Some Reflections on Studies in Humanities Computing

Koenraad de Smedt
University of Bergen, Bergen, Norway

Abstract

In any academic field, research advances tend to percolate naturally to higher education in that field. In recent years, there has been a slow but steady increase in the number of courses and degree programmes in humanities computing. This paper presents some reflections on the status of humanities computing in higher education, in terms of curricula, degrees, and international student and staff mobility. The most important issue is the question of what a humanities computing degree should offer, in view of the wide interdisciplinarity of the field. Different institutions have coped with this question in very different ways. With potentially far-reaching consequences on methodology in the various relevant disciplines, humanities computing is bound to change both what and how humanities students learn. Curriculum innovation that aims to integrate computing in the humanities is a difficult process that requires reflection, cooperation, teacher training, and other supporting actions.

1 Introduction

With a research tradition spanning about half a century, humanities computing has become an established academic field, with its own journals, conferences, learned societies, research centres, and support units. Good educational programmes of humanities computing studies would therefore be a legitimate desideratum. At the moment, there is only a small, but increasing number of universities that have established dedicated courses, degrees, and departments in humanities computing. Alternatively, some universities have integrated, or are integrating, aspects of computing in the various traditional humanities disciplines. In those cases, a few disciplines are typically at the forefront with respect to adopting the new technologies, whereas others lag behind. In general, it can be said that the offerings of computer methods in humanities curricula vary widely between universities and departments.

In all fields of education, the computer has come to stay. Unfortunately, much of the hype in educational information technology is due to a narrow focus on technical, material, administrative, and service aspects, whereas the introduction of innovative scholarly methods in curricula is
less often taken into account. The fact that students are surfing the Web seems to be a cause for enthusiasm for many educators and software developers, but it is questionable if this development is leading toward any appreciable increase in humanities students’ use of dedicated humanities software (such as, for instance, concordancing packages) compared with, say, 10 or 20 years ago.

In the 1970s and 1980s, humanities scholars interested in computing usually learned a programming language thoroughly. They first used punch cards as a tedious input medium, later monochrome character-based displays. Today’s humanities students have access to machines with memories roughly a thousand times larger and equipped with colour displays, but they do not necessarily have better skills in programming or data analysis to tackle relevant research questions in their fields. Still, at several universities, new courses are being introduced that teach the new technologies.

2 A Reflection on Educational Priorities

In the year 2000, at least one university began offering a course in computer literacy including Internet access to humanities students. This kind of initiative comes almost a decade too late and should by now be superfluous at any level above elementary school. Experience teaches that pure computer literacy courses have a very short lifetime. If this kind of teaching is given under the label of computing in the humanities, it can damage the credibility of the department and faculty concerned, which would be a pity for the sake of humanities computing.

Many universities have embarked on integrating Web-based courses into the curriculum. Dedicated Web servers, such as WebCT, support Web-based courses by offering prepackaged course material, providing preprogrammed tests, tracking student progress, allowing students to write notes to the material, and so on. Rarely, however, do these tools incorporate computational support for the creative analysis of a scholarly problem. Also, traditional teaching in humanities subjects is increasingly being supplemented by discussion groups on the Web and so-called MOOs or multi-user domains, which allow interaction in a virtual space. Also here, the focus is mostly on tools that may be useful for communication but that are not supporting specific scholarly methods relevant to the subject matter.

On the one hand, it seems that many universities’ strategic priorities in humanities computing are concentrated on packaging and delivery of traditional teaching materials. Information and communication technologies thus act as a gift wrapper, with colourful multimedia and hypermedia, often in an attempt to make education more efficient and more in line with current fashion. There is, of course, nothing wrong with new ideas on how a subject is to be taught, as long as they start from the question of what is to be taught, rather than the reverse. Furthermore, the question of what is to be taught should be approached in the light of a reflection on why we teach and learn humanities.
On the other hand, academic staff involved in humanities computing are increasingly promoting curriculum innovation. As a consequence, we can fortunately observe ample occasions where academic staff engage in reflections and discussions on the design of humanities computing studies. These reflections and discussions are evident in scholarly publications and at scholarly meetings, particularly those of the ACH-ALLC, often through the presentation of concrete plans for new degrees in humanities computing (Hockey et al., 1997, 2001; Nowviskie and Unsworth, 1999; Unsworth and Butler, 2001).

