Scholarship in interaction: case studies at the intersection of codework and textual scholarship

Zundert, J.J. van

Chapter 8

Working at the Intersection of Computational and Scholarly Styles of Thinking: an Autoethnography

> "What do you think machines have to do with your problem, can you elaborate on that?" – ELIZA
>
> (Weizenbaum 1966)

8.1 Introduction

My thinking is rooted in theoretical scholarship as much as it is in software engineering knowledge and experience, and in the memorable intricacies of personal and institutional politics. It took over fifteen years of de facto interdisciplinary academic work to amass the practical engineering experience, academic scholarship experience, and reflection to produce the insights I have tried to capture on the pages of this dissertation. The way this work came about thus differs from a more usual four-year PhD, and this experience has importantly also influenced my research and results. It is for this reason that it is useful to add an autoethnographic chapter to the body of this dissertation. It allows me to retrace how and why my thinking changed over the years, and this will be useful to understand how textual scholarship and software engineering interact and shape academic practice and convention.

I will not audaciously pretend that my personal experience should be a model for how the interaction between academic software engineers and textual scholars should occur. Ultimately, my motivation is this still relevant remark by Christine Borgman: "Why is no one following digital humanities scholars around to understand their practices, in the way that scientists have been studied for the last several decades?" (Borgman 2009). Although work has been done (e.g. Antonijević 2015), much more ethnographic work is needed to understand how software engineering, computer science, textual scholarship, and institutional politics interact. This type of work is pivotal to inform the agenda setting of the interdisciplinary intellectual work that goes on at the intersection of these domains – because developing an intellectual agenda is itself pivotal for this developing interdisciplinary field as Willard McCarty (2014) pointed out. Without it a field might just linger on merely as a methodological niche, being mostly auxiliary to other fields and over time dissolving into them.

Within the large documentary task Borgman called for this chapter will be just one data point, but I want it to be a richly annotated data point, as I think there is as much value of knowledge in experience as there is in analytical and quantitative reasoning. Towards the end I will draw some conclusions about the interaction between academic software engineering, computer science, and textual scholarship and where I think it should take us from where we are. It would be false pretense to suggest that those conclusions are solely and carefully based on scientific reasoning, because they are certainly also informed by subjective personal experience situated in a specific academic context. It is that experience that I try to document as a token scientific record with this autoethnography.

The application of autoethnography as a method commands a short defense, I suppose. The presupposition that any form of autobiographical documentation can result in valid scientific knowledge, has not gone unquestioned. Amanda Coffey, although acknowledging that the situated self of the investigator necessarily influences the fieldwork of any ethnographer, warns that "it remains debatable as to whether utilizing ethnographic strategies to write autobiography really 'counts' as ethnography at all" and that some would argue "that such texts are not 'doing' ethnography at all, but are self indulgent writings published under the guise of social research and ethnography"

(Coffey 1999:155–156). Obviously, I disagree. I would rather appreciate the ethnographic value of any form of information. My academic background may well play into this, as scholars of historical literature usually have no choice but to appreciate any and all historical documents that may have survived time's relentless jaws – happy with every scrap of evidence they can find. More importantly, as Galison (1999) has pointed out, "Objectivity is romantic": the distanced, disinterested, objective academic observation is an invention of the late nineteenth century connected to the rise of mechanical registration (photographs, phonograph, etc.) of data. But as he concludes: "There is no neutral strategy of machine usage followed by an ethical evaluation of it. The machine is moralized from the get-go." And the most moralized machine may very well be the ethnographer's self.

With that in mind, there is really little difference between the utility of autoethnography and that of methods perceived as more reliable – provided that sufficient explanation is given about how the information was gathered so that the information can be put to use responsibly. We should be careful about the reach of our claims. No, one cannot claim this account here as a universal truth, but yes one may use it as additional information to support my argument for a particular scientific agenda in digital textual scholarship.

As for accountability and explanation, Anderson (2006:378) defines five key elements of analytical autoethnography. The first is that the participant-observer must be a complete member of the social world under study. The second is that the autoethnographer should be aware of a deeper level reflectivity that stems from the reciprocal influence between ethnographers and their settings and informants. That is: both as an involved researcher and observer, the autoethnographer has an influence on and is influenced by the context he is studying. The specific kind of analytic reflectivity applied by the autoethnographer, according to Anderson (2006:382) "entails self-conscious introspection guided by a desire to better understand both self and others through examining one's actions and perceptions in reference to and dialogue with those of others." As a third requirement Anderson contends that an autoethnographer should be visible in his ethnography. Ethnographers have traditionally been an invisible but omnipresent narrator in the texts they produce. "Autoethnographers should illustrate analytic insights

through recounting their own experiences and thoughts as well as those of others. Furthermore, they should openly discuss changes in their beliefs and relationships over the course of fieldwork" (384). Fourth, for an autoethnography to be analytical it should be rooted not just in self-observation but beyond that in dialogue with those in a similar (or the same) situation. Finally, analytic autoethnography is defined by "its commitment to an analytic agenda" (387). Therefore this autoethnography should not be about me and my experience, self-absorbed or indulgent, but it should concentrate on the inferences that may be drawn from one's personal observational data and what these may yield as wider-reaching contentions concerning the social context one is working in.

So, in what follows, I will hold myself to Anderson's rules of the game.

