Representing Aggregate Works in the Digital Library

George Buchanan
Future Interaction Laboratory
Swansea University
Swansea, UK
g.buchanan@cs.ucl.ac.uk

Jeremy Gow
Ann Blandford
UCL Interaction Centre
31/32 Alfred Place
London, UK
j.gow@cs.ucl.ac.uk
a.blandford@cs.ucl.ac.uk

Jon Rimmer
Claire Warwick
UCL SLAIS
31/32 Alfred Place
London, UK
j.rimmer@ucl.ac.uk
c.warwick@ucl.ac.uk

ABSTRACT

This paper studies the challenge of representing aggregate works such as collected poems and journals in heterogenous digital library collections. We demonstrate the complex range of aggregate types and the problems of faithfully representing this in the digital library interface. Aggregates are complex and pervasive, challenge many common assumptions and confuse the boundaries between organisational levels within the library. Critically, we show that existing DL systems can only provide imperfect representation of aggregates, and that alterations to document encoding are not sufficient to create a faithful reproduction of the physical library. The challenge is illustrated by referring to concrete examples, and solutions are demonstrated in a well-known DL system and related to standard DL architectures.

Keywords: Digital Libraries, Collection Building

1. INTRODUCTION

Digital libraries started as collections of documents that were usually of the same genre and form: e.g., a library collection of sheet music, oral history recordings or development literature. Items were distinct works by specific authors, and this simplified the creation of large libraries. In contrast, aggregate documents bring together several discrete items into one "whole": e.g. a collected volume of Irish short stories, or multi-volume books [15].

For many information seekers aggregates play a critical role: a literature scholar searching for a poem by Seamus Heaney; a historian hunting for a memoir knowing that the original is destroyed; a scientist hoping to find a special issue on their topic; a researcher in Victorian literature wishing to obtain Dicken's novels in their original serialised form. In each case, the goal may hide inside an aggregate with no clue to the true content: would you expect to find the only reproduction of the latest poem by Seamus Heaney, an Irish Poet, in a book on Welsh literature?

Without corresponding metadata there is, clearly, no trace of Heaney's tribute to a Welsh academic. However conflating all the metadata for (say) a volume of poetry will reduce search precision through "metadata spamming". Each distinct work should be identified separately to ensure precise matching for searches and consistent browsing and reading.

This paper will demonstrate that existing DL systems cannot be made to fully represent aggregate documents in a heterogenous collection without changes to the underlying components of the DL. Changing the *coding* of ingested documents alone cannot address fundamental behaviors of DL systems – i.e. changes to input documents cannot extend the inherent *functionality* of the hosting DL software. The complexities of aggregates are sufficiently great to mean that 'solutions' based on encoding–only approaches provide at best a partial solution to the handling of aggregates.

1.1 Background

Our study of humanities academics [6] has focussed on the digital provision of 'born paper' literature. While creating collections of material dating from the 5th to 19th centuries CE, we have found significant challenges in representing historic material in a manner consistent with the original.

We have discovered that *aggregate works* are of key importance to humanists and frequently occur in humanities collections. Discovering the correct item that academic references identify is critical to a correct understanding of journal articles, etc. In repositories of historic material – libraries and archives – works originally written or published separately are bound together for the purposes of (physical) long-term storage. When they appear electronically, the DL needs to reflect the physical binding that has been referenced in the past. Reference can be made to items which though the original is now destroyed (e.g. due to war) still exist in part or whole in earlier publications. The item is still referred to directly, but now needs to be identified in regard to published copies, rather than as a reference to the destroyed original. As we shall show, the complications of aggregate forms and representation provide some illogical consequences in the digital domain.

1.2 Aggregates in the DL

It may appear that aggregates are readily stored in a digital library without changing the DL software itself. Intelligent selection of coding of ingested documents can, surely, resolve all possible problems? One common and simple aggregate is the journal. If a collection is built of individual journal articles, then one document can consistently represent one article, an issue of the journal is a set of articles, and a volume a set of issues, etc. It would appear logical that a similar approach should be effective for other aggregates. Unfortunately, that is not the case.

As we will demonstrate during the course of this paper, this simple situation misleadingly avoids the more complex requirements that appear when a heterogenous collection is built. A printed volume may contain several short stories, or simply be a part of a single work, longer articles may be serialised across different issues of a periodical yet form a single whole, and users sometimes search for works that have an independent life and identity, yet appear only as part of larger documents. When only one challenge must be met, e.g. a run of one journal, simple solutions suffice.

Readers will, naturally, use whichever reference data they have, and weakly defined searches will behave very differently if the library distinguishes between individual works, or subsumes them within a larger entity. For a library to support effective searching and browsing, retrieval must be reliable under both criteria.

When a comprehensive and elegant implementation is required for a library that contains a wide variety of aggregate types, ad–hoc document encoding methods (e.g. adding new metadata fields) fail and cannot provide functions that are not already encoded into the DL system.

1.3 Summary

The inherent complexity of aggregate support has already been identified by other researchers (e.g. [8]). However, solutions to the aggregate problems are absent from DL literature, and a systematic analysis of aggregate documents in the DL has yet to be undertaken. We and others have recently addressed models of individual library items during ingest and indexation [10, 5], and document versioning [4]. It is now timely to address aggregates in their turn.

This paper proceeds in five parts: first we enumerate different types of aggregate documents, to clarify the later discussion. Secondly, we discuss how these aggregates could be represented using current DL systems. Subsequently, we review user interaction issues through a user study, and then proceed to identify some solutions and outstanding issues. The paper closes with a discussion of related literature and the course for future research.

2. AGGREGATE STRUCTURES & TYPES

This section introduces sample types of aggregate works in historic literature. First, it is worth clarifying our model of what an aggregate is. We presuppose the existence of some kind of 'document unit'. An ordered series of these may be collected together to form an aggregate, which can in turn form part of another aggregate. So, aggregate works are ordered trees with documents units at the leaves.

