Why XLIFF and Why XLIFF 2?

David Filip
ADAPT Centre, O'Reilly Institute
Trinity College Dublin, University of Dublin
Dublin, Ireland
david.filip@adaptcentre.ie

Abstract

This is to inform the business and decision making communities among the ASLING audience about the high level benefits of bitext and XLIFF 2. Translator and Engineering communities will also benefit, as they need the high level arguments to make the call for XLIFF 2 adoption in their organizations.
We start with a conceptual outline what bitext is, what different sorts of bitext exist and how they are useful at various stages in various industry processes, such as translation, localisation, terminology management, quality and sanity assurance projects etc. Examples of projects NOT based on bitext are given, benefits and drawbacks compared on a practical level of tasks performed. The following is demonstrated: That bitext management is a core process for efficient multilingual content value chains; That usage of an open standard bitext creates a greater sum of good than usage of proprietary bitext formats; and finally: That XLIFF 2 is the core format and data model to base bitext management on.

1 Introduction

XLIFF can be expanded as XML Localization Interchange Format, as the specifications are being developed in US English and the XLIFF TC should expand to (OASIS) XML Localisation Interchange File Format, as the Committee was originally convened and named by a group of Ireland based translation buyers and tool makers (XML can be expanded as eXtensible markup language).
Let us start with the statement that ***XLIFF is the only open standard bitext format*** and actually this whole paper will be just an explanation of that statement and why that matters. To fully explain the above statement we must define and discuss the notions of 1) a bitext format and 2) its management, 3) open standard and finally 4) that there is no other such format and that 5) this is good. In the following, XLIFF means XLIFF Version 2.0, the published OASIS Standard (Comerford et al., 2014a), unless XLIFF 1.2 (Savourel et al., 2008), the OASIS standard superseded by XLIFF Version 2.0, is specifically mentioned. Whenever XLIFF 2 is mentioned, it means any of the XLIFF 2.x Versions, current (Comerford et al., 2014a) (Filip et al., 2016, the XLIFF 2.1, first public review draft) or future, as all those subscribe or will subscribe to the same core object model and are backwards compatible (Saadatfar and Filip, 2015, 2016).

2 Bitext

Let us first define bitext:

> *A structured (usually mark up language based) artefact that contains aligned source (natural language) and target (natural language) sentences. We consider bitex to be ordered by default (such as in an XLIFF file – defined below, an "unclean" rtf file, or a proprietary database representation). Nevertheless, unordered bitext artefacts like translation memories (TMs) or terminology bases (TBs) can be considered special cases of*

bitext or bitext aggregates, since the only purpose of TM as an unordered bitext is to enrich ordered bitext, either directly or through training a Machine Translation engine.

(Filip, 2012) (Filip and Ó Conchúir, 2011)

This is clearly not the classical definition but a (natural) development of the original "bi-text" concept that was first introduced by (Harris, 1988). Melby et al. (2015) explain how Harris – although he first defined bi-text in a psycholinguistic sense, as an alignment of source and target segments in translator's mind – intended that this notion inform the development of translation technology, and so it did. Already Harris himself postulated that a bi-text manifestation can be the authentic result of an actual translation process or a result of a later alignment of the source and target documents – again in the mind of a reviewer or reviser or as an actual interlinear source and target text rendering. Back in 1980s, there were obviously no bitext formats or artefacts (in the sense of the Filip, 2012 definition) around and Harris considered bitext manifestations simply as interlinearly rendered segments of source and target. In translation technology, the pre-segmentation usually is rules (and exceptions) driven, but there can be also statistically or deep learning driven segmenters. Nevertheless, most of the current CAT (Computer Aided/Assisted Translation) Tools do allow for manual re-segmentation of the working bitext by the translator. Also Harris considers bi-text to be primarily ordered and (Melby et al., 2015) explain how in translation technology past and current the unordered translation memory is actually secondary to the ordered bitext or source and target document alignment. Clearly Translation Memories (TM) and bilingual (as a special case of multilingual) Term Bases (TB) or Glossaries are derived and special forms of bitext. These were created from bitext and serve to make better and more consistent bitext and target text in the future.
After this short discussion of the bitext notion, we should be happy enough to use the (Filip, 2012) (Filip and Ó Conchúir, 2011) definition of bitext as an up to date explication of the original Harris notion. Nevertheless, the fact that the notion of bitext is for all practical purposes older than translation technology itself shows how profoundly important bitext is for actually creating natural language translations. We can argue that bi-text (as the psycholinguistic paragon and the potential post hoc re-alignment) exists and plays an important role even in low tech translation workflows that don't make use of bitext in the sense of an actual translation processing and exchange format. It doesn't really matter how bitext is represented (interlinearly, side by side, serialized as rtf, XML or JSON or what have you), since bitext really is just an abstract data or knowledge structure. That's right, the representations don't matter BUT if they do represent the same things (data/knowledge objects) and if they have an effective and efficient way to exchange them (semantic interoperability). Both – the machine readable bitext artefacts and the psycholinguistic instances of bi-text in the mind of translators and revisers – are just manifestations of the same abstract knowledge or data structure that captures the source and target texts as mutually corresponding segments.

3 Bitext Management

Let's look now at Bitext Management:

[Bitext Management is a g]roup of processes that consist of high and low level manipulation of ordered and/or unordered bitext artefacts. Agents can be both human and machine. Usually the end purpose of Bitext Management is to create target (natural language) content from source (natural language) content, typically via other enriching Bitext Transforms, so that Bitext Management Processes are usually enclosed within a bracket of source content extraction and target content re-import.

(Filip, 2012) (Filip and Ó Conchúir, 2011)

Translation industry is highly fragmented and the industrial workflow that produces fit for purpose translations for various translation buyers and target groups is highly distributed. One of the chief reasons for worldwide distribution of typical corporate, public sector or pro bono translation workflows is the best practice of using native speakers of the target language living within the specifically targeted locale. Other reasons include that translation needs tend to have discrete peaks and typical corporate organizations cannot create permanent capacities to translate content. Typically, an organization creates the necessary infrastructure and capacity to interface with external language provision providers. There are some organizations with a statutory flow of translation needs and those normally develop an in-house capacity that can handle the statutory flow. Organizations – such as European Parliament or European Commission, Canadian government et al. – with stable and high statutory flows produce more translations in house than via outsourcing. Nevertheless, even in organizations with low level of inter-organizational outsourcing, a good professional translation is a result of a concerted effort of at least two people, the translator and a reviser. Additional Quality Assurance (QA) steps will occur as spot-checks. It has become a best practice to inform the selection of and ratio of spot-checked translated content by a mixture of automated QA tools reports and human decision making given the relative importance, (lack of) usefulness of a given translated content etc. In any case, the bilingual reviewer performing the spot-check would be extremely inefficient if not presented with an actual bitext artefact that has been the medium and the result of the translation and revision process.

