Appropriate Use Case modeling for humanities documents

Aja Teehan and John G. Keating
National University of Ireland, Ireland

Abstract

This article details a modeling methodology that is appropriate for historical, functional documents that are to be digitally represented and hosted within a software environment for humanities research. The functionality is derived from Use Case modeling that can be undertaken in consultation with the User Group. The Use Cases are an expression of the whole-system model as they embody the interaction of User, with the document, in the software environment. The encoding mechanism largely practiced within the humanities computing community is represented by the TEI, which seeks to provide a set of guidelines for encoding humanities documents. However, TEI offers no guidance in relation to creating an encoding of a document that is supportive of the software environment that will host it, the interaction mechanisms required, or the User. We argue that modeling with recourse to the Logical, the Physical and the Interaction classes enables not just the generation of an appropriate encoding scheme, but also the software to manipulate it. We situate Use Case methodology within Activity Theory and relate this to the humanities computing community. The argument is framed in relation to the creation of a digital edition of an 18th century Spanish Account Book manuscript.

Correspondence:
Aja Teehan,
An Foras Feasa,
National University of Ireland,
Maynooth,
Co. Kildare,
Ireland
E-mail:
aja.teehan@nuim.ie

1 Introduction

As evidenced in the body of text-encoding journal articles, conference papers, and popularity of the TEI, it is an ever growing practice for humanities researchers to act both as text encoder and domain expert for their own digital document creation projects. The TEI (Text Encoding Initiative, P5, 2008) focuses mainly on promulgating the skill of encoding by actualizing chosen aspects of the underlying TEI document datamodel, rather than the knowledge of how to engineer a document-situated datamodel. Few of the text encoding practitioners are taught data modeling or software design, yet their XML encodings must be eventually hosted within some software environment. In some cases the encoded document is completed, and then handed off to a software engineer who independently designs a software environment to manipulate it. This disconnect between humanities scholars and the software environments that house their documents can sometimes result in unappealing and underused systems.

Even prior to the withdrawal of funding from both the Methods Network and the Arts and Humanities Data Service, the Humanities Computing community began to examine itself for 'evidence of value', and how useful, and used, its products were. As the LAIRAH Project noted, it has become apparent that ~35% of humanities computing environments

Literary and Linguistic Computing, Vol. 25, No. 4, 2010.
doi:10.1093/llc/fqq026

remain unused and that though this is comparable to the number of scientific articles that are not cited, the public cost involved in creating these research environments is much larger and more difficult to justify (Warwick *et al.*, 2006, 2008). Furthermore, Warwick's team discovered that few projects had performed user requirement analysis or created formalized user requirement documentation.

During the Digital Humanities 2008 conference, at an ADHO Session 'Digital Resources in Humanities Research: Evidence of Value (2)', Lorna Hughes discussed the results of the evaluation process at the end of the AHRC ICT Methods Network (a formal report has also been published (Hughes, 2008)). She gave an account of how difficult it was to elicit from funded projects what uses they had formally planned and whether those uses had been fulfilled. This lack of analysis tarnishes all humanities computing projects and makes it difficult to argue the case for future funding. While it is self-evident that there are benefits to making digital representations of cultural artifacts available on-line, it is also self-evident that this assertion is not enough; serious consideration needs to be given as to how the best use can be made of these sources (McCarty, 2008).

Bradley argues that 'HC might be more influential if it moved its operations closer to traditional scholarly methods' (Bradley, 2005). This is what we try to achieve with the Primary Use Cases we will discuss; they leverage the domain expert's knowledge. These Use Cases are used in conjunction with an encoding, or more correctly, a modeling, paradigm in order for us to try to 'be in a better position to develop a model of the role of computers that does more to support humanities research' (Bradley, 2005). This modeling paradigm consists of the logical, physical and interaction classes, and models how the document is to be used (not just what it 'is') within the software environment. This type of modeling, which requires detailed knowledge of the interaction that is to be undertaken as well as how to implement that interaction, is of paramount importance and requires software engineering knowledge. In fact, embracing software engineering practice may serve to further develop the humanities-centered role for computers to which Bradley refers.

It is preferable to perform this user requirement and Use Case analysis prior to the creation of the digital artifact, rather than after. In this way, it is possible to inform the process of implementation, thus ensuring that the end-product meets the envisaged needs of the community, be they academic humanities researchers or the public. There are many well-documented methods for gathering user needs for a humanities computing resource, and recommendations regarding these can be found in the report for the LARIAH project. However, in this instance we are specifically interested in a formal methodology for documenting this analysis (UML), and including it in a paradigm that can be used to drive both the design of the software system and the XML encoding of the digital artifacts to be manipulated within that system.