8.2 An Autoethnography: Intellectual Spoils of Interdisciplinarity

8.2.1 About Trying to Retrace Key Experiences

In practice, research is not the smooth, neutral and well-defined process that many people seem to expect it to be. Many forces shape any particular research project besides the researcher(s), the data, the hypothesis, and method (cf. e.g. Latour 1987). In my experience, the power exerted by the politics inherent in institutions, projects, and funding is considerably larger than many researchers in the contexts I worked in acknowledged or seemed to be aware of. So are the politics and ethics of individuals striving towards personal goals. All of these forces together shaped the interaction that took place on the intersection between software engineering, computer science, and textual scholarship that I witnessed over my fifteen odd years of experience in an academic context. However, only traces of this more holistic perspective found their way to the pages of the previous chapters. Therefore I want to recapture some of what I regard as key experiences that shaped my thinking and reasoning in what went before as much as research and scientific literature.

My initial objective in researching the interaction between software engineering and textual scholarship was to prove a point, but the truth is that the actual point eventually found me. In the course of discovery the investigation evolved my style of thinking. My experience is therefore not just one of writing a dissertation, but of becoming a different type of thinker. I started out as a hard-nosed empiricist, and ended up leaning much more towards philosophy. In my experience there is a sensation in science somewhat similar to the Stendhal syndrome, the name for various real physical effects that seeing art may induce in people. There were a number of moments in which my perspective instantly shifted so radically that the sensation became distinctly physical – as if I could feel neurons click into their appropriate places. They seem mostly to have occurred when someone made – in my view – a key remark in research related conversation. I want to record these moments not just because they changed my thinking, but because I increasingly understood that I was part of the problem.

It took time for the problem to find me. But things also needed to be arranged for me so that I could discover the right tools to approach the problem I was causing. I needed one or two helpful gestures and some time to stumble upon the tool which is called Science and Technology Studies before I could actually start making some sense of my evolution of thinking in some slightly systematic fashion. Science and Technology Studies, or STS for short, consists of the critical examination of science, technology, society and the relations between them. Hence STS was a good fit for the subject I had in mind. But it was also a good fit because more so than the arguably somewhat bookishly oriented scholarship of humanities – especially the field of literary studies wherein I mastered – STS pairs theory with the empirical observation of human (and technology) behavior, in other words the stuff that my experiences seemed to be made of.

Long before I was knowledgeable about STS, however, I started my research by doing and making, by building software for humanists. But the results of this haptic thinking did not work very well. All the innovation and technology I offered textual scholars mostly seemed to generate resistance, however many tools and skills I threw at the problem. I had plenty of tools and skills though, and a good dose of positivistic technological thinking to accompany them. I had code repositories, I had user centric agile development meth-

ods, I had test-first programming, I had communication and documentation skills. I even had management backing. But none of this seemed to interest the textual scholars in the least. What remained was frustration and cynicism about all the failing effort, and most of all cynicism toward the lamentable humanists for being resistant to the blessings of modern digital technology and methods. Of course, what I was really lacking – without knowing it – was a proper framework to systematically observe what was happening at a socio-epistemological level, and I had no proper way of reflecting, of systematically trying to understand those observations.

Due to my lack of a systematic reflection at the time I have no formal scientific record of observations from the projects that played at the intersection of research and engineering for the years that passed between roughly 2003 and 2013. Lacking any STS knowledge prior to 2016, the autoethnography I write is necessarily an analytical autoethnography of hindsight and could therefore maybe better be called an analytical autoethnographic reconstruction. In an attempt to mitigate the most pernicious effects of memory distortion and subjectivity I apply the constraints for autoethnography as listed by Anderson (cf. above).

8.2.2 About Sensing but Mislabeling a Real Problem

It took a number of key experiences before I could first see that the problem was not so much one of difficult technology, lack of skills, and resistance to change. These may look like problems at a practical level, but in essence they are symptoms of a more fundamental problem that is socio-technical and foremost techno-epistemological in nature. At the time it was impossible for me to see this because I lacked the language and concepts that are needed to reflect on such issues. I did not know the term "socio-technical", nor the body of critical theory related to it. All that would only come a scant decade later. I thought of software in the usual terms of the industry: users, requirements, functional design, technical design, graphical interface, process, and data. In some sense the social dimension was brought to the table when Willem van den Ende – a very experienced programmer I was working with at the time to develop scholarly software – remarked that "the problem

is never technical, it is always social". Admittedly I still simply assumed, like Willem, that this social problem existed wholly on the side of the humanities scholars as users and had to do with resistance to change and especially technological change.

Around this time, which is 2005, Ronald Haentjens Dekker and I were developing various digital tools at the Netherlands Institute for Information Sciences (NIWI). One was called eLaborate. ELaborate started out as an, in hindsight, over-scoped content management system for humanities and social science data (Beaulieu, Van Dalen-Oskam, and Van Zundert 2012). ELaborate was not itself a tool for computational analysis but rather a tool for data entry. Our idea was to support textual scholars by making the creation of web-based scholarly editions easy. At the same time we would ourselves benefit from the side effect that born-digital editions generated machine-readable representations of historical texts that could also be used for computational analysis. From a methodological point of view this approach seemed completely sound. The idea of web-based digitally born editions that textual scholars could create collaboratively were directly inspired by what we knew from literature around Computer Supported Collaborative Work or CSCW (Greif 1988).

I remember that Karina van Dalen-Oskam, who was leading the project, called a progress meeting for stakeholders and key users where we demonstrated an intermediate release and argued computational approaches to the analysis of textual data. A discussion ensued about the interpretative aspects in which we contended that these were certainly not taken away from the researchers. Rather we pitched eLaborate as a tool supporting and not supplanting conventional research, and argued that the best analytical tool was still the researcher. One of the key users – a literary researcher – nodded in great agreement at that and added: "Indeed! And deep long thinking!" More than anything else, the belligerent tone of voice has made sure I can recall this moment at will ever since. It was the first time I was directly confronted with the fact that creating and offering digital tools could actually provoke not just resistance but outright aggression, born from frustration and misunderstanding between developers and users. To me this still stands out as a key moment because it made perfectly clear to me that this was not a problem that would evaporate because of the technical

quality of the tools we were making. This type of resistance would require more than just the usual documentation and training.

