. Due to the separation of the original file content, workflow information and metadata from the XLIFF data container, the container itself becomes lightweight and easy to manipulate. The development of a file format converter component based on this data container would also be uncomplicated.
Section 1

Original File
<external-file href="http://LocConnect/get_resource.php?id=fcb4c5a8f2"/>

Section 2

Workflow
<external-file href="http://LocConnect/get_resource.php?id=fcb4c5a8f5"/>

Section 3

TMX
<external-file href="http://LocConnect/get_resource.php?id=fcb4c5a8f1"/>

<phase-group>
  <phase phase-name="Project Initiate"
      process-name="project creation and requirement capturing" tool-id="LocConnect" company-name="LRC"/>
  <phase phase-name="Quality Assurance"
      process-name="authoring" tool-id="LKR"
      company-name="LKR"
      contact-email="contact@example.com"/>
</phase-group>

<tool tool-name="LocConnect" tool-id="LocConnect"
      tool-version="2.0"/>

</header>
<body>..</body>
</file>
</xliff>

Figure 26. Improved Data Container
Chapter 6: Conclusions and Future Work

This chapter restates our research questions posed at the beginning of the study, summarises the research objectives and explains how they have been realised by presenting a concrete research methodology. The research yielded a number of important contributions to both the localisation domain and the general computer standardisation community. These contributions and their impacts are summarised below. Finally, the limitations of the research are discussed and suggestions for future research are proposed.

6.1 Research Objectives

The purpose of our research was to explore interoperability issues related to localisation data-exchange standards. We identified that the lack of tool compliance to standards and the limitations of the standards lead to interoperability problems resulting in a significant cost to the localisation industry. There is very little academic literature in this area. The existing limited research approaches are episodic in nature and do not attempt systematic empirical analysis towards addressing interoperability issues related to prominent localisation data-exchange standards. We fill this gap in our research.

Our research sought answers to these two research questions:

RQ1: What are the limitations of existing localisation data-exchange standards and implementations?

RQ2: How can these limitations be overcome to adequately address significant aspects of interoperability in localisation?

These research questions are closely related to each other. Answers to the first research question essentially address some aspects of the second research question. In this research, our main focus was the finding of answers to the first research question. In order to find the answers to the second research question, we investigated how data-
exchange standards can effectively be used to enable successful interoperability among tools and technologies, in addition to addressing identified limitations of standards and implementations.

As stated in Chapter 1 (Sect. 1.2), the primary objective of our research was to improve semantic and syntactic interoperability aspects of prominent localisation data-exchange standards and propose strategies leading to a higher degree of standard-compliance among localisation tools. In achieving this primary objective, we also aimed to meet the following objectives:

1) to identify the degree of prescriptiveness and the degree of coverage of a standard by implementations;
2) to study whether a standard is sufficiently detailed and well enough defined to guarantee interoperability at both semantic and syntactic levels;
3) to identify optimal, standard-based solutions for facilitating data interoperability among tools.

6.2 Conclusions and Recommendations

The conclusions of each of the individual research strategies have been presented at the end of respective chapters (see Sect. 3.7, Sect. 4.9 and Sect. 5.6). In this section, we present conclusions and recommendations drawn from the overall research and also discuss how we have achieved our research objectives presented in the previous section (Sect. 6.2). In Sect. 6.2.1, we present main conclusions of the thesis and then we present other substantive conclusions in Sect. 6.2.2.

6.2.1 Main Conclusion: The Utility of the XML-DIUE Framework

Using the XML-DIUE framework proposed in our research, we illustrated how we can identify levels of tool support for XLIFF features, in Sect. 4.6.3. In Sect. 4.7.2 and Sect. 4.7.7, we discussed how to identify features leading to semantic and syntactic
interoperability issues. These results provide strong evidence that we have achieved our first two research objectives mentioned in the previous section (Sect. 6.1).

In relation to our main research question (RQ1), using the XML-DIUE framework we could not only identify the limitations of the XLIFF standard, but important (and less important) features of the standard for data-exchange interoperability. Therefore, we have not only achieved our objectives, but exceeded them. The results such as the core useful features of the standard (Sect. 4.7.1), how to simplify the standard (Sect. 4.7.4), feature usage patterns (Sect. 4.7.8) and custom extension used (Sect. 4.7.5) provide evidence for this.

As discussed in Sect. 4.8.2, the biggest threat to our research is its low external validity. The nature of our XLIFF corpus makes it difficult for us to generalise the results. However, our XLIFF corpus was significant enough to illustrate the practical utility of our XML-DIUE framework. Therefore, as the conclusion of our thesis, we highlight the practical utility of our XML-DIUE framework, to assist in discovering some of the limitations of XML-based data-exchange standards and their implementations, and to identify other important factors related to features of the standard that are useful for improving data-exchange interoperability. The XML-DIUE empirical framework can be used as a tool to evaluate data-exchange standard usage and thus inform on the development and maintenance of standards.

The empirical results generated by the framework seem useful for identifying important criteria for standard development such as the most commonly used features, the least frequently used features, usage patterns. The findings will also be helpful in identifying compliance issues associated with implementations supporting the standard under study. The framework can also be used for standard development and as a quick reference method to check the status of the actual implementation of elements, attributes, attribute values, and so on. Although valuable resources such as validators and test suites have been reported in other XML document exchange standards (e.g. by W3C), such resources are limited in the localisation domain. Our framework fills this gap. The framework will be helpful for the development of interoperability test suites, for
example by identifying common validation errors (see Sect. 4.7.7). Thus, we believe that this framework will ultimately contribute to improved interoperability among tools and technologies in many verticals.

More importantly, prior to the development of our framework, the XLIFF Technical Committee (TC) had neither access to a large body of data reflecting the actual usage of the standard nor a systematic methodology to evaluate and analyse the usage of the standard. Our research, for the first time, gathered sufficient data and developed the corresponding framework to allow for a systematic analysis of the usage of the standard. Using the current framework, the XLIFF TC can now analyse the actual use of the specification and focus on the aspects of the standard that are obviously more important or less important to the actual stakeholders of the standard.

6.2.2 Other Conclusions and Recommendations

In the previous section, we presented our main conclusion of the thesis. We have achieved our main objective (see Sect. 6.1) by developing and illustrating the utility of the XML-DIUE framework. In this section we present several other important conclusions derived from our research.

Importance of Systematic Comparisons of Standards

To get a deeper understanding of the features of a standard and their advantages and disadvantages, doing a comprehensive systematic comparison with similar standards and developing mappings among them is more appropriate and beneficial rather than merely doing a literature review about the standard. This was informed by both our research (see Chapter 3) and the research by Frimannsson (2005).

Another significant benefit of this methodology is that such comparisons will also help to identify interoperability issues among similar standards (i.e. inter-standard interoperability issues). However, the comparison of standards and the development of the mappings are not trivial, but tedious.
The methodology used in the pilot study may be applicable for similar format comparisons, not only in the localisation domain, but other domains too. Where several similar standards exist for the same purpose, better interoperable standards can be determined by doing a pilot study similar to the one carried out in this research. In particular, the development of the mappings helps to identify incompatibilities between similar standards and also help to identify unique features of the standards. Once the most interoperable standard has been selected using a methodology similar to one we used in our pilot study, that standard can be further analysed using the XML-DIUE framework to discover any further limitations of the standard and its implementations (i.e. to identify intra-standard interoperability issues).

**XLiff as a Messaging Format**

Among the objectives of our research was to find optimal ways of using the standard to enable interoperability. From our pilot study, we understood that XLIFF is being used as a data container in localisation workflows. We investigated this aspect in greater detail in our design and creation experiment (see Chapter 5). From the design and creation of LocConnect, we demonstrated a novel use case for XLIFF: how XLIFF can be used as a messaging format in combination with an open API to enable data-exchange interoperability among distributed localisation components. The research revealed that XLIFF can be used as a messaging format even without using any custom extensions.

The design and creation of the LocConnect was also helpful in identifying detailed requirements of API and the importance of addressing various QoS requirements (such as security, availability, reliability, scalability) related to components and communication channels. We also highlight three major important aspects that were apparent from our research, related to distributed component based localisation processes:

- the importance of a component simulator in the component development and debugging;
• workflow visualisation, real-time service status updates, and component error reporting during the workflow execution phase;
• requirement of properly documented API. Like McConnell (2011) we also advise keeping the API as simple as possible.

The LocConnect interoperability test-bed has been successful in identifying interoperability requirements in an SOA based localisation process. In addition, it has helped us to gain hands-on experience in implementing the XLIFF standard.

**Importance of Addressing Semantic Interoperability Issues**

Especially in the future, it is important for data-exchange formats to cater for the fully automated machine processing of data (rather than for human consumption). However, due to several factors, it is difficult to process localisation data encoded using current version of the XLIFF (v.1.2) by machines automatically. As a result, various interoperability issues may occur when attempting automated processing of XLIFF data in workflows. These factors are further described below.

1. **Custom extensions**

   Custom extensions added by one party may not be interpreted by other parties.

2. **User-defined attribute values**

   The XLIFF specification allows user-defined values to be used in certain attributes using the 'x-' prefix. These user-defined values may only processable by the tool that originally introduced them.

   Two example attributes with some of their user-defined values obtained from our results are given below (see Appendix E for custom values defined for other attributes).

   context-type: x-application, x-attribute-name, x-authorization-scheme
The lack of definitions and semantic agreements of these user-defined values hinders tools’ interoperability and makes it difficult to automatically process them.

3. Other features that allow user-defined values

These features are different from the previously discussed user-defined attribute values, as they do not use special identifiers such as 'x-' to denote that they are user-defined. However, these features allow custom values. Usually, these are the features used for carrying metadata. These features allow any textual value (i.e. any string) to be used with them. For example, the XLIFF attribute category is intended to be used to denote the subject of the translations in an XLIFF file. Any text value is permitted to be used for this attribute. The values found in our corpus for this attribute were: String, Enterprise Software, resourceBundle, Survey Project.

However, in the absence of a proper agreement among tools for these values, interoperability is diminished and the data cannot automatically be processed optimally by different tools due to lack of semantics.

Identification and addressing of these issues

Identification of these features is the first step towards addressing them. Our framework can be used for this purpose. In Sect. 4.7.5, we discussed how our framework can be used to identify custom extensions. In Sect. 4.7.2, we discussed how to identify the features leading to semantic interoperability issues such as data level conflicts and schema level conflicts (Park and Ram 2004) (see Sect. 2.3.2 for more details on semantic interoperability). To solve such semantic interoperability issues, we propose to adapt a methodology like the one proposed by Berges et al (2010) for addressing similar issues in the healthcare domain (see Sect. 2.4.1). For example, tool specific metadata items are mapped to a central ontology that can be used to define the semantics of proprietary data.
Future Trends

Future trends need to be taken into account when developing standards. We highlighted the importance of representing machine processable data in the previous section. In addition to this, as we discussed in Sect. 1.1.2 and Sec. 5.2, localisation processes and standards need to adjust according to current and future technological trends such as the SOA, distributed computing, cloud computing, personalisation and mobile computing. The evolving nature of technologies affects the standardisation process (Ray 2009). Over time, some features of the standard may get outdated. New features need to be created to address new requirements. Failing this, interoperability issues may arise. For example, Heiler (1995) gives an example to illustrate how semantic interoperability issues could arise from using old features of the standard to represent new purposes. Moreover, features of the standard that are never used only increase the implementation complexity of the standard. Therefore, the standard needs to be constantly reviewed while future requirements are being captured and addressed.