If we review current translation practices, from niche literary and art theoretical translations, over highly customized marketing transcreations, through product documentation in various areas, print manuals and user assistance, high volume knowledge bases, user generated and social media content, to crowdsourced and wiki based translation efforts, the need for bitext is ubiquitous; albeit it is not present in all cases in the form of an articulate exchangeable artefact that can be instantiated more than once without the loss of identity or integrity. For instance in a literary translation workflow there is rarely a tangible manifestation of bitext. Bi-text exists in the original Harris sense in the mind of the Translator as she works through the bulk of the book, treatise or what have you. Then this unique authentic alignment is lost and the Translator herself often struggles to recreate that unique initial alignment that she had in mind when originally creating the segment translation, when doing (multiple rounds) of self-correction or self-revision. In the next step, in good publishing houses, there comes a Translation Editor who performs a bilingual review and that person too needs to perform the task of re-aligning the source and the target as created by the Translator. Even in this simplest editorial workflow, there is arguably a lot of time waste, as even a single person workflow involves repeated re-creation of bi-text. Even tech hostile partisans of artistic translation or transcreation should see that the Translator would benefit from fixing their first bitext rendition of the source and target and that existence of such a fixed rendition would greatly facilitate the self-revision process. The waste entering with the second person in the process, the Translation Reviewer or Editor who is supposed to discuss choices and options of the translation with the Translator is beyond argument.

Some may argue (Melby et al., 2015) that pre-segmentation imposes a certain interpretation onto the Translator and may limit or preclude creativity in the process. In the workflows of industrial localisation, the weight of this argument is very limited and it is well addressed by the option provided by most current CAT tools to change the pre-segmentation at their working time. Workflows differ in how this is handled in the subsequent steps and obviously there is value and economy in actually preserving the segmentation choices and possible reordering of segments that the Translator performed during the translation. In the section

6 Overview of the XLIFF data structur, it will be shown how XLIFF supports re-segmentation and reordering of segments within higher level logical text units and exchanging of these modifications within an ecosystem based XLIFF roundtrip while maintaining alignment of corresponding source and target segments.
In industrial localisation, there is a non-negotiable requirement to exchange bitext between and among workflow agents, human or machine; the time-loss when distributing source text and translation suggestions to pools of single target language translators, the need for industrial revisers to instantly see the corresponding segments between source and target content simply cannot be served in any other way than through persistent bitext.
Historically, this requirement had been fulfilled by proprietary (albeit ad hoc or *de facto* standardized) formats such as the "unclean" rtf (rich text format). The early *de facto* standard bitext interchange formats had been driven by Trados, as the leading CAT Tools vendor. This is discussed with somewhat more detail in 5 Evolution and Adoption of Bitext formats.
Various types of organizations based on their priorities and availability of resources choose one of three possible approaches to bitext management. This partially depends on their current level of Localization Maturity (DePalma et al., 2006).
At very low levels of maturity, customers – and often also translators – don't know that they could benefit from working with an actual bitext representation in their translation process (Fig 1).

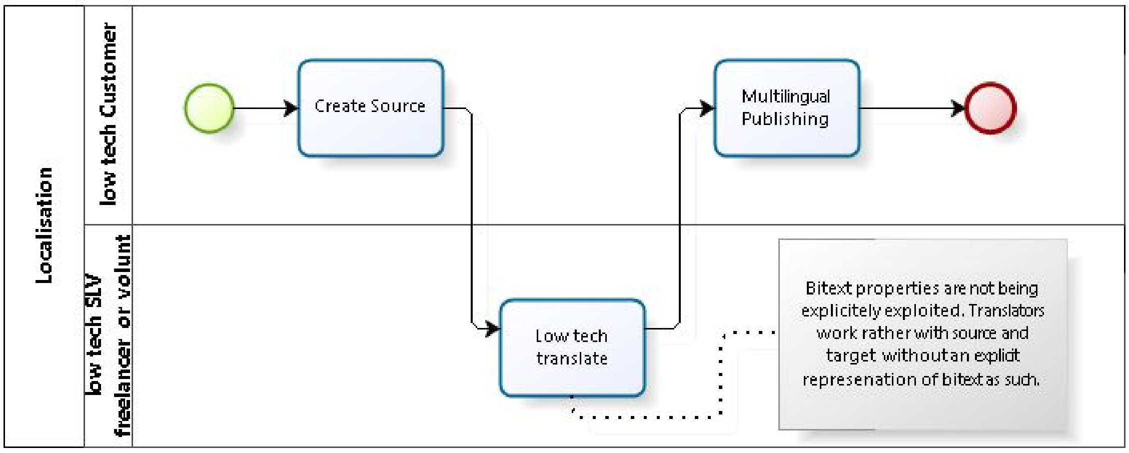
Localisation
low tech Customer
low tech SLV freelancer or volunt
Create Source
Multilingual Publishing
Low tech translate
Bitext properties are not being explicitely exploited. Translators work rather with source and target withoutan explicit represenation of bitext as such.

Fig 1: No bitext (just bi-text)

In case customers know about bitext they don't know language service providers or freelancers that could manage bitext for them. At low levels of maturity, the return on investment into having stable bitext representation isn't clear because it isn't measured or otherwise quantified in the organizations. Translators are either afraid to invest into CAT Tools that they could use to manage bitext or are not technically savvy enough to use free open source or cloud offerings.
In case of translators who use bitext privately (without telling their customers), we are drifting towards the second model (Fig 2). To be fair to the translator, the low tech customer usually doesn't care enough to know what the translator did or had to do, so that they could afford to provide their documentation translation at 4 cents per word or less and not perish in the process.

