Computer Science and Information Systems 11(1):393–417 DOI: 10.2298/CSIS121223014L

Slicepedia: Content-Agnostic Slicing Resource Production for Adaptive Hypermedia

Killian Levacher, Seamus Lawless and Vincent Wade

Trinity College Dublin
{killian.levacher,seamus.lawless,vincent.wade}@scss.tcd.ie

Abstract. The production of resources supporting the needs of Adaptive Hypermedia Systems (AHSs) is labor-intensive. As a result, content production is focused upon meeting the needs of resources with higher demand, which limits the extent to which numerous and diverse content requirements of AHSs can be met. Open Corpus Slicing attempts to convert the wealth of information available on the web, into customisable information objects. This approach could provide the basis of an open corpus supply service meeting more diverse and unpredictable content requirements of AHSs. This paper takes a case study approach, focusing on an educational sector of adaptive hypermedia, to test out the effect of using Slicepedia, a service which enables the reuse and customisation of open corpus resources. An architecture and implementation of the system is presented along with two user-trial evaluations, involving 91 participants, which suggest that slicing techniques represent a valid content production supply for AHSs

Keywords: slicing, open corpus reuse, adaptive hypermedia

1. Introduction

Adaptive Hypermedia Systems (AHSs) have traditionally attempted to respond to the demand for personalised interactive learning experiences through the support of adaptivity, which sequences re-composable pieces of information into personalised presentations for individual users. While their effectiveness and benefits have been proven in numerous studies [15], the ability of AHSs to reach the mainstream audience has been limited [2]. For example, in Adaptive Educational Hypermedia Systems (AEHSs), this has been in part due to their reliance upon large volumes of one-size-fits-all educational resources available at high production costs [8]. Although extensively studied, solutions proposed so far (section 2) do not address the fundamental problem directly, which is the labor-intensive manual production of such resources. As a result, content creation is naturally focused upon addressing the needs of predictable resources in higher demand. However as the number of internet users grows, so does the diversity of user information needs. Since 2006, the number of internet users has more than doubled [1], with an increasing number of individuals accessing the web through mobile devices, from more diverse backgrounds (different languages, cultures etc...). This growing demand for more ephemeral niche online user-experiences is driving the necessity to target increasingly smaller content requirement niches, in even greater volumes, covering larger ranges of diverse information needs. As a consequence, the difficulty involved in predicting user information needs in advance also rises [20]. Such resource requirements are precisely those which traditional content production approaches desist due to restrictive constraints and prohibitive production costs.

In parallel to these developments, the field of Open Adaptive Hypermedia System (OAHS) has attempted to leverage the wealth of information, which has now become accessible on the World Wide Web (WWW) as open corpus information. However, when open corpus reuse has been achieved, it is traditionally performed *manually* [6] or at best using conventional Information Retrieval (IR) approaches [21]. These techniques suffer because they only provide *one-size- ts-all*, *un-customised* delivery of results, with limited control over the granularity, content format or associated meta-data of resources targeted for *prede ned* reuse use-cases by AHSs (section 2). Open Corpus Slicing (OCS) techniques [14] on the other hand, aim to automatically convert native open corpus resources into customisable content objects meeting various specific AHS content requirement needs (topic covered, style, granularity, delivery format, annotations) which are unknown in advance. We believe that, in order to serve a more diverse and unpredictable range of AHS content requirements, the cost intensive manual production and/or adaptation of resources must be augmented (and if possible replaced) by the automated repurposing of open corpus content[1]. In other words, in the same way copper or gold are mined as raw materials, processed and further reused for various purposes, open corpus resources should be harvested as preliminary resources and converted into reusable *tailored* content packages, serving specific content requirements needs of individual AHSs on demand. One-size-fits-all delivery of open corpus content to AHSs would be equivalent to the reuse of raw copper material without any post-excavation modifications. A fully-automated, on-demand, content production system based upon OCS techniques could thus theoretically address the need for such a content supply paradigm described previously. This paper builds upon previous OCS research and presents experiments conducted towards investigating the use of such techniques to achieve this goal. AEHSs are cited as being the most successful but also most expensive systems to develop content for [17], this paper hence takes a case study approach investigating how educational AHSs in particular can be supported by an automated content production service based upon OCS techniques.

Contribution: The rest of this paper presents Slicepedia, a content supply service, based upon slicing techniques, that leverages open corpus resources to produce large volumes of right-fitted information objects. An i) architecture and implementation of the system is presented, followed by ii) two user-trial evaluations (involving over 91 contestants), both applied in authentic educational scenarios.

2. Background and Related Work

The reliance of AEHSs upon the ability to access large volumes of diverse educational resources has been a problem investigated by many researchers. The reuse and sharing of existing resources, through encapsulation standards such as Learning Object Models (LOMs), followed a natural need by the community to reuse previously produced resources, which subsequently led to the creation of many Learning Object Repositorys (LORs)[2]. As these repositories grew in size, research focusing upon improving the search and discovery of existing resources [5] naturally emerged as consequence. Tools

[1] In order to deal with the significant challenges of open corpus slicing and reuse, copyright and digital rights management issues are deemed beyond the scope of this paper

[2] NDLR: www.ndlr.ie, MERLOT: www.merlot.org

improving production efficiency [16], by re-purposing existing material already available in LORs into new learning resources, or automatically combining existing resources into new courses [18], ultimately emerged as both the size and search capabilities of LORs improved. Although these solutions certainly do reduce the production time and costs of learning resources, none of them directly address the fundamental issue which is the initial labor-intensive production of such content. When production efficiency is taken into account, improvements through re-purposing are still carried out manually [17]. This results in a costly content production paradigm, limited in terms of volume of resources available.

OAHS attempts to surpass these volume supply limitations through the incorporation of open corpus resources through either manual incorporation [6] techniques, automated [23] and community-based [3] linkage or IR approaches [10]. In contrast with previous solutions, these techniques offer a much cheaper alternative supply of content, with limitations, in terms of volume of resources available, significantly lower. However even when relevant open web resources are retrieved, IR techniques suffer because they only provide un-tailored, document level, delivery of results, with limited control over topics, granularity, content format or associated meta-data. Open-corpus material, in its native form, is very heterogeneous. It comes in various formats, languages, is generally very coarse-grained and contains unnecessary noise such as navigation bars, advertisements etc...[22]. Open corpus resources, originally produced for a pre-define purpose, are generally incorporated by these systems in their native form, as "one-size-fits-all" documents. However, the reuse potential of such resources (complete with original menus, advertisements), is far less adequate than the reuse of selected parts of the article, de-contextualised from their original setting, at various levels of granularity, with associated meta-data and in a delivery format of choice [10]. This situation is even more inappropriate if large and diverse ranges of smaller and more specific niche content requirements are to be served (section 1).

In order to clarify what we mean by a diverse and unpredictable niche content requirements, lets consider the following open and user-driven AEHSs use case scenario. Suppose Alice wishes to improve her grammar skills in Portuguese and decides to use a portal specifically built for this purpose. The system provides e-assessments consisting of traditional gap filling exercises for a given piece of native language text (figure 4a). It provides Alice with a list of various languages Λ, grammar skills Γ and reading level difficulty to choose from, which she selects accordingly to her training needs. So as to sustain learner motivation, the portal additionally provides the ability to select topics of interest, among a large list Θ, which training exercises should also cover. Whenever Alice starts her training, the system searches for resources on the web fulfilling the combined content requirements $\Sigma\{\Lambda\ \Gamma\ \Theta\ \}$ and converts these into grammar e-assessments. The system continuously records the sets of mistakes μ performed by Alice and includes this additional variable to its content requirement combination Σ in order to address the required subset of grammar points of immediate importance. As Alice progresses, the set of requirements Σ evolves and so does the content supplied. The portal can supply Alice with as much content as needed for her training. As pointed out by Steichen et al. [20], the production of educational resources a-priori of any learner interaction, generally assumes that the type and quantity of resources needed for a particular AEHSs is known in advance of system deployment. In the use case presented above however, the num-

ber of content requests is unknown in advance and the content requirement combination possibilities for Σ are very large. For this reason, only a handful of deliveries will ever occur for most individual content requests possibilities. This situation portrays a resource distribution scenario, which could only be sustained by an automated content production service guaranteeing the on-demand provision of:

- P_i) large volumes of content,
- P_{ii}) at low production costs,
- P_{iii}) suitable for a large range of potential activities

OCS techniques [13] aim to automatically right-fit open corpus resources in order to match various content requirements. As pointed out earlier, open corpus material in its native form is very heterogeneous. It comes in various formats, languages, is generally very coarse-grained and contains unnecessary noise such as navigation bars, advertisements etc. OCS techniques provide the ability to harvest, fragment, annotate and combine relevant open corpus fragments and meta-data in real time. Hence, although the quality of content supplied (with respect to relevance) is important of course, this aspect is addressed by traditional retrieval systems. OCS techniques instead aim at improving the quality, in terms of appropriateness for further AHS consumption, of open corpus resources identified as relevant. More recent approaches for example [20] attempt to semi-automatically tailor open web content through the use of hand-crafted resource specific algorithms. Such approaches to open corpus reuse are referred to, throughout this paper, as content-specific OCS techniques. Although these techniques provide higher control over the granularity and reuse of such resources, only a pre-defined set of open web resources can be targeted for reuse, for pre-determined AHSs and niche content requirements. Whenever new resources or different niche content requirements emerge, existing algorithms must be changed, or replaced altogether, which still requires a significant level of manual labor. These techniques, in essence, attempt to deal with the challenge of open corpus heterogeneity by reducing the degree of unpredictability and variability inherent in such resources, which limits the scale of reuse with respect to the range of AHSs served and open corpus resources targetable.