Our research addresses some of the above aspects: first, by providing the XML-DIUE framework (see Chapter 4) to review the current usage of the standard (e.g. to identify never used XLIFF features); secondly by developing the first SOA based interoperability test–bed (see Chapter 5) which can be used to explore future standardisation requirements; and also by proposing a zipped data container for XLIFF for efficient bandwidth and storage utilisation.

Importance of a Reference Implementation of the Standard

Results of our research show that (see Sect. 4.7.6) there is a lack of open-source software that implements XLIFF. According to Tiemann’s (2006) definitions of different levels of open standards, XLIFF can be categorised into “Open Standard Level 1” (see Sect. 2.6.2) as there are open-source software packages such as Okapi Rainbow tools, Virtaal and OmegaT which can interoperate with the XLIFF standard. However, we highlight the importance of achieving at least the “Open Standard Level 2”. While the Open Standard Level 1 guarantees the existence of an open-source implementation of the standard, Open Standard Level 2 goes one step beyond to this by providing a dedicated open-source reference implementation of the standard. To further support our
recommendation to achieve Open Standard Level 2 in XLIFF, we quote Tiemann’s definition below:

Open Standards Principle 2 aims beyond merely giving users a guaranteed exit strategy: it provides the ability to review the actual workings of the standard in an implementation. While there clearly is a distinction between a standard definition and a reference implementation, a reference implementation can be extremely valuable in identifying gaps or hidden assumptions that may underlie a standard definition. The availability of reference implementations of both web servers and web clients proved indispensable in corroborating or impeaching claims of competing proprietary vendors in reference to HTTP and HTML standards, for example. The IETF’s Best Practices paper ... generalizes this principle, arguing that standards without implementations are not trustworthy, and implementations that cannot be examined cannot be verified.

(Tiemann 2006)

As pointed out by Tiemann, among the benefits of having a reference implementation of a standard is the ability to identify limitations of the standards including underlying gaps and hidden assumptions. In addition, we also highlight the importance of having a reference implementation to carry out interoperability tests among tools. The existence of a fully-fledged reference implementation significantly simplifies the interoperability testing process by eliminating the need to cross-check all possible combinations of tools (Shah and Kesan 2008; 2009). Moreover, the existence of a reference implementation may also help to understand the processing requirements of the features of the standard more precisely.

In this context, it is important to note that one of the contributions of our research is LocConnect, which is also an open-source reference implementation of the XLIFF standard (see Sect. 6.3).

**Importance of an Object Model**

From our pilot study, we realised the importance of having an Object Model (OM) (e.g. similar to the OM found in the LCX format) that describes how to implement and
manipulate the standard programmatically. A properly documented OM helps to achieve consistency of implementations and thereby helps to address non-orthogonal issues of representations like the ones discussed in our literature review (see Sect. 2.8.2). Moreover, an OM may also help to clear any dubious processing requirements of features (especially for the tool developers) and simplify the standard implementation process. Therefore, we highlight the importance of an Object Model for XLIFF, as it will contribute towards improving the standard conformance in tools.

**Defining “Support”**

The term “support” is a widely used term related to XLIFF implementations (tools). A few examples are given below:

- *the tool supports XLIFF*;
- *the tool supports feature X (e.g. tool supports in-line mark-up elements)*;
- *the tool partially supports feature X*.

As discussed in detail in Sect. 2.8.2, lack of a proper definition for this term leads to many confusions. It can also be seen that the term “support” has been used by the tool developers just as a marketing slogan. Therefore, we recommend that one of the first steps towards addressing interoperability is the proper definition of term “support” in these contexts.

### 6.3 Research Contributions

This research offers several contributions. Our substantial contribution is the proposed XML-DIUE framework that can be used to evaluate the usage of localisation data-exchange standards. The XML-DIUE framework provides the following key benefits:

XML-DIUE helps to identify

- limitations of standards and implementations,
- important features of standards for successful data-exchange interoperability.
In addition, our research also provides several contributions at different levels to different stakeholders. These contributions are presented below.

**Contribution of knowledge to the XLIFF TC:**

- The XLIFF symposium is the most relevant forum to present our research results. The findings of our research were presented to the XLIFF community on two main occasions: the first XLIFF symposium (Wasala et al 2010) and the third XLIFF symposium (Wasala et al 2012). Our research was well received by the XLIFF community on both occasions. Results of our research have direct impact on the XLIFF standard and these are further described in Sect. 6.4.
- Through the design and creation of the LocConnect interoperability test-bed, (see Chapter 5), we presented and proved a novel use-case for XLIFF. That is the use of XLIFF as the main messaging format (i.e. the payload) of SOA architecture to enable interoperability among distributed localisation components. The LocConnect itself is a substantial contribution not only as an open-source reference implementation of XLIFF, but also as a research platform for studying the interoperability requirements of distributed localisation components.
- The researcher joined the XLIFF TC in early 2012 and has been serving as a voting member contributing to the XLIFF standardisation work.
- The knowledge gained from the current PhD research on localisation and related standards has helped the researcher to apply learned concepts into a number of other interesting localisation research projects. These projects were especially related to the real-time localisation of websites and desktop software using micro-crowdsourcing. The XLIFF standard has been implemented in various prototypes developed as part of these projects.

**Contribution of knowledge to Microsoft:**

- The pilot study revealed the advantages, disadvantages and unique feature of both the LCX and XLIFF file formats. The proposed schema mapping between these formats and the developed prototype converter will be a valuable
contribution to Microsoft. At the time of writing this thesis, Microsoft is in a phase of adopting XLIFF in their localisation process.

**Contribution of knowledge to tool developers and standard consumers:**

- Tool developers and standard consumers will also benefit from our XML-DIUE framework. The XLIFF corpus analysis yields a number of important results for both of these groups. For example, by identifying important features for cross-organisation data interoperability, to identify features that are likely to be deprecated in future (see Sect. 4.4 for details).

**Contribution of knowledge to the localisation domain and other fields in general through publications:**

- During different stages of our research, we published and presented several research papers in various peer-reviewed journals and conferences. First, we presented our pilot study (details provided in Chapter 3) at the first XLIFF symposium (Wasala et al 2010). Later, a journal paper was published with more details on the same study (Wasala et al 2012b). Then, two journal papers were published detailing LocConnect, our design and creation experiment (Wasala et al 2011a; Wasala et al 2011b). We also contributed to a book chapter (Aouad et al 2011) discussing the role of LocConnect in the SOLAS platform. Subsequently, LocConnect has been presented and demonstrated in almost all CNGL events held since it was developed, along with being demonstrated at international conferences such as FEISGILTT 2012 and AGIS 2010 (Filip et al 2012; O’Conchuir 2010). In 2012, the University of Limerick declared LocConnect as an Invention Disclosure (Wasala and Schäler 2012). In addition to peer-reviewed papers, our pilot study and LocConnect has been presented in various national and international events through posters (see the full publication list in Appendix I for details). A video on the SOLAS platform, including the LocConnect orchestration engine, has been published by the Localisation Research Centre, and can be accessed at: http://www.youtube.com/watch?v=iU3bp6yPojQ&t=47m15s. Next, the findings of the XLIFF corpus analysis experiment were presented in detail at the third
XLIFF symposium (Wasala et al 2012a). Finally, a journal paper is being prepared on our XML-DIUE framework and is currently in its final stages. We hope to publish it in a high-impact journal related to computer standards.

- As previously mentioned, the knowledge gained from this research could be applied in several other projects. It is worth noting that a number of academic publications have also resulted from these projects, including a book chapter contribution and proceedings of various conferences (see Appendix I for more information on our publications).

6.3.1 Resources Produced by this Research

The following section presents important resources which have emerged from our research.

Resources Produced from the Pilot Study

Several sample XLIFF and LCX files were generated as a part of our pilot study. Both XLIFF and LCX files were generated from the same source file. Therefore, this resulted in several parallel XLIFF and LCX files. These files will be useful in future comparative studies. Moreover, the XSLT mappings developed to convert between XLIFF and LCX formats will serve as an XLIFF filter for the LCX format. The prototype converter developed as a part of this research will serve as a first step towards implementing a fully-fledged converter.

Resources Derived from the XML-DIUE Framework

One of the substantial resources generated by this research is the XLIFF corpus constructed as a part of our main experiment. The XLIFF corpus includes 3179 files. The files were mainly collected through contributions from several CNGL industrial partners and by crawling openly available XLIFF files on the web (see Sect. 4.3.1 for details on the XLIFF corpus construction). This is the first XLIFF corpus ever reported in the literature. While the provision of access to the industrial contributions to this corpus for future research is subjected to the conditions imposed by them (i.e.
anonymity and non-disclosure), a significant portion of the corpus is made up with open-source files (2108 files); hence that portion can be readily published.\textsuperscript{43}

It is also worth noting that the XLIFF corpus will be a significant asset as it also represents a huge Translation Memory (TM), where other research could also be carried out (for example, research related to TM leveraging and segmentation issues).

Significant resources in addition to the corpus emerged from this research, including the XLIFF database derived from the corpus (i.e. the parsed XLIFF repository containing about 1 GB of data) as well as the technologies to crawl, pre-process, bulk validate, and construct the database, and solid data analysis criteria. These resources can be used in the future to evaluate and improve the XLIFF standard further, while the technologies developed can be used for the evaluation of similar standards.

**Resources Developed from the Design and Creation Strategy**

The main outcome of the design and creation strategy is the development of the LocConnect interoperability test-bed (see Chapter 5). It will serve as a platform for future research in the area of distributed, component based localisation. LocConnect will be further developed as a commercial grade orchestration engine, yet it will be released to the public as open-source\textsuperscript{44} software, promoting further research and innovation.

**6.4 Impact of the Research**

- The pilot study and its results were presented at the first XLIFF Symposium in Limerick. We presented advantages and disadvantages of both the LCX and XLIFF file formats, interoperability issues associated with the XLIFF standard and proposed improvements to both file format standards. The research and the results were well received and discussed by the XLIFF community.

\textsuperscript{43} At the time of writing this thesis, a decision has not been taken on publicly releasing the XLIFF corpus produced by this research.