1.1 Use Cases

A Use Case acts as a blueprint for the system design and typically depicts the steps an actor takes while interacting with the software in order to achieve some meaningful goal or task, goal being higher level and task being lower level. These explicit steps are expressed in a formalized diagram using UML (Unified Modeling Language) and can be used by the software engineer to create a supportive software environment. In a wider context, UML has been used to model and generate XML schema and documents for many years (Carlson, 2001), and has also been propounded in humanities computing both as a data modeling (Hayashi and Hatton, 2001) and document modeling tool (Kimber and Heintz, 2000).

The Use Cases model the interaction of the researcher with the document within the environment; they go beyond modeling what the document is, to how it will be used. We are particularly interested in this aspect because these analyses can then be used to build a software environment that encapsulate the functionality and interactivity required by the researcher. In order to achieve this the XML encoding of the source must support these whole-system Use Cases, including the interaction mechanisms.

In the remainder of this article we will refer to the digital edition of the Alcalá Account Book manuscript for examples (Fig. 1) (Keating *et al.*, 2008). The Alcalá Project was originally proposed as a digital humanities project to mark humanities collaboration between the University of Alcalá de Henares (UAH), Spain and the National University of Ireland, Maynooth (NUIM), Ireland. The source was to be a Spanish 18th century account book recording the monthly expenses of the Royal Irish College of Saint George the Martyr. In 8 weeks, the source manuscript was chosen, encoded and made available in a web based, dual language, searchable, and interactive environment. More importantly, it was developed to aid the historian in answering historically pertinent research questions that are specifically prompted by the functionality and information contained in the historical object, an account book.

In the digital edition of the Alcalá Account Book manuscript a transcription and translation are provided for the manuscript and are presented along with facsimile images of the original manuscript on a page-by-page basis. In addition, it is possible to transfer specific expenses to a datasheet for accounting operations to be performed upon them. A typical example of the goal a user might wish to accomplish using the original manuscript might be, 'calculate how much was spent on bread at the college in 1778'. At a lower level of abstraction, this