From September 1996 to October 2000, a consortium of European universities carried out a SOCRATES/ERASMUS thematic network project called 'Advanced Computing in the Humanities' (ACO*HUM). Although limited in terms of which humanities subjects were covered in depth, the project addressed far-reaching questions such as the following (Orlandi et al., 1999):

- What is the extent to which the nature of humanities scholarship changes as a result of the incorporation of formal and computational methods?
- How does the adoption of formal and computational methods have an impact on humanities education?
- What are the theoretical and practical problems involved in the integration of advanced computing in humanities curricula, across the various disciplines involved?
- Can a common approach be identified offering formal and computational methods for all humanities students?
- In which respects should humanities computing be taught differently from any other sort of computing?
- Is humanities computing some kind of niche, separate from other humanities education, or is it being made available to all humanities students?

The network’s reflection on computing and humanities education was anchored to a consideration of the changing needs of humanities graduates in society (De Smedt, 1999). Whereas traditional humanities graduates increasingly fail to find jobs, there is a shortage of workers who have integrated both computing and humanities competencies in their curriculum. The following is an attempt to categorize new profiles for humanities computing graduates:

1. graduates who can integrate new computing methods in traditional humanities jobs: archives, libraries, museums, publishing sector, and culture and entertainment sectors;
2. graduates who can integrate humanities knowledge in traditional information technology jobs: the communication industries with applications in telecom, media, car manufacturing, distance education, etc.;
3. graduates who can develop new humanities information technology: basic and applied research in text encoding, human language industries, information retrieval, computer games, digital art, etc.
3 Competencies in Humanities Computing

Clearly, the competencies implied by the new profiles for humanities graduates are not acquired by placing a computer in front of every student. More important than the use of machines are new ways of thinking, based on computational methods. Central in these methods are the formalization and quantification of scholarly problems. Working from either formal or quantitative representations, computational methods assist in generating new knowledge. Computational modelling, simulation, and statistics are examples of such computational methods. For any of these methods to make sense to students, they must learn to apply them to the specific objects of study in a discipline.

In that context, it is not superfluous to consider the various disciplines related to humanities computing. Being an interdisciplinary concept by definition, humanities computing is not exclusively a matter of the humanities, although most of the relevant teaching is situated there. But even the humanities do not represent a clearly delineated group of disciplines. The extent of humanities as a collective term for a group of disciplines is one that varies somewhat with the different academic and cultural traditions in different places, as well as with the various names given to the faculties and schools.

A case in point is linguistics, which at many European universities is usually situated in humanities faculties, close to literature, and indeed often in the same department, whereas at other universities, notably US ones, linguistics is often situated outside the humanities. Another aspect to be considered is the place of computer science, which cannot be ignored, as it provides important premises, and indeed advances, for humanities computing. At some institutions, computational linguistics is carried out in computer science departments, whereas at other institutions, computational linguistics is situated in linguistics and literature departments, at the heart of humanities faculties. A survey carried out by ACO*HUM (De Smedt et al., 1999, pp. 97–102, 138–54) brought these and other important differences between various European institutions to light.

What then are the computational methods that students in the various humanities disciplines learn? In linguistic studies, some computational methods in curricula consist of formal linguistic modeling (parsers, formal grammars, etc.) whereas others deal with quantitative empirical studies (corpus linguistics). In literary and textual studies, some scholarly methods involving computational approaches are text encoding, edition philology, and statistical approaches in formal stylistics. Students in some history departments learn to use data analysis methods for designing and mining electronic archives. Scholars in history of art learn to work with, for instance, image recognition and classification.

Although this brief list of examples is only illustrative, it does raise some questions. Given that the different humanities each have their own objects of study as well as their own goals and perspectives, we can ask: is there a single canon of computing methods that every humanities student
must learn? On the one hand, experience has proven that it is a pointless effort to teach computing to a humanities student without integrating it into the student’s domain of expertise. On the other hand, it is an equally pointless effort to teach computing methods in any discipline without an understanding of what computing itself is all about. The question then is, how far should the teaching of humanities computing differ from the teaching of computer science? At this point, it is useful to distinguish between various categories of humanities computing. These could be tentatively given the following names.