\textsuperscript{44} Development work is still on-going; therefore an official release date has not been decided.
Subsequently, the XLIFF TC recorded and published specific contributions from individual presentations of the first XLIFF symposium. The specific takeaways from our research by the XLIFF TC committee are available at: https://wiki.oasis-open.org/xliff/Consolidated%20Takesaways%20from%20First%20XLIFF%20Symposium [accessed 11 Feb 2013].

Lieske (2011) presents an article titled “Insights in to future XLIFF” in Multilingual magazine summarising the important points of the first XLIFF symposium. It worth noting that some of the highlighted proposals such as the zipped data container and binary data representation mechanism towards improving the XLIFF standard in this article are direct contributions of our research.

- We presented our main experiment, the implementation of the XML-DIUE framework and its results, at the third XLIFF symposium. This symposium was attended by many XLIFF TC members as well as tool vendors. Subsequently, the XLIFF Promotion and Liaison Sub-Committee (XLIFF P&L SC) have decided to implement our framework. It has already started to collect XLIFF files with the aid of XLIFF TC, with the intention of constructing a representative corpus. The comments by the XLIFF TC on our presentation can be found here: https://wiki.oasis-open.org/xliff/Consolidated%20Takesaways%20from20Third%20XLIFF%20Symposium [accessed 11 Feb 2013].

- In our pilot study, we compared Microsoft’s internal localisation file format: LCX with the open standard XLIFF. Through our publications and presentations, our research has provided an opportunity for Microsoft to engage with the XLIFF community. In addition, our research has influenced Microsoft’s decisions in adopting the XLIFF standard in their workflows.

- The development of LocConnect posed several substantial impacts. First, LocConnect can be introduced as the technological breakthrough that led to the Service Oriented Localisation Architecture Solution (SOLAS). The SOLAS platform includes various localisation components (ranging from content pre-
processing to crowdsourced translation voting) developed using various technologies by various researchers in the CNGL project. LocConnect is being used as the orchestration engine of the SOLAS platform. LocConnect motivated fellow researchers of the CNGL project to develop components and integrate them into the SOLAS platform. This was not only helpful in identifying interoperability issues among distributed components, but also in developing relationships and improving collaborations among researchers from different universities. Currently, LocConnect connects over eight localisation software components distributed across three universities in Ireland: University of Limerick, Dublin City University and Trinity College Dublin. As previously mentioned, ever since it has been developed, it was demonstrated at various international and national events and has been successful in attracting the attention of academia as well as industry.

Discussions on SOA based approaches for localisation are only starting. LocConnect, along with the SOLAS platform, is the only reported platform discussed in the literature which demonstrates the enactment of an end-to-end localisation workflow using distributed component technologies. LocConnect is the only reported platform that uses XLIFF as the main messaging format to enable communication and interoperability between distributed component technologies.

LocConnect has been deployed publicly for testing purposes and can be accessed via: [http://193.1.97.50/locconnect/index.php](http://193.1.97.50/locconnect/index.php) [accessed 13 Mar 2013].

6.5 Limitations of the Research

*no study is perfect and ... the real challenge is to create, design and conduct high-impact, credible studies*

*(Perry et al 2000)*

Perry et al (2000) point out that designing high-impact credible empirical studies involves managing trade-offs in a manner so that they maximise the accuracy of
interpretation, research relevance and impact subject to resource constraints and risks. The
limitations are often related to the latter two criteria. In this section we describe the
various limitations of our research.

- Limitations of our individual research strategies were discussed at the end of
each of the respective chapters: see Sect. 3.7.3, Sect. 4.8.2 and Sect. 5.6.3. The
major limitation of our research is the low external validity due to the lack of
representativeness of the XLIFF corpus. This was discussed in detail in
Sect. 4.8.2.
- Due to time and resource accessibility constraints, in our pilot study, we chose
only to compare two standards. Comparison of multiple standards could lead to
more insights into features of standards as well as interoperability issues among
them.
- A very few research activities have been reported in relation to the bottom-up
analysis of standards, especially using corpus-based approaches. We
summarised related literature in Sect. 4.1.1. The lack of research in this area
compromises the ability to generalise and compare findings of our research.
- The current research mainly focuses on a bottom-up analysis of standard
constructs. Our research does not take into account the future requirements of
standards. Therefore, the bottom-up approach may not cover some aspects that
could be captured by top-down approaches. Moreover, there can be additional
requirements for successful tools’ interoperability, beyond what has been
covered by the standard. While our design and creation experiment partially
investigated this issue, not all the requirements could be captured by it.

6.7 Future Work

Our research has opened up several interesting future research opportunities. These are
discussed in the following sections.
6.7.1 Improvements to the XML-DIUE Framework

As highlighted in Sect. 6.2.1, our core contribution with this research is the XML-DIUE framework. Therefore, future work should mainly focus on the further development of the framework. In the following, we propose several enhancements and extensions to it.

Expansion of the XLIFF Corpus

The XLIFF corpus should be continually expanded by collecting new files. New sources have to be identified where XLIFF files could be obtained. Openly available files should be continually crawled. LocConnect could also be used to collect XLIFF files for the XLIFF corpus subjected to permissions from the users of LocConnect. We propose the intervention of XLIFF TC in the corpus building process. Future research should also focus on improving the representativeness of the corpus. For this purpose, one possibility is to create or identify several different source files first and then disseminate them to the XLIFF standard implementers who will be requested to convert them into XLIFF. A corpus can be constructed out of the converted files and it can then be analysed using our framework.

It is also important to take into account the organisation of files in the corpus and improvements to the repository. Recording of different metadata related to individual XLIFF files such as the generated date, crawled date, modified date, and stage in the workflow which the file was used, and tools used to produce the file, could provide valuable information for further analysis. Moreover, the choice of a data persistent XML-enabled data repository (instead of the current relational database) would be more suitable to store XLIFF content, especially when analysing frequently used patterns (for example, XML snippets).

Improvements to the Standard Usage Evaluation Metrics

Future work should also focus on fine-tuning the existing standard usage analysis metrics and the identification and proposal of new metrics. As pointed out earlier (Sect. 4.4), one of the major limitations of the existing metrics is the need of manual analysis and human intervention in the decision making process. Future work should
focus on the proposal of new metrics or improving the existing metrics that lead to results that require minimum human intervention for interpretation and the decision making process. While most of the standard usage analysis metrics proposed in this research are generic and applicable to many standards, domain-specific and standard-specific metrics could also be identified.

**Different Analysis**

Different analysis can be performed on the selected standard using the XML-DIUE framework. As the corpus grows, longitudinal studies could be conducted to examine the trends of feature usage over time. Moreover, analysis of XLIFF files used in different phases of the localisation workflow may yield interesting findings. Other possibilities include the application of XML-DIUE separately on different versions of the standard, and the use of XML-DIUE for evaluating similar standards and to compare and contrast features of those.

**Evaluation of XML-DIUE**

The XML-DIUE framework is solely based on the analysis of a corpus constructed using authentic files of a selected standard. As discussed in Sect. 4.8.2, the corpus building process is a never ending task. The practical use of the framework is to apply it to evaluate various usage characteristics of features of XML-based data-exchange standards and the standards keep on evolving. These factors make it difficult to evaluate the effectiveness and the utility of the framework. However, the framework could be evaluated by conducting a pre-survey followed by a post-survey of the standard developers (e.g. XLIFF TC) as described below.

- First, a pre-survey would be conducted prior to the deployment of the XML-DIUE framework to determine the current methods in use to evaluate the usage of the standard.
- In the next step, XML-DIUE would be deployed and used in the standard development process for a certain considerable period of time (for example, 6 months).
Finally, a post-survey would be conducted to assess the usefulness of the XML-DIUE framework where opinions could also be sought on the utility of different metrics and proposed improvements to those. A comparison of pre- and post-surveys would validate the utility of the framework.

In addition to the above, the application of the framework to evaluate similar standards in the localisation domain as well as other domains may also increase the validity of the framework.

XML-DIUE and Interdisciplinary Research

Data mining is the analysis of (often large) observational data sets to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful to the data owner.

(Hand et al 2001)

The proposed XML-DIUE is a first step towards adapting knowledge on corpus linguistics to improve standards. From the literature review, we identified that there is a paucity of research on corpus based studies in relation to the standardisation work. Our research has laid the foundation for much other interdisciplinary research towards improving XML-based data-exchange standards. Many opportunities exist to explore how to utilise the existing knowledge in the fields of Natural Language Processing (NLP), Computational Linguistics and Corpus Linguistics in aid of the development and maintenance of modern XML based standards.

The research could benefit significantly from the utilisation of Data Mining techniques and algorithms to develop more fine-grained, precise standard usage analysis metrics. Future research could also expand upon using XML Information Retrieval techniques to improve standard usage analysis metrics.
Finally, the XML-DIUE has opened up many possibilities for conducting research on similar data-exchange standards used in other domains.

### 6.7.2 Other Possibilities

Future work should also focus on addressing the limitations of the pilot study and the design and creation experiment. Limitations and possible future work to address these limitations of the above strategies have been discussed in Sect. 3.7.3 and Sect. 5.6.3 respectively.

The research could be complemented by additional research concerning the following aspects:

- research could be carried out “top-down” as identified by Frimannsson and Lieske (2010) to address existing issues of the standard, as well as to capture future standardisation requirements. In aid of top-down analysis, the framework proposed by Mykkänen and Tuomainen (2008) could be used.

- As pointed out by Shah and Kesan (2008), there might be some requirements beyond what is covered by the standards to enable successful interoperability between tools. These requirements can be identified by performing interoperability tests among tools. For this purpose, the methodology proposed by Shah and Kesan (2008; 2009) could be adopted. An alternative would be to perform additional interoperability tests with more data following the work proposed by Anastasiou and Morado-Vázquez (2010).

Future work should also not only consider addressing existing interoperability issues and capturing new standardisation requirements, but should also focus on how to ensure standard-compliance in the long-term. As highlighted in Sect. 2.8.4, the only known serious academic work in this area is the IGNITE project (LocFocus 2006; Schäler 2007; Schäler and Kelly 2007). Unfortunately, it is currently not operational. Future
research should augment this work and focus on methodologies to ensure standard-compliance, by means of conformance test suite development, conformance testing, interoperability testing and tool certification as discussed in Sect. 2.8.4.

6.8 Final Remarks

To the best of this researcher’s knowledge,

1) this is the only academic empirical research reported in the localisation domain for addressing interoperability issues related to localisation data-exchange standards.

2) XML-DIUE is the only reported systematic practical empirical framework, which constitutes standard usage analysis metrics for evaluating the usage of XML-based standards, mainly for improving data-exchange interoperability.

This will be a first step toward academic studies on using empirical evidence to improve standards. Therefore, we believe our research has made both substantial and significant contribution, not only to the localisation domain, but also to many other verticals.

Our research shows that a standard like XLIFF does not necessarily mean interoperability; having a functioning standard committee like in XLIFF does not necessarily lead to an efficient and lean standard. Our research highlighted major areas of improvement.