In this paper we restrict ourselves to textual examples. Units may themselves be documents, or parts of documents – where aggregation stops and internal document structure commences is not necessarily clear, and often depends on the library users' needs and library administration resources. Such issues are beyond the scope of this paper.

In the following discussion, we enumerate some features of aggregate works. For clarity each aggregate type is given an identifier 'AGn' . Note that the features are not all mutually exclusive, so a single aggregate work may exhibit several.

Homogenous Aggregation (AG1) The example above of journal articles aggregated in issues can be termed *homogenous aggregation*. Each aggregated unit is of the same type. However, many journals also contain editorial content and indexes. Articles themselves may be, say, original academic contributions or reviews of other literature (e.g. published books). Therefore, the pure homogenous collection is actually rather rare, as our examples below illustrate.

Heterogenous Digital Forms (AG2) Though an aggregate work may be logically homogenous, its digital form may vary internally. This can arise where an aggregate has been digitised over a period in which digitisation practice shifted. For example, we encountered this situation while handling content from the Oxford Text Archive[1]: some works were initially encoded in COCOA/Tact transcripts, but later parts were both scanned and supplied with a TEI transcript. Such heterogeneity clearly complicates ingest.

Serial Aggregation (AG3) Forms of aggregation emerge from the requirements of publishing. Journals – a common aggregation – appear at (semi–)regular intervals. This also happens with larger works published over many years: e.g. Sir John Fortescue's "History of the British Army" was published in twenty parts over a twenty-five year period (in the case of the first edition) and thus each part has its own year of publication. This work is even more complex as some parts are multi-volume themselves, and some parts refer to the same historical period yet were printed at different dates.

Binding Aggregation (AG4) Emerges from the technological limitations in printing: a work was published as one item, bound in separate volumes: though clearly one work, digitally we may need to treat the work as being of separate parts in order to support accurate retrieval (e.g. from references that distinguish a volume of the work).

Composite Aggregation (AG5) A key historic form of aggregation *within the same work* was its serialisation within a larger aggregate work. Unlike **AG3** where the parts are printed independently, *Composite aggregation* occurs when the parts of a work appear mixed with other items. 19th century fiction, including famous works by Charles Dickens & Sir Arthur Conan Doyle were published in this way: e.g. Sherlock Holmes appearing in *The Strand* magazine. Though now available as consolidated works, their original printed form is the aggregation of the separate columns and pages in the containing periodical. Thus, what is from one point of view a part of a newspaper is at the same time a

[1] http://www.ota.ox.ac.uk/

part of a novel. This is an important historical aggregation which is of strong interest to humanities researchers.

Containing Aggregation (AG6) A work may be small and only appear within larger works. However, those larger works are not simply aggregates themselves. This is a point where aggregation and internal structure collide. A practical example can be found in the poems found in A.A. Milne's stories of Pooh: the poems have a "life" of their own, independent of the holding work. Such contained works may not be by the same author: e.g. T.S. St.Clair's "A residence in the West Indies and America" contains a chapter this is an independent work by his brother on Martinique. The book as a whole is not a composite **AG5**, but it does contain in its entirety an article not by the author, and often referred to directly without reference to the enclosing work.

Heterogenous Aggregation (AG7) As noted at the beginning of this section, heterogenous aggregation is more commonplace than we may think — even within modern journals. In what may be termed 'simple' heterogenous aggregation a work is created from units of diverse types. For instance, newspapers and journals contain articles of different types that may need to be distinguished in the DL interface (obituaries, articles, reviews). Similarly, libraries bind tracts – small texts printed without hardback bindings – into sets by size. It may be necessary to reconstitute such arbitrary aggregations in digital form for reference purposes.

Supplementary Aggregation (AG8) A common form of aggregation in historical literature is where an original work is supplemented by further material, possibly by another author. For example, a volume of memoirs may be published with a foreword and/or additional notes. These are also known as *augmented works*, but aggregation may be a more appropriate model where the additional material is substantial, or referenced independently.

Incomplete Aggregation (AG9) Some aggregates are incomplete, either because they were not fully published or because a collection may be incomplete. Some works were published with the intention of being part of a series, but the series was never completed, leading to "missing" volumes.

Variable Aggregation (AG10) Different versions or editions of an aggregate work may bring together different material, or different versions of the same material: e.g. the New Testament is an aggregation, where earlier forms often omits books found in modern versions and vice versa.

As noted above, the boundary between dealing with external and internal document structure is not fixed, and many of these "aggregation" issues may also occur *within* a given document. For example, a journal or diary of historic interest may be considered to be a homogenous aggregate of diary entries, perhaps with supplementary hetergenous content from letters received by the author. What is important, from the view of a DL system, is that the treatment of internal and external aggregation are treated consistently in the DL architecture and also in the user interface, to ease the task of readers and librarians alike. Having surveyed some of the important features of aggregate works, we now look in more detail at their use in digital libraries.

3. REPRESENTING AGGREGATES

This paper is motivated by our need to represent historic literature in a digital library. The section will first identify existing tools within DL systems that could be used to support aggregation. Second, simple forms of aggregation are studied that will establish foundations from which design choices can be formed for more complex situations. The section finally examines the representation of complex aggregate types from Section 2. The problems that emerge from simple renderings of these aggregates are identified, and where solutions have been achieved these are reported.

3.1 DL Tools for Aggregation

Before discussing aggregate support in detail, we will briefly review the existing tools within DL systems that could be exploited. We have used, and focus upon, the well–known Greenstone DL system [19], but we will also refer to other systems, namely DSpace [16] and Fedora [9].

Aggregation is essentially a hierarchical structure, where a leaf node may be a whole work or, more rarely, a specific part of one. In using existing DL tools, we must thus necessarily focus on DL facilities that support hierarchical organisation and composition. As noted in [1], there are a few key elements that underpin most DL systems and most DL protocols. Two structural elements are key to handling aggregates: *documents* and *classifiers*.