When Karina and I, and later also Ronald, moved to the Huygens Institute (which then still went under its former name "Constantijn Huygens Institute for Intellectual History and Textual Scholarship") the social science modules of eLaborate were transferred to other partners, but development of the core functionality that was concerned with text transcription and annotation was continued by me and Ronald. While online content creation now seems a rather mundane basic affordance of the Web, in 2005 "Web 2.0" technology was still quite new. Google had only just acquired an upstart that later was to turn into Google Docs (Wikipedia 2020). At that point in time eLaborate was at the leading edge of development with regard to web based authoring and collaboration, especially in the textual scholarship field.

The director of the institute, Henk Wals, perceived digital publishing as a viable and adequate means to supplant the exclusively print-based high-quality scholarly editions and reference works that were produced by the scholars in the institute. Reshaping the scholarly output as web-based publications made sense from a high level managerial perspective. It was expected that it would reduce cost, and that it would do so without lowering quality. Rather the opposite: digital editions would improve the support and affordances for scholarly editing, and with that their quality. At the same time it would allow the Huygens Institute to reposition itself as an institute at the forefront of digitally innovative scholarship.

ELaborate found some enthusiastic users and supported several projects successfully. Most notably a group of volunteer transcribers produced the transcriptions needed for a web publication of an early print late Middle Dutch translation of *De proprietatibus rerum* by Bartholomaeus Anglicus (1485 CE). The edition launched on 5 March 2010 (Werkgroep Middelnederlandse Artesliteratuur n.d.). In hindsight it looks like eLaborate offered value mostly to scholars and editors outside or only semi-related to the Huygens Institute. Indeed for such scholars eLaborate offered a free web based transcription environment with collaborative aspects – i.e. working on the same shared document – that were an improvement above mailing around copies and versions of documents. And possibly in time eLaborate

would also offer them a free or at least affordable outlet for the resulting editions. However, the welcome the tool received from textual scholars working in the institute was at best lukewarm.

Ronald and I as developers and Karina and I as scholars did not really grasp why textual scholars in the institute were reluctant to work with eLaborate. Although the tool had its shortcomings we were positive that the benefits – i.e. no version conflicts, a shared working environment, and the ease of working anywhere – should be ample enough to live with the downsides, which would be temporary anyway as development would continue. We mostly still figured that technophobia and resistance to change were the main causes of the less than expected success of the tool. We saw confirmation of that in the fact that we had a few willing users, a large group of skeptical scholars that might be convinced still, and a few scholars – mostly older ones – that outright refused to use eLaborate or any software that progressed beyond email and word processor. To us that seemed like a normal "demographic" make up of a group adopting a new technology.

8.2.3 About Getting it Wrong

What I was not able to see at that time was that from the perspective of most textual scholars – who had never had any experience with computers apart from using a word processor, browsing the internet, and using e-mail – eLaborate meant a complete disruption of their well established workflow. For us putting text directly on the web was a logical next step given our knowledge, use, and experience of Web 2.0 technologies. To editors who had done no digital work prior to that time at all, it looked like a reckless proposition to hand over the stability of print publication and a methodology sophisticated during centuries rather than decades to a technological fad with very doubtful claims to sustainability. More importantly we have never, I think, fully appreciated or considered how reductive the work model of eLaborate appeared in a methodological sense.

Through teaching and practice a scholar develops a professional understanding of the structural and analytical concepts that pertain to the practice of scholarly editing. The scholar learns to think in great detail

about such concepts as "document", "text", "transcription", "author", "context", "audience", and about the relations between them. As much through education as through experience a highly personalized systematic praxis develops for finding, selecting, reading, transcribing, annotating, and contextualizing historical sources. Eventually a textual scholar develops an unparalleled individual literacy in observing and describing select historical sources. On a conceptual level there are many commonalities between these individual systems. All editors, for instance, take notes and make transcriptions. But how these notes are taken, how transcriptions are made, and how, for instance, annotations are categorized, is primarily left to the scholarly editor. That is not to say there are no rules and no boundaries. Several well-known standard works introduce the work of the scholarly editor in fascinating detail (e.g. Mathijsen 2003; Greetham 1994). But a recurring motif in these works is a full recognition of the uniqueness of any text worth editing as to materiality, content, provenance, historical contexts of use, and interpretation. No two texts are the same, and therefore scholarly editors can get only so much mileage out of standardized rules for their craft. At some point all scholarly editors are confronted with the heterogeneity of their material. At that point it is the responsibility of the editor to account explicitly for the choices that were made in the editing of the text, and then to proceed conscientiously to a representation that the editor deems adequate.

In hindsight, again, eLaborate failed to acknowledge the idiosyncratic aspects of scholarly editorial work. It also only supported a rudimentary one-dimensional linear notion of text as a series of characters. It lacked a convincing way of delineating the multidimensional structures found in real world documents and texts. It did have very advanced annotation facilities that allowed for arbitrary and overlapping annotations of text. But that did not sufficiently compensate the lack of easily adjustable layout of published texts. In all this meant that eLaborate must have felt to scholarly editors as forcing an oversimplified model of text on them, one that would never allow for the subtleties found in source texts that they felt needed to be captured and expressed. Arguably editors would have been able to overcome many of the shortcomings they felt were part of eLaborate, either by workarounds or by acquiring some literacy in HTML (the markup language in which webpages

are written). Attempts at training and teaching however fell short and demand was insufficient to create a critical mass of eLaborate users inside the institute.