Finally, as we mentioned in Sect. 2.8.4, many other activities are on-going to address interoperability issues related to localisation standards. Our research only addressed a fraction of the overall issues. Therefore, consistent with Choudhury (2012) we also acknowledge the need for a coherent strategy to connect various efforts by different parties towards improving the semantic and syntactic interoperability aspects of prominent localisation data-exchange standards and propose strategies leading to a higher degree of standard-compliance among localisation tools.
Bibliography


Track, FEISGILTT 2012 (collocated with Localization World 2012), Seattle, USA, 17-19 Oct.


Folmer, E. and Verhoosel, J. (2011) State of the Art on Semantic IS Standardization, Interoperability & Quality, UT, CTIT, TNO en NOiV.


Leahy, P. (2011) 'Enabling Translation is the role of a Developer/Author', in 16th Annual LRC Internationalisation & Localisation Conference Limerick, Ireland, 22-23 Sept, University of Limerick.

Natural Language Processing, Boulder, Colorado, 1621956: Association for Computational Linguistics, 47-55.


LISA (2008b) 'Term Base eXchange (TBX)', [online], available: http://www.lisa.org/Term-Base-eXchange.32.0.html [accessed 6 Jul 2009].


McConnell, B. (2011) 'Building a common RESTFul API for translation services', *TAUS User Conference 2011* [online], available: [http://www.youtube.com/watch?v=9yXFapKW6Zg] [accessed 30 Jan 2013]


Microformats (n.d.) 'About Microformats', [online], available: [http://microformats.org/about] [accessed 15 Aug 2012].


204


Soderstrom, E. (2003) 'Casting the standards play - which are the roles?', accepted for The 3rd Conference on Standardization and Innovation in Information Technology, 22-24 Oct, 253-260.


Appendix A: The Localisation Research Centre (LRC) and the Centre for Next Generation Localisation (CNGL)

Localisation Research Centre (LRC)

Established in 1995, The Localisation Research Centre is the only dedicated research and educational centre that exclusively focuses on localisation. The LRC has established partnerships with digital publishers, localisation tool developers, Localisation Service Providers (LSPs), the EU commission, as well as other academic institutions worldwide (LRC 2010; Anastasiou 2009). Among the main activities of the centre are: publishing the “Localisation Focus” journal, the only peer-reviewed, indexed journal which focuses exclusively on Globalisation, Internationalisation, Localisation and Translation (GILT) related research; hosting the annual LRC Localisation conference; offering a Master’s degree programme in Multilingual Computing and Localisation, and finally conducting world-class research in the localisation field.

Centre for Next Generation Localisation (CNGL)

The Centre for Next Generation Localisation (CNGL) is an Academia–Industry partnership that conducts research related to “machine translation, localisation, multilingual content search, and personalisation, as well as driving standards in content and localisation service integration” (CNGL 2013). Comprised of over 100 researchers from four universities in Ireland (University of Limerick, Trinity College Dublin, Dublin College University, University College Dublin) and 13 top industry leaders in the field including Microsoft, Intel, and Symantec among others, CNGL undoubtedly is the largest research project in the aforementioned areas of research. CNGL, which was operational since 2007, is currently in its second phase. In the first phase, the research within CNGL was divided into four main tracks with localisation being one of them (Anastasiou 2009). The current research was carried out as a part of the localisation track, which was led by the LRC and also with the support of Microsoft European Development Centre (EDC), Ireland.
Appendix D: File Format Conversion Tool User Guide

D.1 Limitations of the Converter

The tool can only process LCL files generated by the LocProject Editor. The tool assumes that LCX to LCX-XLIFF conversion is carried out soon after generating the LCL file. Currently, the tool does not support the conversion of following LCX elements and their sub elements: <Prop> <Cmt> <PsrProps> <Bin>. Furthermore, the tool is only capable of converting LCL files or LCX-XLIFF files that represent Win32 resources. The tool can only convert LCX-XLIFF files generated by itself. The tool uses Saxon47 as its internal XSLT parser.

D.2 Converter Usage

A screenshot of the developed tool is given in Fig. D1.

Figure D1. A Screenshot of the File Format Converter48

---

48 Note that Microsoft has not endorsed this tool. It was developed by the researcher as a proof-of-concept for the proposed mapping.
1) Clicking on the “Open-Source Document” link will open a standard file open dialog where, the user can select either an LCX (*.lcl) file or XLIFF (*.xlf) file for the conversion.

2) Next, the user should specify the target file path for saving the converted file. This can be done by clicking on the “Select Target Document” link.

3) Then, by simply clicking on the “Convert” link, the tool will convert the file into the target file format. The tool will first examine the extension of the opened file. If it is an LCL file, the tool will convert the file into the LCX-XLIFF format and save it in the specified location. If the opened file is an XLIFF file, the tool will convert the file into the LCX format and save it in the specified location with the LCL extension.

Note: During the conversion process, a pop-up window will appear with a message displaying “Processing”. This message will be automatically closed after processing a file. Please allow the message to be automatically closed before start converting another file. If the user does not specify the target file path, the converter will create the target file in the same folder as the source document, using the name of the original file, but with the extension of the converted file.

The status area will show any errors occurred during the conversion process. Otherwise, a message will be appear in the status area saying “Document converted successfully”.

The XSLT style sheets used by the tool can also be configured manually. See Fig. D2.

![Figure D2. XSLT Configuration Dialog](image_url)
Appendix F: Attributes and their Values

The XLIFF attributes and their typical values found in the current corpus are given in the table below (Table F1).

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Typical Values/Formats Found in the XLIFF Corpus</th>
</tr>
</thead>
<tbody>
<tr>
<td>category</td>
<td>String, Enterprise Software, resourceBundle, Survey Project</td>
</tr>
<tr>
<td>charclass</td>
<td>String</td>
</tr>
<tr>
<td>company-name</td>
<td>String, universities, ids</td>
</tr>
<tr>
<td>contact-email</td>
<td>String, valid email addresses</td>
</tr>
<tr>
<td>context-type</td>
<td>database, linenumber, sourcefile, task, x-FontDefId, x-RegionDefId, x-application, x-attribute-name, x-authorization-scheme, x-breadcrumb, x-breadcrumb-entry, x-button, x-category, x-column, x-devicestate, x-dialogpath, x-flash-chart, x-interactive-report, x-item, x-linkdescription, x-linksToTextId, x-list, x-list-entry, x-list-of-values, x-lov-entry, x-navigation-bar-entry, x-object-name, x-object-type, x-page, x-page-process, x-page-validation, x-po-autocomment, x-po-trancomment, x-po-transcomment, x-popup-lov-template, x-region, x-region-definition, x-report-column, x-screenshot, x-section, x-series, x-shortcut, x-tab,</td>
</tr>
</tbody>
</table>