*Humanities computer literacy.* A large number of courses at European universities are dedicated to the provision of basic computational skills for humanities students. These are usually geared towards specific disciplinary needs: For instance, a student of Russian needs to know how to write, display, and print Cyrillic; this is even today not an altogether trivial task. As long as the use of computers is related to skills only, they do not influence the way in which scientific results are gained. At this level we are simply talking about the application of tools.

*Humanities computer applications.* A much smaller number of courses, and a substantial number of research projects, use strongly computer science-based methods (such as database technology, applied to an information analysis of some specific problem area). Alternatively, they use computation-intensive methods (such as statistics) to gain scientific results, which could not be gained without the tools employed. At this level, therefore, we talk about the application of methods.

*Humanities computer science.* An even smaller number of courses and projects, finally, deal with the study and the development of computational methods themselves, aiming at their improved understanding. It might be expected that their results are relevant for a wide range of disciplines.

4 Commonalities and Differences in Humanities Computing Programmes

Whereas many institutions of higher education have pursued a discipline-wise integration of computing in humanities curricula, under labels such as *computational linguistics* or *historical informatics*, other institutions offer dedicated courses or programmes under the wide-ranging label of *humanities computing*, *digital humanities*, etc. One would expect that the latter deal with humanities-wide computational methods. In practice, however, one finds considerable differences between the way different institutions have implemented such programmes.

One current programme offered at Groningen, for instance, is strongly based on formal methods in logic, language, and cognition. Its curriculum includes natural language processing and language technology, corpus linguistics, logic and logic programming (Prolog), formal and mathematical methods, information analysis, cognitive modelling and neural networks, and statistics. By way of contrast, a very different programme in humanities computing offered at Bergen focuses on cultural
aspects of information and communication technology. Its curriculum is geared towards historical and technical developments as well as social consequences of information and communication technology, use of the Internet, multimedia and hypermedia, a cultural and critical perspective, digital culture and computer games, practical computer use and programming, and statistical and quantitative methods.

It would be presumptuous to say that one of these two approaches is better than the other. It is significant, however, that the overlap between these instances of humanities computing programmes is small and that the two programmes cater to rather different types of knowledge and skills. Although in name not discipline-specific, they nevertheless differ clearly as to the profiles of their student audiences. Whereas the Groningen program is strongly geared to students of language and logic, the one at Bergen seems more appropriate for students interested in media, culture, and aesthetics.

Of course, it is good to have a certain variety between programmes at different institutions and in different countries, as variety provides richness. Specializations are, however, mostly useful at advanced levels of study. At an introductory level, widely different approaches with strong biases tend to form niches and may discourage student mobility. One would expect that programmes in humanities computing at the bachelors and even taught masters levels would aim at developing some common foundation of knowledge that is applicable in a wide range of humanities disciplines.

The need for a balance between humanities-wide methods and their discipline-bound applications is explicitly recognized in the following two examples of planned study programmes. The MA in Digital Humanities that is planned at the University of Virginia (Unsworth and Butler, 2001) aims at providing students with experience in recognizing and articulating problems in humanities computing and working collaboratively to solve them. The design of the degree is motivated by the convergence of many aspects of our cultural heritage in the digital melting pot, and will prepare students to meet the cultural need of managing this migration. Although the motivation for the programme is truly humanities-wide, the course of study includes discipline-specific electives that provide each student with in-depth course work in the student’s particular subject area.

Another new programme is the MA in Humanities Computing at Alberta, which emphasizes interdisciplinarity and co-operation rather than niche formation. The masters degree is offered with a large range of specializations: Applied Linguistics, Arts and Design, Chinese Literature, Classics, Comparative Literature, Drama, East Asian Studies, English, French, German, History, Italian, Japanese Literature, Latin American Studies, Linguistics, Music, Philosophy, Political Science, Religious Studies, Russian, Spanish, and Ukrainian. This shows not only that the term ‘humanities’ is very broadly read by the thirteen participating departments, but also that it could offer a real opportunity for intense interdisciplinary co-operation (Unsworth and Butler, 2001).
5 Issues in the Design of Humanities Computing Programmes

Within the ACO*HUM project, an attempt has been made to identify some core elements of humanities computing knowledge and skills. Two main foci of humanities computing have been distinguished that could be of value to students throughout the whole range of the humanities:

- formal methods in the humanities;
- textual scholarship.