Content-agnostic OCS on the other hand, attempts to go beyond these limitations by slicing open-corpus material fully-automatically S_i) for AHSs built independently of the slicer, using techniques which do not require any predefined knowledge of either S_{ii}) the resources targeted or S_{iii}) the reuse purposes. Such techniques have shown to provide similar performance in comparison to content-specific alternatives [14]. It is still unclear, however how the suitability of content generated automatically by these techniques compares with respect to content manually produced for a specific purpose within an educational scenario. If such a technique is to be used as a basis for more scalable content supply services, which can be used to match larger ranges of diverse niche content requirements, the suitability of the content produced (P_{ii}) from a user's point of view as well as it's production cost (P_i) must be examined.

3. Slicepedia Anatomy

3.1. Overall Anatomy of a Slicer

As depicted in Figure 1, a slicer is designed as a pipeline of successive modules, consisting of four steps, each analysing open corpus resources and appending specific layers of meta-data to each document. Resources openly available on the WWW, in multiple languages and with unknown page structures, are gathered and then transformed, on demand, into reusable content objects called slices. AEHS (or so-called slice consumers) thus use the slicer as a pluggable content provider service, producing slices matching specific unique content requirements (topic, granularity, annotations, format etc.). A slicer can also provide the ability for an open-corpus resource to be ”focused” on an arbitrary topic, meaning only parts of a resource covering this topic are delivered in a slice [14].

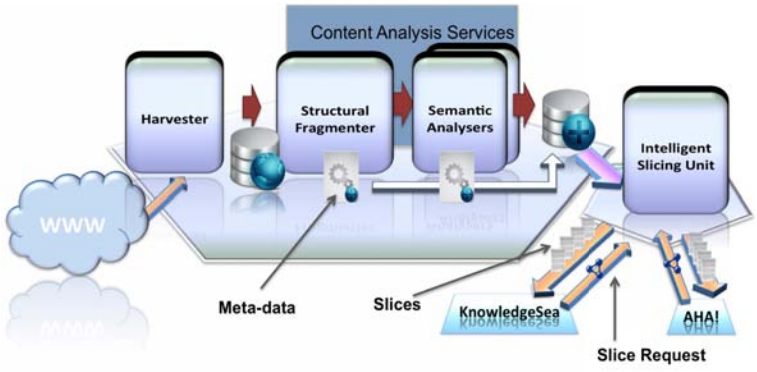


Fig. 1. Slicer Anatomy

Although the architecture presented in this paper could of course be implemented using various content analysis techniques with different levels of automation (see [14] for alternative implementations), the aim of a slicer ultimately is to provide a fully automated service, pluggable within multiple AHS slice consumers. In order to support niche content requirements (section 1), a slicer should process large volumes of open corpus resources, without any prior knowledge of structure, domain, or language used by each document and possess the ability to support diverse AEHS content requirements. For this reason, whenever the use of a slicer is referred throughout this document, it implicitly refers to a slicer implemented using content-agnostic techniques. The anatomy of a slicer consists of the four following components:

1) Harvester: The first component of a slicer pipeline aims to acquire open corpus resources, from the web, in their native form. Standard IR systems or focused crawling techniques [10] are used to gather relevant documents, which are then cached locally for further analysis.

2) Structural Fragmenter: Once resources have been identified, each individual document is analysed and fragmented into structurally coherent atomic pieces (such as menus, advertisements, main article). Structural meta-data, such as the location of each

fragment within the original resource, is extracted and stored in the meta-data repository. This phase is critical since, as mentioned previously (section 2), maximising the reuse potential of a resource involves the ability to reuse selected parts of documents, which in turn, depends upon correctly identifying individual sections of pages to produce individual structurally coherent fragments. Any structural inconsistencies (such as parts of original menus erroneously broken in two and merged with paragraphs or various titles merged together) produced at this stage, will have a direct impact upon the quality of reuse, regardless of the performance of subsequent steps. Any algorithm used within this component must be fully automated and deal with unknown document structures in multiple languages.

3) Syntactic/Semantic Analyser: Once an initial set of fragments is produced, each is analysed by standard Information Extraction (IE) and Natural Language Processing (NLP) algorithms with the intention of producing discerning syntactic and/or semantic meta-data, supporting the identification and selection of such content for reuse by individual slice consumers. Such meta-data might include, writing style, topic covered or the difficulty level of content. Since what constitutes discerning meta-data is highly dependant upon the reuse intentions of each slice consumer, an inappropriate provision of such data can lead to very low reuse scenarios across consumers. Hence great care must be taken during the selection of suitable annotators in order to support the broadest needs possible of targeted consumer use cases. Additionally, this phase is also essential with respect to the level of granularity control provided to slice consumers. The ability to focus (or constrain) a resource to only cover a chosen topic is clearly dependent upon the capacity to accurately match selected parts of single resources with appropriate concepts. Moreover, as the targeted document space of a content-agnostic slicer is by definition open, the resulting open content model available to slice consumers shouldnt be restrained by any predefined subjective domain model. For this reason, Slicepedia disentangles domain modelling from its open content model and instead provides anchors to linked-data concepts as a foundation for any domain model chosen by individual slice consumers. Additionally, all fragments and annotations produced by Slicepedia are also available as linked-data. The intention is to reduce low reuse effects related to the subjective nature of discerning meta-data, and support collaboration with reusable annotations across institutions.

4) Slice Creator: Once individual fragments and meta-data annotations are available, the slicer is ready to combine these individual elements into slices. The array of possible adjustments (such as the extent of control over granularity, formats and annotation) a slicer can offer upon a set of open corpus resources is referred to as its Content Adaptation Spectrum (CAS). Whenever slice requests[3] are received, an intelligent slicing unit combines fragments together, along with relevant meta-data, into customised slices. Since a slicer is to be used as a pluggable service within a variety of slice consumers, if deployed mainstream, a slicer should support a range of delivery formats. A slice is therefore defined as: *customised content generated on-demand, consisting of fragment(s) (originating from pre-existing document(s)) assembled, combined with appropriate meta-data and right- tted to the speci c content requirements of a slice consumer (with various application-speci c content-reuse intentions)*. Slices can contain other slices and can be reused or re-composed with many slices.

[3] An example request is provided in section 3.2

3.2. Slicepedia Slicer Implementation

Since the architecture of a slicer is designed as a pipeline, components can only be run successively in the numerical order specified above. Nevertheless, the content-agnostic slicer pipeline implemented as part of this research can be run in various modes based upon requests received. For this reason, an additional component, named the Slicepedia Requests Coordinator (SRC) module (see Figure 2), is responsible for coordinating each component together based upon what Slicepedia request is being performed. Slicepedia can be requested to either R1) prepare open-corpus resources for reuse or R2) generate slices (slice request). When performing a slice request, if slices are to be generated from open-corpus resources, which havent yet been prepared, Slicepedia performs both the preparation of resources and slice requests all at once (R3).