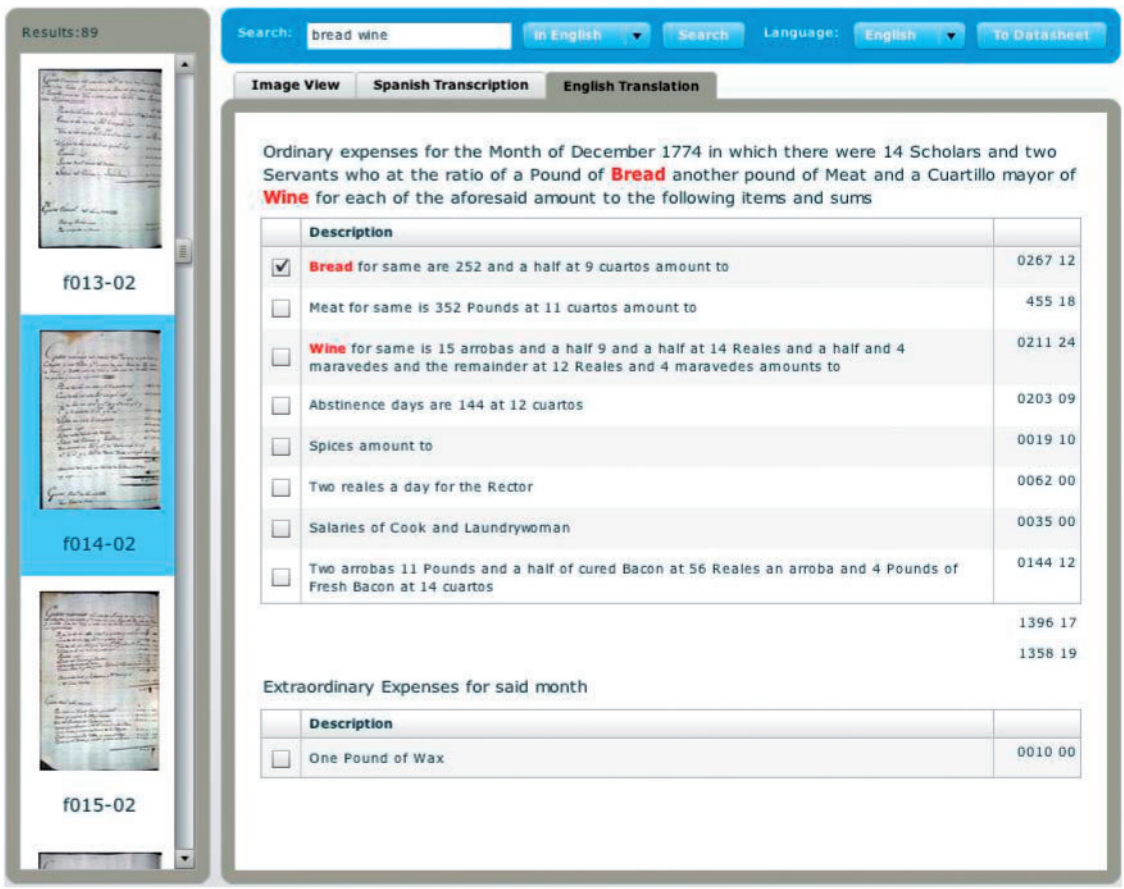

Fig. 1 Bread and wine used as keyword filters; items of interest selected in the English translation on one resultant page

Use Case requires that the User performs six steps. If the User speaks only English then they must (1) select the translation as the version to search, (2) enter bread as the keyword for the search, (3) examine the returned facsimiles from 1778, (4) use the checkbox to select the entries in the account book pertaining to bread, (5) transfer the pertinent entries to the datasheet for calculation, and (6) switch to the datasheet view to read the total. The interaction here is non-trivial and the XML encoding has been designed to support it.

2 Activity Theory and Use Cases

Activity Theory (AT) is a paradigm developed by Leont'ev and Rubinshtein (Leont'ev, 1947) that considers human activities as complex, socially situated phenomena. It has roots in Vygotsky's sociocultural psychology, and theorizes that a subject's engagement and interaction with their environment results in the production of tools; these tools are exteriorized forms of mental processes. The mental processes are manifested in accessible and communicable tools that are useful for social interaction, i.e. subjects having an objective use tools to produce some outcome. In AT driven methodology, any activity (task) can be broken into actions, which are further subdivided into operations, so from a design context, AT can provide the designer with an understanding of the steps necessary for a user to carry out a task.

An Foras Feasa has determined that AT is a useful tool for modeling the activities of digital humanities, and the UML-specification humanities computing tools that provide users with opportunities for meaningful engagement with these tools, i.e. the engagement with digital humanities. In particular, we favor the approach of Engeström (Engeström, 1987) who proposed an AT that further developed the original scheme by Leont'ev; it contains three interacting entities—the individual, the object, and the 'community'. Here AT rejects the isolated human as an adequate unit of analysis, focusing instead on the cultural and technical mediation of human activity (rules, etc.), as in Fig. 2. More recently, Nardi and Kuutti see AT as

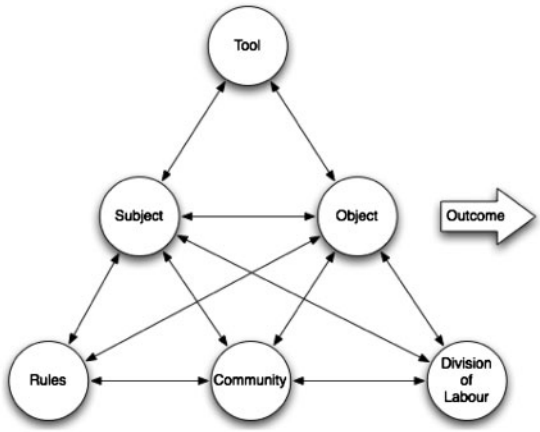

Fig. 2 Expanded Scandanavian AT (Engeström, 1987)

'. . . a powerful and clarifying descriptive tool rather than a strongly predictive theory' (Kuutti, 1991, 1996; Nardi, 1996). Nardi has argued that Human Computer Interaction theory has 'largely ignored the study of artefacts, insisting on mental representations as the proper focus of study' and AT is seen as a way of addressing this deficit (Nardi, 1996).

2.1 An Example Activity: Document Encoding Process

AT provides us with a framework for examining the tools that we use when trying to achieve outcomes, and how we interact with those tools. Software Engineers use Use Cases to model activity, i.e. to specify software tools for scholars to achieve outcomes. AT tells us that the tools must change depending on the outcomes. If our activity is the process of encoding a document, then the objective we have for that document dictates the form that must be taken by the encoding tool. The encoding approach (process) and encoding (product) must be adapted in order to support the object and outcome, as in Fig. 3.