Classifiers provide the most commonplace support for hierarchical structures over a set of documents, and they are supported by each of the key DL systems in question. Each of the three systems provide classifiers within a collection of documents: e.g. Greenstone, provides lists, alphabeticised lists (as used in encyclopaedia), and full hierarchies. All these options can be viewed as some form or other of hierarchy. Each item in the hierarchy typically has a human–readable description, and (optionally) an abstract code. A classification node will also contain a list of its children – be they individual documents or sub–classifications.

Documents, on the other hand, have a wider range of representation. Whilst each system allows metadata to be applied to each document, the DL systems provide varying metadata schemes – e.g. MARC or Dublin Core. However, this variation is of only limited significance in handling aggregates.

A key feature of documents is the concept of internal structure. Fedora and DSpace primarily approach documents as bytestreams [9]. Though this facilitates the handling of content in binary format, it reduces the concept of the document to a single package of metadata, together with a set of discrete binary sources. In contrast, Greenstone [19], views documents themselves as a hierarchical structure, and a document's source may be either whole or part of one or more files. These distinctions carry into the treatment of documents during indexation.

In Greenstone, a search may be performed over the parts of all documents, or across the whole documents alone. In contrast, only whole documents are indexed in DSpace. Greenstone also stores metadata for each part of a document, as well as the document as a whole. This distinction is a critical difference within the context of this paper: where document

parts can be indexed separately, we acquire a second tool besides classification for representing aggregation.

Note that DSpace can use the Lucene search engine for document content indexation. Though Lucene does provide for document parts to be separately indexed, the authors have found no evidence that there is the necessary counter–part within DSpace itself to model internal document hierarchy.

3.2 Traditional Solutions

We commence our examination of providing aggregate support by examining the indexation of multi–volume works in the physical library using card indexes. The traditional approach taken for aggregates which are indexed separately by parts is to add metadata to each part of the aggregate, indentifying either its parent aggregation, or its sibling parts. In a DL, this classical approach can be realised in two ways: either by a simple reinstantiation of the metadata tactic, adding metadata items to each part, or by a hyperlinked reference in the same manner. In the latter case, the metadata may retain the previous textual content, but this is enriched by using hypertext links within the metadata.

However, this enriched approach, though placing less effort upon the reader, suffers a number of problems which should drive us to find a superior solution.

One immediate issue is ensuring that the link is a permanent and reliable reference to the aggregate parent or other parts. The use of DOI or similar permanent references is a partial remedy, but itself suffers shortcomings. Variations in access rights to different instances of the same part document cause complications: e.g. the DOI may resolve to a copy of the part which is not available to this particular reader. Permanence is less easily resolved than we can readily realise within present technologies and practices.

A second issue is more durable: encoding aggregate data within the catalogue as metadata upon the parts is, simply, an ineffective approach. There is no single canonical instance of the aggregate data: rather, it is embedded in a complex network of links that may have simple inconsistencies or errors in the representation of a specific aggregate. In consequence, extracting the data on an aggregate suffers all the problems of crawling a network to extract data: it is often slow, prone to error and possibly incomplete. No sound solution can countenance such hazards.

Both these issues have been addressed in classic hypertext by the concept of *linkbases* [7] – databases of links that may be applied to any document upon retrieval. Linkbases provide a proven solution to the maintenance issue, and will also provide a support for more complex situations. However, none of the three key DL systems that we have chosen to examine provide a proper equivalent of a linkbase, and so we cannot use this approach with extending or modifying the DL system software.

To summarize, though we may wish to reproduce the helpful cross–references in the library catalogue, we must seek to do so in a discrete, well–structured and dependable form which is better suited to a scientific approach to the problem of representing and storing aggregate works systematically.

3.3 Simple Cases

Some common forms of aggregation are already well understood in the digital domain. The online access of academic journals is now commonplace. Each issue is treated as a homogenous aggregation of articles, and each volume as a homogenous set of issues. Such regular structure is easily represented in any DL: each article is stored as a separate document; each issue is recorded as a node in a hierarchy; and each issue node is in turn a child of a volume node. Frustratingly, this treatment proves sub-optimal in certain circumstances: when a search is run, the result list is of documents, and only inspection of discrete document metadata can identify where, for example, a particular issue (e.g. topical special issue) of the journal frequently matches the search. Sec. 4 will show that this is a real problem for users.

Users in the humanities often need to search for particular forms of article, e.g. they may seek reviews of books on a topic. This then results in a need to search by genre. Some systems such as JSTOR[2] permit this as an advanced option. This effect is readily achieved by adding the genre as a metadata property of the collection: the collection remains, in indexation terms, homogenous, but selectivity can be achieved by adding metadata criteria to particular searches. Thus, simple heterogeneity can be represented whilst retaining a simple underlying digital representation of the material.

Thus, we can readily represent chronologically serial (**AG3**) aggregates with simple heterogenous (**AG7**) or homogenous (**AG1**) material using standard DL software. Using a combination of hierachical classification and metadata, the regular hierarchy of the journal format is easily mirrored through browsing hierarchies and genre distinctions between articles can be made using document metadata. This underlying solution can be reproduced in the DL interface – e.g. including a genre option in the search tools available to the user.

This solution is, however, not complete: search results may display the parent aggregate of each hit, but the aggregates themselves are not indexed as documents, and functional changes to the DL software would have to be made to consolidate related parts within the result list if this were desired. No DL system would distinguish 'aggregate' from 'classifier' hierarchies, leading to confusion by the reader. As with the traditional solution, a the reader must work to tie together parts of the same aggregate.

Furthermore, DL systems frequently distinguish between 'classifiers' and 'documents' when it comes to indexation. Documents are indexed, whereas classifiers are themselves a form of index, and properties such as their titles, etc. are not indexed. This readily leads to shortcomings during interactive search: if a user has the title of an aggregate that they wish to find, the search will often fail, as the title is, from the viewpoint of the DL, of a classification, not of a document. An apparent solution to this dilemma is to include the title of the aggregate within the title of the document it contains. However, this in turn causes difficulties where a document occurs within more than one aggregate. Again, in some cases a simple approach proves effective, yet it clearly appears suboptimal as complexity increases.