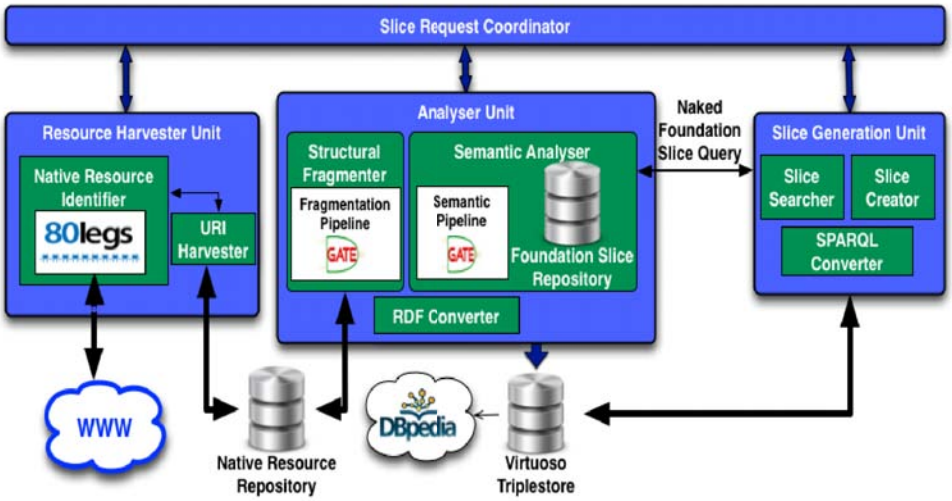


Fig. 2. Slicepedia Implementation

For the purpose of this research, the ***harvesting module*** used as part of Slicepedia consisted of the 80Legs[4] web crawler. With respect to the ***fragmentation component***, plain fusion Densitometric Content Fragmentation (DCF) [9] was selected as the fragmentation approach of choice for this content-agnostic slicer. The ability of DCF algorithms to fragment pages regardless of the meaning or structure of xml tags used and without the need for any rendering, allows it to process virtually any xml-based documents at very high speed. These algorithms proceed to fragment resources by converting individual web pages into a one dimensional array of so-called text density values. Text density values are computed based upon the number of words (treated as mere tokens) contained within leaf nodes of a page's Document Object Model (DOM) tree structure. Fragment-

[4] http://www.80legs.com/

ing pages thereafter simply consists in detecting variations in text density and correlating them with fragment boundaries[5].

A prior detailed analysis of densitometric algorithm [12] revealed that it could fragment parallel corpora in multiple languages with a predictable degree of accuracy. However, limitations of this fragmentation algorithm with respect to content type (such as forum or product pages) were also discovered. For this reason, this experiment considered a subset of the WWW as an open corpus target, consisting of any article type pages (news, tutorial or encyclopaedia pages). Although, this clearly represents a limitation with respect to the aim of improving the automated reuse of any open corpus resource available on the WWW, considering the wide diversity and quantity of such pages currently available, this subset was deemed sufficient for the purpose of experiments performed for this research.

Fragmentation NLP Pipeline		
1		ANNIE English Tokenizer
2		ANNIE Sentence Splitter
3		**BoilerPipe**
	a	Densitometric Fragmenter
	b	Boilerplate Detector
4		Fragment Sequencer
5		Annotation Transfer
6		**Internal Fragment Pipeline**
	a	Fragment Type Identifier

Annotation NLP Pipeline	
1	ANNIE English Tokenizer
2	ANNIE Sentence Splitter
3	Part-Of-Speech Tagger
4	GATE Morphological Analyser
5	Verb Group Chunker
6	Verb Feature Analyser
7	Concept Annotator
8	Reading Level Analyser

Table 1. Slicepedia NLP Pipelines

The set of NLP algorithms used to support the underlying operation of the fragmenter module (see table 1), were assembled using a real-time controller[6] available as part of the the Generral Architecture for Text Engineering (GATE) framework. A GATE controller consists of a pipeline of successive NLP algorithms each accomplishing a specific predetermined task. Each of these algorithms are packaged as Processing Resources (PRs) in order to be executable within a controller. GATE PRs can also be constructed specifically for a given controller using Java Annotation Patterns Engine (JAPE) grammar rules. These rules consist of Java functions executed whenever arbitrary annotation patterns are recognised within the pipeline.

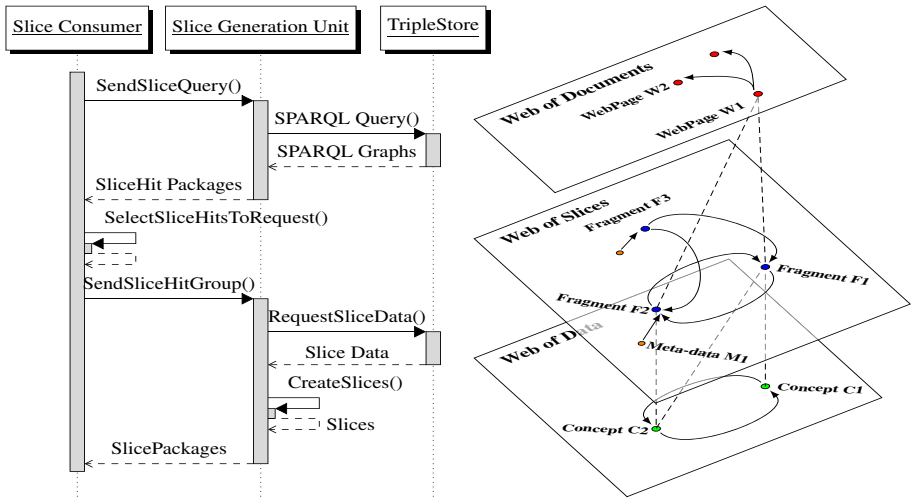
As in most NLP systems, and since the equation underlying DCF algorithms relies partially upon the total amount of tokens contained within a specific tag, the first PR contained within this controller consists of a tokenizer provided as part of the A Nearly-New Information Extraction System (ANNIE) set of GATE NLP algorithms. This tokenizer outputs annotations pointing to the start and end of each token identified within the content processed. Following the tokenizer, is the ANNIE English sentence splitter. This domain and application-independent algorithm consists of a cascade of finite-state transducers, which use the token annotations produced earlier and identify sentence boundaries within

[5] For a more detailed explanation of these algorithms, the readers are referred to the original paper
[6] http://gate.ac.uk/sale/tao/splitch7.html

the text. Each sentence identified is assigned an annotation of type sentence (pointing to the start and end of each sentence). Directly following the sentence splitter is the DCF algorithm mentioned earlier. This fragmentation algorithm was used as part of a library named Boilerpipe[7]. This library also includes a boilerplate detection algorithm which produces two types of annotations, selected based upon whether it considers fragments to be boilerplate content or not. A set of JAPE[8] grammar rules was written specifically for the purpose of this slicer to create the fragment sequencer. The latter identifies each non-boilerplate fragment as a valid foundation fragment and assigns sequence numbers to each one based upon their position within the original native resource. The slice creation module, (last module in the pipeline) later on uses these sequence numbers to produce slices. Once each valid foundation fragment has been identified and sequenced, the annotation transfer module identifies all original mark-up contained within valid fragments, as well as those produced within this pipeline, and saves them within a Slicepedia annotation category. A fragment type identifier (built using JAPE grammar rules) finally assigns each fragment a type annotation (paragraph, table etc...) based upon the mark-up annotations contained within each fragment transferred previously.

As for the fragment module, the GATE pipeline used as part of the ***annotator module*** uses a tokenizer and sentence splitter. For the purpose of this research, and as a consequence of reuse vehicles needs selected for this evaluation (see section 4.2), the algorithms used as part of this module consisted mostly of syntactic analysers (with the exception of Dbpedia annotations see below). Prior initial[9] results focusing upon semantic-related evaluations of the slicer, can be found in a previous publication [14]. A Part-Of-Speech (POS) tagger [7] is then used to assign a POS tag to each token. Following the latter are a morphological analyser[10], verb group chunker[11] and verb feature analyser. As will be described in section 4, the first user-trial scenario carried out for this evaluation involved grammar exercise tasks. The role of these algorithms hence consisted in identifying the tense, infinitive form and mode of each verb encountered in each fragment. The verb feature analyser consists of a JAPE transducer specifically developed for Slicepedia. It produces the non-finite form of each verb group identified by the verb group chunker by taking into account the lemmas and verb features produced earlier by previous algorithms. Concepts mentioned within each fragment were identified using the AlchemyApi[12] concept tagging service and associated with Dbpedia[13] instances. The reading-level difficulty expressed as Flesh Reading scores was determined using the open source Flesh annotator[14]. All annotations and fragments produced by the two GATE pipelines were stored within a Sesame triple store[15] using the Resource Description Framework (RDF) metadata language.