Here we see that three classes of encoding are required to model three different aspects of a single document to allow for different scholarly objectives and perspectives. Thus, the encoding is always the tool, but it is not invariant. Tools change depending on the activity. By analyzing and adapting the tool we are adapting both the

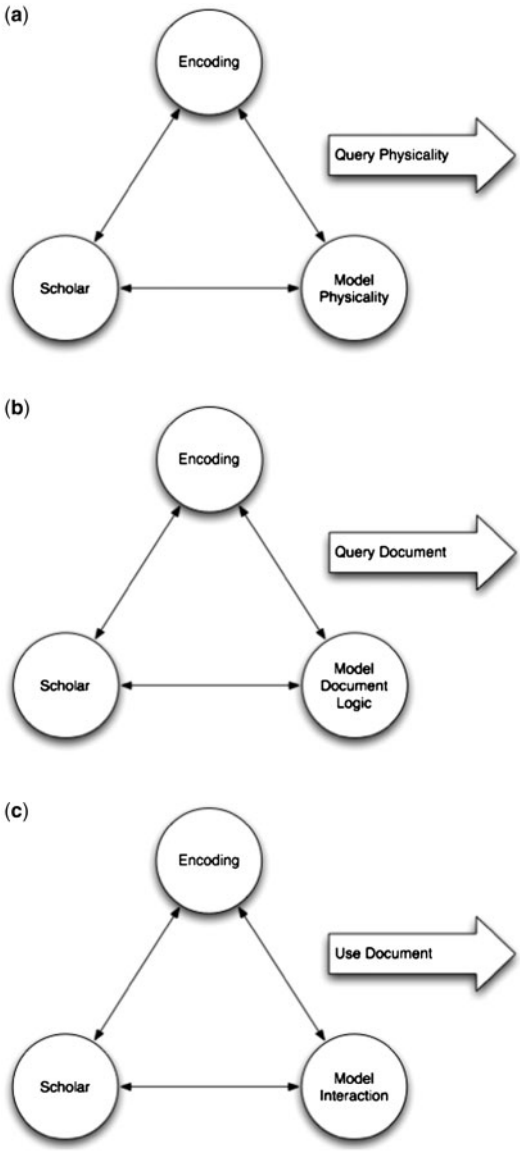

Fig. 3 (**a**) Encoding tool, Class 1, allows scholar to query physicality of the document. (**b**) Encoding tool, Class 2, allows scholar to query contents of the document. (**c**) Encoding tool, Class 3, allows scholar to interact with (use) the document

process of encoding (externalized mental processes) and the product of that process. At this level of abstraction, the object is the model of the document, which is being created and manipulated by the scholar, using the encoding as the tool to aid in this creation and manipulation.

This form of knowledge building is often referred to as a hermeneutic spiral, where the understanding of the document that is gained in one round of modeling informs the subsequent understanding of the document upon its re-examination by the scholar. However, the term 'spiral' is uni-directional and tends to move us from the outside of the spiral inwards towards the centre, an end point. AT allows us to more accurately theorize about the process of understanding that is undertaken.

In fact, each node of the AT triangle has an effect on the other two nodes. The tool (the encoding) has an effect on both the scholar (their understanding and ability to understand) and the model the scholar is creating. The level of understanding and the accuracy of the model depends on the tool being 'fit for purpose'.

A tool, any tool, is more than just a product, it also encapsulates a process. Thus, an encoding tool is more than just a product; it encapsulates a process, or an approach. TEI can sometimes be made to apply in unforeseen circumstances by coincidence, abuse, extension or customization, but it remains a predefined product that encapsulates a predefined process; it is one perspective, albeit multivariate, on what documents 'are', and it embodies one perspective of how they are 'used'. Indeed, the confines of this underlying abstract data model has been hinted at most recently by Piez at the Digital Humanities 2010 conference (Piez, 2010). If a given project's perspective on the documents and their uses are not encapsulated within TEI then it is not the most suitable tool for encoding in that project. Instead, a custom designed tool would be beneficial as it encapsulates, and has been specifically adapted to, the particular needs of the encoder (the 'Subject' in AT), along with the characteristics of his objective (the 'Outcome' in AT) and the source he is working with (the 'Object' in AT).

2.1.1 *Primary use cases*

In a scenario such as that of the Alcalá project, we design our system and digital documents with recourse to Primary Use Cases. These are aligned to the original reason for creating the document

and capture the data that the original creator wished to convey. As they exploit the data within the context of the document itself they support a wide community of researchers. Primary Use Cases document the steps a user undertakes while interacting with a system in order to answer queries that relate to this data. For example, medical records were created in order to provide medical information regarding a patient and their treatment.