214
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Typical Values/Formats Found in the XLIFF Corpus</th>
</tr>
</thead>
<tbody>
<tr>
<td>x-text category</td>
<td></td>
</tr>
<tr>
<td>coord</td>
<td>-0;-0;-0;-0, 0;0;243;119</td>
</tr>
<tr>
<td>count</td>
<td>new, total, word count</td>
</tr>
<tr>
<td>css-style</td>
<td>String</td>
</tr>
<tr>
<td>ctype</td>
<td>bold, italic, lb, link, underlined, x-accelerator, x-c-param, x-font, x-fontFormat, x-html-a, x-html-b, x-link, x-newline, x-sup, x-superscript, x-tab, x-winformatmsgparameter</td>
</tr>
<tr>
<td>datatype</td>
<td>PLAINTEXT, cpp, cstring, database, html, javalistresourcebundle, javapropertyresourcebundle, plaintext, po, xhtml, xml, xsl, javaproperties, ICUResourceBundle, JSSkeleton, pofile, resource, resource message, resx, s2x, skeleton, text, winres, x-IGD-XML, x-STR#, x-application/json, x-application/x-gettext, x-application/xml+DTD, x-ICU-resourcebundle, x-infoXmlReference, x-localization, x-ms-word, x-office, x-sample, x-sdlfilterframework2, x-soy-msgbundle, x-test, x-text/html, x-text/xml, x-undefined, xliiff</td>
</tr>
<tr>
<td>href</td>
<td>#, /opt/Maxprograms/Swordfish/skl/Swordfish_manual/swordfish.odt.41660.skl,</td>
</tr>
<tr>
<td>Attribute</td>
<td>Typical Values/Formats Found in the XLIFF Corpus</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>C:</td>
<td>C:\fr_issue\scripts\browser_automation\en_US\Converted\about.html, E:\dotnet.goglobalnow.net\work\prdlg.rc, cv_skeleton.skl, file:///C:/MyFolder/MyProject/article.htm, <a href="http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32003L0098:EN:HTML">http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32003L0098:EN:HTML</a></td>
</tr>
<tr>
<td>id</td>
<td>Null, NMTOKEN, TestArray_2, DESTINATION, 850&lt;br&gt;Long sentences, sentences spanning multiple lines. e.g. A line OFFSET is a required '+' or '-' followed by a positive integer. After any flags comes an optional field width, as a decimal number; then an optional modifier, which is either E to use the locale's alternate representations if available, or 0 to use the locale's alternate numeric symbols if available.</td>
</tr>
<tr>
<td>match-optional</td>
<td>no</td>
</tr>
<tr>
<td>match-quality</td>
<td>100, 100%, 78.46, fuzzy, String, Guaranteed</td>
</tr>
<tr>
<td>maxbyte/maxheight</td>
<td>1, 30, NMTOKEN</td>
</tr>
<tr>
<td>maxwidth</td>
<td>-1, 100, NMTOKEN</td>
</tr>
<tr>
<td>Attribute</td>
<td>Typical Values/Formats Found in the XLIFF Corpus</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>mimetype</td>
<td>application, image, text/internal-tek</td>
</tr>
<tr>
<td>mtype</td>
<td>seg, protected, x-test</td>
</tr>
<tr>
<td>name</td>
<td>...(document.getElementById(_, SourceLanguageId, _ski_416103, o1615827</td>
</tr>
<tr>
<td>origin</td>
<td>Uploaded TMX File: Jade_FR_FR.zi:9:12, /Filesystem Projects/000_English/00957 HR, Policies/00957_000_Sep05_partA_Title&amp;TOC. doc, Gaya</td>
</tr>
<tr>
<td>original</td>
<td>null, /org/com/test.fil</td>
</tr>
<tr>
<td>phase-name</td>
<td>adhoc Export, Creation, NMTOKEN, PM review (4), Post-Editing (6), Project Coordinator Review (31), Quality Assurance, String, _ski_4838, create, pl, p2, prep1, rev1099, rev1290, rev1967, rev337, rev396, rev705, review, trans,</td>
</tr>
<tr>
<td>pos</td>
<td>open, close, after, before</td>
</tr>
<tr>
<td>process-name</td>
<td>!!!TEST_WF_BA, Extraction, String, Terminology Management, WorldServer Asset Export, authoring, commit, development, leverage, preparation, translation</td>
</tr>
<tr>
<td>product-name</td>
<td>nxliff, LocWorks Studio, String, View:LoggerControlView[us.esme] of Component:LoggerControl[us.esme], XalanC, delim, iManager, ice, ldaphdlr, ldif, mergetl, messages, mpRealityAdmin, ndscheck, schhdlr</td>
</tr>
<tr>
<td>Attribute</td>
<td>Typical Values/Formats Found in the XLIFF Corpus</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td>purpose</td>
<td>Location, Match, information, location, location x-test match x-abcdef, location x-test x-aaa match</td>
</tr>
<tr>
<td>reformat</td>
<td>yes, coord font</td>
</tr>
<tr>
<td>resname</td>
<td>null values, !Sepcials-test=&quot;&quot;&amp;&gt;</td>
</tr>
</tbody>
</table>
| restype    | Button, button, caption, combobox, comboboxitem, ctext, defpushbutton, dialog, dlginit, edit, , file, groupbox, icon, label, list, listbox, listitem, ltext, menu, menuitem, popupmenu, pushbutton, radio, rtext, separator, table, string, Static, accelerator, acceleratortable, array, bin, check, dialoginfo, dialoginit, dialoginitentry, editfieldimport, informationMessage, inientries, int, menuentry, menupopup, , radiobutton, rectangle, resourcegroup, static, stringtable, stringtableentry, summary, title, token, tokensgroup, version, versionstring, x-LABEL_KEY, x-Text, x-UI_MESSAGING_KEY, x-comboboxex32, x-dc:description, x-dc:subject, x-dc:title, x-dialogex, x-edittext, x-gettext-domain-header, x-gettext plurals, x-glb80, x-icu-array, x-icu-binary, x-icu-integer, x-icu-table, x-ingame-string-text, x-keyval, x-meta:keyword, x-meta:user-defined, x-misc, x-mru80, x-msctls_progress32, x-msctls_trackbar32, x-msctls_updown32, x-office:annotation, x-page, x-paragraph, x-
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Typical Values/Formats Found in the XLIFF Corpus</th>
</tr>
</thead>
<tbody>
<tr>
<td>pr_arrayclass.8.32, x-</td>
<td></td>
</tr>
<tr>
<td>pr_bitmapbutton.8.32, x-</td>
<td></td>
</tr>
<tr>
<td>pr_super_shadowbox.8.32, x-</td>
<td></td>
</tr>
<tr>
<td>pre, x-prglb, x-scrollbar, x-static, x-sysanimate32, x-sysdatetimepick32, x-syslistview32, x-systabcontrol32, x-systreeview32, x-text:h, x-text:index-title-template, x-text:note, x-text:p, x-toolbarwindow32, x-wined80, x-wpbmp80, x-wpbtn80, x-wpclr80, x-wpcnt80, x-wpcombo80, x-wpcombobox80, x-wpeditfield80, x-wpfne80, x-wplistbox80, x-wpnslb80, x-wppdowncombo80, x-wptrbox80, x-wpsbox80, x-wpsqf80, x-wpstatic80, x-xml, x-{a65535ab-129c-4f65-a423-647217c79ccc}, x-{d27cdb6e-ae6d-11cf-96b8-444553540000}</td>
<td></td>
</tr>
<tr>
<td>state</td>
<td>final, needs-review-110n, needs-review-translation, needs-translation, new, signed-off, translated, needs-review, updated, user:translated, x-reviewed</td>
</tr>
<tr>
<td>state-qualifier</td>
<td>exact-match</td>
</tr>
<tr>
<td>Attribute</td>
<td>Typical Values/Formats Found in the XLIFF Corpus</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>no, oc, pl, pt, pt-PT, ro, sl, tbd, th, tk, uz-UZ-Cyrl, uz-UZ-Latin, x-zz, zh-CHT, zh-CN, zh_CN</td>
<td></td>
</tr>
<tr>
<td>tool</td>
<td>Ektron, Idiom WorldServer 9.0.5, Idiom WorldServer 9.2.0, LKR, NAIL.LUI 1.1.0.26065, NAIL.LUI 1.5.0.28574, NAIL.LUI 1.6.0.21409, PASSOLO 3.0, Rainbow v2.00, Snap-On Ireland, CATFile_Translation Utility, String, TM Search, TM-ABC, genrb, manual</td>
</tr>
<tr>
<td>tool</td>
<td>Ektron, Maxprograms, Mindark PE AB, World Wide Web Consortium - Internationalization Tag Set Interest Group; OASIS XLIFF TC</td>
</tr>
<tr>
<td>tool-id</td>
<td>Ektron, IGD-2-XLIFF, JavaPM, P25CBF2C2, Swordfish, benten, blancoNLpackGenerator, genrb-3.3-icu-4.0, <a href="http://www.mindark.com/dbexport/1.0">http://www.mindark.com/dbexport/1.0</a>, nlpack, pleiades, prev, tbd, xliiff</td>
</tr>
<tr>
<td>tool-name</td>
<td>AgdaVS export tool, Benten, Ektron, IGD-2-XLIFF, ITS Translate Decorator, Maxprograms JavaPM, Okapi.Utilities.Set01, Pleiades, Swordfish III, blancoNLpackGenerator, genrb</td>
</tr>
<tr>
<td>translate</td>
<td>yes, no</td>
</tr>
<tr>
<td>unit</td>
<td>word</td>
</tr>
<tr>
<td>xml:lang</td>
<td>&quot;&quot;, DE, EN, EN-US, ENGLISH, FR-FR, GERMAN, PL, de, de-de, de_DE, en, en-GB, en-US, en-us, en_US, es, es-ES, fr, fr-</td>
</tr>
<tr>
<td>Attribute</td>
<td>Typical Values/Formats Found in the XLIFF Corpus</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>xml:space</td>
<td>default, preserve</td>
</tr>
<tr>
<td>date</td>
<td>04/02/2009 23:24:18</td>
</tr>
<tr>
<td></td>
<td>11/06/2008</td>
</tr>
<tr>
<td></td>
<td>2001-04-01T05:30:02</td>
</tr>
<tr>
<td></td>
<td>2006-11-24</td>
</tr>
<tr>
<td></td>
<td>2007-01-01</td>
</tr>
<tr>
<td></td>
<td>2010-03-16T21:58:27Z</td>
</tr>
<tr>
<td>match-quality</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td>78.46</td>
</tr>
<tr>
<td></td>
<td>fuzzy</td>
</tr>
<tr>
<td></td>
<td>String</td>
</tr>
<tr>
<td></td>
<td>Guaranteed</td>
</tr>
<tr>
<td>state</td>
<td>final, needs-review, needs-review-110n, needs-review-translation, needs-translation, new, signed-off, translated, updated, user:translated, x-reviewed</td>
</tr>
</tbody>
</table>

Table F1. Attributes and their Typical Values
Appendix G: XLIFF Usage Patterns Derived for Individual Organisations

Figure G1. Pattern Derived for Eclipse
Figure G2. Pattern Derived for Company A
Figure G3. Pattern Derived for Company B
Figure G4. Pattern Derived for Company C
Appendix H: LocConnect: System Architecture, Features, API and Installation Guide

H.1 Systems Architecture

This section describes the systems architecture of LocConnect in detail.

LocConnect is a web-based, client-server system where LocConnect acts as a server. Components attached to LocConnect are clients. The design is based on a three-tier architecture as depicted in Fig. H1. The implementation of the system is based on PHP and AJAX technologies.

![Figure H1. Three-Tier Architecture of LocConnect](image)

**User interface tier** - a client-based graphical user interface that runs on a standard web browser. The user interface provides facilities for project management, administration and tracking.
**Middle tier** - contains most of the logic and facilitates communication between the tiers. The middle tier mainly consists of a workflow engine and provides an open API with a common set of rules that define the connectivity of components and their input output (IO) operations. The components simply deal with this interface in the middle tier.

**Data Storage tier** - uses a relational database for the storage and searching of XLIFF and other resource data. The same database is used to store information about individual projects.

Important components of the systems architecture are described below.

**User Interface**

Web-based graphical user interfaces have been developed for:

- capturing project parameters during project creation;
- tracking projects (to display the current status of projects);
- post-editing translations;
- configuring the server and localising the interface of LocConnect.

During project creation\(^{49}\), a web-based form is presented to a user (see Fig. H2). This form contains fields that are required by the Workflow Recommender\(^{50}\) to generate a workflow. Parameters entered through this interface will be stored in an XLIFF file along with an uploaded source file (or source text) and resource files. The project is assigned a unique ID through this interface and this ID is used throughout the project’s lifecycle.

\(^{49}\) Note that LocConnect project’s can also be created through an API call (see Sect. H.3.6) by-passing the UI and the web-form.

\(^{50}\) Workflow Recommender is a separate component developed by a fellow CNGL doctoral researcher, as a part of his PhD.
The project-tracking interface reflects the project’s workflow. It shows the current status of a project: *pending*, *processing*, or *complete* in relation to each component. It displays any feedback messages (such as errors, warnings) from components. The current workflow is shown in a graphical representation. Another important feature is a log of activities for the project. Changes to the XLIFF file (the changes of metadata) during different stages of the workflow can be tracked. The project-tracking interface uses AJAX technologies to dynamically update its content frequently (see Fig. H3).
At the end of a project’s lifecycle, the user is given the option to post-edit its content using the built-in XLIFF post-editor interface. It displays source strings, translations, alternative translations and associated metadata. Translations can be edited through this interface. The Post-editing component also uses AJAX to update XLIFF files in the main datastore. See Fig. H4 for a screenshot of the post-editing interface. A preliminary target file preview mechanism has also been developed and integrated into the same UI.
A password-protected interface has been provided for the configuration of the LocConnect server. Through this interface various configuration options such as LocConnect database path, component descriptions can be edited. The same interface can be used to localise the LocConnect server itself (see Fig. H5 for a screenshot of the administrator’s interface).
Figure H5. Administrator’s Interface

The user interfaces were implemented in PHP, Javascript, XHTML and use the Jquery library for graphical effects and dynamic content updates.

Middle tier: Application Programming Interface (API)

The LocConnect server implements a Representational State Transfer (REST) (Fielding 2000) based Application Programming Interface (API) to send and retrieve resources, localisation data and metadata between components through HTTP-GET and HTTP-POST operations using proper Uniform Resource Identifiers (URI). These resources include:

- localisation projects;
- XLIFF files;
- resource files (i.e. files such as TBX, TMX, SRX);
- resource metadata (metadata to describe resource file content).
The LocConnect API provides functions for the following tasks:

- creating a LocConnect job (new_project method);
- retrieving a list of jobs pending for a particular component (list_jobs method);
- retrieving an XLIFF file corresponding to a particular job (get_job method);
- setting the status of a job. The status can be one of the following: pending, processing, or complete (set_status method);
- sending a feedback message to the server (send_feedback method);
- sending processed XLIFF files to the server (send_output method);
- sending a resource file (e.g. a non-XLIFF asset file) to the server (send_resource method);
- retrieving a resource file from the server (get_resource method);
- retrieving metadata associated with a resource file (get_metadata method).