[2] http://www.jstor.org/

Classification is, itself, a source of further complications. In Fedora, DSpace and other common DL systems, it is quite normal for a document to be included in more than one classification. However, classifications are hierarchical trees – each node belonging to only one parent. When an *aggregate* is itself treated as a document, it can appear in one or more classifier nodes. When an aggregate is itself represented as a classifier, then a dichotomy emerges: a classifier node (all or part of the aggregate) must have multiple parents. The solution to this is to replicate the aggregate hierarchy in the leaves of the topic hierarchy. This duplication leads to many easily anticipated problems: adding or removing the aggregate from a classifier becomes a more complex process, and ensuring consistency if the aggregate is modified in any way.

A second issue also emerges: that the multiple instances of the aggregate are not explicitly related. Therefore, it may not be clear that the elements of two separate classifications that represent the same aggregate are, in fact, the same.

Finally, whatever functionality is available in the DL interface to relate items in a search result list to their parent classifications, there cannot be a proper semantic distinction made in the live DL system between those parts of a classification hierarchy that are actually members of a classification, and which parts represent aggregational hierarchies. If aggregates are never represented in a topical classification, then a distinction can be made between "topical" hierarchies and (e.g.) "journal" hierarchies, but this in turn means that aggregates again disappeear from the topical ones!

In many simple libraries of homogenous aggregates such as journals, there is little or no use of classifiers for topic hierarchies for the journals or, say, an issue as a whole. Where there classifiers are used, it is on individual articles. This avoids the complications we have just described, however it is quite clear that, for example, a collected volume of poetry, a composite such as the *Gang of Four*, or a multi-volume history is indeed an entity in its own right.

We cannot therefore simply use topic hierarchies as a substitute for separate representation of aggregates in any of the DL systems under consideration.

The 'aggregates as classifiers' solution is considerably superior to the classical approach and readily realised in any majar DL system. However, it is clearly not a perfect solution, and unless the role of aggregates is merely as navigational classifiers (often the case in collections of journals), problems emerge both in targetting the aggregate as a whole during search (e.g. the name of a poetry collection) as classifiers are not search targets, and also in consistency and maintenance when adding the aggregate to a topical hierarchy.

3.4 Conflicting Structures

Though the regular form of journal collections results in few problems, it is easy to identify problematic situations that differ in only small, yet critical, details. If the aggregation hierarchy is not the same for all library content, or is even potentially ambivalent, problems rapidly multiply.

We have already noted in the previous part the problems that emerge when a document occurs within more than one aggregate. Another difficulty frequently emerges when a collection is built of pieces of literature. The scale of each item varies from a short novella to a multi-volume "epic". In such situations, if we faithfully replicate the physical text in digital form (critical for supporting referencing to particular pages) some texts will be formed from separate volumes, whilst the majority are contained in only one. Conversely, a single volume may contain several discrete works and be known to the community both separately and as a combined volume. The concept of 'volume' thus becomes problematic: it may be both a container of or a part of a work.

Indexing a collection by volume only conflates works in the same volume, whilst indexing by works only conflates volumes of the same work – leaving reader to separate the parts during retrieval. Clearly, neither solution is optimal: the natural conclusion is to index by whichever unit (work, volume) is the smaller and aggregate upwards to unify elements of the same item. The underlying storage can be represented in different ways in the library interface: e.g., matches against a single search for separate volumes of the same work can be unified in the search result list, and retrieval made by volumes. This option is available within documents in Greenstone (see [19] for details), and can be achieved in DSpace with careful configuration.

During browsing, however, the contradictory use of volume (as a part or as an aggregate) will still emerge. One can distinguish the part-of and aggregate-of styles of volume through a three-level hierarchy and using discriminating labels for top and bottom levels. Many items are represented by only one item at each level, and as reported in [17] such simple single-child relationships should be pruned so that unnecessary interaction is minimised. Thus, to improve the interactional efficiency, the experienced hierarchy becomes irregular. Bringing together different elements from the same hierarchy node in search result lists remains a problem (though this can be achieved in Greenstone).

The difficulty of consistent "levels" is not simply esoteric: where full–text indexation is provided (e.g. in Greenstone or DSpace with Lucene enabled), the search engine provides search results from the items at one particular level. Every item at a particular "level" will be treated as of the same "type" as the others. Clearly, well–considered ingest will produce a family of indexes. Nonetheless, the administrator must consider which items are peers of others, and an appropriate default setting. No single setting under the current model will provide matching services for multiple levels (e.g. for a collection of poetry and for its consistuent poems) at one moment without the duplication of items.

3.5 Difficult Cases

So far, we have examined simple cases of aggregation that can be partially resolved in the DL interface. Now we turn to more complex cases that create increasing degrees of difficulty. For reasons of space, we focus here on *containing aggregation* (**AG6**) and *composite aggregation* (**AG5**).

Composite aggregates (**AG5**) represent particularly problematic structures. Serialised fiction such as Conan Doyle's *Gang of Four* disrupts DL assumptions in its original form. One serial – the *Strand* magazine – contains another.