One cause that sunk eLaborate was our assumption that all scholarly editorial work could be reduced to one model of text and text processing. Surprisingly, maybe, we had arrived upon a one-dimensional model exactly because we understood the problem of text not being one-dimensional. We were well aware of the post hoc rationale of the Text Encoding Initiative's[1] XML[2] based hierarchical perspective of text by DeRose, Durand, Mylonas and Renear (DeRose et al. 1990). More importantly we were convinced that the TEI's hierarchical model – even if it was a de facto standard in the field of digital scholarly editing – was not a good fit for text that is in essence multidimensional (cf. Buzzetti 2002; Buzzetti and McGann 2006). Text has various layers of structure and meaning that are interconnected – chapter structure versus narrative structure for instance, or literal versus symbolic meaning, or the way syntax contributes to semantics. Because the meaningful elements of such layers and the layers themselves essentially are networked and not neatly nested inside of each other like Russian dolls, it seemed to us that a model that demanded an exact hierarchical structure of text was unfit to capture so many useful structures.[3] That is why we specifically chose to model texts as strings of characters that could be arbitrarily annotated while annotation could be stacked and overlapped in any combination. Note however, that we also did not provide a means of networking annotations – although we did provide a means, at first crude, of categorizing. The upshot of these choices was, I suspect, that we inadvertently created an impression with the textual scholars in the institute that we were working from very simplistic

[1] The Text Encoding Initiative or TEI is the de facto leading standard for marking up digital texts according to scholarly conventions. The TEI Consortium maintains and governs its guidelines. (Cf. https://www.tei-c.org)

[2] XML or eXstensible Markup Language is TEI's primary digital technology to markup text. (See https://en.wikipedia.org/wiki/XML for background and https://www.w3schools.com/xml/xml_whatis.asp for a quick primer.)

[3] We were also acutely aware that it is technically possible to express any other structure than a single hierarchy in XML. The way the TEI and XML are designed however, and especially the rationalization of these by DeRose et al., reveal a clear predilection for a strict hierarchical modeling.

assumptions about the complexity of text.

There are two opposing forces at work here. One is a force directed at abstraction and generalization, which is generated by the engineers and digital humanities researchers in this story. The other force is driving towards concreteness and specificity, and is produced by those who identify most as textual scholars.

A key moment that made me appreciate this opposition occurred when I returned – more or less – to the institute after a three year period where I had been largely uninvolved with the institute's developments because I was managing a large project that, for the most part, was externally oriented. Mariken Teeuwen (a classicist at the institute) was one of the researchers with a more benevolent attitude towards eLaborate. Although she judged that many things were still suboptimal she did use eLaborate as a primary tool during the collaborative editing of Martianus Capella's *De nuptiis Philologiae et Mercurii* (Teeuwen 2008; Teeuwen 2011). The text of "Martianus", as we colloquially got used to calling it, has been preserved in a fascinating document. The space between the lines of the core text bristles with glosses and annotations, while the margins are covered with more elaborate commentaries and Tironian notes (refer to figure 4 in chapter 7 for an example). While talking about the pros and cons of the tool, Mariken explained to me how she had actually used the tool to describe the text of Martianus. This use was quite counter to the intended use, which was to transcribe all the text on the page of a document into a single transcription window and subsequently to highlight the annotations in the original and mark them as an "annotation" (figure 8.1). Instead, Mariken's solution was to "lift" the original annotations from the base text in the window and to put them in full into the windows that were meant to hold annotations by the current editors, and specifically not the annotations of eleventh century scholars on Martianus' text (cf. figure 8.2). Mariken's creative use was born from the very practical consideration that it would have been cumbersome in the scheme originally intended to relate contemporary annotations to the medieval text. It was of course not the case that Mariken and her team did not understand the intended use of the annotation facility. It was simply that the creative use as a more direct description of the original document made a whole lot more sense to them than the envisaged use.

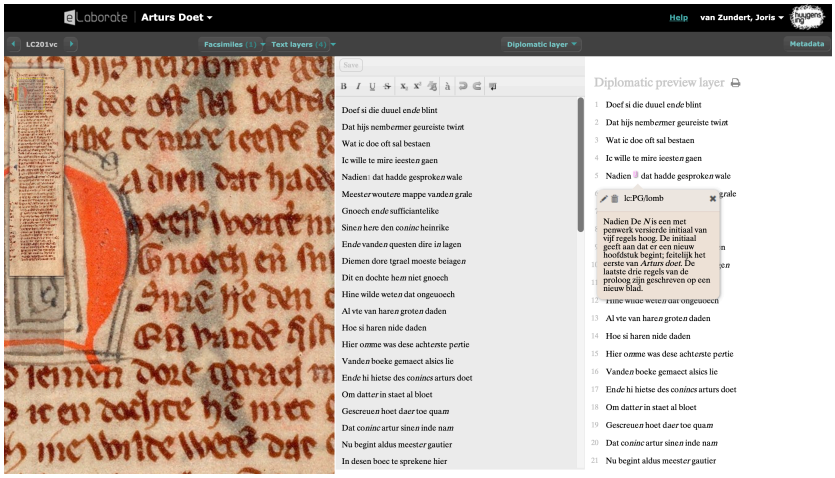

Figure 8.1: The "engineers envisaged use" of eLaborate's annotation function; the annotation box (pop up) is used to harbor an annotation by the scholarly editor.