A complete description of each REST-based function is provided below.

Creating LocConnect jobs (programmatically): new_project method

The alternative method for creating LocConnect jobs is by using its interface (see Sect. H.3.6). This method takes four arguments: component ID, project name, project description and data. It will create a LocConnect project and return an XML with a LocConnect job ID assigned to the project. The IDs are alphanumeric and consist of 10 characters. The component ID is a string (usually, a short form of a component’s name, such as WFR for Workflow Recommender). Data is usually assumed to be in XLIFF format, however other UTF-8 encoded text based resources can also be used.

This method uses the HTTP POST method to communicate with components.

Obtaining available jobs: list_jobs method

This method takes a single argument: component ID. It will return an XML containing the IDs of jobs pending for any given component. See example below.
This method uses the HTTP GET method to communicate with the LocConnect server.

**Retrieving the XLIFF file corresponding to a particular job: get_job method**

This method takes two arguments: component ID and job ID. It will return a file corresponding to the given job ID and component ID. Usually, the file is an XLIFF file, however it can be any text-based file. Therefore, the returned content is always enclosed within a special XML mark-up: `<content>..</content>`. The XML declaration of the returned file will be omitted in the output (i.e. `<?xml version="1.0"?>` will be stripped off from the output).

```xml
<content>
<xliff version="1.2" xmlns="urn:oasis:names:tc:xliff:document:1.2">
  <file original="hello.txt" source-language="en" target-language="fr" datatype="plaintext">
    <body>
      <trans-unit id="hi">
        <source>Hello world</source>
        <target>Bonjour le monde</target>
      </trans-unit>
    </body>
  </file>
</xliff>
</content>
```

This method uses the HTTP GET method to communicate with the LocConnect server.

**Setting current status: set_status method**

This method takes three arguments: component ID, job ID, status. The status can be pending, processing or complete. Initially, the status of a job is set to pending by the LocConnect server to mark that a job is available for pick up by a certain component.
Once the job is picked by the component, it will change the status of the job to *processing*. This ensures that the same job will not be re-allocated to the component. Once the status of a job is set to *complete*, LocConnect will perform the next action specified in the workflow.

This method uses the HTTP GET method to communicate with the LocConnect server.

**Sending feedback message: send_feedback method**

This method takes three arguments: component ID; job ID; feedback message. Components can send various messages (for example error messages and notifications) to the server through this method. These messages will be instantly displayed in the relevant job tracking page of the LocConnect interface. The last feedback message sent to the LocConnect server before sending the output file will be stored within the LocConnect server and it will appear in the activity log of the job. The messages are restricted to 256 words in length.

This method uses the HTTP GET method to communicate with the LocConnect server.

**Sending a processed XLIFF file: send_output method**

This method takes three arguments: component ID; job ID; and content. The content is usually a processed XLIFF file. Once the content is received by LocConnect, it will be stored within the LocConnect datastore. LocConnect will wait for the component to set the status of the job to *complete* and move on to the next step of the workflow.

This method uses the HTTP POST method to communicate with the LocConnect server.
**Storing a resource file: send_resource method**

This method takes one optional argument: resource ID and two mandatory arguments: resource file and metadata description. The resource file should be in text format. Metadata has to be specified using the following notation:

**Metadata notation:** 'key1:value1-key2:value2-key3:value3'

e.g. 'language:en-domain:health'

If the optional argument resource ID is not given, LocConnect will generate an ID and assign that ID to the resource file. If the resource ID is given, it will overwrite the current resource file and metadata with the new resource file and metadata.

This method uses the HTTP POST method to communicate with the LocConnect server.

**Retrieving a stored resource file: get_resource method**

This method takes one argument: resource ID. Given the resource ID, the LocConnect server will return the resource associated with the given ID.

This method uses the HTTP GET method to communicate with the LocConnect server.

**Retrieving metadata associated with a resource file: get_metadata method**

This method takes one argument: resource ID. The LocConnect server will return the metadata associated with the given resource ID as shown in the example below:

```xml
<metadata>
    <meta key="language" value="en"/>
    <meta key="domain" value="health"/>
</metadata>
```
This method uses the HTTP GET method to communicate with the LocConnect server.

Implementation details of the LocConnect’s API functions are given in Sect. H.3.

**Component-Server Communication Process**

A typical LocConnect component-server communication process includes the following phases.

**Step 1: list_jobs**

This component calls the list_jobs method to retrieve a list of available jobs for that component by specifying its ID.

**Step 2: get_job**

This component uses get_job to retrieve the XLIFF file corresponding to the given job ID and the component ID.

A component may either process one job at a time or many jobs at once. However, the get_job method is only capable of returning a single XLIFF file at a time.

**Step 3: set_status – Set status to processing**

This component sets the status of the selected job to *processing*.

**Step 4: Process file**

This component processes the retrieved XLIFF file. It may send feedback messages to the server while processing the XLIFF file. These feedback messages will be displayed in the job tracking interface of the LocConnect.
**Step 5: send_output**

This component sends the processed XLIFF file back to the LocConnect server using `send_output` method.

**Step 6: set_status**

This component sets the status of the selected job to `complete`. This will trigger the LocConnect server to move to the next stage of the workflow.

**Middle tier: Workflow Engine**

A simple workflow engine has been developed and incorporated into the LocConnect server to allow for the management and monitoring of individual localisation jobs. The current workflow engine does not support parallel processes or branching. However, it allows the same component to be used several times in a workflow. The engine parses the workflow information found in the XLIFF data container (see Sect. 5.4) and stores the workflow information in the project management datastore. The project management datastore is then used to keep track of individual projects. In the current setup, setting the status of a component to `complete` will trigger the next action of the workflow.

**LocConnect Datastore**

The database design can be logically stratified in 3 layers:

- main datastore holds XLIFF files;
- project management datastore holds data about individual projects and their status;
- resource datastore holds data and metadata about other resource files;
- the main datastore is used to store XLIFF files corresponding to different jobs. It stores different versions of the XLIFF file that correspond to a particular job.
Therefore, the LocConnect server also acts as a revision control system for localisation projects.

The project management datastore is used for storing the information necessary to keep track of individual localisation jobs with respect to localisation workflows. Furthermore, it is used to store various time-stamps such as job pick-up time, job completion time, by different components.

The resource datastore is used to store various asset files associated with localisation projects. The asset files can be of any text-based file format such as TMX, XLIFF, SRX, TBX, XML. The components can store any intermediate files, temporary or backup files in this datastore. The files can then be accessed at any stage during workflow execution. The resource files (i.e. asset files) can be described further using metadata. The metadata consists of key-value pairs associated with the resource files and can also be stored in the resource datastore.

SQLite was chosen as the default database for implementing the logical data structure in this prototype, for a number of reasons. Firstly, it can be easily deployed. It is lightweight and virtually no administration is required. Furthermore, it does not require any configuration.

LocConnect’s installation and configuration guide is given in Sect. H.4.
H.2 Features of LocConnect

H.2.1 Key Features

Key features of the LocConnect interoperability test-bed are summarised below.

**Common XLIFF based Data Layer and Application Programming Interface**

LocConnect implements a common XLIFF-based data store corresponding to individual localisation projects. The components can access this datastore through a simple API. Furthermore, the common datastore can also hold various supplementary resource files related to a localisation project. Components can manipulate these resource files through the API.

**Workflow Engine**

The orchestration of components is achieved via an integrated workflow engine that executes a localisation workflow generated by another component.

**Live User Interface (UI)**

One of the important aspects of a distributed processing scenario is the ability to track progress along the different components. An AJAX-powered UI has been developed to display the status of the components in real-time. Moreover, LocConnect’s UI has been developed in a manner that allows it to be easily localised into other languages.

**Built-in post-editing component (XLIFF editor)**

In the present architecture, localisation project creation and completion happens within LocConnect. In a typical localisation workflow, the last step is to post-edit content. Therefore, an online XLIFF editor was developed and incorporated into LocConnect in order to facilitate post-editing of content after project completion.
Component Simulator

In the current experimental setup, only a small number of components, most of them developed as part of the CNGL research at the University of Limerick and other participating research groups, have been connected up. The Workflow Recommender, Mapper, Leveraging Component and a Translating Rating component are among these. A component simulator was, therefore, developed to allow for further testing of interoperability issues in an automated localisation workflow using the LocConnect framework.

H.2.2 Business Case

While the LocConnect environment supports the ad-hoc connection of localisation components, it can also serve as cloud-based storage for localisation projects. These and other key advantages of LocConnect from a business point of view are highlighted below.

Cloud-based XLIFF and Resource File Storage

LocConnect can simply be used as a cloud-based XLIFF storage. Moreover, due to its ability to store resource files (e.g. TMX, SRX), it can be used as a repository for localisation project files. As such, LocConnect offers a central localisation data repository which is easy to backup and maintain.

Revision Control System

During a project’s life cycle, the associated XLIFF data container continuously changes as it travels through different localisation components. LocConnect keeps track of these changes and stores different versions of the XLIFF data container. Therefore, LocConnect acts as a revision control system for localisation projects. LocConnect provides the facility to view both data and metadata associated with the data container at different stages of a workflow.
**In-built Online XLIFF editor**

Using the inbuilt online XLIFF editor, users can post-edit XLIFF content easily. The AJAX-based UI allows easy inline editing of content. Furthermore, the online editor shows alternative translations as well as useful metadata associated with each translation unit.

**Access via Internet or Intranet**

With its single click installer, it can easily be deployed via the internet or an intranet. LocConnect can also act as a gateway application where LocConnect is connected to the internet while the components can safely reside within an intranet.