Researchers using these records should thus, as a primary concern, be able to query the records for medical data, rather than, for instance, presentational or graphological items. This differentiation and its implications for text encoding has been documented by Buzzetti (Buzzetti, 2002), who states 'we sense a ''basic ambiguity'' which prevents us from discriminating between the formal properties of the ''representation'' of information, and the formal properties of the ''information'' represented, that is, between the data considered as ''information coded in a special way'' and data considered as the model or the ''abstract structure'' of the information'.

Primary Use Cases relate to the semantics of the document, rather than any presentational or representational features. Thus XML tags that might be used in an encoding based upon this analysis would likely use words such as 'patient', 'prognosis' and 'medication', rather than 'strike-through', 'underline', or 'paragraph'. The encoding would then semantically structure these components, based upon an analysis of their documentary context, including ordering or location if pertinent. This type of semantic encoding has parallel applications in business document engineering, which seeks to access and rationalize the underlying meaning of certain sections of collections of business documents—the aim being to design rationalized and modular documents that can be accurately generated (Glushko and McGrath, 2005).

2.1.2 *Secondary Use Cases*

Secondary Use Cases are predicated upon the needs of the individual researcher, the needs of the community being of secondary concern. That is, the digital object being created is designed to serve the research needs of the individual encoder. In this scenario, it is quite legitimate that the Primary Use Cases are ignored. Secondary Use Cases can, quite literally, span any area of research interest and are not related to the original reason for creating the document. In the medical record example above, Secondary Use Cases might support querying for language patterns or prosopographical information.

Supporting Primary Use Cases often results in indirect support of Secondary Use Cases. For example, medical data is usually associated with an individual; encoding a collection of medical records subsumes encoding the individual's data and thus prosopographical research is facilitated indirectly. By performing document modeling on the information that was originally encoded in the document, the largest possible audience is provided with a digital resource.

2.2 TEI Use Cases

TEI-encodings fulfill a limited number of primary Activities (or Use Cases), and provide a solution set for those activities, for example, producing preservable and machine-readable texts (a metadata heavy tool; but it is getting 'lite-r'). This reflects the needs of the individual scholars most closely associated with TEI, historically, those who have been involved in scholarly editing and publishing, and subsequently their Secondary Use Cases (encapsulated by the needs of the Special Interest Group for a module).

This print and editing-oriented legacy, and its ramifications, have most recently been documented by Schmidt (Schmidt, 2010). While it is universally acknowledged that the TEI guidelines have evolved over the course of a long discussion within and through humanities computing professionals, it seems overly proscriptive to imply that this is the only valid approach, and that it encapsulates all possible uses and perspectives.

We are interested in producing Human Usable Documents for wide community use, which require a different kind of encoding than that offered by TEI; our activities are different to those encapsulated in TEI (tools), but are similar to others in the text encoding community. In order to support our activities, we must use the most appropriate tool. While we do not attempt to implement each of the possible Use Case scenarios in the software

environment, we are not limited to those Use Cases previously conceived by an encoding scheme that has not been informed by the documented activities our users wish to undertake.

3 A Model Framework: Logical, Physical, and Interaction Classes

Based on our investigations, we propose a modeling framework that encapsulates both AT and Use Case methodology. The Use Case from our Alcalá project, 'calculate how much was spent on bread at the college in 1778' is a Primary Use Case, and the steps detailed above provide a good example of how the software environment should support the User. However, there is no detail about how the steps should be achieved by the software; instead everything is from the User's point of view. Additional steps must be performed by the software environment, for instance, the first step, usually depicted in a corresponding Interaction Sequence Diagram, now becomes (1a) Interface presents translation, transcription and facsimile image of first page (1b) User selects translation as the version to search (1c) Interface presents translation on the screen.

This poses a problem for the researcher charged with encoding the document. The above Use Case requires information from three different classes: the logical, the physical and the interaction classes. The logical class is a model of the content of the document, e.g. monthly expenses; the physical class is a model of the document e.g. pages; the interaction class is a model of how the User interacts with the document and, by extension, how the User interacts with the software environment's representation of that document. The encoding, though only part of the whole system model, must support the functionality in the Use Case and thus must support the three classes: logical, physical, and interaction.