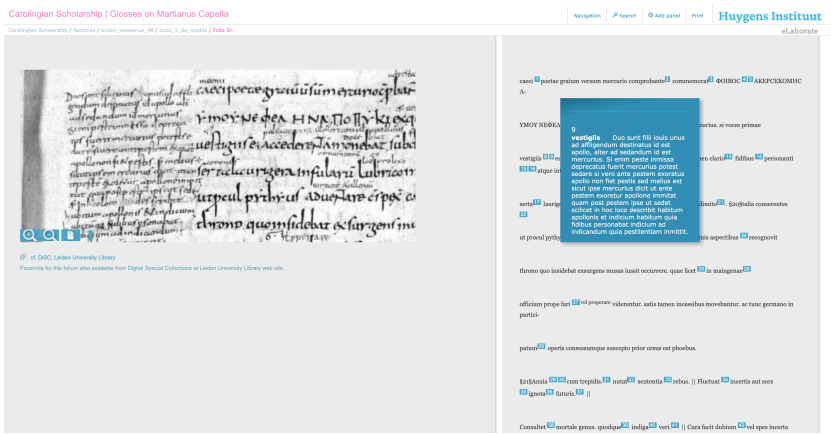

Figure 8.2: The "creative scholarly use" of eLaborate's annotation function: the annotation box (pop up) is used to harbor an annotation that was found in the margin of the transcribed document.

What I realized myself at that moment, if not for the first time then at least for the first time lucidly, was that code creators and textual scholars have opposing interests in the same technology. Engineers are trained to support automation by abstraction. That is: if there is a possibility to support a repetitive task or to capture an often occurring pattern in data they will create a digital function or object that will act as a model for that task or that pattern. Because many unique instantiations of the task or pattern can then be captured by the same digital object this is – from the engineer's perspective – a tremendous scaling and efficiency benefit. Ideally all annotations belong to the class (or category) of "Annotation". In this way the propensity deeply ingrained by training in engineers to draw abstractions from concrete objects in the interest of automation is also, by definition, reductive. Engineers err on the side of broad inclusion. In this sense we, as engineers, had expected that all metatext (i.e. all elements, information, and metadata that pertain to the text but are not actually in the text) would be represented by Annotation Objects. Conversely we had expected that all things in the text would be in a Text Object. This approach to modeling information is exactly opposite to how the textual scholars in the institute were thinking about text. Through their training and teaching they have understood that every text worth editing deserves the best scholarly description possible. In this approach there is no such thing as "abstracting away": all words, all characters, all individual structures of all categories within a text ideally should be explicitly labeled and named. In this approach of course an annotation is still an "object of category Annotation". However, that is not the essence of it, as engineers might reason. For scholars the fact that an annotation is an instance of the "class" annotation is its least important aspect. Far more important are the specifics: where, for which (part of the) text, by whom, what is it saying, when was it made, etc. This is why the eLaborate model applied differently than intended made for a better description of the text that Mariken was working on, because it lend the description more "immediacy" and therefore seemed scholarly more precise. Textual scholars prefer a more specific and more precise description of the specific historical source over some abstraction in service of digital (re)mediation. Their primary concern is not digital mediation but a precise scholarly description of the text. The automation, modeling, or digital support of the workflow is at best a "nice to have".

From a high-level overview and once again with the aid of hindsight it may be easy to gauge that these forces in principle do not have to work against each other but should be perfectly complementary, keeping textual scholarship highly precise and specific, yet allowing it to be digitally mediated. Applied carefully, respectfully, and with mutual understanding, digital tools might have had this effect. At the time we were initially developing eLaborate however, we probably created the impression that we were doing away with almost all specificity in text description. To be entirely clear: we were not. We intended for the specifics to be described precisely in various annotation objects. The textual scholars however preferred to see all specific textual constructs they perceived in texts visualized in the interface as distinct visual objects or representations. That is: they would have liked to see visual representations and digital objects expressing "page", "paragraph", "verse", "annotation", "line", "sentence", "chapter", "speech", "character", etc. etc. Instead, what they got with the first version of eLaborate was "facsimile", "transcription", and "annotation". This created a misunderstanding about the overly simplicity of the model used. We intended the scholars to use the annotation object as a dynamic modeling object so that they themselves might allocate any object, pattern or structure they found important to mark out in the original text and document. That is: we would have them selecting a paragraph, hit the "annotate" button and have them type in "paragraph" plus the reason why it got annotated. The reasons why the model was as generalized as it was are thus clear and proper from an engineering perspective. But the impression from the scholarly perspective, equally clear and proper, was of a rather underwhelming model, almost entirely reductive with regard to the rich multidimensionality of text.

Much of this relates to what is also described in chapter 3. In that chapter another example where the desired specificity clashes with the scaling benefits of abstraction serves to underpin the definition of an analytical notion of paradigmatic regression. What may be useful to infer from these cases is that "creative use" is an important signal that should not be neglected. Rather, as for instance Willard McCarty also argues, the moment a tool breaks down it reveals important information (McCarty 2005:41–43), in this case on the misalignment between models and goals of engineers and scholars.

8.2.4 About a Dialogue That Did Not Happen

Would there have been ways to better appreciate these different perspectives of programmers and scholars? The idea that the OHCO (Ordered Hierarchy of Content Objects) approach was fundamentally flawed was deeply entrenched among us, the technicians – we thought so both as scholars and as developers. Chapters 4 and 7 of this book should also make abundantly clear that I, notwithstanding XML's clear and great utility, remain convinced of this. I remember that we understood that textual scholars wanted more clearly defined text objects and visualization of these in the tool's interface (pages, columns, notes, etc.). We met them halfway by providing curated lists of annotation categories so that any text could be annotated as being a page, a column, etc. But working with these categories was often found tedious and cumbersome. I think we were all stuck at a purely functional level with respect to the tool at hand. The discussion between the scholars and developers always converged on missing functionality, the ease of use, how things looked, the rigid workflow etc. As far as I can recall, a real scholarly-technical discourse on text models never truly emerged in that time.