[7] https://code.google.com/p/boilerpipe/
[8] http://gate.ac.uk/sale/tao/splitch8.html
[9] Additional research specifically investigating the use of slicers with higher semantic capabilities is currently being carried out
[10] http://gate.ac.uk/sale/tao/splitch21.html
[11] Based upon grammar rules defined in [4]
[12] http://www.alchemyapi.com/
[13] http://dbpedia.org/
[14] http:// esh.sourceforge.net/
[15] http://www.openrdf.org/



(a) Slice Search Procedure (b) Web of Slices

Fig. 3. Slice Generation

Slice requests received by Slicepedia are thus converted to SPARQL queries by the slice searcher module within ***slice generator unit*** and submitted to the triple store in order to identify any matching fragment/annotation combinations (figure 3a). All potential fragment/annotation combinations are returned by the slice generator to the slice consumer as Slicehits. A Slicehit describes the specifications (or blueprint) of a potential slice that can be delivered to the AHS as a response to a request. Slice consumers select the most adequate slice to produce, along with arbitrary annotations available for this slice and provide the relevant SliceHit back to the slice generator unit. The slice creator module thereafter identifies the relevant fragments constituting the slice solution selected and appends these to each other (with an order based upon the sequence numbers generated by the fragmenter module). Annotations requested by the slice consumer are subsequently inserted in the resulting combined fragments. At this stage, whenever a slice consumer requests slices to be focused upon an arbitrary concept, combined fragments are trimmed based upon the AlchemyAPI Dbpedia annotations available for each fragment and concepts selected [14]. Since the recomposition of fragments was performed linearly based upon their original order, with the exception of content focus related requests, the Dbpedia annotations did not affect in other ways the recomposition of fragments. Additional research investigating alternative types of semantically aware recompositions are currently in progress. For the purpose of this experiment slices were output in xml format. The result of this process consists of a so-called *web of slices* (figure 3b) constructed with RDF annotations, and available publicly as linked-data (web of data), pointing to annotations produced by Slicepedia and fragments, in turn pointing to original native web pages (web of documents). The Slicepedia implementation presented above therefore provides an open corpus CAS to slice consumers consisting of three right-fitting dimensions (Content Style, Granularity and Annotation Type) including ten adaptation variables (content style, topics covered, reading difficulty, verb chunk tense, number of annotations, paragraph/word number, topic focus, annotation focus, delivery format) that can be arbitrarily

combined to suit various content requirements over any relevant open corpus resources identified. A slice request hence could consist of the following: *slices should originate from only a speci ed list of urls and have a granularity ranging from 3 sentences up to 3 paragraphs. They should cover the topics of whale migration, atlantic ocean or hunting and should have a Flesh reading score ranging from 45 to 80. They should not contain any tables or bullet points lists but be focused on the speci ed topics (ie: exclude content not on these topics). Slices should contain between 7 and 15 annotations consisting of verbs conjugated at the past perfect continuous and should be delivered in xml format*.

4. Evaluation Design

4.1. Aim and Hypothesis

As discussed in section 2, in order to supply large diversities of niche content requirements, a content production service should guarantee the provision of large volumes of content (P_i), at low production costs (P_{ii}), suitable for arbitrary activities (P_{iii}). Although the first condition (P_i) is necessarily achieved through the selection of an open corpus reuse strategy, the performance of a content agnostic slicer with respect to the two other conditions is yet to be examined. For this reason, this evaluation focuses, as a first step, upon investigating the suitability and cost of content produced automatically by a content agnostic slicing system (fulfilling S_i, S_{ii}, S_{iii}) with respect to content manually produced for the same activity.

The following two experiments presented in this section were hence performed upon a sample of independently selected niche requirements and open corpus resources (S_{ii} & S_{iii}), sliced and recomposed within an independent AHS (S_i). Assuming positive results were measured, this would indicate content production systems based upon content agnostic OCS techniques could also scale for any niche requirement using any open corpus content. Any issues detected at this early stage would only be amplified within a large scale deployment making any scaling pointless. The hypotheses tested within this evaluation are therefore as follows:

- H1: Content manually & automatically produced achieves similar suitability results, from a users point of view (P_{iii})
- H2: Automated slicing offers a reasonable production cost solution for large volumes of content, in contrast with a manual production which is unsustainable (P_{ii})
- H3: Content produced by a slicer can be successfully recomposed within an independent AHS system (S_i)

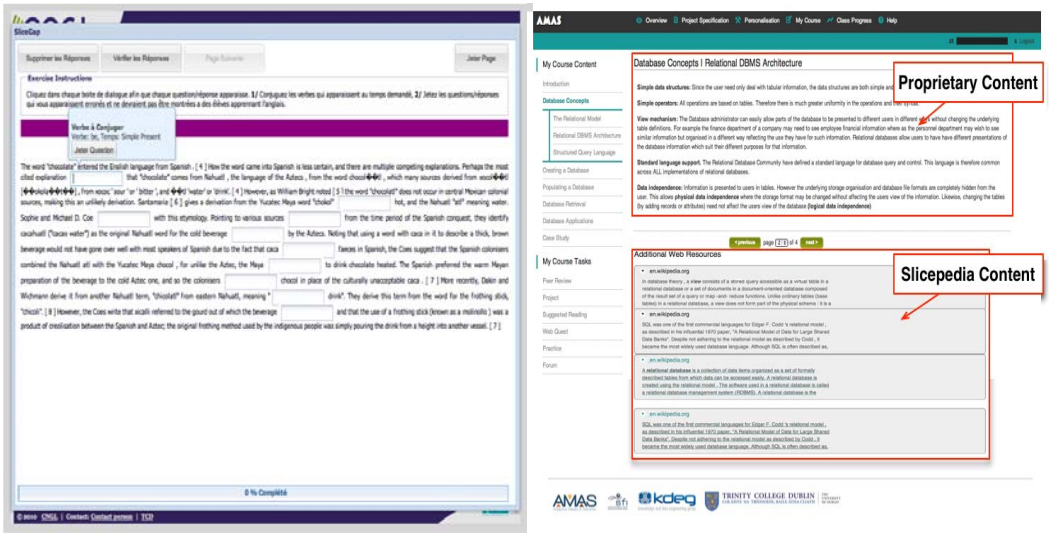
4.2. Reuse Vehicle Selection

Since content consumed by AHSs is ultimately presented to people, any content production measurement should consider user experience as critical. Furthermore, as the aim is to evaluate content produced by two different methods, slices should initially be assessed individually, with interface complexity and re-composition kept to a minimum (figure 4a).

For this reason, in order to perform the first experiment presented in this paper, a simplified version of the language e-assessment use-case application presented in section 2, to be named Slicegap, was implemented. Its purpose in this experiment is to evaluate the

content output of the slicer and hence must be seen in priority as a "content reuse vehicle" (i.e. not discussing educational aspects). This reuse vehicle also temporarily removes any interference potentially caused by AHSs slice consumer user interfaces.

This application represents a well known approach to grammatical teaching, via a computer adaptive testing tool, similar in nature to the item response assessment system introduced by Lee et al. [11]. For this reason, it provides a good example of the kind of educational application which could avail of third party content. In this application, users are presented with series of individual open corpus resources (involving no re-composition), repurposed (manually or automatically) as traditional gap filler exercises. Verb chunks items are removed and replaced by gaps, which users must fill according to particular infinitives and tenses specified for each gap. Answers provided are then compared to the original verb chunks and users are assigned a score for each specific grammar point. Slicing open corpus content is known to affect the reading ow and annotation quality of content produced [14]. An evaluation of content performed within the context of a language learning task (by individuals with varying levels of competency in the English language) should hence make participants very sensitive to the reading ow properties of content. Moreover, since grammar e-assessment units are created according to verb chunk annotations, any annotation quality issue would provoke major assessment errors, making any training over such content pointless and time consuming to users. While the Slicegap "reuse vehicle", used in this first experiment, provides the ability to assess content presented to users using minimal interface complexity, it nevertheless does not technically represent an authentic slicer/slice consumer relationship as described in section 3. While the open corpus resources provided as an input to Slicepedia were indeed unknown to its designers (S_{ii}), both the Slicegap slice consumer and language use-case scenario built and selected for the purpose of this experiment were known in advance by Slicepedia designers. For this reason, the previous slicer/slice consumer relationship comes short of fulfilling requirements (S_i) and (S_{iii}). The Adaptive Media and Services for Dynamic Personalisation and Contextualisation (AMAS) AHS [19], was therefore selected for a second experiment. Since the underlying architecture of AMAS represents a typical example of that used in most modern AHSs, this slice consumer appeared to provide a valid representative example of a large range of contemporary systems. It was built independently of Slicepedia (S_i), and offers a reuse scenario within the use-case of online personalised Structured Query Language (SQL) courses for undergraduate students, unknown in advance by Slicepedia designers (S_{iii}). As can be seen in figure 4b, in contrast with the previous experiment, this slice consumer recomposed both proprietary content (built by arbitrary educators specifically for the purpose of this course) with slices delivered by Slicepedia. Hence, in comparison with the previous experiment, results for the second experiment were measured through an independent AHS user interface. Slicepedia designers had no input regarding how slices delivered to AMAS would be recomposed or reused. They did not have any control either over what open corpus resources would be targeted for reuse nor what slice content requirements would be specified by the AMAS team (requirement A). For all these reasons, while the first experiment aimed at evaluating the first two hypothesis stated in the previous section, this experiment aimed at evaluating the third hypothesis instead.