During this project we wished to provide a clickable interface for the User where each expense could be selected by 'click' for manipulation. Without recourse to the interaction class component of this Use Case, the intuitive interaction offered by the clickable facsimile would have been foregone. The expense figures required for any query can be directly manipulated on the facsimile image. The User can click on those manuscript account book expenses that they wish to interact with. The expense items are simultaneously selected on the facsimile, in the translation text, and in the transcription text; they can then be sent to the datasheet for further manipulation. This simultaneous selection imparts to the User the sense that all the versions are integrated, and are representative of the original encoding. The interaction becomes more intuitive, closer to the usability of the original document, but enhanced.

Failure to support the interaction class in the XML encoding will result in the software engineer being unable to fulfill the interaction Use Cases. In this instance it was necessary to record the area of the image that corresponded to each expense. We captured the x and y coordinates of each relevant point on the perimeter of the expense area; anything within the set of points is an expense. This produced a string of x and y coordinate pairs for each expense, for instance, '42,406, 48,586, 994,566, 996,386'. The next step is to associate these coordinates with their expense as it is encoded in the XML thus allowing the expenses to be highlighted and selected on the facsimile image. It is not possible to provide this level of interaction without an analysis of the Interaction Class requirements, and subsequent implementation of support for these requirements within the XML encoding. Thus, the humanities researcher who encodes XML must also be aware of the interaction class and the requirements of the software engineer.

Furthermore, the encoding of the logical model must also take cognizance of the User's requirements. As is widely recognized, choices need to be made between which perspectives upon, and characteristics of, the document will be encoded (Pichler, 1995). A single document can be researched in many different ways, for instance a historian may be interested in the social history captured in the Alcalá Account Book manuscript or they may be interested only in the prosopographical information that can be gleaned from it. In relation to XML encoding specifically, tags are created to give context to content, and segmentation

of the document facilitates the decision as to what should be contextualized. These decisions can all be derived from the Use Case analysis.

While encoding the information the encoder must always be mindful that the Use Cases can be fulfilled. For instance, in the Alcalá Account Book encoding each of the expenses was labeled separately and broken down into its description and the sum spent. This allowed us to manipulate the figures separately on the datasheet so that mathematical operations could be performed. Without this separation of the sum spent we would have been unable to contextualize the figures as 'money' and thus would have been unable to fulfill the Use Case, 'how much was spent on bread in 1778?'

Although it is both possible and necessary for a researcher skilled in humanities to be the primary articulator of Use Cases that encompass both the physical and logical classes, it is more difficult to articulate those parts of the Use Case that derive from this interaction class and a dialogue should be opened here with a practitioner knowledgeable of Computer Science and Software Engineering.

4 Discussion

Use Cases can often be described using terminology such as 'objectives', 'aims', 'requirements', 'tasks', and 'actions'. Like these terms, they operate at various levels of abstraction; the appropriate level can be applied for each part of a project. Use Cases have some well documented problems associated with them (Firesmith, 1999), most deriving from application at an inappropriate level of abstraction. An oft-cited problem is that Use Cases can only be successfully used when the modeler has a full understanding of the problem domain, in this case, some humanities data or object. This would limit their usefulness to cases when it is possible to fully understand the humanities data or object in question before the digitization takes place.

Transactional documents, that is functional documents or records, are always created in some context and can thus be understood. This is not necessarily true for creative humanities objects, such as novels, which do not have a definitive purpose. These are less definable and thus are sometimes digitized to aid in the investigation of their meaning. In this instance, the Use Cases that pertain to the function of the document, and thus the elements of the Logical Class are not easily elicited or agreed upon. At this level of abstraction, this type of Use Case is less useful.

The argument is subsequently made that Use Cases are less useful in general when the aim of the digitization is to promote prima facie discovery or investigation of the humanities object. On the contrary, 'encoding for discovery' must be subjected to the same rigorous processes that 'encoding of meaning' is. That is, unless the aims of the investigation can be formally documented prior to the investigation it is likely that the investigation will be only partially successful. If a researcher is truly unable to enunciate the aims of their XML encoding, it is likely that XML encoding is not a useful tool in their scenario.

Use Cases can also suffer from the same drift that applies to requirements. To overcome these problems it would also be important to combine this approach with an iterative design process, as opposed to a sequential. This would ensure that the Use Cases could also be updated to reflect the most current set of requirements for the project and help to avoid, 'the biggest iteration of all, going back to the start' (Dominick, 2000).

The Use Case is just one tool available to the humanities computing researcher from the arsenal of software engineering paradigms, for instance Rapid Application Development (Martin, 1991) and Participatory Design (where the end users are actively involved as consultants in the design of the software ecological system) would be valuable (CPSR, 2008). Situating the design and use of Use Cases within these software engineering paradigms would be even more beneficial to the humanities computing community.