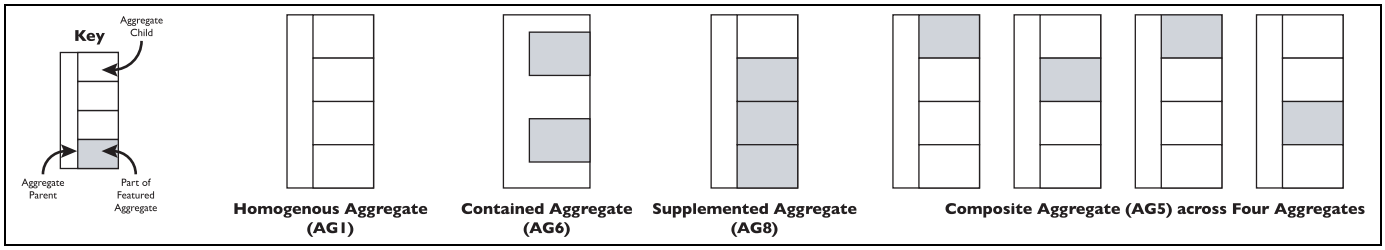

Figure 1: Different aggregate types

If we naively record each magazine as a single document, then the reader would need to map their information need (to read the Gang of Four in its original context) to particular editions of the correct publication – detailed knowledge that the reader may well not have. The reader must would also have to search for, and inspect, the particular editions that contained part of the story.

A better approach would be to extract and record the elements of the story as one DL document, tidily avoiding the problem for a searcher specifically looking for the Gang of Four, but conversely divorcing it from its original context: to connect each article with its context in the original magazine, the user must in fact engage in the 'hunting' of articles we apparently just avoided. Such contextual interpretation is the nub of many practices in humanities research. Clearly, an optimal approach allows both the recovery of the original composited piece, and the magazines of which it was part.

How can this be done in current DL systems? We noted above the use of classifiers to create simple aggregate structures in (say) Greenstone. However, each part of the "Gang of Four" is a child of two aggregates: the issue of the Strand in which it appears, and also of the serialised story itself.

Problems rapidly emerge if we look more closely at the DL interface. Let us take Greenstone as an example, as it is extensively documented and functionally rich. When the user views the document page for part of the story, we can happily configure Greenstone to present the parent classifiers (both the Strand and Gang of Four).

Documents are, however, viewed within a context: e.g. the user has navigated through the Strand magazine issue/article hierarchy. Thus, one of these parent hierarchies will represent the user's context for the document. For a poem, there may be many more contexts (containing aggregates) than the two we see in this example. Though Greenstone has the ability to present the hierarchical information of containing aggregates, it does *not* have the ability through reconfiguration alone to identify the "current" aggregate at run time. This shortcoming is shared with the other DL systems.

The document view alone is not the only one in which configuration and ingest coding fail to achieve the full function that one might wish. For example, returning to the "Gang of Four", if a user were to search for this story, would the presentation of its separate parts as distinct items in the result list be optimal? Given that a search by title should match all elements of the aggregate, it seems clear that a better solution would be to return the aggregate itself as a single item. Returning to our observations on the functionality of DL systems above 3.1, hierarchy classifiers are not themselves indexed, and therefore we cannot reconstitute the single item without either ingesting a separate, distinct document to represent it or changing the functionality of the DL system. Given the duplication that would occur by representing both the parts and the whole in the same index, adjustments to the DL system are to be preferred.

4. USER STUDY

In this section we move from the technical issues underpinning the indexation of aggregates in the digital library to the interactional concerns that surround their use in practice. The paper so far has identified shortcomings in aggregate representation in the DL system. However, it is clearly also important to identify how people use aggregates in practice, and to what degree existing DL interfaces support common information seeking strategies.

To explore the interactional issues surrounding the use of aggregate documents, we undertook a brief user study. We recruited 6 postgraduate students in Computer Science at Swansea University, 2 library staff studying postgraduate humanities research and 8 postgraduates & staff in humanities subjects. Participants were given a simple information retrieval task, in a field related to their own, that involved searching a familiar online DL: JSTOR for the humanities and the ACM Digital Library[3] for Computer Science. Users used a large display to minimise screen size effects.

We wished to see the degree to which the researchers would extract the identity of relevant aggregates, rather than seeing the consitituent parts as unrelated fragments. The two library systems use rather different representations for aggregations in their interfaces, and many researchers have argued that humanists emphasise scholarship when compared to their scientific peers [18]. The protocol thus included a pre-study questionnaire to establish the degree of use of aggregate and non-aggregate printed works (e.g. monographs, journals). During our evaluation of the results, we scrutinised the outcomes for differences between the groups.

Participants described their strategies in a talk-aloud protocol as they interacted with the retrieval engine. After their task was completed, we elicited further information in a semi–structured interview. The searches performed by the participants were recorded, together with details of the documents returned, opened and selected as relevant.

[3]www.jstor.org and www.acm.org/dl respectively

Space does not permit a full discussion of our hypotheses or findings, so only a selection of the key details follow.

4.1 Findings

Much of what we observed reflects existing knowledge of information seeking. Our investigation revealed a mix of skill and interaction problems within the participants' use of the DL. These issues contributed to a low recognition of larger-scale aggregates that were relevant to the chosen task.

Our study included the elicitation of practice with physical literature as well as electronic material. Of the 16 participants, 13 reported normally checking the table–of–contents of any journal issue that they retrieved physically. In contrast, only one reported doing so in electronic documents, and even they commented that "often I forget, or I can't figure out how to get it". This amply demonstrates that there are differences between the physical and digital.

This was reflected in their online behavior: During retrieval, users focussed on the descriptive metadata in each matching document that they expected to contain relevant information. In particular, these included the author, document title and extracted prose content (e.g. abstract). A low level of attention was paid to indexation material such as the journal title, with diminishing attention being paid to publication material such as page numbers and volumes – notably only at most two journals were ever cited for any search, whilst often there were several related titles. Journal titles were primarily used to evalute relevance.

Eleven researchers specifically reported the significance of special issues of journals, yet only three observed special issues in their search results, though at least one occurred during fourteen sessions, and one participant received articles from three such issues. Relevance clearly played a role, with one session including the rejection of an identified special issue because it was "off topic".

Users often missed the frequent appearance of a journal issue in a search result list. One humanities researcher, failed to spot the regular appearance of a special issue of "Poetics Today" despite two articles appearing next to each other, one of which they judged highly relevant. Two users searched for individual journals by name, having seen them cited in relevant works, yet neither received search results focussed on that title. This was because the searches were interpreted as "full text" searches, and subsequently matched references within other documents.