**Enhanced Revenues**

The LocConnect-centric architecture increases data exchange efficiency as well as automation. Due to increased automation, we would expect lower localisation costs and increased productivity. This promotes data recycling by allowing to reuse previous XLIFF data containers.
### H.3 Application Programming Interface (API)

#### Description

**H.3.1 Get Job Method**

<table>
<thead>
<tr>
<th>Function</th>
<th>GetJob(id, com)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Fetch a file corresponding to a particular job ID</td>
</tr>
<tr>
<td>Base URL</td>
<td>locConnect/get_job.php</td>
</tr>
<tr>
<td>(HTTP Request)</td>
<td></td>
</tr>
<tr>
<td>Query String</td>
<td>id = job ID</td>
</tr>
<tr>
<td>Parameters</td>
<td>* id is an ascii based alphanumeric string (max 15 chars, lowercase)</td>
</tr>
<tr>
<td></td>
<td>com = component</td>
</tr>
<tr>
<td></td>
<td>* com should be uppercase ascii string (max chars 6)</td>
</tr>
<tr>
<td>Example URL</td>
<td>locConnect/get_Job.php?id=880a0796e7&amp;com=LMC</td>
</tr>
<tr>
<td>Response Format</td>
<td>&lt;content&gt;.. (file content goes here)</td>
</tr>
<tr>
<td>(XML)</td>
<td></td>
</tr>
</tbody>
</table>
| Example Response: | <content>  
|                   |   <xliff version="1.1">
|                   |     <file original="NoName" source-language="en" datatype="plaintext"> |
|                   |       <body> <trans-unit approved="yes" id="asciiopt ">
|                   |         <source>Fields</source>
|                   |       </trans-unit>
|                   |     </body> </file> </xliff>
|                   |   </content>             |
| Error Response:   | <error>
<p>|                   |   &lt;msg&gt;                  |</p>
<table>
<thead>
<tr>
<th>Component</th>
<th>Either component:&lt;com&gt; not found or job:&lt;job id&gt; is not available for this component</th>
</tr>
</thead>
</table>
| PHP Example | ```
<?php
header("Content-Type:text/html; charset=utf-8");
require_once 'HTTP\Request2.php'; // uses Pear
$request = new HTTP_Request2('http://localisation.ie/locConnect/get_job.php', HTTP_Request2::METHOD_GET);
$request->setHeader('Accept-Charset', 'utf-8');
$url = $request->getUrl();
$url->setQueryVariable('id', '7985aecae0'); //set the job id here
$url->setQueryVariable('com', 'LCM'); // set your component name here
// This will fetch the given job from the CNLF server and store content in $file variable;
$file=$request->send()->getBody();
echo $file;
?>
``` |
| Notes | getJob(id, com) returns a utf-8 encoded XML response containing the file to be processed.
xml declaration of the content will be omitted. (i.e.
<table>
<thead>
<tr>
<th>XML</th>
<th><code>&lt;?xml version=&quot;1.0&quot; ..?&gt;</code> will be stripped off in the output.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>Uses GET Method</td>
</tr>
</tbody>
</table>
### H.3.2 Fetch Job Method

<table>
<thead>
<tr>
<th>Function</th>
<th>fetchJob(com)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Get a list of available jobs for a particular component</td>
</tr>
<tr>
<td>Base URL (HTTP Request)</td>
<td>locConnect/fetch_job.php</td>
</tr>
<tr>
<td>Query String Parameters</td>
<td>com = component</td>
</tr>
<tr>
<td></td>
<td>* com should be uppercase ascii string (max chars 6)</td>
</tr>
<tr>
<td>Example URL</td>
<td>locConnect/fetch_job.php?com=WFR</td>
</tr>
</tbody>
</table>
| Response Format (XML) | <jobs>
|             |     <job> jobid </job>
|             | </jobs> |
| Example Response: | <jobs>
|             |     <job> 16674f2698</job>
|             |     <job>633612fb37 </job>
|             | </jobs> |
| Error Response: | <error>
|             |     <msg>
|             | Either component:<com> not found or not pending jobs available for this component |
|             | </msg>
|             | </error> |
| Client Implementation Example (in php) | <?php
|             | header ("Content-Type:text/html;
|             | charset=utf-8");
|             | require_once 'HTTP\Request2.php'; // |

245
uses Pear

$request = new HTTP_Request2('http://localisation.ie/locConnect/fetch_job.php', HTTP_Request2::METHOD_GET);
$request->setHeader('Accept-Charset', 'utf-8');
$url = $request->getUrl();
$url->setQueryVariable('com', 'WFR'); // set your component name here
// This will get a list of pending jobs from the CNLF server and store them
.jobs variable;
.jobs=$request->send()->getBody();
echo $jobs;
?>

Notes:

fetchJob(com) returns a utf-8 encoded XML response containing a list of pending jobs for a particular component
xml declaration of the content is omitted. (i.e. <?xml version="1.0"..?> will be stripped off in the output.
Uses GET Method
H.3.3 Set Job Status Method

<table>
<thead>
<tr>
<th>Function</th>
<th>setJobStatus(id, com, msg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Sets the job status of a particular component</td>
</tr>
<tr>
<td>Base URL</td>
<td>locConnect/set_status.php</td>
</tr>
<tr>
<td>(HTTP Request)</td>
<td></td>
</tr>
<tr>
<td>Query String</td>
<td>com = component</td>
</tr>
<tr>
<td>Parameters</td>
<td>* com should be uppercase ascii string (max chars 6)</td>
</tr>
<tr>
<td></td>
<td>id = job ID</td>
</tr>
<tr>
<td></td>
<td>* id is an ascii based alphanumeric string (max 15 chars, lowercase)</td>
</tr>
<tr>
<td></td>
<td>msg = status</td>
</tr>
<tr>
<td></td>
<td>* msg should be one of: complete, pending or processing status</td>
</tr>
<tr>
<td>Example URL</td>
<td>locConnect/set_status.php?id=16674f2698&amp;com=WFR&amp;msg=pending</td>
</tr>
<tr>
<td>Response Format</td>
<td>&lt;response&gt;</td>
</tr>
<tr>
<td>(XML)</td>
<td>&lt;msg&gt;Status Updated&lt;/msg&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/response&gt;</td>
</tr>
<tr>
<td>Error Response:</td>
<td>&lt;error&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;msg&gt;</td>
</tr>
<tr>
<td></td>
<td>Status was not updated. Ensure job id:&lt;jobid&gt; is correct and component:&lt;com&gt; has been assigned that job.</td>
</tr>
<tr>
<td></td>
<td>&lt;/msg&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/error&gt;</td>
</tr>
<tr>
<td>Client Implementation</td>
<td>&lt;?php</td>
</tr>
</tbody>
</table>
Example (in php)

```php
header("Content-Type:text/html; charset=utf-8");
require_once 'HTTP\Request2.php'; // uses Pear
$request = new HTTP_Request2('http://localisation.ie/locConnect/set_status.php', HTTP_Request2::METHOD_GET);
$request->setHeader('Accept-Charset', 'utf-8');
$url = $request->getUrl();
$url->setQueryVariable('com', 'WFR'); // set your component name here
$url->setQueryVariable('id', '16674f2698'); // set job id here
$url->setQueryVariable('msg', 'pending'); // set status id here

// This will get the server response
$response=$request->send()->getBody();
echo $response;
?>
```

Notes:

xml declaration of the content is omitted. (i.e. `<?xml version="1.0"?>` will be stripped off in the output.

Uses GET Method
### H.3.4 Send Feedback Method

<table>
<thead>
<tr>
<th>Function</th>
<th><code>sendFeedback(id, com, msg)</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Send feedback of a particular component corresponding to a specific job</td>
</tr>
<tr>
<td>Base URL</td>
<td><code>locConnect/set_status.php</code></td>
</tr>
<tr>
<td>(HTTP Request)</td>
<td></td>
</tr>
<tr>
<td>Query String</td>
<td></td>
</tr>
<tr>
<td>Parameters</td>
<td></td>
</tr>
<tr>
<td><code>com</code> = component</td>
<td></td>
</tr>
<tr>
<td>* com should be uppercase ascii string (max chars 6)</td>
<td></td>
</tr>
<tr>
<td><code>id</code> = job ID</td>
<td></td>
</tr>
<tr>
<td>* id is an ascii based alphanumeric string (max 15 chars, lowercase)</td>
<td></td>
</tr>
<tr>
<td><code>msg</code> = feedback</td>
<td></td>
</tr>
<tr>
<td>* feedback should be <strong>url encoded string</strong> containing not more than 250 (utf-8) characters.</td>
<td></td>
</tr>
<tr>
<td>Example URL</td>
<td><code>send_feedback.php?id=16674f2698&amp;com=WFR&amp;msg=Globalsigt workflow generated</code></td>
</tr>
<tr>
<td>Response Format (XML)</td>
<td></td>
</tr>
<tr>
<td><code>&lt;response&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;msg&gt;</code> Feedback Updated<code>&lt;/msg&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/response&gt;</code></td>
<td></td>
</tr>
<tr>
<td>Error Response:</td>
<td></td>
</tr>
<tr>
<td><code>&lt;error&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;msg&gt;</code></td>
<td></td>
</tr>
<tr>
<td>Feedback was not updated. Ensure job id:<code>&lt;jobid&gt;</code> is correct and component:<code>&lt;com&gt;</code> has been assigned that job.</td>
<td></td>
</tr>
<tr>
<td><code>&lt;/msg&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/error&gt;</code></td>
<td></td>
</tr>
</tbody>
</table>
| Client Implementation Example (in php) | <?php
code

header ("Content-Type:text/html; charset=utf-8");
require_once 'HTTP\Request2.php'; // uses Pear
$request = new HTTP_Request2('http://localisation.ie/LocConnect/send_feedback.php', HTTP_Request2::METHOD_GET);
$request->setHeader ('Accept-Charset', 'utf-8');
$url = $request->getUrl();
$url->setQueryVariable ('com', 'WFR'); // set your component name here
$url->setQueryVariable ('id', '16674f2698'); // set job id here
$url->setQueryVariable ('msg', 'Globalsigt workflow generated'); // set your component’s feedback here

// This will get the server response
$response=$request->send()->getBody();
echo $response;
?>

| Notes: | xml declaration of the content is omitted. (i.e. <?xml version="1.0"..?> will be stripped off in the output. Uses GET Method |
### H.3.5 Send Output Method

<table>
<thead>
<tr>
<th>Function</th>
<th>sendOutput(id, com, data)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Send output of a particular component corresponding to a specific job</td>
</tr>
<tr>
<td>Base URL</td>
<td>locConnect/send_output.php</td>
</tr>
</tbody>
</table>
| Query String Parameters | com = component  
* com should be uppercase ascii string (max chars 6)  
id = job ID  
* id is an ascii based alphanumeric string (max 15 chars, lowercase)  
data = file content  
* file content can be either xml or text content (in utf-8 encoding). |
| Response Format (XML) | <response><msg>Output Accepted</msg></response> |
| Error Response: | <error><msg>
Output was not updated. Ensure job id: <jobid> is correct and component: <com> has been assigned that job.
</msg></error> |
| Client Implementation Example (in php) | `<?php
header("Content-Type:text/xml");
require_once 'HTTP\Request2.php'; //uses PEAR
$request = new
` |
HTTP_Request2('http://localisatin.ie/localConnect/send_output.php');
$data = "<methodCall>
  
  <methodName>foo.bar</methodName>
  
  <params>
    
    <param>
      <value><string>Hello, world!</string></value>
    </param>
    
    <param>
      <value><int>42</int></value>
    </param>
  </params>

</methodCall>";
$request->setMethod(HTTP_Request2::METHOD_POST)
  ->addPostParameter('id', '880a0796e7')
  ->addPostParameter('com', 'LKR')
  ->addPostParameter('data', $data);

try {
    $response = $request->send();
    if (200 == $response->getStatus()) {
        echo $response->getBody();
    }
} else {
    echo 'Unexpected HTTP status: ' . $response->getStatus() . ' ' .
    $response->getReasonPhrase();
}

} catch (HTTP_Request2_Exception $e) {
    echo 'Error: ' . $e->getMessage();
}

Notes:
xml declaration of the content is omitted. (i.e. <?xml
version="1.0"?> will be stripped off.

**Uses POST Method**

Output can contain utf-8 encoded strings.
### H.3.6 New Project

<table>
<thead>
<tr>
<th>Function</th>
<th>createNewProject(com, pname, pdesc, data)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Create a new locConnect job via direct API call</td>
</tr>
<tr>
<td>Base URL</td>
<td>locConnect/new_project.php</td>
</tr>
</tbody>
</table>
| Query String Parameters | com = component (mandatory)  
* com should be uppercase ascii string (max chars 6)  

  | pname = project name  
* pdesc=project description (optional)  

  | data = file content (mandatory)  
* XLIFF content assumed, although other text based contents are supported. (in utf-8 encoding). |
| Response Format   | <response><msg>124D1334E</msg>  
</response> |
| Error Response:   | <error><msg>(error)</msg></error> |
| Client Implementation Example (in php) | <?php  
header ("Content-Type:text/xml");  
require_once 'HTTPRequest2.php';  
//uses PEAR  

$request = new HTTP_Request2('http://localisatin.ie/locConnect/new_project.php');  
$data="<methodCall>
"  
"  
<methodName>foo.bar</methodName>
"  
"  
</methodCall>"; |
"<params>
  <param><value><string>Hello, world!</string></value></param>
  <param><value><int>42</int></value></param>
</params>"

$request->setMethod(HTTP_Request2::METHOD_POST)
  ->addPostParameter('pname', 'New Project')
  ->addPostParameter('com', 'MYCOM')
  ->addPostParameter('data', $data);

try {
  $response = $request->send();
  if (200 == $response->getStatus()) {
    echo $response->getBody();
  } else {
    echo 'Unexpected HTTP status: ' . $response->getStatus() . ' ' . $response->getReasonPhrase();
  }
} catch (HTTP_Request2_Exception $e) {
  echo 'Error: ' . $e->getMessage();
}
xml declaration of the content is omitted. (i.e. `<xml version="1.0" ..?>` will be stripped off.

**Uses POST Method**
# H.3.7 Send Resource

<table>
<thead>
<tr>
<th>Function</th>
<th>sendResource(type, data, metadata, desc)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>storing supplementary files (i.e. resources) in LocConnect</td>
</tr>
<tr>
<td>Base URL (HTTP Request)</td>
<td>locConnect/send_resource.php</td>
</tr>
<tr>
<td>Query String Parameters</td>
<td>type: lmc, tmx, xli ff (identifier for the resource) e.g. file type/extension</td>
</tr>
<tr>
<td></td>
<td>metadata: metadata to describe the resource in the following format: &quot;attribute1:value1-attribute1:value2-attribute3:value3&quot; (i.e. a list of attribute : value pairs in a single string separated by hyphens)</td>
</tr>
<tr>
<td></td>
<td>desc: description about the resource/files</td>
</tr>
<tr>
<td></td>
<td>data = file content</td>
</tr>
<tr>
<td>Response Format (XML)</td>
<td>&lt;response&gt;&lt;msg&gt;(resource ID)&lt;/msg&gt;&lt;/response&gt;</td>
</tr>
<tr>
<td>Error Response:</td>
<td>&lt;error&gt;&lt;msg&gt;(error)&lt;/msg&gt;&lt;/error&gt;</td>
</tr>
</tbody>
</table>
| Client Implementation Example (in php) | ```php
<?php
header("Content-Type:text/xml");
require_once 'HTTP\Request2.php';
// uses PEAR

$request = new HTTP_Request2('http://localisatin.ie/locConnect/send_resource.php');
$data="<methodCall>
" .
"<methodName>foo.bar</methodName>
";
``` |
"<params>
  