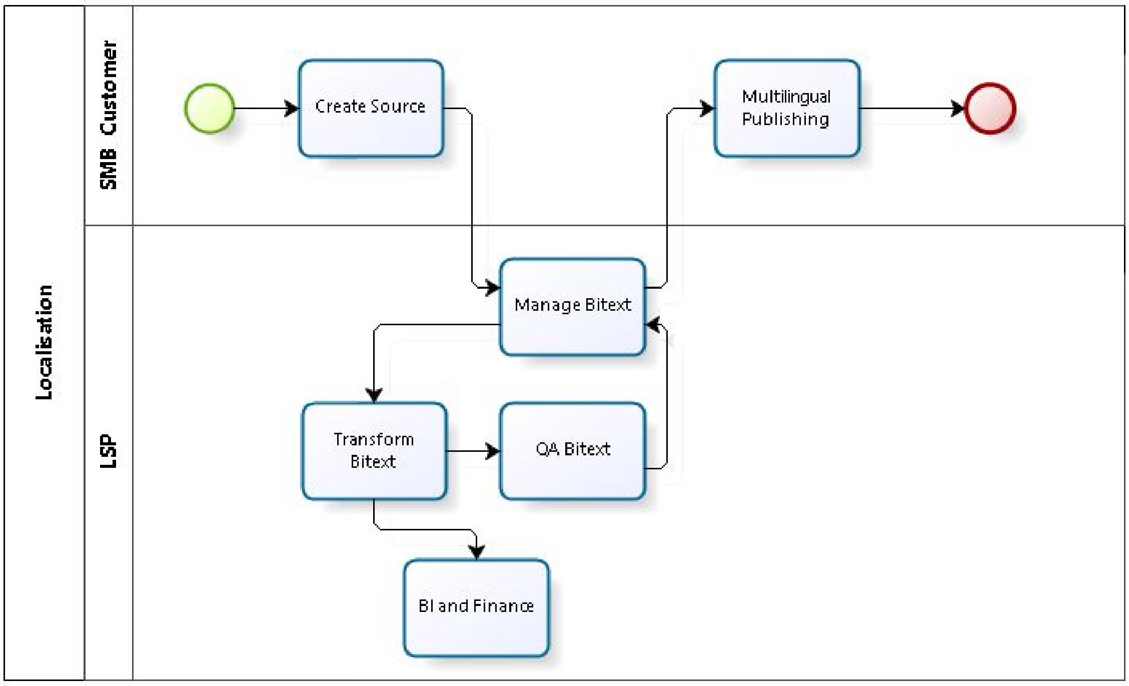

Fig 2: SMBs don’t manage bitext

Small and medium businesses (SMB) or other customers at average levels of maturity are usually aware that their language service provider benefits from explicit bitext management and they want to share the benefit. However they usually don’t care enough or don’t have enough resources to be able to manage the bitext themselves and they rely on their Language Service Providers (LSP) to do that for them.
Technology agnostic service providers in this model tend to use standard based interoperable solutions since they don’t want to manage multiple tools stacks for different customers and want to be able to effectively exchange work packages among their in-house staff (usually experienced revisers and proof readers) and freelancers (usually the translator performing the bulk of the jobs).
However, large service providers with their own technology and tool stacks are tempted to secure their revenue streams by vendor lock-in. Therefore, it is difficult to make the case for large service providers to use the standard bitext management format. This is a complex argument based on enlightened self-interest that brings us to the third model (Fig 3).

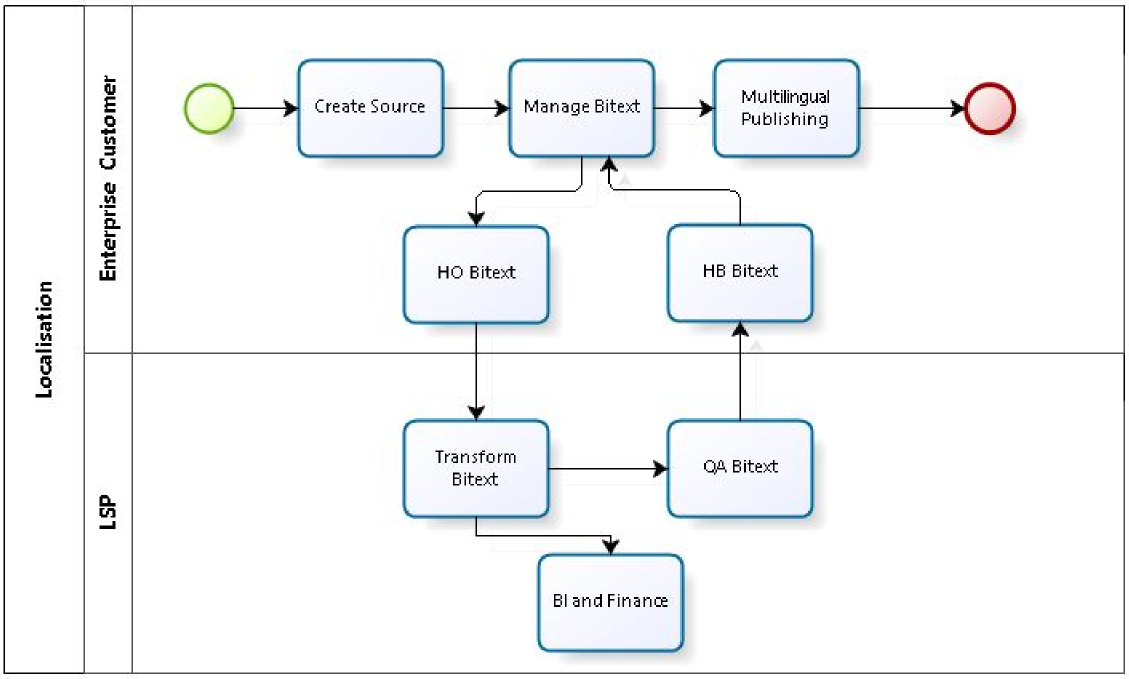

Fig 3: Enterprises do manage bitext

Large service providers usually crave for huge orders from large localisation services buyers. However, the very largest buyers are usually at fairly high levels of localisation maturity and definitely do understand the value of managing bitext and the role of managing it themselves in prevention of vendor lock-in. These very large buyers actually have the power to participate in standards development and the weight required to effectively request that their providers – no matter how big – play nice and along the standard bitext format. However, even large buyers have had and used to employ other options than employing an open standard bitext format. Obviously, automation is easiest in controlled environments and this led Microsoft to actually developing and maintaining a proprietary CAT Tool and bitext management solution for over 20 years. Only in 2011/2012, Microsoft decided against the ongoing maintenance cost of a technology tool stack outside of their core business that they would prescriptively enforce within their vendor base and decided to immediately employ XLIFF 1.2 in Windows user assistance localisation, to participate in the XLIFF 2 standardisation project and in general to require compliance within their vendor base in a technology agnostic and descriptive way based on the open standard.
Microsoft by the way donated their XLIFF 2.0 open source implementation, the so called XLIFF Object Model (King, 2015; Microsoft News, 2015; Petersen, 2015) to the localisation community and that code has been reused since by toolmakers building XLIFF 2 compliance in .NET environments.