From these foci, a list of more precise areas of study could be developed. For textual scholarship, the following preliminary list of core curriculum components has been suggested (Ore et al., 1999, p. 88):

- information analysis, modelling, and system design;
- information retrieval and filtering, including the use of large digital resources, such as bibliographies, reference works, and corpora;
- representation of character sets and writing systems;
- text mark-up (including the TEI);
- automated text analysis;
- electronic (non-critical) editing and publication (including XML and HTML);
- digitization, and image manipulation;
- tabular data, including spreadsheets and simple databases;
- the creation and management of digital resources, including metadata and resource discovery and preservation;
- the role of computing in society and its implications for cultural life and the transmission of cultural heritage;
- statistical methods;
- design of complex relational database systems.

Remarkably, fundamental issues in text encoding seem to be missing from some older humanities computing curricula, although in most newer ones, text encoding courses are clearly offered. Text encoding seems to create the foundation for almost any use of computers in the humanities. Although many courses discuss text formatting from the viewpoint of practical purposes only, students rarely seem to be given enough theoretical background to allow reflection on the nature of the encoding, on the levels of encoding, or on the consequences of the basic designs of markup systems. As a result, for instance, many humanities computing students have never imagined that there is an alternative to a document having a strictly hierarchical SGML-like markup structure. This may seem like a trivial detail, but in fact it is a case in point in that a purely instrumental approach to humanities computing falls short of the generally desired goal, that students should develop a critical perspective in the course of their higher education.
6 Pedagogical Benefits of Research Advances

One organizational and pedagogical problem with text encoding, as with other elements of possible humanities computing topics, is that, on the one hand, there is little room in the curricula of the different disciplines to explore such topics thoroughly, and on the other hand, a separate humanities computing course risks teaching methods *in vacuo*, not sufficiently related to the different scholarly traditions and objects of study.

Another problem is that teaching of some computational methods, notably statistics and quantitative methods, seems to lag significantly behind the research advances made in relevant fields. Some instances that could be mentioned are the fields of stylistics and dialectology, where many members of the teaching staff are still staunchly resisting the adoption of computational methods. Nevertheless, great progress has recently been made in applying quantitative computational techniques in these fields, not only in terms of research results, but also in visualizing these results in insightful ways that can benefit students (e.g. Nerbonne *et al.*, 1999).

On a similar note, it has to be mentioned that advances in computation need not be a burden on teaching in the traditional disciplines, but can be a help in practical instruction. A case in point is the well-known computer program *Tarski’s World* (Barwise and Etchemendy, 1993), which was awarded the 1997 Educom medal. The program allows the student in philosophy and logic to construct and test logical sentences relating to a world represented by simulated three-dimensional objects on a screen. By doing this kind of exercise, students learn how to write well-formed sentences in the language of logic and to determine the truth of such sentences. Barwise and Etchemendy argue that it is advantageous for the student to quickly identify and correct his or her own misconstruals of the language, rather than waiting for classroom help or, as happens all too commonly, completing the entire course without the problem being noticed.

Despite the deceptive simplicity of *Tarski’s World*, the game-like nature of the simulation allows a flexibility that is hardly possible to achieve in a book. Students can evaluate given sentences with respect to any given world. Conversely, students can construe a world that makes a set of sentences come true. In addition, students can make infinitely many new sentences to express facts about a world, and test these. The creative nature of the exercises is found enjoyable by students and stimulates the understanding of logical reasoning. Briefly, here we have a program that is not simply a replica of a book with some multimedia giftwrap. This learning tool is truly powerful: it has an underlying engine that embodies the laws of logic itself. Computation, in this case, contributes to making students of philosophy and logic understand how logic works by means of a simulation game.