Overall, documents were evaluated individually, though two participants with library experience browsed through a specific journal. This was yet another strategy reported in the physical world (reported by 12 users) that did not reappear electronically. When users endeavored to use aggregates to locate material, the libraries failed to give effective results.

4.2 Summary

These findings reflect the outcomes of prior research on the (metadata) content of document summaries in search result lists: readers seek for titles and narrative content first to inform their acceptance or rejection of a hit as relevant. At the point of inspecting a search hit, the reader's attention is not focussed on detecting patterns within the result set.

As readers in electronic environments recall and read individual articles without retrieving their aggregating parent – e.g. the journal issue – there is a much reduced opportunity for encountering nearby related material. It may be noted, however that this opportunity for "serendipity" is not without cost, as in the physical library, the reader must travel to the shelf holding the journal and then locate the article within in. An approach that mirrors the physical world may prove less acceptable to the user due to apparently unnecessary impedance and effort.

If we wish to support the identification of rich sources on a topic (e.g. a special themed issue of a journal), then this needs to be brought to the user in a way that does not intrude into the relevance judgement task and it also demands little attention or effort.

5. SUPPORTING AGGREGATES IN DLS

We have demonstrated the technical and utility limitations of current DL systems, and found that new functionality is required to fully support heterogenous aggregates in DL systems. This section turns to implementing improved features. We have extended a standard DL system – Greenstone – to incorporate aggregate support. The methods used throughout this section can be readily adapted to other systems.

Two key architectures of DL systems can be noted: what we may term 'bytestream' architectures [9] that treat documents as a collection of metadata and corresponding binary content that is delivered to the reader; and 'full text' architectures that treat documents as metadata and organised text (where the content is textual) [19]. 'Bytestream' architectures generally do not index the full text of the document, whereas full text architectures both index the text and are aware of its internal structure. Clearly, particular systems and particular installations may vary. For example, DSpace is often installed as a simple bytestream architecture, but can optionally be used with the Lucene XML indexer to index some or all of the content of a collection [16].

5.1 Indexing Aggregates

We studied the complications of composite aggregates (AG5) above. In this case there are two key options for full-text indexation: first, each separate aggregate is indexed independently and as a whole – resulting in duplication of text; second, each unique unit of text is recorded only once, but is referred to by each aggregate that uses it. The duplication approach can readily be achieved with any digital library, though it is clearly wasteful of space. The second approach is unfortunately not as straightforward to exploit as it may at first appear: additional indexation structures need to be present in the DL. We will discuss this in the context of the MG indexer used in Greenstone – though much of this discussion would also apply to Lucene.

Like many full-text search engines, MG [20] can create multiple indexes for different parts of each document: the text is indexed once, and each internal document structure mapped to a single span across the text index (see. Fig. 2). However, this model of a continuous and contiguous stream of

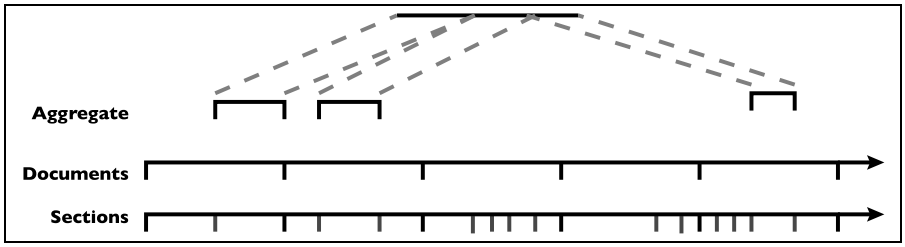

Figure 2: Indexation of an aggregate

blocks does not support composite aggregation which combines separate, discrete blocks of text. A supplementary map structure links an aggregate document to its text blocks; the document weight for each aggregate must also be calculated. MG is also typical in indexing every document in sequence with its full text. This adds a second complication: when an aggregate is to be processed, it has no content of its own – that is found in its constituents. Either the data to link it to its constituents can be transmitted to the search engine at this point, or it can be omitted and processed in a separate phase of indexation.

We chose this second approach, and added a third indexation phase to MG's two–phase design. The new phase records the map structure and calculates document weights for aggregates. In the first two phases (dictionary creation, compression and indexation) document text is indexed and compressed. The only change made is that aggregation data is used to identify when a document or document part is a duplicate of another. Where duplication occurs, second and subsequent occurrences are marked, and not passed for processing. Instead, in the second (compression and indexation) phase the document's pointer to its content in the index is made to point to the first occurrence. All multiple occurrences are temporarily stored in a lookup table. When the third, aggregation, phase is run, this receives a hierarchical set of identifiers for each aggregate. Each set identifies which documents or document parts consititute the aggregate. The aggregate record is then completed by translating the input document identifiers with their offsets into the compressed index, and the document weight for the aggregate is then calculated by first creating a word list for the *whole* document and calculating its weight accordingly. For faster processing, MG's approximate weighting system [20] can be used to create an aggregate document weight.

When searches are run, they can use either the original MG index (retrieving simple documents only), or the third-level index to return aggregates. In the latter case, results can be of aggregates only or both aggregates and documents.

There is IR research on detecting similar and contained documents [3]. However, in our context references are explicit and the task therefore different. IR duplicate research may well prove useful in identifying aggregation.

5.2 Classification and Aggregates

In Sec.3.3 we reported that simple aggregates can be represented in existing DL systems through using classification structures. We also noted that simple consolidation of different parts of a document can be achieved in search result lists when using Greenstone. We have just reported a more sophisticated approach that is superior when composite aggregation occurs – i.e. when an item occurs in more than one aggregate. It is worthwhile pointing out the different advantages of these two approaches.