4 Open Standards

What does that mean for a standard to be open? Please note that in standards, “open” is not the antonym to “closed” as in case of Software; open in standards is opposed to proprietary. Openness in standards doesn’t come as a binary notion, either open or proprietary; there are shades of grey with regards to standards openness. There are two major conceptual components to openness in standards: i) licensing and related cost ii) transparency of creation and availability. Rosen (e.g. Rosen, 2004, aka oslbook), Krechmer (2006), (Cargill, 2011) and

others provide rather clear criteria what a standard needs to be like to be considered open. However, the thinking on standardization develops over time and is informed by ongoing IPR (Intellectual Property Rights) litigation. Also the proprietary standardization has a substantially longer tradition than open standardization; the very notion of open standardization actually started to form as late as 2004 and actually no SSO (Standards Setting Organization) with truly open policies in both aspects existed before 2004. The thinking on openness in interoperability standards has been linked with the related but distinct notion of Open Source development.
Let us review what Openness Requirements are put forward by OSI (Open Source Initiative, 2012), the most influential legal framework around Open Source and Permissive Licensing of Open Source Software (OSS)

> ***The Requirement***
>
> *An "open standard" must not prohibit conforming implementations in open source software.*
>
> ***The Criteria***
>
> *To comply with the Open Standards Requirement, an "open standard" must satisfy the following criteria. If an "open standard" does not meet these criteria, it will be discriminating against open source developers.*
>
> 1. ***No Intentional Secrets:*** *The standard MUST NOT withhold any detail necessary for interoperable implementation. As flaws are inevitable, the standard MUST define a process for fixing flaws identified during implementation and interoperability testing and to incorporate said changes into a revised version or superseding version of the standard to be released under terms that do not violate the OSR.*
> 2. ***Availability:*** *The standard MUST be freely and publicly available (e.g., from a stable web site) under royalty-free terms at reasonable and non-discriminatory cost.*
> 3. ***Patents:*** *All patents essential to implementation of the standard MUST:*
>   1. *be licensed under royalty-free terms for unrestricted use, or*
>   2. *be covered by a promise of non-assertion when practiced by open source software*
> 4. ***No Agreements:*** *There MUST NOT be any requirement for execution of a license agreement, NDA, grant, click-through, or any other form of paperwork to deploy conforming implementations of the standard.*
> 5. ***No OSR-Incompatible Dependencies:*** *Implementation of the standard MUST NOT require any other technology that fails to meet the criteria of this Requirement.*

The above requirements are strictly speaking only met by standards developed under the Non-Assertion IPR mode. Even RF (Royalty Free) standards developed at W3C or OASIS might have theoretically an issue with the requirement #4, as traditional obligations of participants in committees and working groups chartered with the RF IPR mode don’t preclude license agreement conclusion or even negotiation. In practice however, such negotiations rarely happen or are replaced by granting of a Non-Assertion covenant even in case the IP was originally released for the scope of the standardization project chartered as a classical RF committee. The simple reason for that is that no one cares to spend money on expensive lawyers negotiating a licence grant at no cost.

To summarize, the only absolutely save IPR mode from the OSI point of view is the Non-Assertion that is being used only by OASIS and that from 2008, so that only the more recently formed Technical Committees could have actually been chartered under this IPR mode and it would be very difficult to impossible for older initiatives to re-charter with this progressive IPR mode. Nevertheless, the RF IPR mode – that is the only option at W3C and the most widely adopted option at OASIS – is generally safe in practice. But in case of standards from all other traditional standardization bodies such as ISO, Unicode, ETSI, IEEE etc. the IPR mode is invariably (F)RAND and thus openness of the standard for the practical purposes of Open Source development needs to be secured by legal frameworks extraneous to the original standardization framework, which constitutes a non-negligible legal risk for open source implementers and adopters.
Importantly, all of the above openness requirements are covered by the two aforementioned conceptual components. The requirement #1 *No Intentional Secrets* is in its description a requirement of Transparency of development and maintenance. 2# is a requirement of transparent publishing mixed with the licensing cost requirement. Interestingly, OSI does not preclude the potential cost of buying a copy of a standard, which is different from paying royalties dependent on the usage of IP used essentially in the standard. 3# is the requirement of no royalties in somewhat more detail. #4 is the most problematic and in essence it is an extension of the transparency of publishing requirement. #5 is again an extension of the development transparency. Secrets or hidden cost must not occur through dependencies. Hence we are going to review in a little bit more detail how XLIFF and related standardisation fares with respect to transparency of creation and availability as well as licensing and related cost.

4.1 Transparency of Standards Development and Publishing

This paper does not intend to provide an exhaustive overview how standards happen to be developed at a range of SDOs (Standards Developing Organizations) or SSOs (Standards Setting Organizations); we are going to use SSO onwards as a simple short term deemed synonymous with SDO, Standards Organizations, Standardization organizations and other similar terms in currency. Rather we want to demonstrate that the producer of XLIFF, OASIS XLIFF TC can serve as a paragon of transparency and persistent availability of open standards among SSOs in general. Transparency of the OASIS Technical Committee Process can be only rivalled with that of W3C. Not only all public review stages of an OASIS Standard Track Work product have persistent public URIs, but the whole standards creation process is fully publicly visible and that since the very inception of standardization projects at the stage of chartering new Technical Committees. All OASIS working lists including the ones dedicated to committee formation are publicly and persistently archived and every message ever sent to those lists has a persistent public OASIS URI; moreover, the mailing lists are effectively searchable through indexing services such as http://markmail.org/. All OASIS SVN repositories are publicly visible and accessible at https://tools.oasis-open.org/version-control/browse/, all OASIS GitHub repositories (Cover, 2016) that will be used to conduct Technical Committees' chartered work will be available at https://github.com/oasis-tcs/. This location will soon include OASIS XLIFF OMOS repositories for the work on the Abstract XLIFF Object Model (https://github.com/oasis-tcs/xliff-omos-om) and for the work on JLIFF (https://github.com/oasis-tcs/xliff-omos-jliff), the JSON serialization of the said Abstract Object Model. No OASIS TCs were conducting chartered work on GitHub at the time of writing this. Everyone can perform a full depth checkout of any of the SVN or GitHub repositories and inspect every line of specification, validation artefacts or sample code since the inception of each of the OASIS standardization projects.

Write access to OASIS working lists, wikies, and repositories (SVN or GitHub) is restricted to the TC members and active OASIS (TC) credentials are required to contribute. This restriction does not impede transparency; on the contrary, this restriction underpins the SSO's IPR Policy. OASIS TC members were authorized by the Primary Representatives of their employers (or are authorized as individual members) to contribute IP towards a chartered standardization effort as scoped in the TC Charter at the inception of the project. Charter, especially its parts defining scope and IPR are essential for the TC identity. If IP contributed to OASIS TCs' chartered work came from contributors who had not agreed to OASIS IPR policy or who had not been authorized to contribute IP towards standardization in the specific area scoped by the specific TC Charter, it would undermine the legal safety of the standardization deliverables and could lead to inability to publish work products due to copyright or patent infringement claims, or worse implementers of published work could be sued by copyright or patent owners for infringing on their IP by implementing the standard. This shows that transparency and IP management must strike a fine balance between flexibility and red tape to provide a legally safe framework for transparent development of Open Standard deliverables.