(a) Slicegap (b) AMAS

Fig. 4. Reuse Vehicles Interfaces

4.3. Content Batch Creation

The manual content production activity selected for the Slicegap experiment deliberately involved only the repurposing of existing content (in the spirit of educator repurposing activities described in section 2). This decision aspired to replicate a manual content production scenario (used as a baseline) with minimal production time requirements. Our assumption was that any content authoring activities would always depict higher time requirements. Additionally, as the aim of a content-agnostic slicer consists in automating the reuse of any open corpus resource available on the WWW, the need to initially harvest a set of open corpus pages, arbitrarily selected by independent users and relevant for the purpose of a grammar e-assessment use-case, was necessary.

For this reason, the 80Legs web crawler used as part of the harvesting module in Slicepedia was temporarily ignored within the context of this experiment and a simple URL list-harvesting feature was used instead. This offered tighter control over the resources supplied to the slicer and allowed for a direct comparison between manual and automated reuse of identical open-corpus resources over a common subset of resources. Hence, prior to conducting the experiment, a group of five independent English teachers, from two different schools[16], were asked to arbitrarily select nine pages of their choice from the web (combined total of forty-five pages) and perform the following tasks (figure 5).

Slicegap Step 1) Page Identification: English teachers were first asked to arbitrarily select three sets of pages. Any page, constituting the first set, could be selected as long as they consisted of either encyclopaedia or news articles respectively (see section 3.2). The two other sets of pages could be of any source as long as they respectively covered an

[16] E.L.T.A. (Dublin, Ireland) and H.E.L.L.O. (Rouen, France) School of Languages

arbitrary set of concepts, or contained a set of ten verbs conjugated at an arbitrary tense. The entire set of pages identified were subsequently used as Native Content Batch (NCB).

Slicegap Step 2) Content Extraction and Annotation: They were then asked to select fragments, of any size, from the first set of pages harvested, which they felt could be adequate for English grammar exercises, and annotate each verb present within these extracts with the relevant tense and infinitive.

Slicegap Step 3) Content Focus: Finally, they were asked to extract fragments from the two other sets of pages, by ensuring these fragments were correctly focused respectively upon verbs annotated with tense Θ (such as present perfect) and content covering individual concept Γ (such as "human rights"). Content not about these topics was discarded). The Manual Content Batch (MCB) combines content produced within the last two steps.

Slicegap Step 4) Content-Agnostic Slicing: Open-corpus resources identified by teachers (MCB), were harvested in their native form and then sliced by Slicepedia to produce content batch Content-Agnostic Batch (CAB). This content batch represents a true set of open corpus resources since the origin and structure of pages selected were unknown in advance of slicing. Content adaptation spectrum characteristics (including topic focus requirements) of fragments manually produced and arbitrarily chosen by teachers, were used as arbitrary niche content requirement parameters for the slicer.

Slicegap Step 5) Slice Conversion: Slices present in all batches were subsequently converted into grammar e-assessment pages.

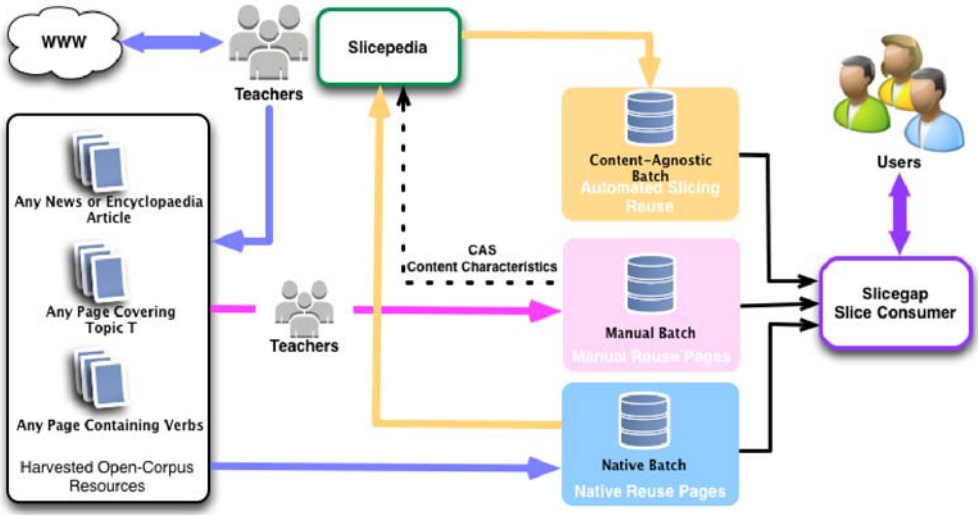


Fig. 5. Slicegap Content Batch Creation

With respect to the AMAS experiment, in addition to the proprietary resources already present within the AMAS AHS, Slicepedia content was incorporated within this AHS via the following four steps (figure 6).

AMAS Step 1) Open-corpus Repository Specification: AMAS designers were initially requested to provide an arbitrary list of online repositories containing resources they wished to target for slicing. The list provided to Slicepedia ultimately consisted of six websites, mostly consisting of online SQL tutorials.

AMAS Step 2) Slicepedia slicing: Slicepedia thereafter crawled, harvested and processed resources from these repositories (a total of 336 pages) in order to prepare them for reuse according to a Slicepedia request of type R1 (see section 3.2)

AMAS Step 3) Slice delivery: Finally, once all pages were processed, slices where generated (using Slicepedia requests of type R2) and delivered to AMAS in real-time based upon predefined SQL topics contained within the course and selection choices made by users.

AMAS Step 4) Slice recomposition: Groups of slices received per topic were finally recomposed by the AMAS system into an accordion type interface (see figure 4).

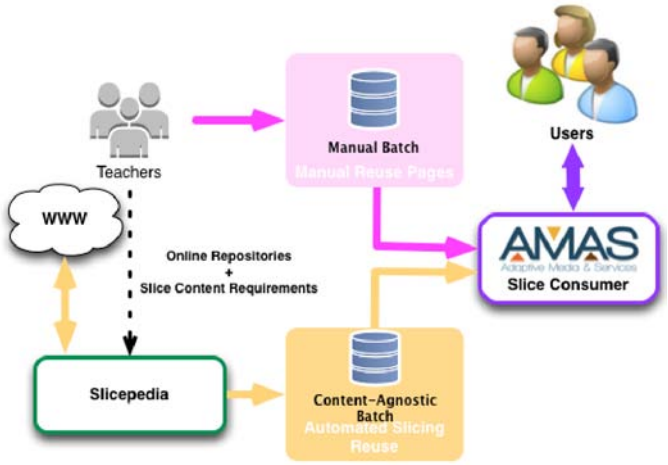


Fig. 6. AMAS Content Batch Creation

4.4. Evaluation Scenarios

Following the creation of content batches for both experiments, two independent user-trials (Slicegap and AMAS trials) were performed. As is explained in further details below, the Slicegap user-trial, which was made available online to the public, consisted of a task-based evaluation using real-life linguistic assessment needs. The AMAS trial on the other hand, used a different group of users consisting of undergraduate students, presented with an SQL personalised AHS platform, as part of a university database course. The evaluation carried out using the Slicegap trial aimed at addressing hypothesis H1 and H2 of this research, while the AMAS trial focused upon hypothesis H3.

Slicegap Scenario: Slicegap users were asked to perform a set of tasks in a single session without any interruption. Task completion time, in addition to user interaction,

was tracked throughout each activity. The trial performed consisted of a traditional language learning activity encountered in most textbooks. Each user was initially asked to select their interface language of choice and specify their level of ability in the English language (which was then used to separate users within Trainees and Experts categories). They were subsequently invited to perform successively the same activity using each of the three content batches generated for Slicegap. Each batch was associated with a unique colour and users were unaware of either how the content was created or what content batch was being presented to them at each task. Each original page (regardless of the batch it belonged to) was only shown once to each user. Users were asked to read the text presented to them and fill in any blanks encountered (ten gaps on average per page) with the appropriate verb and tense specified for each case (figure 4). The ability to provide users with accurate grammar assessment questions is highly sensitive to the precision of the annotations delivered by the slicer, therefore the previous task aimed to evaluate both the annotation quality and overall reuse appropriateness of the slices presented. However, this task did not necessarily require users to read the entire content presented to them (only the sentences containing gaps). For this reason, users were then asked to summarise what they felt the entire piece of content was about (as in traditional textbooks) in their own native language. In order to balance any effect of order bias, each user was presented content batches according to a Latin square design distribution.Since previous results related to the quality of content produced by a slicer appeared to indicate such a process can affect the quality of original resources with respect to readability and super uous content presented [14], the suitability of content within this experiment was measured based upon both of these two dimensions in addition to the ability for such content to support the activity performed by users (i.e correctly assess an individual). Finally, since non-native English speakers were also invited to perform the experiment, the Slicegap interface was also translated into Spanish and French.