It is not a given that eLaborate would have been more successful if we had been able to leverage the more fundamental discussion, because it is not at all a given that we would have convinced the textual scholars that the eLaborate model offered more freedom and was thus in principle better for expressing multi-dimensional aspects of text. However, it was certainly not going to happen without such a discussion. And we might have acknowledged that the discussion kept hovering at an insufficient level of reflection earlier than we did, because there was a small group of textual scholars within the institute that started to work on TEI-XML transcriptions at around the same time. Scholars that were hitherto not using digital methods or techniques, or were even until that point actively eschewing them, started to experiment with XML-tools and the TEI text model. This, I think, should have signaled to us that the problem was not resistance to digital tools per se. Because why would they turn to Oxygen (a "word processor" for XML) at all in that case? Entrenched as our assumptions were, we interpreted it as resistance and competing solutions, not as non-verbal moves in a more fundamental discourse on text models.

8.2.5 About What Method Meant

I may be getting a little ahead of my narrative however, because I am quite sure that at that point in time I would not have been able to consider the problems that eLaborate met from a reflectional point of view with regard to text models. One reason for this is that I had not at all appreciated what "method" meant to the textual scholars in the institute. Moreover, I had not quite figured out how to relate the concept of method to the work of software developers or scholars. My understanding of "method" was no more than "the way things are done". As an analytic frame that is not very useful.

Much later Matt Burton (then a recent PhD in STS turned postdoc, now lecturing at Pittsburg University) would relate theory, method, and technique in the clearest way I have ever heard: "Method is theory plus a technique." That is: method is the systematic application of a technique to verify or falsify a theory one has about certain research data. The neat function of this definition is that it ties together three of the most important aspects of research work in a clear and reciprocal relation. And also only much later I would come across a wonderful phrase by Federica Frabetti: "I propose a reconceptualisation of the digital humanities as a field that can and should try to understand the digital in terms other than [...] the technical ones[...]. Such an understanding must be pursued through a close, even intimate, engagement with digitality and with software itself" (Frabetti 2012).

Understanding that an intimate exploration needed to be undertaken and having a clearer picture of the relation between theory (model), method, and technique would have been extremely useful. If so, we might have reflected more adequately on what the differences between eLaborate's model of text and other text models meant. But as it happened, in our software development context "method" meant foremost agile development principles. "Agile" is a set of development principles that tries to maximize customer value and user centric design in order to deliver best fit functionality even in contexts with highly volatile user requirements (Martin 2003). In an academic context new to digital methods user requirements change quickly and often. Therefore we chose "agile" as our development strategy. But "agile" puts the focus very much on function. Users describe requirements in the form

of "stories", e.g. : "If I press a button with a red cross the corresponding column is closed." This meant that in every meeting between scholars and developers it was not model that was foregrounded but rather functionality. The point of the methodology should have been to lead us to a discourse on text model, but the specific method was such that it would focus effort elsewhere.

I think that at the time we were developing eLaborate, method – in a research sense – was for us simply equal to digital technology. I was convinced that any tools we would offer textual scholars would mean an improvement simply because they were going to be digital and web-based. Those techniques were going to fast-forward textual scholarship method in the institute to the twenty-first century. Thus better tools would lead to better method, where "better" was somewhat naively defined as "digital" and not, as it ought to be, in terms of some new hermeneutic perspective or a specific epistemological gain. When prodded for exact benefits we kept returning to a "work anywhere, collaborate anywhere, explore anywhere" mantra – and we would add the affordances of the computer as an analytical tool once the texts would be machine-readable. The epistemological gains of computer-supported collaborative work (CSCW) seemed clear to us. Also nobody seemed to deny the uses and gains of being able to find historical resources online. But whether the method used to put historical resources online was also scholarly valid and well argued was a question that we had only a feeble, circular sort of argument for.

However, I am getting ahead of myself again. I keep meaning to return to three key remarks that were made before all of this, before I realized that the rationales of engineers and textual scholars created opposite dynamics. I think that without those remarks I would have been unable to identify the problem as such. Although these remarks will sound simple, almost self-evident, they strongly influenced the reflectivity of my thinking.

8.2.6 About Becoming More Reflective

At the time, in 2009, I was managing a large project called Alfalab, which was a Royal Academy subsidized project to leverage CSCW as a means

to strengthen the collaboration between the humanities institutes of the Academy (cf. Van Zundert, Zeldenrust, and Beaulieu 2009). Apart from offering an opportunity to researchers of different institutes to work together, the project was a trial balloon for an envisioned grouping of the humanities institutes by the Royal Academy management. The current Humanities Cluster in Amsterdam is what resulted from these plans and early trials. The carrot that drew the institutes – that traditionally have fiercely guarded their autonomy and independence – was substantial funding from the Academy for digital or computational projects that the institutes would bring to the table. A core working group of six to eight senior researchers and project leaders would run the project. The intention was that the individual projects would use tools and data located within other institutes, thus showing the potential of digital methods and techniques to leverage cross-institutional uses of data and technology. The ideal was that something of a humanities lab would emerge where researchers would be able to choose datasets and tools to process these sets with. Hence the name Alfalab, "alfa-wetenschappen" (transl. "alpha-sciences") being a colloquial designation for the humanities in the Dutch language.