4.2 Availability of Standards under Royalty Free Conditions

Despite different schools of thought, there is a wide consensus among standardizers and IPR lawyers that standards purporting to be open need to give access to the embedded essential IPR (patented or not) at (F)RAND ((Fair) Reasonable and Non-Discriminatory) or more implementer friendly conditions. Rosen (2004), OSI and others argue that IT standards in the Internet age should be RF rather than just any sort of (F)RAND.

Again this paper does not aim to provide an overview on how SDOs manage Patent encumbrance of their standards. Rather we want to show that also in this aspect OASIS XLIFF TC and XLIFF OMOS TC are at the forefront of openness, ease and clarity of licensing conditions for potentially included patented IPR.

OASIS XLIFF TC Charter (XLIFF TC, 2014) was first put forward in late 2001 in the original Call For Participation (Best, 2001), the charter has been "clarified" four times since (2002, 2005, 2006 and 2014). Clarification here means performing editorial changes to the charter mainly to keep it up to date with specific developments. However, two things can never change in the process of subsequent clarifications, should the TC preserve identity: (1) The statement of purpose (including the high level technical scope) and (2) the IPR mode. Hence the original OASIS XLIFF TC needs to stay on the same IPR Policy, with which it had started back in 2001, i.e. the RF on RAND OASIS IPR Policy (OASIS Board of Directors, 2010: Sections 10.2.1 and 10.2.2).

Should one of these essential components of the Charter change, the original TC ceases to exist and a new one is being formed by the process of Rechartering. All IPR must be committed again to the newly formed TC if Rechartering took place. This is to make absolutely sure that the IP was provided and remains available under the same IPR Policy and for the same purpose. For instance the XLIFF TC in its Charter committed to conducting its work via development of XML Vocabularies, hence a "sister committee" – the XLIFF OMOS TC – had to be chartered when the wider stakeholder community agreed (Filip, 2015) that XLIFF needs an explicit abstract data model and an option to develop non-XML serializations, prominently a JSON serialization. XLIFF OMOS TC took the most progressive available RF IPR policy, the so called Non-Assertion IPR Policy (OASIS Board of Directors, 2010: Section 10.3) that had not been formulated at the time of the original XLIFF TC formation. Thanks to this progressive IPR mode, XLIFF OMOS can launch Open Source projects on GitHub under the OASIS umbrella.

5 Evolution and Adoption of Bitext formats

First Computer Aided Translation (CAT) Tools were just scripts over Rich Text Format (rtf). rtf had the advantage that it was widely supported by office suits and was not proprietary as the internal word-processing formats of the office suit producers. So the first technological representations of bitext were interlinear and the target translations were hidden from sight of the normal user not armed with the special script (or the skill to manipulate the visibility of the invisible style) making the other part of the bitext visible. This was a fairly reasonable idea and the first generation of these tools helped to build up the Translation and Localisation industry. Unfortunately, rtf (its implementations rather than the format itself) was not great for handling different scripts and the rtf based tools caused lot of headache in localisation engineers working between scripts or across text flow directionality.
XML started to be used in Localisation very early, as UTF-8 encoded XML provides a very robust and suitable data model for handling data in multiple languages. Pretty much all current and legacy localisation standards are indeed XML vocabularies. TBX was started as SGML but moved to XML also very early. Albeit XLIFF was available in some form or other as early as 2002, the first XLIFF Version that achieved a full OASIS ratification and the Standard publication status was XLIFF Version 1.2 in early 2008. It should be noted that the localisation providers were in general either working on rtf based tool stacks or using the Trados proprietary TTX format that – although not publicly documented – was widely understood and implemented also by tools other than Trados. The industry was immature (even more so than now) and there was little interest in an open standard format as long as the market leader Trados remained independent from major Language Service Providers. All changed when SDL made a series of tools acquisitions between 2005 and 2008 starting with the Trados acquisition in 2005. The buyers and some service providers decided to back the open standard to make sure that there was a viable bitext format outside of the direct control of a single Language Service Provider. This led to the XLIFF 1.2 success story but also to all lessons learnt from XLIFF 1.2 adoption between 2007 (before OASIS approval) until 2010 and beyond. In 2010, the XLIFF TC made the decision not to work on any other 1.x versions (not only not to add features in a 1.3 Version, even not to finish a 1.2.1 hotfix or 1.2 errata) and started pursuing the XLIFF 2.0 design.

6 Overview of the XLIFF data structure

As explained in **2 Bitext** and **5 Evolution and Adoption of Bitext formats**, XLIFF is a bitext format. In **6 Open Standards**, we explained how XLIFF is an Open Standard (transparently developed, maintained, available and Royalty Free). Now is the time to show how it is fit for purpose.
It so happened that the development of Localisation standards largely coincided with the advent and rapid rise of XML, the eXtensible Markup Language (Bray et al., 2008) and so XLIFF, as well as several other Localisation standards, happens to be an XML Vocabulary. XLIFF 1.2 as well as XLIFF 2 are XML vocabularies. Nevertheless, there is a specific data structure behind the XML serialization and the specifics of the serialization as XML or something else, or even as different XML styles is something accidental. It stops being accidental though, as soon as a tool, tool chain or an ecosystem of tools starts relying on the particular serialization. This kind of reliance on a serialization for *processing* purposes can be dangerous for tools. It is important to stress that XLIFF in its exact XML serialization has never been intended as a processing format, it is the *interchange* bitext, it advances through distributed workflow steps, but does not prescribe the internal processing representation