4.1 Are Use Cases Redundant?

Use Cases are sometimes considered to be the expression of the obvious through highly formalized means, the implication being that the administrative overhead incurred is not justified for the benefit that is produced. It may seem obvious to state that

without identifying and then isolating the required pieces of information for a question, you cannot answer that question. However, this is a very basic, and very valuable, step that is missing from many digitization projects. For instance, a digital repository of 'The Chymistry of Isaac Newton' (Newman, 2006) offers diplomatic transcription, normalized version and correlated facsimile image for many documents, including Newton's most complete laboratory notebook. The documents are fully keyword searchable. The 'ultimate goal is to provide complete annotations for each manuscript and comprehensive interactive tools for working with the texts' (Newman, 2006).

There is no doubt that this is a very valuable source. However, the encoding does not provide for the functionality that one would initially expect of such a collection, nor can this functionality be added later, without significant recoding. For instance, the element tags are drawn from the prose, figures, linking, analysis, names/dates, and transcription tag sets of the TEI. These tags could support queries such as 'display the diagram called X'. In addition, there could be support for Use Cases based on the editorial transcription tag set, for instance, 'display all pages that have words which have been struck-through'. However, the only Use Cases based on the Logical class will be supplied by the names/dates module, for instance, 'list the pages where person X is referred to'. There is no support in the encoding for implementing a contextual search for obvious logical model elements such as 'experiment', 'apparatus', 'chemical', 'method', or 'conclusion'. One might expect to be able to answer queries such as 'how many experiments did Newton conduct using the chemical copper?', but this is not possible.

Though already a very rich and rewarding source, it would benefit greatly from this type of functionality. Furthermore, once the Use Cases were elicited and described, the additional work involved in encoding this type of functionality would have been minimal, and mainly confined to the document modeling and XML-framework building stage. As it is, the 'comprehensive interactive tools' that the creators envisage will prove difficult to implement, being conceived of as an additional, rather than an integral part of the digitization. The project may be stymied in its aims as it cannot support the logical element-based searches outlined above, and will also be unable to support interaction class use cases that need to be embedded in the XML encoding. It may seem obvious that to build a system one first has to define precisely what the requirements of that system are, but this is not the widespread practice within humanities computing.

Using a software engineering methodology such as Use Case analysis, coupled with a humanities computing methodology based upon an analysis of the logical, physical and interaction classes, would have enabled the designers of the project to design and implement an encoding and software environment that supported the projected activities of their users.

5 Conclusion

Use Cases do not, of themselves, guarantee the quality of the digital artifact. They function as a tool to aid the improvement of the whole-system software environment so that the main requirements of the User can be satisfied. The creation and implementation of the Use Case still requires skill and knowledge, and still depends completely on the writer of the Use Case, the software engineer and the encoder.

In relation to ascertaining the appropriate level of abstraction McCarty posed the question, 'For us in the digital humanities, when and how does it matter that we know directly what's in the cellar?' (McCarty, 2008). We contend, 'that from the outset (when it matters) researchers should know how, at least at a detailed pattern level, what they want to do, now and in the future (how it matters)' (Keating, 2008). This detailed pattern level is exemplified by the knowledge of how to create Use Cases. The researcher who wishes to operate at a lower level of abstraction and actually encode the humanities document must first have this high-level knowledge. In order to create an encoding scheme based around a document they should, in addition, have knowledge of the logical, physical and interaction classes. Only then can they appropriately apply that knowledge at the skill-level in an encoding.

Both the problem domain (humanities research) and the software engineering required to create appropriate Use Cases are very demanding of the researchers involved. Both areas demand high levels of expertise and understanding; it is unusual to find this level of specialization in one person. The solution is not to promote the assimilation of software engineering skills within humanities disciplines, but rather to promote the dialogue between the experts at a suitable design and abstraction level—that of the Use Case.

Acknowledgements

We would like to thank the staff of the Russell Library for their continued support.

Funding

This work was partially funded by the Higher Education Authority's PRTLI Cycle 4 and the National University of Ireland, Maynooth's President's Fund.

References

Bradley, J. (2005). What you (fore) see is what you get: Thinking about usage paradigms for computer assisted text analysis. *Text Technology*, **14**: 1–18.

Buzzetti, D. (2002). Digital Representation and the Text Model. *New Literary History*, **33**: 61–88.

Carlson, D. (2001). *Modeling XML Applications with UML: Practical e-Business Applications*, Redwood City: Addison-Wesley Professional.