As far as linguistics is concerned, similar developments are closing the gap somewhat between traditional linguistics curricula and computa-
tional linguistics. Among the available CL tools usable in education are several systems for the automatic analysis (or parsing) of sentences (De Smedt et al., 1999). Such tools not only perform sentence analysis but also present the sentence structures visually in graphical ways that linguists are used to, including representations in the form of hierarchical tree structures and feature-value matrices. The educational benefits are considerable compared with students doing grammar on paper. Without the need to develop programming skills, students using these tools quickly learn the relation between a sentence and its syntactic structure as defined by a grammar. It is therefore hardly surprising that such tools are not only used by computational linguists, but have found their way into general linguistics curricula.

If various scholarly tools become more refined, accessible and user friendly, it would not be unreasonable to imagine that, in a foreseeable future, literature students will routinely use on-line concordances and automated keyword extraction to verify an analysis of the role of single women in eighteenth-century French literature; that history students will, without hesitation, use on-line archives and censuses to tackle an assignment about the relation between age, marital status, and profession in nineteenth-century Norway; and that students in history of art will simply use automated comparative visual search techniques to track the borrowing of motifs across national boundaries in seventeenth-century painting (e.g. Vaughan, 1997). Will humanities computing then slowly seep into the humanities disciplines to the point where it is no longer recognized as something special?

7 Levels of Computing Knowledge

When considering what kind of knowledge humanities students need regarding computing methods, I would like to distinguish between the use of tools, the application of methods, and the development of methods (Ore et al., 1999).

The use of tools takes place on two levels. First, at the survival level, beginning scholars learn to identify the formal requirements for their fields of interest. They are quickly confronted with an increasing amount of information that cannot be searched or retrieved other than by using tools that inherently refer to formal methods. Second, at the basic level, they have to identify and understand the coincidence of tools and problems to be solved by them, in that order. As an added value, they learn which tools should be applied to specific kinds of problems, what skills are additionally needed, and how to acquire them.

At an advanced level, the application of methods requires an abstraction of the material of the investigation as well as of the questions to be answered. Formal procedures provide for the deductive explanation of underlying structures and processes that go beyond the treatment of individual cases. The added value of formalization at this level is a deeper, more abstract understanding of the field, in which the application of a method to a problem is mediated by a formal analysis of the problem.
For instance, the student of computational linguistics has to go beyond the level of the general linguistics student who needs to learn the relation between a sentence and its structure. If the computational linguistics student also knows how the method for establishing such a relation works, then an insight into underlying structures and processes is in order, and this in turn creates further requirements for pedagogical approaches. For example, a visual representation of the step-by-step growing search space of a top-down parser referring to a left-recursive rule can help the student understand the limitations of that particular parsing mechanism (Black et al., 1999).

At an expert level, the investigation is no longer application oriented, but adds value by developing new methods for the explanation of human activities in some way. Usually this will take the form of new models, the workings of which need new algorithms. This development of new methods provides an added value by affecting the other levels, leading to a spiral of progress in the understanding of the field.

Although the development of new methods should be left to the experts, we should still consider if knowledge at the other two levels needs to be taught only to a small group of dedicated humanities computing students, or if it should be incorporated in all humanities curricula. My very formulation of this question may suggest that I would prefer the last option, and indeed I do, although I think it would be a point for discussion as to how far one can go in this.

In any case, it is a legitimate question, in how far humanities computing is a scientific field or a technology. Various study programmes reflect different positions on this question. The postgraduate MA in Electronic Communication and Publishing, based in the School of Library, Archive and Information Studies at University College London, has taken in students since 1997 (Hockey et al., 2001). The programme is successful in attracting students and delivering graduates to employers in the publishing industries. Although the programme’s theoretical component bears a clear relation to humanities, the practical instruction mainly covers relevant IT systems to create and manage Web sites. The MA, therefore, seems to represent in the first place a technical and professional study rather than a scholarly discipline. It would therefore fall outside of the humanities not so much because of its subject matter but because of its approach.

In contrast, the long-established M.Phil. in History and Computing at the University of Glasgow is a one-year taught postgraduate course providing training in the application of computer-based methods to substantive problems in history (Hockey et al., 2001). Instruction in this course emphasizes historical interpretation as much as methods and techniques. Here, the focus is on how to use the computer as a scholarly tool for analysis and modelling so as to enhance historical interpretation and contribute to key historiographical debates. Technical matters are subordinate to methodological ones. The M.Phil. does not deliver IT professionals but rather humanities scholars.