(serialization) in each and every tool of the ecosystem, as long as the semantic interoperability has been preserved and a valid XLIFF exists at input and output of each *processing* step.
The XLIFF 2 specifications are largely written as for an abstract data model but the instantiation specifics are only given for one particular XML serialization that is the XLIFF format itself. But the XLIFF format still contains industry wisdom that can be abstracted into a generalized Localisation Interchange Object Model that will guarantee interoperability between arbitrary serializations based on the same Object Model.
The root element of an XLIFF file is `<xliff>` and the registered extension for the XLIFF media type `xliff+xml` is `xlf`. The root element holds at least one `<file>` element that can optionally hold a recursive structure of `<group>` elements. Logical units of content are held in `<unit>` elements that can be children of `<group>` or `<file>` elements. If you are not an XML partisan you may wonder why bother with the angle brackets and you are right. To give the abstract overview of the XLIFF data model, we don't need to rely on the XML specifics of its serialization. In fact, the `<xliff>` element is a top level or root element that allows to declare what type of XML vocabulary is to be expected in the file at hand, namespaces, version and very few other things that have payload impact such as the source or target language or white space handling. `<file>` elements represent actual extracted files (or high level content nodes) and the files (or nodes) to which the targets will be merged. `<group>` elements do correspond to actual groupings to be preserved from the source to the target documents etc. So I can easily drop the angle brackets in the following and describe the XLIFF data structure in an apparently natural language.
The most interesting feature of XLIFF is its inline data model. Given by the extensive experience dealing with transformations from one natural language to another, localisation industry was and is in a unique position to define a native format agnostic data structure that is also capable to carry business critical metadata as annotations. From the theoretical point of view the most interesting predecessor of the XLIFF inline data model is the Text Encoding Initiative, which is to the present day a vigorous movement producing a widely adopted XML serialization that supports multiple approaches to the inline data model. In fact, TEI covers the whole logical space when it comes to encoding overlapping annotations in a natural language text. TEI is largely serialization centric but it in fact strives to solve data model issues when it comes to practical problems of text encoding. The TEI encoded text needs to carry – for the most varied scholarly purposes – different sets of metadata that have overlapping scope within natural language units and segments. TEI postulated a while ago that there are basically only three different serialization options: 1) duplicate the text and annotate each copy with well-formed annotations of one and only one data type. 2) define non-well-formed annotation methods using pseudo-spans delimited by empty (from the XML DOM point of view) markers. 3) use a standoff method. The third option however doesn't solve the whole issue unless it is employed together with 2) or an extraneous "offset" method that would be capable of selecting partially overlapping spans with a reasonable persistence (too persistent can be as bad or even worse than unstable or transient).
We are not going to explore all of this in detail. Suffice it to say that XLIFF 1.2 had used a less than ideal mixture of options 1) and 2); whereas XLIFF 2 is a full-fledged option 2) combined with option 3) implemented as standoff metadata containers referencing to or (less so) being referenced from well-formed as well as non-well-formed spans within the inline payload.
Interestingly XLIFF 2.0 is the first ever version of the XLIFF bitext format that defines a specific fragment identification mechanism that allows for external, internal and custom referencing, which among other things allows for a workable standoff solutions. Standoff methods are used mostly for XLIFF 2 modules data, but also for arbitrary extensions and core notes/commenting mechanism. Extension and module metadata containers are allowed on

three structural levels, the file level, the group level and the unit level. Modules and extensions that have data on the unit level can reference arbitrary spans of the inline payload within the same unit as the payload to which the metadata applies. To address cases where no suitable spans had been delimited, modules and extensions can extend the core inline annotation mechanism and use it for marking suitable spans within the unit in scope.
XLIFF inline content can contain plain text (PCDATA) and inline elements in any order, whereas well-formed pair codes (`<pc>`) and markers (`<mrk>`) have an exhaustively defined equivalence with non-well-formed pseudo-spans delimited by empty start (`<sc/>`) and end (`<ec/>`) code pairs or empty start (`<sm/>`) and end (`<em/>`) marker pairs. Importantly, pseudo-spans can overlap not only other pseudo-spans but also a strictly defined subset of well-formed spans. As result, the XLIFF inline data model cannot be represented by a tree graph. The well-formed spans (other than the pair code and markers) that do not limit formation of overarching pseudo-spans are the segment (`<segment>`) and ignorable (`<ignorable>`) spans that are used to represent segmentation within a logical unit of text. Moreover, orphaned native codes can be represented with orphaned XLIFF inline start or end codes, while orphaned start or end markers are not allowed. The rationale is that annotators (Enrichers) are in control of their own markup while Extractor might not be able to access corresponding opening or closing native markup for valid processing reasons.
Apart from the above described spanning methods, XLIFF has an empty placeholder (`<ph/>`) inline element and an empty code point (`<cp/>`) representation element for representing XML illegal Unicode characters.
The fact that XLIFF inline data model is not of a tree type, codes and annotations can survive the change of the DOM structure on the sub-unit level. Although segment and ignorable qualify as structural elements in the XLIFF 2 specifications, it is a fluid structure and CAT Tools (Modifiers) can and are expected to change this structure according to their segmentation rules or according to the human translator preferences. Thus finally the translator has the power to encode their bi-text rendition as an actual bitext artefact. In case the structure is changed, the original rendition can still be stored within the Change Tracking Module. Finally, XLIFF 2 inline data model brings a target order attribute that allows for thema-rema reordering of target content within logical units (paragraphs), a feature so important when moving between Germanic and Romance languages.

6.1 Why XLIFF 2?

This paper is supposed to be consumable by translator and business decision making audiences, hence we will omit a detailed comparison between XLIFF 1.2 and XLIFF 2 in this rationale. Why was it necessary to develop XLIFF 2.0?
Although XLIFF 1.2 is a widely adopted format and is the format behind several spectacular localisation interoperability success stories, it has – due to its old age – a number of inherent limitations that could not simply be fixed by another "dot release". Making of XLIFF 2.0 signified that there was a substantially new version that strived to build on all the good old things but has to part from backwards compatibility due to some irredeemable issues of XLIFF 1.2. XLIFF TC indeed carefully studied the usage of XLIFF 1.2 and listened to implementers feedback in XLIFF Symposia held ever since 2010. XLIFF TC (Promotion and Liaison Subcommittee) produced two editions of the XLIFF 1.2 State of the art report; Morado Vázquez and Filip (2012), Filip and Morado Vázquez (2013) analysed in detail, which XLIFF 1.2 features were largely supported by implementers and which less so. (Filip and Wasala, 2013a) brought a frequency analysis of a large (albeit not demonstrably representative) corpus of XLIFF 1.2 files as currently used in the industry, Filip and Wasala (2013b) analysed those findings in greater detail and also brought one of the major advances in addressing conformance of XLIFF Agents that was later adopted in XLIFF 2. (Morado