Where aggregates are represented by classifiers, they are usually not known to the underlying search engine — particularly for a DL with a componentised services architecture [14, 5]. Thus classifier nodes do not have a document weight and cannot reliably be used for ranking. Conversely, where knowledge of aggregates is supported in the text indexer, different problems emerge: for example, if a specific document that appears in several aggregates matches a search, how should it be displayed? In its own right, or within an aggregate? If within an aggregate, which one? If the aggregated document is a good match, and its aggregating parents a poorer match, then the apparent solution differs from where the opposite applies.

In both cases, there are interactional concerns: e.g. the default interface of Greenstone allows the user to choose which level of document structure to search at (within documents, sections or paragraphs, etc.). Given the variable numbers of levels that occur in aggregates, and the fact that the structure of aggregates differs, this simple approach breaks down, e.g. the 'volumes' labelling clash referred to in Sec.3.4. The manual selection of search granularity also places a burden on the user. It will take further exploration and research to find optimum solutions for such problems.

5.3 Architecture

We observed above 3.3 that classifiers, though an adequate representation of aggregates in some circumstances, suffered from significant shortcomings when topic hierarchies included aggregate works, and also in terms of their display during browsing and searching. Our solution to this problem was to add a new "Aggregate" item type to Greenstone, incorporating many of the features of the existing "Classifier" objects. Aggregates are hierarchical trees, as is the case with classifiers, but can be referenced directly by (multiple) classifiers as if they were documents. This means that the aggregate structure is only stored once in the DL system, avoiding the maintenance and efficiency problems that emerged when classifiers were used to represent aggregates.

Aggregates are recalled as if they were documents during searching and browsing, exploiting the improvement to the search engine described immediately above. However, simple reverse look–up mechanisms can be used to simulate a similar behavior using an unmodified search engine.

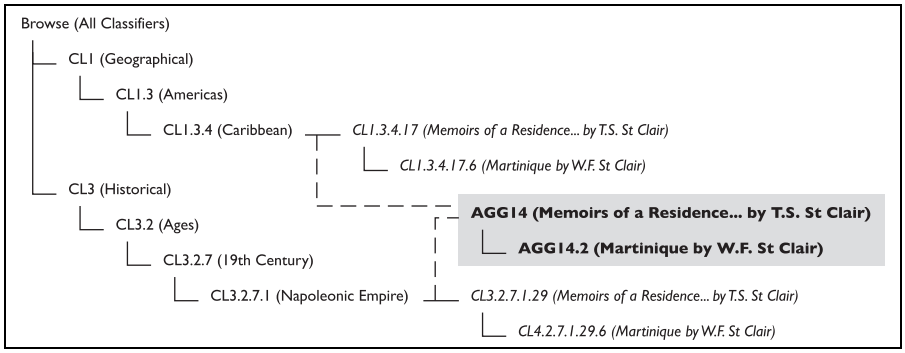

Figure 3: Representing an aggregate using classifiers in *italic* and using a specific aggregate object *bold* – the latter approach is clearly superior.

Given the broad similarity in DL systems observed in [1], the model is readily transferrable. As with FRBR support [4], this "extension" to Greenstone is stand–alone, modular and immediately adaptable by other DL systems. It also supports reference of an aggregate part to an item stored in any common DL system.

6. INTERACTION DESIGN

Readers in a library naturally get the "whole volume" containing an aggregate; readers in a DL only get the part which they retrieve. Delivering the whole aggregate in a DL leads to confusion and poor retrieval performance (due to naive up–and–down scrolling), which eliminates the direct parallel with the volume–centred retrieval of physical libraries. We therefore need to discover a different solution that recovers the benefits of the physical environment in the digital domain. Elaine Svenonius [15] is typical of many librarians in claiming that serendipity is, in fact, not an incidental feature of the traditional library, but rather a direct product of its structure and organisation. Aggregation is one such means of placing related material together so that "serendipity" is in fact a product of co–location. The simple method of adding metadata to documents for aggregation seen in our user study clearly failed to reproduce this effect.

Aggregation also complicates delivering text for reading: e.g. when a PDF file contains three works, each work could be indexed individually by the DL system. If the user now wishes to read the material, in current DL systems the entire PDF will be delivered. As observed in [2], users often inspect only the top of any digital document. If the visible head of the document does not match the user's expectations (e.g. the title of a constituent work rather than the aggregate name used as a search term) it is likely that they will incorrectly conclude that an error has occurred or the document they wish is not available. Therefore, a comprehensive handling of aggregates must deliver material for reading in a way that is consistent with the collection index. For file formats such as XML this is straightforward, whereas PDF for example is complex. Where delivery of the whole aggregate is necessary or desirable, information should appear in the DL interface to ensure that the user's expectations are aligned with what they will receive if they download the work. Where DLs provide material in different digital formats (e.g. HTML, PDF or Word of the same document) then this interface idiom may be adopted for delivering content of different scope (work, volume, etc.). The cautions over confusions in download formats noted in [2] will almost certainly apply.

7. RELATED WORK

The difficulties of the representation of aggregate works in digital libraries has already received attention. Hickey and O'Neill [8] note a number of problems in applying the Functional Requirements for Bibliographic Records (FRBR) [12]. They propose treating aggregates as published volumes of more than one work, and to avoid recording aggregates as works in their own right. This introduces an inconsistency with the accepted FRBR model where every published volume (*manifestation*) is an instance of a single work.

Two standard electronic document formats allow for the representation of aggregate works: TEI [11] and METS [10]. In each case, aggregates are achieved by pointers, be they between content of the same file or to separate files, to create a whole. TEI primarily uses pointers between parts of the aggregate, whereas in METS a document contains references to part or whole other METS documents – these parts then form sections of the current document.

Aggregates have been poorly represented in DL systems to date. Though it is conceivable that Fedora's object-based architecture [9] may be able to represent aggregation, we have not been able to discover any published coverage of this issue. Popular systems such as DSpace [16] and Greenstone [19] have focussed on treating collections as sets of objects, with a hierarchical classification structure. Aggregates can be represented using the classification structure, as we demonstrated in Sec. 3.3, but at the loss of consistent treatment of aggregates across both searching and browsing.