In practice this turned out to be very difficult to achieve. Of the six "demonstrators" (i.e. example applications, prototypes, and pilot tools) only one integrated several resources from different institutes to any substantial degree. It was very difficult to come up with research questions that were both useful or interesting and that crossed institutional borders. But even given such questions it turned out to be even harder to curate and service highly heterogeneous legacy data in such ways that different partners would be able to reuse them. And although the idea of tools and data from a high level might have seemed promising, in practice very few really generalizable tools and generic digital infrastructure could be thought of. Most research questions would require highly specific one-time-use workflows, specialized data curation, and bespoke analytical computational tools. Therefore the project proved a success for the individual partners, but it mostly put in sharp relief how onerous it is to collaborate across institutional and disciplinary boundaries. The project demonstrated that digital infrastructure, data, and tools are no magic facilitators of such collaborations.

Given the aims of the project the group comprised foremost researchers

and/or scholarly-informed engineers that were at the forefront of digital methods and techniques within their institutes or even in their fields. Two of the researchers involved were Anne Beaulieu and Smiljana Antonijević. Rather in contrast to the others they were not digitally-inclined humanities researchers, nor engineers, but social scientists. Their affiliation was with the then existing Virtual Knowledge Studio that was essentially the STS centre of the Royal Academy (consult Wouters et al. 2013 for examples of its scientific output). Their task was to observe the project and report on the integration of digital methods in the various research flows. They also helped the group by informing them of knowledge and research on innovation diffusion, CSCW, epistemology, evaluation of science etc. – important reflective knowledge that was expected to support achieving Alfalab's aims.

The project leaders would gather on a semi-weekly basis in one of the offices of the Royal Academy to plan and discuss the progress of the various projects. These meetings also provided ample opportunity to discuss problems in a wider perspective. A recurring trope amongst the digitally-inclined researchers were stories about what was experienced as resistance towards digital methods or technophobia in their various institutes. We often talked about how we might lessen the anxiety of researchers towards digital tools, how better training and teaching might be provided to mitigate the clear lack of knowledge and skills that most researcher showed with regard to computational techniques. We also would sometimes be baffled and mystified about how Luddite some of our otherwise so well educated humanities colleagues seemed to be. We even considered if maybe different but mutually exclusive styles of thinking were to blame for the tremendous obstacles we experienced in introducing humanities researchers to digital and computational technologies: were some conventional humanities researchers seriously incapable of abstract thought and could they only relate to text, for instance, as a linear thing?

In short: our discourse was clad in assumptions that the fault ought to be sought with the scholars. Until Anne, taking part in one of our meetings, iterated a number of other possible causes ending her list with: "...or maybe these tools are just not good enough." A remarkably unremarkable remark at first sight. But to someone caught in an echo chamber it was a very effec-

tive means of turning the argument back to oneself, in other words: to make it more reflective. It was simply a question we, or at least I, had never considered. I had always assumed that by default the software we built and offered to scholars of humanities subjects was going to be an improvement. Looking back, I have a hard time understanding exactly why I thought this. Apparently the digital nature of these tools brought them well beyond questioning. And certainly their web-based nature (in most cases) gave them an aureola of reaching beyond anything scholars had seen before. That all of that might do actually nothing at all to improve the scholars' methods, I had simply never considered. I think Anne's remark, as simple as it may sound today, caused me to get into a more reflective stance of thinking: for the first time I started questioning what I was doing, rather than making assumptions about what others were doing for which reason.

8.2.7 About Finding a Method

Later in the project Anne made another remark that has distinctly influenced the way I think and approach problems. We were in the more final stages of preparing a scholarly contribution to an edited volume (i.e. Van Zundert et al. 2012). I was satisfied with how the writing was proceeding, as did most of the (six) authors. But at the point I was for the most part satisfied Anne remarked: "I want to go over the text one more time. There's an imbalance in who's talking about whom, the dynamic is rather forcefully from engineer to researcher, and I don't want this to be another technology push article." I had never before perceived what I was doing as pushing technology on anybody. I simply expected that the clear benefits of the tools I presented would convince any reasonable person that they were better tools than the tools of the trade currently in use. The "push" remark unsettled me a little. The effect of it was profound and, judging now, for the better in that I started to think about how my work was being perceived by the scholars I hoped to convince. Until then I was in a pretty deterministic state of mind: if the tools were genuinely better, they would eventually simply win out against obsolete methods. But what if my tools were not noticeably better, and what if I was making not the slightest positive impression on the scholars I was trying to convince? In short: Anne's remarks made me see that convincing was not

a matter of (technical) merit alone. But the takeaway for me in her words was not merely that, nor that we needed to "build support within the scholarship community", nor that we needed "better PR". All true enough, but the salient point for me was seeing the merit of systematic reflective thinking.

To be honest I had interpreted most of what I had seen from STS people until that time as academically glorified Luddism. It seemed to me that their stance was pre-cooked and definitely anti-technology. But because of Anne's remarks I understood theirs was indeed a sincere method. A method to examine power structures, networks of influence, and behavior. From the perspective of the scientists being observed by social scientists, ethnography is meta-research, which in my academic context met with sincere indifference most of the time. When discussed, meta-approaches even met with derision from a good number of colleagues whose judgement I normally find important and insightful. But indeed: what if our tools were just not good enough? I started to think that maybe I stood to gain much from reflectively examining my own motives and actions, from examining how what I did impacted the humanities researchers I asserted to aid.[4] Whether I was part of a solution or a problem I did not yet know, but I had figured out that I was part of an equation that itself needed some critical reflection.