Vázquez and Filip, 2014) are the first State of the Art report exploring the XLIFF 2 implementations.
In general, XLIFF 1.2 suffered from ailments common in many first generation standards. We can name a few based on Cargill's taxonomy of standardization failures (Cargill, 2011). For instance, overreliance on compromise is one of issues largely present in XLIFF 1.2, the result of which is the inline markup "salad" (courtesy of Rodolfo Raya) that allowed at least two different styles of inline markup without actually defining their equivalence or saying what to do in case one uses their style of choice and someone else another. So we ended up with implementers with very fierce arguments for using just their own subset of XLIFF 1.2 features. Interoperability Now! (see e.g. Andrä et al., 2011) was one other classic Cargill failure, an attempt of three toolmakers and one localization buyer to standardize an XLIFF 1.2 profile outside of standardization bodies. This particular profile attempt failed because it missed the market opportunity and failed to draw together a representative critical mass of stakeholders. In 2010/2011 IN! proclaimed that they don't want to wait for XLIFF 2.0 and will create (surprise, surprise) Interoperability Now! However, they ended up creating a pseudo-standardization body with a mailing list and bi-weekly meetings that in the end arrived at a release around the time when XLIFF 2.0 started settling into technical stability; late 2013, when subsequent XLIFF 2.0 public reviews took place (Comerford et al., 2013, 2014b). All IN! feature requests were in fact covered with XLIFF 2.0 except one feature covered in XLIFF 2.1 (via the ITS module that was before available as an extension).
The original XLIFF 1.x – of which only XLIFF 1.2 made it to an OASIS Standard back in 2008 – was intended as a fire and die format, a bitext intended for interchange between two agents, an Extractor/Merger and an Editor "beyond the wall".
XLIFF 2.0 can afford not to be backwards compatible because XLIFF is a transient interchange format, most of its usefulness expires when the target becomes stable throughout an arbitrary number of bitext transforms, including QA checks, engineering checks etc. After the target content is successfully merged and published in the native format but in the new target language and locale, the primary role of XLIFF has ended.
However, in many organizations XLIFF has a second life. Because it is an extremely metadata rich bitext format, XLIFF stores can have multiple business functions that go far beyond the primary role of the interchange facilitating bitext. This is however out of scope of this paper.
The single biggest issue of XLIFF 1.2 is that the core is too big, in fact everything in XLIFF 1.2 is core. This makes the whole standard unwieldy and let's implementers to decide what are the most important parts of the standard. Effectively, the only universally adopted characteristic of XLIFF 1.2 was the bitext aspect. Thus the standard was infallibly delivering on its promise to keep source and target aligned during industrial bitext transformations.

XLIFF 2 applied an acid test on what is core and what isn't:

> *The core of XLIFF 2.0 consists of the minimum set of XML elements and attributes required to (a) prepare a document that contains text extracted from one or more files for localization, (b) allow it to be completed with the translation of the extracted text, and (c) allow the generation of* Translated *versions of the original document.*
>
> *The XML namespace that corresponds to the core subset of XLIFF 2.0 is* `"urn:oasis:names:tc:xliff:document:2.0"`.

(Comerford, Filip et al., 2014)

The core namespace and the functionality covered by the core elements and attributes is about 20% or less of what had the XLIFF 1.2 provided. Not much, you might say, but in fact, this is an extremely good news for interoperability. There is a 100% lossless interoperability promise guaranteed by the tight and non-negotiable core. At the same time XLIFF 2.0 defined eight (8) optional modules for advanced functionality in certain areas:

Translation Candidates Module allows for inclusion of relevant Translation Memory or Machine Translation suggestions within each unit. Glossary Module in turn is for inclusion of relevant Terminology subsets, or for sourcing new bilingual term glossaries. Format Style Module enables CAT Tools to create simple HTML previews of the content and metadata. The Resource Data Module replaces the bin-unit functionality of XLIFF 1.2 and allows for inclusion of external binary data either as context or localizable payload. Change Tracking Module allows for simple content tracking within units and notes. However in practice the usefulness is limited to simple comparison tasks, such as comparing raw MT with the post-edited version, storing a note or QA history for auditing purposes. This module cannot serve as a traditional full-fledged versioning (for that you can manage your XLIFF store on an XML database or a classic version control such as git) since it is optional and the core can be edited by tools that don't support this module. Usefulness of this module could be enhanced in controlled workflows where usage of the Change Tracking module could be mandated. Size and Length restriction module is an extensible framework for defining any sort of content limitations given by the target systems storage or display capability restrictions. It offers two standard profiles for the two most common use cases: restriction by storage size or number of code points of the target content. Validation Module is a simple mechanism to encode and enforce some common QA rules patterns. Metadata module is an extensibility mechanism without the need of using namespaces not defined in XLIFF. One cannot expect transparent machine to machine interoperability based on this module's data, but the general logic of this module allows to group different types of metadata and the data structure is one of key and value pairs. Therefore all agents supporting this module should be able to at least display the grouped key-value pairs. Since this is an extensibility mechanism, users of the module must observe the general extension functionality restriction of the XLIFF 2 architecture, i.e. that extensions must not implement any features that are available through core or other modules.

In XLIFF 2.1 a large new module was added, the ITS Module. The number of features added and the general expressivity added through this module can be compared to at least six modules in XLIFF 2.0. The module fully defines all elements and attributes to natively support six (6) ITS (Filip et al., 2013) data categories: Allowed Characters (restricting character sets for instance for legacy Unicode incapable systems), Domain (for conveying of topic or specialization of content, useful for instance for selecting of a domain trained MT engine or subject matter expert translators), Locale Filter (for conditional profiling of content for various locales in monolingual XLIFF files, i.e. before the targets in one of the relevant locales are added and in fact informing the workflow or process to do so). Localization Quality Issue and Rating are two categories that serve for injecting of QA metadata during the process of performing QA. Notably, MQM originates from the list of issue types in the Localization Quality Issue data type and is in effect the only machine readable MQM profile as of now. Text Analysis data category lets enrichers include data from disambiguation and entity recognition services.

Four (4) ITS metadata categories (Localization Note, Preserve Space, Translate and External Resource) were fully expressible in XLIFF 2.0, therefore the XLIFF 2.1 ITS module only informatively describes how these categories correspond to native XLIFF features.

Five (5) ITS metadata categories have partial overlap with XLIFF 2.0 features. For Language Information, MT Confidence, Provenance and Terminology, the XLIFF 2.1 ITS module defines how to extend the existing XLIFF 2.0 features with ITS module defined elements and attributes to fully express them. ITS Storage Size data category would be fully expressible within the Size and Length Restriction Module framework as a third party profile, however

the profile itself has not been specified at this point due to lack of current industry interest in expressing this profile.
Five (5) ITS data categories are not expressed as metadata when extracting content into XLIFF, instead the category data is fully "consumed" by the extraction behaviour. The categories are: Directionality (the directionality provisions of ITS 2.0 were obsoleted by recent changes in Unicode and HTML5 that are supported by the XLIFF 2 data model), Elements within Text, ID Value, Locale Filter (in the standard use case this category is not needed as metadata in a bilingual XLIFF file) and Target Pointer (there is a simple rule that points ITS Processors to XLIFF target content).

7 Conclusion

We have seen how bi-text and bitext are critical in producing fit for purpose translations in varied contexts. After looking in more detail at industrial localisation, we concluded that interoperable bitext representations are necessary for effective localisation operation at translation buyer organizations above a certain size and maturity level and most language service providers and freelancers. Existence of an open standard bitext format allows sound competition in the marketplace and helps translation buyers to prevent vendor lock-in. Technology agnostic vendors and freelancers benefit as they are not forced to maintain parallel tool stacks. We have reviewed at a conceptual level the XLIFF data model and shown that it is a fully expressive and adequate vehicle for interoperable bitext representation in the ever growing translation and localisation market.