One may hope that practice and experience from library science would be helpful. However, the historic need to find and recover texts via bound volumes has emphasised the approaches we have seen in DL systems. Aggregates are generally indexed by part where the parts are discrete works: e.g. the British Library[4] binds brief tracts together in volumes, but each tract in a volume is indexed separately. Conversely, multi-volume works are usually, but by no means universally, indexed with only one entry. In the case of the British Library, practice here varies from work to work.

[4]www.bl.uk

Svenonius [15], p. 103, notes that there are two potential routes to relating aggregates with their constituent parts: first, formal linkage structures; second, providing descriptive aggregation (meta-)data for each item. The latter approach, though informal and easy to apply, leaves much of the retrieval work with the user, and greater room for mismatches between the descriptive data and the corresponding description of the part or aggregate in the catalogue index. This paper has demonstrated that the link–based approach also has its limitations, and that specific functions are needed in the DL system to implement it effectively.

8. CONCLUSION

This paper introduced a number of different forms of aggregate works in the digital library. It demonstrated that simple types of aggregation are easily supported in DLs, though with key shortcomings in their representation in the DL interface. More complex forms of aggregation htat occur frequently in historic literature map less readily to existing DL architectures and interfaces. We reported on changes to Greenstone and MG that widen the forms of aggregation that can be successfully represented in DLs. We briefly reported on the complications of integrating these changes into the DL user interface, and some corresponding solutions. Though aggregates have only recently started to receive attention in the digital domain, a considerable amount of work will be required to move from the initial steps represented here and in other projects such as the IFLA Working Group on Aggregates [13] to a complete solution. In our case, we wish to investigate further the appropriate interactions to support the occurrence of aggregates in search result lists, and the location of desired aggregates in the course of information seeking.

9. ACKNOWLEDGEMENTS

This research is supported by EPSRC Grant GR/S84798.

10. REFERENCES

[1] D. Bainbridge, G. Buchanan, J. McPherson, S. Jones, A. Mahoui, and I. H. Witten. Greenstone: A platform for distributed digital library applications. In *ECDL '01: Proc. 5th European Conference on Digital Libraries*, pages 137–148, London, UK, 2001. Springer.

[2] A. Blandford, H. Stelmaszewska, and N. Bryan-Kinns. Use of multiple digital libraries: a case study. In *Proc. ACM/IEEE-CS Joint Conf. on Digital libraries*, pages 179–188. ACM Press, 2001.

[3] A. Broder. On the resemblance and containment of documents. In *Procs. Compression and Complexity of Sequences*, pages 21–29, 1997.

[4] G. Buchanan. Frbr: enriching and integrating digital libraries. In *JCDL '06: Procs. 6th ACM/IEEE-CS joint conference on Digital libraries*, pages 260–269, New York, NY, USA, 2006. ACM Press.

[5] G. Buchanan, D. Bainbridge, K. J. Don, and I. H. Witten. A new framework for building digital library collections. In *JCDL '05: Proc. ACM/IEEE Joint Conf. on Digital Libraries*, pages 23–31, 2005.

[6] G. Buchanan, S. J. Cunningham, A. Blandford, J. Rimmer, and C. Warwick. Information seeking by humanities scholars. In *Proc. 9th European Conference on Digital Libraries*, pages 218–229. Springer, 2005.

[7] H. Davis, W. Hall, I. Heath, G. Hill, and R. Wilkins. Towards an integrated information environment with open hypermedia systems. In *ECHT '92: Proceedings of the ACM conference on Hypertext*, pages 181–190, New York, NY, USA, 1992. ACM Press.

[8] T. B. Hickey and E. T. O'Neill. FRBRizing OCLC's WorldCat. *Cataloging and Classification Quarterly*, 39:239–251, 2005.

[9] C. Lagoze, S. Payette, E. Shin, and C. Wilper. Fedora: An architecture for complex objects and their relationships. 2005.

[10] Library of Congress. *Metadata Encoding and Transmission Standard (METS).*

[11] C. Sperberg-McQueen and L. Burnard, editors. *Guidelines for Electronic Text Encoding and Interchange*. TEI P3 Text Encoding Initiative, Oxford, 1999.

[12] Study Group on the Functional Requirements for Bibliographic Records. *Functional requirements for bibliographic records*. K.G. Saur, 1998.

[13] Study Group on the Functional Requirements for Bibliographic Records. Minutes of the FRBR review group's meeting, Aug 2005.

[14] H. Suleman and E. A. Fox. Designing protocols in support of digital library componentization. *Proc. European Conference on Digital Libraries*, 2458:568–582, 2002.

[15] E. Svenonius. *The Intellectual Foundation of Information Organization*. Digital Libraries and Electronic Publishing. MIT Press, 2000.

[16] R. Tansley, M. Smith, and J. H. Walker. The DSpace open source digital asset management system: Challenges and opportunities. In *Proc. European Conf. on Dig. Libs.*, pages 242–253. Springer, 2005.

[17] Y. L. Theng, E. Duncker, N. Mohd-Nasir, G. Buchanan, and H. Thimbleby. Design guidelines and user-centred digital libraries. In *Proc. 3rd European Conf. for Digital Libraries, ECDL*, pages 125–134. Springer-Verlag, 1999.

[18] S. S. Wiberley. Habits of humanists: Scholarly behavior and new information technologies. *Library Hi Tech*, 9:17–21, 1991.

[19] I. H. Witten, S. J. Boddie, D. Bainbridge, and R. J. McNab. Greenstone: a comprehensive open-source digital library software system. In *Proc. ACM conf. on Digital libraries*, pages 113–121, 2000.

[20] I. H. Witten, A. Moffat, and T. C. Bell. *Managing gigabytes (2nd ed.): compressing and indexing documents and images*. Morgan Kaufmann, San Francisco, 1999.