  </params>

"</methodCall>";

$request->setMethod(HTTP_Request2::METHOD_POST)
    ->addPostParameter('desc', 'resource desc')
    ->addPostParameter('metadata', 'domain:test')
    ->addPostParameter('data', $data);

try {
    $response = $request->send();
    if (200 == $response->getStatusCode()) {
        echo $response->getBody();
    } else {
        echo 'Unexpected HTTP status: ' . $response->getStatusCode() . ' ' . $response->getReasonPhrase();
    }

catch (HTTP_Request2_Exception $e) {
    echo 'Error: ' . $e->getMessage();
}
?>

Notes:
If you specify an ID then the sending resource will overwrite the current resource with the specified ID (this allows components to update an existing resource). Otherwise if you leave it blank (i.e. ""), it will generate a resource ID, assign it to the new resource and send the ID to you in the response. The format of the ID is same as LocConnect job IDs.

Uses POST Method
**H.3.8 Get Resource**

<table>
<thead>
<tr>
<th>Function</th>
<th>getResource(id)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Retrieving supplementary files (i.e. resources) from LocConnect</td>
</tr>
<tr>
<td>Base URL</td>
<td>locConnect/get_resource.php</td>
</tr>
<tr>
<td>Query String Parameters</td>
<td>Id: resource identifier</td>
</tr>
<tr>
<td>Response Format (XML)</td>
<td><code>&lt;response&gt;..&lt;/response&gt;</code></td>
</tr>
<tr>
<td>Error Response:</td>
<td><code>&lt;error&gt;&lt;msg&gt;(error)&lt;/msg&gt;&lt;/error&gt;</code></td>
</tr>
</tbody>
</table>

**Client Implementation Example (in php)**

```php
<?php
header("Content-Type:text/html; charset=utf-8");
require_once 'HTTP\Request2.php'; // uses Pear
$request = new HTTP_Request2('http://localisation.ie/locConnect/get_resource.php', HTTP_Request2::METHOD_GET);
$request->setHeader('Accept-Charset', 'utf-8');
$url = $request->getUrl();
$url->setQueryVariable('id', '16674f2698'); // set resource id here
// This will get the server response
$response=$request->send()->getBody();
echo $response;
?>
```

**Notes:** Uses GET Method
H.4 LocConnect Installation and Configuration Guide

H.4.1 Installation Requirements

- A web server running PHP (5.2.13 or higher) with PEAR, PDO and PDO SQLITE extensions ENABLED.
  - Apache 2.2 or higher is recommended for the web server.
  - PHP installation instructions available at: www.php.org

H.4.2 Recommended Settings

Recommended PHP Settings

max_execution_time = 240
max_input_time = 60
memory_limit = 256M
display_errors = Off
post_max_size = 20M
magic_quotes_gpc = Off
magic_quotes_runtime = Off
magic_quotes_sybase = Off
default_mimetype = "text/html"
default_charset = "utf-8"
file_uploads = On
upload_max_filesize = 20M
extension=phppdo.dll
extension=phppdo_sqlite.dll
extension=phpsqlite.dll
extension=phpxmlrpc.dll
extension=php_xsl.dll
extension=php_zip.dll

Recommended Apache Settings
EnableMMAP off
EnableSendfile off
Win32DisableAcceptEx

Folder and File Permissions
Inet user / Apache user should have full control over (i.e. including write permission) following files and folders:
files: conf.php and conf-back.php
folders:
1) LocConnect installation folder
2) LocConnect upload folder (See Sect. H.4.3 ‘Installation’ below)

H.4.3 Installation
Step 1: Create a folder to store uploaded files in the server. This folder is known as ‘upload’ folder in this document. Grant ‘read/write’ permissions to this folder.
Step 2: Copy/Move the iLocConnect-<version>-<language>.php into the desired folder within the web root.
Step 3: Execute iLocConnect-<version>-<language>.php through the browser. (e.g. iLocConnect-v27-en.php)
Step 4: Delete iLocConnect-<version>-<language>.php script from the web server.
Step 5: Configure LocConnect by editing necessary parameters in the conf.php (see Configuration section).
This can also be carried out by log-in onto the Admin panel (request Admin password from author).
Edit the BASE_UPLOAD_PATH to include the correct path to the folder created in step one. Usually the BASE_UPLOAD_PATH is the first line of the conf.php.

Step 6: Launch LocConnect by visiting index.php.

H.4.4 Configuration

Step 1: Open conf.php in a text editor.

Step 2: Configure LocConnect Database path.

e.g. define('BASE_DB_URL', './');

e.g. define('BASE_DB_URL', 'http://www.yourdomain.com/db/en/');

Step 3: Configure components (See Adding New Components section).

Step 4: Set-up language packs and respective directories to store them. Modify $languages array to include installed language packs (see section on Language Pack Installation).

$languages = array(
    "en" => ".",
    "es" => "/es" );

The two letter language code and the directory of language pack have to be specified in the above manner.

H.4.5 Adding New Components

Step 1: Open conf.php in a text editor.

Step 2: Enter component short and long names in the $arr variable as given below.

Example:

$arr = array(
    "LKR" => "Localisation Knowledge Repository",
    "WFR" => "Workflow Recommender",

263
"LMC" => "Localisation Memory Container",
"RT" => "Translation Rating",
"MT" => "Machine Translation" );

Step 3: Include three 100x101 (px) png images and name them in the following convention:

<component-short-name>-red.png
<component-short-name>-grey.png
<component-short-name>-green.png

Example:

WFR-red.png, WFR-grey.png, WFR-green.png

### H.4.6 Adding New Language Packs

Step 1: Create a folder with two letter language code

Step 2: Install relevant LocConnect language pack into the created folder (e.g. es) - see LocConnect installation instructions

Step 3: Configure conf.php (especially $language parameter).

### H.4.7 Security

Restrict access to Sqlite database through .htaccess, by adding following lines:

Do this for all language packs.

<files dbTemp.sqlite>
order allow,deny
deny from all
</files>
Appendix I: Publications, Presentations, Posters and Achievements

I.1 Publication from PhD Research

I.1.1 Invention Disclosures


I.1.2 Book Chapters


I.1.3 Peer-Reviewed Journal Papers


I.1.4 Conference Proceedings and Presentations


265


I.1.5 Demonstrations and Posters


in *CNGL Localisation Innovation Showcase*, Microsoft European Development Centre, Dublin, Ireland, 10 Nov, (Live Demo).


### I.1.6 Journal Papers in Preparation


### I.2 Other Related Publications during PhD Studies

#### I.2.1 Book Chapters


#### I.2.2 Peer Reviewed Journal Papers

I.2.3 Conference Proceedings and Presentations


I.2.4 Achievements


- Won the LRC Best Scholar Award in 2009, Localisation Research Centre, University of Limerick, Ireland.

- Voting Member, XLIFF Technical Committee (2012 – 2013).


- Member of the Programming Committee of the 4th XLIFF Symposium, 2013.