Computer Professionals for Social Responsibility. (2008). *Participatory Design*. http://www.cpsr.org/issues/pd/ (accessed 10 September 2010).

Dominick, P. G. (2000). *Tools and Tactics of Design*. New York: Wiley & Sons.

Engeström, Y. (1987). *Learning by Expanding*. Orienta-Konsultit: Helsinki.

Firesmith, D.G. (1999). *Use Case Modeling Guidelines, Proceedings of International Conference on Technology of Object-Oriented Languages*. Santa Barbara, California, 184.

Glushko, R.J. and McGrath, T. (2005). *Document Engineering: Analyzing and Designing Documents for Business Informatics & Web Services*. Cambridge, MA: MIT Press.

Hayashi, L. and Hatton, J. (2001). *Combining UML, XML and Relational Database Technologies. Proceedings of the IRCS Workshop on Linguistic Databases*. Pennsylvania, Philadelphia.

Hughes, L. (ed.) (2008). The AHRC ICT Methods Network: Final Report. http://www.methodsnetwork.ac.uk/redist/pdf/finalreport.pdf (accessed 10 September 2010).

Kimber, W. E. and Heintz, J. (2000). Using UML to Define XML Document Types. *Markup Languages: Theory & Practice*, **2**(3): 295–320.

Keating, J. (2008). Signs of times present and future. *Humanist Discussion Group*, **22**: 219.

Keating, J. G., Teehan, A., and Gallagher, D. (2008).The Alcalá Account Book Project. http://archives.forasfeasa.ie (accessed 10 September 2010).

Kuutti, K. (1991). Activity theory and its applications to information systems research and development. In Nissen, H. E., Klein, H. K., and Hirschheim, R. (eds), *Information Systems Research: Contemporary Approaches & Emergent Traditions*. North-Holland, Amsterdam, pp. 529–50.

Kuutti, K. (1996). Activity theory as a potential framework for human–computer interaction research. In Nardi, B. M. (ed.), *Context and Consciousness: Activity Theory and Human Computer Interaction*. Cambridge, MA: MIT Press, pp. 17–44.

Leont'ev, A. (1981). *Problems of the Development of Mind*. English translation. Moscow: Progress Press (Russian original 1947).

Martin, J. (1991). *Rapid Application Development*. New York: Macmillan.

McCarty, W. (2008). Signs of times present and future. *Humanist Discussion Group*, **22**: 218.

McCarty, W. (2008). Evidence in and evidence of "Evidence of Value", AHRC ICT Methods Network Expert Seminar. http://staff.cch.kcl.ac.uk/~wmccarty/essays/McCarty,%20Evidence%20of%20value.pdf (accessed 10 September 2009).

Nardi, B. M. (1996). Studying context: a comparison of activity theory, situated action models and distributed cognition. In Nardi, B. M. (ed.), *Context and Consciousness: Activity Theory and Human Computer Interaction*. Cambridge, MA: MIT Press, pp. 69–102.

Newman, W. R. (ed.) (2006). The Chymistry of Isaac Newton. http://www.dlib.indiana.edu/collections/newton (accessed 10 September 2010).

Pichler, A. (1995). Transcriptions, Texts and Interpretation. In Johannessen, Kjell S. and Nordenstam, Tore (eds), *Culture and Value*, Beiträge des 18. Internationalen Wittgenstein Symposiums. 13–20 August 1995 Kirchberg am Wechsel. pp. 690–695.

Piez, W. (2010). *Towards Hermeneutic Markup: An architectural outline*, Digital Humanities 2010 Conference Abstracts. King's College London: Office for Humanities Communication Centre for Computing in the Humanities.

Schmidt, D. (2010). The Inadequacy of Embedded Markup for Cultural Heritage Texts. *Literary and Linguistic Computing*, **25**(3): 337–356.

Warwick, C., Terras, M., Huntington, P., and Pappa, N. (2008). If you build it will they come? The LAIRAH Study: Quantifying the use of online resources in the arts and humanities through statistical analysis of user log data. *Literary and Linguistic Computing*, **23**(1): 85–102.

Warwick, C., Terras, M., Huntington, P., Pappa, N., and Galina, I. (2006). *The LAIRAH Project: Log Analysis of Digital Resources in the Arts and Humanities Final Report to the Arts and Humanities Research Council*. London: University College London.

TEI Contributors. Text Encoding Initiative Guidelines. (2008). P5. http://www.tei-c.org/release/doc/tei-p5-doc/en/html/index.html (accessed 10 September 2010).