AMAS Scenario: With respect to the second experiment, and as mentioned above, undergraduate students were presented with an SQL personalised AHS platform during one semester. This authentic case study was overall very similar to a previous study carried out by Staikopoulos et. al. [19]. For this reason, readers wishing to get detailed information about this system are referred to the original paper. As part of this course, students were required to complete various database related tasks (such as designing and implementing a database). For each activity, students were presented with SQL related proprietary material covering topics required to be mastered, which they could study in order to perform these tasks. In contrast with the original use-case presented by Staikopoulos however, the platform also incorporated content dynamically generated by Slicepedia on demand for each arbitrary topic and recomposed by AMAS as can be seen in figure 4. This experiment was run by the designers of the AMAS system independently from Slicepedia designers.

Following the set of tasks performed as part of both the Slicegap and AMAS experiments, users were presented with a questionnaire (one for each content batch in the case of Slicegap), answered using a ten and five point Likert scale for the first and second experiments respectively. The questionnaire provided to AMAS users, also contained additional AMAS related questions, which are not further discussed in this paper. Since a number of users who performed the Slicegap experiment were non native speakers, questionnaires for this experiment were also available in Spanish and French.

5. Evaluation Results

The rest of this section presents a summary of the findings observed throughout these experiments in relation to each hypothesis. The Slicegap experiment involved a total of 41 users, divided into two groups (Experts (63%) and Trainees (37%)), with most users using the English interface (en=66%, non-en=34%), while the AMAS experiment involved a total of 50 university students. Results obtained for hypothesis H1 and H2 were obtained from the Slicegap trial while results obtained for hypothesis H3 were obtained during the AMAS experiment.

H1: As pointed out in section 4, previous experiments indicated that the ability to remove super uous content from native resources had a significant impact upon the quality of reuse with respect to the activity performed. Hence, users were asked directly, for each content, whether *in addition to the main content, a lot of material displayed on the page was irrelevant to the task (such as advertisement, menu bar, user comments..)*. Figure 7 presents the results, using a Likert scale ranging from one to ten (with ten meaning full agreement), obtained by comparing pages from the same source (see section 5). Content batch NCB, containing pages in their native forms, achieved the worse score with a mean equal to 5.67, while manual MCB and automatically sliced CAB batches achieved much better scores (MCB=2.13, CAB=2.53) with paired t-tests confirming results are indeed significant ($p = 0\ 001$). Differences measured between MCB and CAB batches however were insignificant ($p = 0\ 423$) which would indicate that content automatically produced achieved similar performance to content that was manually produced.

When observing whether this performance had any impact upon the activity performed (both by asking users directly or by measuring the time taken to perform exercises), differences measured between native resources and the two other batches (figure 7) were statistically significant in both cases ($p <= 0\ 01$), which appears to confirm both the assumption stated by Lawless et al [10] earlier (see section 2) and results obtained previously [14]. Differences between MCB and CAB batches on the other hand, were again statistically insignificant ($p >= 0\ 278$). Nevertheless, when plotting together both the super uous removal performance and time taken to perform each exercise, batch MCB depicts the best performance in both cases, which would reaffirm results obtained in [14] concerning the positive impact of the super uous content removal on the reuse of resources.

While previous measurements aimed at estimating any decrease in quality of the content delivered from the point of view of visual layout, the ***readability of content*** produced for reuse is of course critical. For the purpose of this experiment, and for consistency purposes with previous results obtained [14] readability was decomposed into i) the ability of content to be easily understood, and ii) the extent to which the ow of reading is broken. Figure 8, depicts the results obtained for the first component measured. The first pair of questions directly asked users their opinion with respect to how easily content presented to them could be understood. The second pair relates to measuring any impact upon the difficulty in summarising content. As can be seen, results obtained across content batches are very similar. In both pairs of questions MCB and CAB batches interchangeably achieve slightly better scores with the highest difference between the two obtained for the first question ($Q1 - dif = 1\ 16$). However when performing paired t-test on these two content batches, differences appeared to be insignificant ($p > 0\ 143$). The same applies when comparing both MCB and CAB batches with the native content batch (NCB). Differences

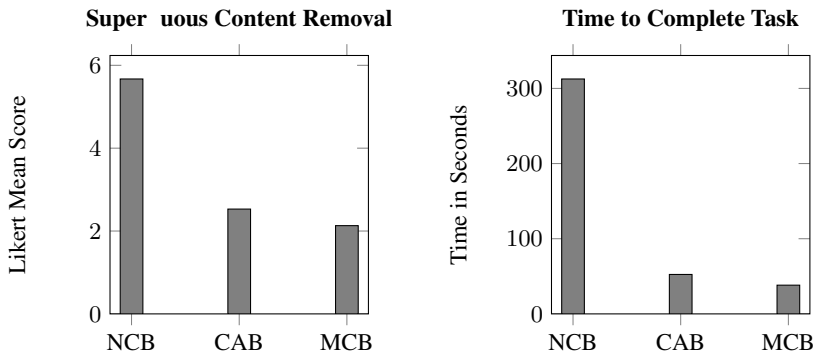
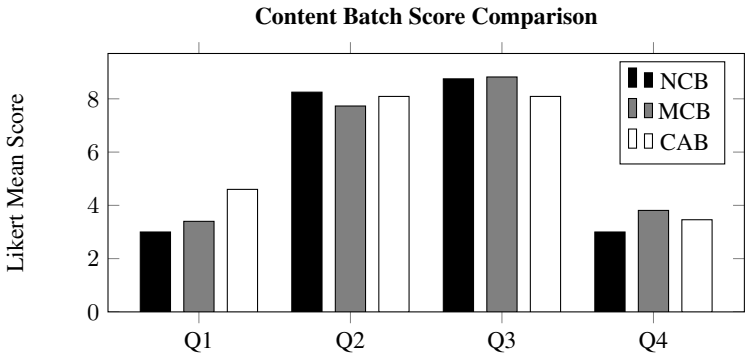


Fig. 7. Super uous Content Removal Analysis per Content Batch

measured between content batches appeared to be statistically insignificant with an exception occurring for question Q1 in which a difference of 1.6 between NCB and CAB could be measured. These results appear to reinforce the results obtained previously [14]. In contrast with the latter however, differences measured this time appear to be significant for confidence intervals of 95% ($p = 0\ 028$).

As a sanity check, and in order to make sure all sentences within the content presented were read (regardless of whether they contained gaps to be filled), users were asked to summarise the meaning of the content presented to them. Among all the summaries provided by users across content batches, only less than 5% were rated as weak. Despite being on topic, these answers were too short (less than 5 words) to determine whether the overall meaning of the content was correctly understood or not. Nevertheless, the overall quality of summaries submitted clearly correlates with the low values measured in figure 8. Since close to a third of users, who participated in the trial where non-native speakers, a Pearson correlation analysis between answers submitted to each question and contestant language level, was also performed. Correlations measured for each variable combination ranged between -0.2 and 0.2, which indicates no direct correlation between answers provided and the level of users. Additionally, when users were asked whether they felt *their language skills had any negative impact upon [their] ability to correctly summarise content*, means measured across content batches were all inferior to 2.30 (CAB = 2.29 and MCB = 2.19).

With respect to the second component of readability (reading ow), there exists, to the best of the authors knowledge, no automated mechanism to reliably estimate whether pronouns or topics mentioned within a fragment referred to earlier subjects mentioned in omitted parts of the original content. Hence users were asked directly whether this situation occurred or not for all content presented to them (Q5). As can be seen, content batch MCB achieved a good performance across all questions, however the difference between content batch CAB and both MCB and NCB (1.07 and 1.03 respectively) could be measured for question Q5. These differences were statistically significant when performing paired t-tests ($p =< 0\ 037$) for 95% confidence intervals. These results suggests content-agnostic slicing can indeed slightly deteriorate the ow of reading of sliced content. However when asking users if this situation had a negative impact upon the read-

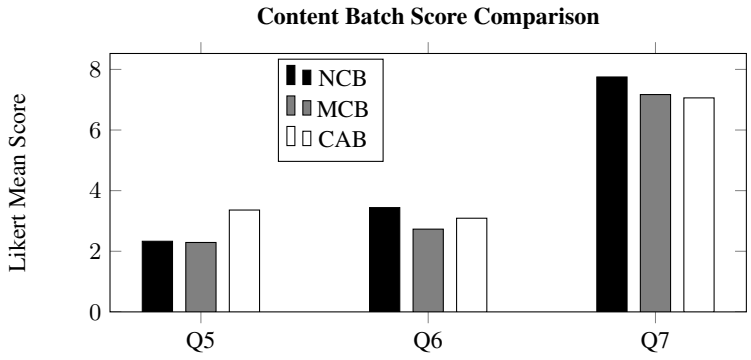


Q1 I felt this content was difficult to read
Q2 I felt I could understand easily what this content was about
Q3 I felt it was easy to provide summaries of the content
Q4 I felt the understandability of the content (as opposed to my language skills) had a negative impact upon my ability to correctly summarize the content

Fig. 8. Understandability

ability of content presented to them (Q6 & Q7), no statistical difference could be noticed between contents ($p > 0\ 300$). Hence, although a slight decrease in the reading ow of content sliced using a content-agnostic approach could be noticed, the resulting impact upon the overall readability of content delivered was marginal. The results obtained for both dimensions of readability appear to indicate that any differences measured between content produced using a manual or automated reuse approach were insignificant. Results measured between the NCB and CAB content batches however appeared to suggest decreases in reading quality caused by the automated reuse of content are large enough to be significant.