This brings to the forefront another aspect of the debate, namely, on
Some Reflections on Studies in Humanities Computing

the one hand, how far the integration of computing in humanities education has a positive effect on employability of graduates, and on the other hand, whether this implies that the humanities are on the road from liberal studies towards professional studies. Although the former seems to be a case and clearly would be a benefit, the latter does not seem a necessary consequence and is unlikely to receive unequivocal support. Universities still mostly pride themselves as providing an academic rather than a professional education. Although computing tends to enhance the applicability of approaches, the new methods are equally important for making purely academic advances in the field.

8 Concluding Remarks

In recent years, several humanities computing courses and degrees have been established, and more are being planned (a prominent example is the degree at King’s College London, starting 2002–2003). This process undoubtedly is a consequence of a real need, and at the same time offers opportunities for the humanities to renew itself. Clearly, humanities as a discipline is not untouched by the integration of IT methods. The formal and quantitative approaches that underlie many computing methods are making the humanities more exact in the way knowledge is produced, and these new methods are thereby somewhat reducing the fundamental difference from the natural sciences, which are sometimes called the ‘exact sciences’.

The telescope was invented in 1608 and was initially thought useful in war. Galileo obtained one, improved it a little, and used it to challenge existing ideas about the Solar System. Although a magnificent new technology in itself, the telescope was hardly a scientific tool until Galileo used it to create new knowledge. Today, if we wish to give humanities a credible future, we cannot be content to teach students to use computers merely to access and communicate existing knowledge. Those humanities computing programmes that take a purely instrumental approach to computing miss the spirit of Galileo.

New knowledge is being produced in traditional humanities subjects using a variety of computational methods that were not necessarily conceived for the humanities. But now, humanities scholars use, for instance, statistical techniques that for a long time were predominant in the social sciences; learning models that have their roots in psychology, neurology, and artificial intelligence; and Monte Carlo modelling, which was initially used to predict the fate of neutrons moving through uranium. Advances in humanities computing will be measured by the degree to which we succeed in stimulating students to make creative use of computational methods to discover new knowledge, not merely by the amount of Web pages that students read and write.

Still, it is necessary to consider both what and how humanities computing should be taught. Even at an introductory level, textbooks are hardly effective tools for learning humanities computing beyond the level of absorbing facts. Students need hands-on experience, experimentation,
and exercise. In an ideal world, students would have access to the same scholarly tools as researchers in their disciplines. Thanks to the Web, this is, in fact, technically possible today, but practical and organizational problems remain.

For one thing, curricula are changing too slowly in what are perhaps the most paper-based areas of study. Unless we want humanities computing to remain a niche, computing methods must be explicitly recognized as a way to transform traditional humanities disciplines. Furthermore, it must be recognized that rapid developments in humanities computing require a conscious effort to update the competencies of the teaching staff. In other words, the teachers have to be taught, through training and retraining programmes consisting, for instance, of short courses and workshops, conference attendance, on-line forums and help-desks, and access to on-line pedagogical resources and documentation.

Supporting actions that target these logistical needs are indeed worth mentioning. Especially in the UK, there has been a long tradition of support centres for humanities computing, in particular the Arts and Humanities Data Service (AHDS) and the various organizations that it incorporates. An example of a newer, international and network-based initiative is the Joint European Website for Education in Language and Speech (JEWELS) started by ELSNET in 2000. JEWELS is intended as a permanent, growing catalogue of courses and educational resources and tools, partly supervised by editors.

As more and more study programmes in humanities computing are created, the reflection and discussion on useful designs only increases. The ACO*HUM project has perhaps been the first major international project entirely dedicated to investigating the status quo, the needs, and the opportunities in humanities computing education. As a direct result of stimuli from this project, the ALLC, ACH, ELSNET, EACL, and other organizations have already taken an interest in relevant educational matters. It is to be hoped not only that more efforts will be undertaken in the near future, but also that higher education institutions themselves will take the consequences to heart and prepare to implement useful strategies.

References


Some Reflections on Studies in Humanities Computing