Acknowledgements

I wish to thank Alan Melby and Olaf-Michael Stefanov for introducing me to the JIAMCATT network and making me work on the deep reasoning for "Why bitext?" and "Why open standard bitext?".
This research was supported by the Theme E of the SFI ADAPT Centre at the Trinity College Dublin (SFI Research Centres Programme Grant 13/RC/2106).
BPMN 2.0 compliant model exports were created with Bizagi Modeler 64-bit Version 3.1.0.011. Thanks to Bizagi for developing their free Process Modeler.

References

Andrä, S.C., Coffey, D., Filip, D., Reynolds, P., Ugray, G., 2011. Interoperability Now! Presented at the MemoQ Fest, Kilgray, Budapest.
Best, K.F., 2001. [tc-announce] OASIS TC Call For Participation: XML LocalisationInterchange File Format TC.
Bray, T., Paoli, J., Sperberg-McQueen, C.M., Maler, E., Yergeau, F. (Eds.), 2008. Extensible Markup Language (XML) 1.0 (Fifth Edition), Fifth. ed, W3C Recommendation. W3C.
Cargill, C., 2011. Why Standardization Efforts Fail. J. Electron. Publ. 14. doi:http://dx.doi.org/10.3998/3336451.0014.103
Comerford, T., Filip, D., Raya, R.M., Savourel, Y. (Eds.), 2014a. XLIFF Version 2.0, OASIS Standard. ed, Standard. OASIS.
Comerford, T., Filip, D., Raya, R.M., Savourel, Y. (Eds.), 2014b. XLIFF Version 2.0 [csprd03], Committee Specification Draft 03 / Public Review Draft 03. ed, Standard. OASIS.
Comerford, T., Filip, D., Raya, R.M., Savourel, Y. (Eds.), 2013. XLIFF Version 2.0 [csprd02], Committee Specification Draft 02 / Public Review Draft 02. ed, Standard. OASIS.
Cover, R., 2016. GitHub Repositories for OASIS TC Members' Chartered Work [WWW Document]. OASIS. URL https://www.oasis-open.org/resources/tcadmin/github-repositories-for-oasis-tc-members-chartered-work (accessed 9.29.16).

DePalma, D.A., Beninatto, R.S., Sargent, B.B., 2006. Localization Maturity Model (paid research No. Release 1.0). Common Sense Advisory, Inc., Lowell, MA.
Filip, D., 2015. [xliff-comment] Draft Meeting Minutes from FEISGILTT 2015 Infosessions (Meeting mInutes of Record). OASIS XLIFF TC, Berlin.
Filip, D., 2012. Using Business Process Management and Modelling to Analyse the Role of Human Translators and Reviewers in Bitext Management Workflows, in: IATIS 2012. Presented at the IATIS 2012, Belfast.
Filip, D., Comerford, T., Saadatfar, S., Sasaki, F., Savourel, Y. (Eds.), 2016. XLIFF Version 2.1, Committee Specification Draft 01 / Public Review Draft 01. ed, Standard. OASIS.
Filip, D., McCance, S., Lewis, D., Lieske, C., Lommel, A., Kosek, J., Sasaki, F., Savourel, Y. (Eds.), 2013. Internationalization Tag Set (ITS) Version 2.0, Recommendation. ed, Recommendation. W3C.
Filip, D., Morado Vázquez, L., 2013. XLIFF Support in CAT Tools (Subcomittee Report No. 2), XLIFF State of the Art. OASIS XLIFF TC.
Filip, D., Ó Conchúir, E., 2011. An Argument for Business Process Management in Localisation. Localis. Focus 10, 4–17.
Filip, D., Wasala, A., 2013a. Addressing Interoperability Issues of XLIFF: Towards the Definition and Classification of Agents and Processes. Presented at the FEISGILTT 2013, Localization World, London.
Filip, D., Wasala, A., 2013b. Process and Agent Classification Based Interoperability in the Emerging XLIFF 2.0 Standard. Localis. Focus, Special Standards Issue I 12, 61–73.
Harris, B., 1988. Bi-text, A New Concept in Translation Theory. Lang. Mon. 54, 8–10.
King, R., 2015. Microsoft/XLIFF2-Object-Model. Microsoft.
Krechmer, K., 2006. Open Standards Requirements. Int. J. IT Stand. Stand. Res. IJITSR 4, 43–61. doi:10.4018/jitsr.2006010103
Melby, A., Lommel, A., Vázquez, L., 2015. Bitext, in: Routledge Encyclopedia of Translation Technology. Routledge, pp. 409–424.
Microsoft News, 2015. Microsoft XLIFF 2.0 Object Model is now available on GitHub [WWW Document]. Microsoft News Cent. Blog. URL https://blogs.microsoft.com/firehose/2015/11/11/microsoft-xliff-2-0-object-model-is-now-available-on-github/ (accessed 10.16.16).
Morado Vázquez, L., Filip, D., 2014. XLIFF 2.0 Support in CAT Tools (Subcomittee Report No. 3), XLIFF State of the Art. OASIS XLIFF TC.
Morado Vázquez, L., Filip, D., 2012. XLIFF Support in CAT Tools [1st ed.] (Subcomittee Report No. 1), XLIFF State of the Art. OASIS XLIFF TC.
OASIS Board of Directors, 2010. Intellectual Property Rights (IPR) Policy | OASIS.
Open Source Initiative, 2012. Open Standards Requirement for Software [WWW Document]. Open Source Initiat. URL https://opensource.org/osr (accessed 9.29.16).
Petersen, P., 2015. XLIFF 2.0 Object Model is now Open Source on GitHub [WWW Document]. Microsoft Lang. Portal Blog. URL https://blogs.technet.microsoft.com/terminology/2015/11/11/xliff-2-0-object-model-is-now-open-source-on-github/ (accessed 10.16.16).
Rosen, L., 2004. Open Source Licensing Software Freedom and Intellectual Property Law [WWW Document]. URL http://www.rosenlaw.com/oslbook.htm (accessed 5.29.12).
Saadatfar, S., Filip, D., 2016. Best Practice for DSDL-based Validation, in: XML London 2016 Conference Proceedings. Presented at the XML London 2016, XML London, London.
Saadatfar, S., Filip, D., 2015. Advanced Validation Techniques for XLIFF 2. Localis. Focus, Special Standards Issue II 14, 43–50.
Savourel, Y., Reid, J., Jewtushenko, T., Raya, R.M. (Eds.), 2008. XLIFF Version 1.2, OASIS Standard. ed, Standard. OASIS.
XLIFF TC, 2014. OASIS XML Localisation Interchange File Format Technical Committee [Charter].