Statistical t-test analysis, of both the ***number of mistakes*** performed by users as well as the time required to perform e-assessments, revealed that any differences measured were statistically insignificant ($p > 0\ 100$). When trainees were asked whether, for content batch CAB, *"the number of erroneous assessment units presented was tolerable"*, a mean score of 7 out of ten was measured. When asked whether *"Overall, I felt this content was adequate to perform a grammar training exercises"* both content achieved very similar scores (MCB=8.33, CAB=8.57, p=0.536) with t-tests again suggesting any difference observed was insignificant. However, when plotting this data on a graph(figure 10), a pattern can be observed for both the trainee as well as expert group. In both cases, the number of mistakes and time taken to perform e-assessments, upon content created automatically, appears to be higher than for the content produced manually. This would suggest users do make more errors when using content from the CAB batch. Although automated verb chunk annotation recall accuracies measured outperformed those produced manually (Recall A=0.92, M=0.68), manual annotation precision was slightly higher than automated (Precision M=0.89, A=0.86), which could explain this minor increase in errors observed. Finally, although the difference in errors between content batches was slightly higher for trainees in comparison to experts group, an independent t-test indicated this difference was insignificant (mean dif. E=5.80%, T=7.48%, p=0.891). This indicates that



Q5 I felt that some pronouns within the text were referring to earlier subjects mentioned in paragraphs/sentences which were not included within the content presented.

Q6 I felt that missing parts of original paragraphs/sentences, had a negative impact upon the readability of the content presented to me (in terms of pronouns referring to earlier subjects mentioned in omitted paragraphs/sentences).

Q7 I felt that although parts of original paragraphs/sentences were missing, the content presented was easily readable. (with respect to pronouns referring to earlier subjects mentioned in omitted paragraphs/sentences).

Fig. 9. Reading Flow

users from the expert group didn't appear to use their language skills to compensate content suitability differences between both batches. Overall, these results suggest that although a pattern of slightly lower performances on assessments automatically generated was observed, this difference was insignificant and didn't appear to affect trainees more than the experts group of users, nor did it appear to decrease the perceived usefulness of the content for the assessment task performed.

H2: In relation to production costs, since production cost is dependent upon local prices of manual labor as well as individual server specifications, an estimation of production time was considered instead a better proxy for production cost differences. For this reason, the time required to produce content batches was measured for both the educators and slicer. Since the set of pages used as preliminary resources was purposely manually harvested from the web (section 4), no automated harvesting was performed by the slicer during this experiment. Nevertheless, in order to provide a fair comparison with a manual production approach, an estimation of time required by the slicer to perform this task was necessary. Teachers were asked to only harvest open corpus resources matching specific criteria combinations (such as topics covered, tenses etc...). Hence, an automated harvesting time estimation based upon traditional IR services would have created an unfair advantage towards the slicer, since these techniques only provide keyword searches with little guarantee that harvested resources satisfy these criteria. For these reasons, the Open Corpus Content Service (OCCS) focused crawler [10] was considered a fairer option since it provides the means to specify content to be harvested based upon a wider range of constraints (including topics covered) and also guarantees resources harvested, if any, meet these sets of constraints. Time measurements obtained for this experiment reveal that when no particular content requirement was specified, teachers took an average

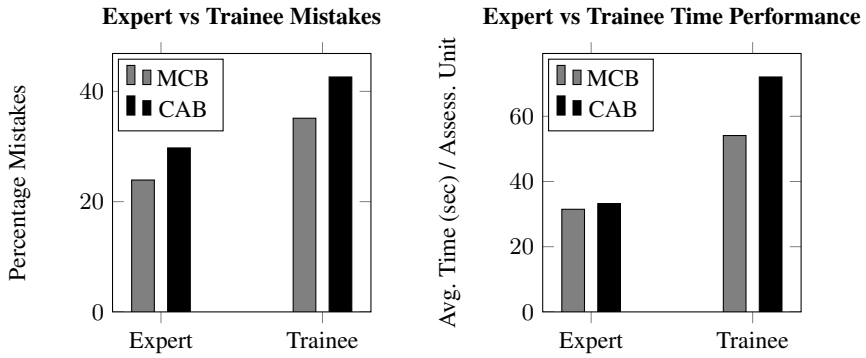
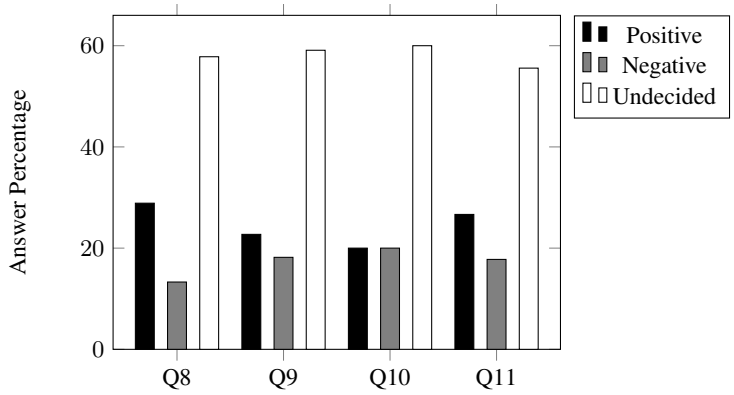


Fig. 10. Manual vs Content-Agnostic Results

of 3.75 minutes to harvest open corpus resources and extract arbitrary fragments suitable for grammar exercises. Requesting resources to be on a specific topic, only slightly increased the average time measured (4 min) whereas requesting resources to possess verbs conjugated at particular tenses nearly tripled the time needed (10.5 min). These results follow common sense, since the ability of humans to identify topics covered within resources is much more straight forward than for machines, however the reverse is also true when dealing with specific fine grained requirements such as verb tenses. Teachers on average took 4.25 minutes to annotate fragments (189 words in average, 14 annotations per fragments) leading to a total time ranging from 8 min to 14.75 min to produce these resources. This would be the equivalent of between 1.5 to nearly 3 years of manual labor necessary to produce a hundred thousand of such resources. According to Lawless et al. [10], a time performance of 149,993 valid resources harvested in 43h was measured for the OCCS system (without any cpu parallelisation). This is equivalent to $17\ 2 * 10^{-3}$ minutes of harvesting time necessary per page. Summing extraction, annotation and slice creation time performed on a 2.8GHz machine leads to a total of $5\ 4 * 10^{-1}$ minutes necessary to produce each page. Assuming no parallelisation was used during the slicing process (section 3), this already represents a difference of up to 96% production time increase with respect to it's manual production equivalent. Although automated and manual production time are clearly not directly comparable, one can assume in most cases, server costs per time unit to be much lower than labor costs. Considering the low server production time measured in comparison to the manual tasks, automated content production cost can be inferred to be also much lower than a manual approach and hence more adequate for large number of resources produced.

H3: Figure 11 presents the results obtained with respect to the content produced by a slicer for recomposition within an independent AHS system. As can be seen, results obtained for questions related to the Slicepedia content, recomposed within AMAS, depict a very large proportion of undecided users. This situation occurred as well on many other AMAS related questions presented to users. Although the authors of this paper chose a ten point likert scale to push users to make a judgement during the Slicegap experiment, the AMAS questionnaire on the other hand presented users with a five point likert scale. This decision might have induced users not to make any decision and choose a more neutral



Q8 The Additional Web Resources were of similar quality to the rest of the content presented within the website.
Q9 The Additional Web Resources were missing portions of content, making them hard to understand.
Q10 The Additional Web Resources contained irrelevant material which should have been removed to make them more easily understandable
Q11 The Additional Web Resources were beneficial in learning the topics covered by the course.

Fig. 11. Slicepedia resources within the AMAS platform

position overall. It transpired that, based upon user comments analysed while processing the results, some students had not noticed the Slicepedia content on the platform at all. This situation occurred depending on the screen size used by each student. In some circumstances, the Slicepedia content appeared below the level of the screen and required scrolling. Hence an additional trial run, with the AMAS interface taking into account screen sizes, is currently underway. Nevertheless, the results obtained do provide some indication into how Slicepedia content was perceived overall within the AMAS AHS. When asked whether the resources provided by the slicer *were of similar quality to the rest of the content presented* to them (Q8), 29% of contestants provided a positive answer in contrast with only less than half the amount (13%) responding negatively. This is a very positive results, suggesting the quality of content automatically produced by Slicepedia could perform reasonably well in comparison to that manually produced in advance for a dedicated platform. On the other hand, when asked whether *missing portions of content [made resources] hard to understand* (Q9), a slightly higher proportion of students agreed with this statement in opposition to those who disagreed (23% and 19% respectively). Although this represents only a 4% difference, this results appears to conform with reading quality decrease measurements obtained earlier for the Slicegap experiment. When asked whether these resources *contained irrelevant material* (Q10), an equal amount of users responded both positively and negatively. Finally, when asked whether these resources *were bene cial in learning the topics covered* (Q11), 27% of users responded positively with only 18% disagreeing. Despite a large number of undecided users, these results appear to indicate that the incorporation of Slicepedia content within this independent AHS provided a positive experience to users, both in comparison with proprietary content as well as in terms of support for this learning experience. This suggests a slicer can indeed

be built to support the production of content for reuse use-cases unknown in advance by its designers. These results also appear to reinforce previous results obtained in relation to possible decreases in reading quality.

6. Conclusion

When taking into account content batch production costs, the usage of automatically generated resources provides a clear significant advantage in contrast to those manually produced with respect to scalability. Hence, within the context of high speed low cost production environments, one could easily assume any content produced with unsatisfactory suitability to be discarded and rapidly replaced, which could compensate for any potential decrease in quality. Such quality differences were observed with respect to readability decreases. However, while content automatically generated appeared to provide a quality slightly lower than that manually produced, differences measured were most of the time statistically insignificant. Furthermore, results obtained when incorporating content produced by a slicer into an independent AHS indicate such an approach could indeed support the production of content for independent AHSs.

The ability of open corpus content-agnostic slicing techniques to produce large volumes of content on-demand, at very low costs and with a suitability comparable to manually produced resources for independent AHSs, would thus appear to represent a promising content production approach for the adaptive hypermedia community. As the experiments presented in this paper only took into account specific aspects of content quality (i.e. reading ow, annotations...) within specific educational content reuse scenarios, further empirical research will be required in order to validate this approach within a more diverse range of use cases. Additionally, if DCF algorithms are to be used in the future for OCS purposes, content dependency limitations discovered in a previously [12] should be addressed.

References

1. ICT FacTs and Figures. Tech. rep., International Communication Union (2011)
2. Armani, J.: VIDET : a Visual Authoring Tool for Adaptive Websites Tailored to Non- Programmer Teachers Jacopo Armani. Educational Technology & Society (2005)
3. Brusilovsky, P., Chavan, G., Farzan, R.: Social adaptive navigation support for open corpus electronic textbooks. In: Proc. of the 3rd int. conf. on Adaptive Hypermedia and Adaptive Web Based Systems (2004)
4. Cobuild, C.: English Grammar (1999)
5. Diaz-aviles, E., Fisichella, M., Kawase, R., Nejdl, W.: Unsupervised Auto-tagging for Learning Object Enrichment. In: Euro. Conf. on Tech. Enhanced Learning (2011)
6. Henze, N., Nejdl, W.: Adaptation in Open Corpus Hypermedia. IJAIED'01: Int. Journal of Artificial Intellligence in Education pp. 325 – 350 (2001)
7. Hepple, M.: Independence and Commitment: Assumptions for Rapid Training and Execution of Rule-based POS Taggers. In: ACL'00:Proc. of the 38th Annual Meeting of the Association for Computational Linguistics (2000)
8. Jednoralski, D., Melis, E., Sosnovsky, S., Ullrich, C.: Gap Detection in Web-Based Adaptive. In: Proc. of the 9th int. conf. on Web-based Learning (2010)
9. Kohlschütter, C., Nejdl, W.: A Densitometric Approach to Web Page Segmentation. In: Proc. of the 17th int. conf. on Information and knowledge management (2008)

10. Lawless, S.: Leveraging Content from Open Corpus Sources for Technology Enhanced Learning. Ph.D. thesis, Trinity College Dublin (2009)
11. Lee, Y., Cho, J., Han, S., Choi, B.u.: A Personalized Assessment System Based on Item Response Theory. In: Proc. of the 9th int. conf. on Web-based Learning (2010)
12. Levacher, K., Lawless, S., Wade, V.: An Evaluation and Enhancement of Densitometric Fragmentation for Content Slicing Reuse. In: CIKM'12: Proceedings of the 21rst ACM Conference on Information and Knowledge Management (2012)
13. Levacher, K., Lawless, S., Wade, V.: Slicepedia: Automating the Production of Educational Resources from Open Corpus Content. In: EC-TEL'12 (2012)
14. Levacher, K., Lawless, S., Wade, V.: Slicepedia: Providing Customized Reuse of Open-Web Resources for Adaptive Hypermedia. In: HT'12: Proc. of the 23rd Conf. on Hypertext and Social Media (2012)
15. Lin, Y.l., Brusilovsky, P.: Towards Open Corpus Adaptive Hypermedia : A Study of Novelty Detection Approaches. In: Proc. of the 19th Conf. on User Modeling, Adaptation and Personalization (2011)
16. Madjarov, I., Boucelma, O.: Learning Content Adaptation for m-Learning Systems : A Multimodality Approach. In: ICWL'10. pp. 190–199 (2010)
17. Meyer, M., Rensing, C., Steinmetz, R.: Multigranularity reuse of learning resources. Transactions on Multimedia Computing, Communications and Applications (2011)
18. Savic, G., Segedinac, M., Konjovic, Z.: Automatic generation of E-Courses based on explicit representation of instructional design. Computer Science and Information Systems (2012)
19. Staikopoulos, A., Keeffe, I., Rafter, R., Walsh, E., Yousuf, B., Conlan, O., Wade, V.: AMASE : A framework for composing adaptive and Personalised Learning Activities on the Web. In: Proc. of the Int. Conf. on Web-based Learning (2012)
20. Steichen, B., Connor, A.O., Wade, V., Oconnor, A.: Personalisation in the Wild Providing Personalisation across Semantic , Social and Open-Web Resources. In: HT'11: Proc. of the 22nd int. conf. on Hypertext and Hypermedia. pp. 73–82 (2011)
21. Taylor, P., Aroyo, L., Houben, G.j., Vdovjak, R., De Bra, P.: Embedding information retrieval in adaptive hypermedia : IR meets AHA ! New Review of Hypermedia and Multimedia (June 2012), 37–41 (2007)
22. Weissig, Y., Gottron, T., Darmstadt, T.U., Mainz, J.G.u.: Combinations of Content Extraction Algorithms. In: Workshop Information Retrieval. pp. 1–4 (2009)
23. Zhou, D., Truran, M., Goulding, J.: LLAMA: Automatic Hypertext Generation Utilizing Language Models. In: Proc. of the int. conf. on Hypertext and hypermedia (2007)

Killian Levacher is currently pursuing a PhD in the School of Computer Science and Statistics in Trinity College Dublin, Ireland, and holds a B.A.I and MSc in Computer Science from the same university. His research interests, for which he received multiple awards, include open corpus content slicing, adaptive hypermedia, semantic web and eLearning technologies. Killian is also actively involved in entrepreneurial activities, applying novel slicing and personalisation technologies in the field of language elearning.

Seamus Lawless is an Assistant Professor in the discipline of Intelligent Systems in the School of Computer Science and Statistics in Trinity College Dublin. He is also a Principle Investigator in the SFI-funded Centre for Next Generation Localisation (CNGL) project and is a senior researcher in the EU FP7 CULTURA project. His research interests are in the areas of information retrieval, information management and digital humanities with a particular focus on adaptivity and personalisation. Samus is a co-founder

of Emizar, a CNGL and Trinity College Dublin spin-out company, which specialises in delivering personalised customer support solutions for web self-service.

Vincent Wade is Professor of Intelligent Systems at the School of Computer Science and Statistic, Trinity College Dublin and holds BSc., MSc and PhD degrees in Computer Science. In 2002 he was awarded Fellowship for his contribution to research in Adaptive Systems. He is Director of CNGL - a world leading research centre focused on Global Intelligent Content and is Academic Director and Founder of the Learnovate Centre for Learning Technology. Vincent has authored almost 300 publications in peer reviewed international journals and conferences and has led over 18 European Union/International Research projects, with significant research funding awards (50m) over the last 12 years. Vincent also previously founded three (spin out) technology companies (Empower The User, Emizar and Wripl) in the areas of Soft Skills Training and Simulation, Customer Care Management, and Personalised Web Services.

Received: December 23, 2012; Accepted: December 5, 2013.