As a result of Anne's remarks my interests shifted in two ways. The first was that I got more interested in the actual work of textual scholars rather than in what my master's education had taught me about literary and documentary

[4]A tangential note on reflectivity in computing. Interestingly there is a technique called reflectivity in computer programming. A computer language can be used to interrogate objects active in the language itself, that is: a programmer can make code that examines the properties from some other objects in that language or code. E.g. a programmer might ask from an object what functions it supports, what variable types it expects for these functions, etc. The technique is not very often used and leads a pretty academic existence, but it does have significant application in program language design, performance testing, code profiling etc. The application of such techniques is often called "meta-programming". Thus the domains of computer science and software engineering do encompass reflective techniques and thinking, but these meta-modes are very seldom applied in my experience. Meta-programming is a highly specialized niche. I cannot help but wonder if the little reflectivity I found in these domains and the level of adequate interaction with scholars are related.

scholarly editing: how they did their work, how they motivated it, what they perceived as their method, etc. Although I still assumed that digital technology, also the technologies I was working on, would improve their methods, I began to be more aware of what could be a problematic technology push indeed. Maybe the textual scholars that we were trying to support did not experience our attempts as supportive at all, but as academic politics to push a poorly motivated technology onto them. The second shift followed from that: I developed a more analytical reflective approach to the problem and now got interested in the ways digital technology would affect and change the ways in which these textual scholars were working.

8.2.8 About Identifying the Real Problem

The thought that just maybe our tools might not be good enough was reinforced a little later as the result of another event. We accompanied the launch of the "demonstrators" that the Alfalab project yielded with a symposium on digital humanities. One of the invited speakers was Annamaria Carusi who has a highly interdisciplinary profile in humanities, philosophy, medicine, and in digital approaches bridging these domains. For us she was therefore an example of successfully working at the intersection of the humanities and digital technology. Her address on digital tools and humanities theory (Carusi 2011) was critical. The main thrust of her argument was that digital tools in science, medicine, and the humanities were doing poorly because computer scientists and software engineers had taken little notice of the humanities aspects that are involved with developing such tools. To underscore her argument she stated: "They have been working on this for more than forty years, and what have they gotten us? Predicate logic! Something we knew already for about two thousand years! Thanks to philosophers – like Aristotle." The statement was arguably intentionally hyperbolic. But it is actually a quite fair depiction of how far computer science has managed to come: the overwhelming majority of general-purpose computer languages (Java, C, Python, Ruby, and all their ancestors and derivatives) are indeed rooted in first-order logic. Only much later I would appreciate that this makes them a poor fit for hermeneutics oriented humanities computation (cf. Gamut 1991:75–76). Nevertheless, Annamaria's contention and Anne's

remarks were eyeopeners for me: I was to genuinely question the capabilities of the tools we made.

Key remarks are not made only during research meetings proper. They may crop up in any context. During drinks following a presentation and networking event, for instance. They are not all voiced as constructive dialectic either. Sometimes people voice contentions that seem so utterly flawed that it is hard to decide where to start pointing it all out. Imagine a small circle at a scientific reception. A director of the Royal Academy, a director of one of the Academy's institutes, a renowned scientist in the field of natural language processing, one of computer science, and me. We got caught up in a discussion about the perceived slow take up of digital and computational methods in the humanities. At some point one of the scientists resolutely waved aside the more convoluted point one of the others was trying to make. He argued it was just natural to have some early adopters, a large chunk of people that would only slower adopt the new technology, and a number of laggards that would maybe or maybe not convert much later. "Exactly," said one of the directors, "I firmly believe this resistance is an issue that will pass with time. It's a generational thing." "Indeed," nodded the computer science guy, "the older researchers may well let this treat pass them by, but the next generation will take to it with great ease." Obvious this was just talk at some reception – maybe even in part the wine talking – but there was asserted profundity in the statements by these leaders of fields and institutes. What struck me was the complete and unshakeable believe in the superiority of digital approaches as compared to conventional scholarship. I would only later learn that this is technological determinism at its most positivistic. But it was the generational thing that drew my more immediate attention. To understand why I, at least intuitively, knew they were blatantly wrong about that I have to digress a bit.

8.2.9 About the Myth of the Digital Native

As it happens I have always had a propensity to be intrigued by people *not* getting a computer, or software. I was raised in a family where the computer was already a technology that was well integrated in household live from the

mid 1970s onward. My father was a self-taught expert and later professional in the hardware and software that would be used in secondary school teaching. The home were I grew up contained early self-built microcomputers, a PET CBM[5], later C64's[6] and the first IBM personal computers[7]. I played *Pong* before I had ever touched a tennis racket. I knew how to operate a punch card reader before I took my first multiple choice test that applied punch cards to register the answers. When *WarGames*[8] was released I was eleven and I found it kind of cool that I was actually programming computers, although I recognized I was not even close to anything like cracking systems – but I understood the basics of code in a few computer languages (Basic, TurboPascal, and assembly language[9]).

In that time, a youth spent in a household close to several computers, thanks to my father, also meant walking around in a laboratory for the observation of first contact between humans and computers. People would come visit because they knew my father had some of these things that were interesting and apparently useful. Some asked if they could be educated a bit by my dad, others found a convenient place to use word processors while not at work, and there were my father's friends and colleagues who fed their mutual professional hobby. From these observations I can report the two things that computers and software excel at: pissing people off and convincing people they absolutely suck at operating a computer.[10]

Some twenty years and two generations of students later, during the latter days of my university education, nothing had changed. Computers still excelled at pissing people off, they just did it at an exceedingly large scale, and with impressive devotion. Software had dramatically improved its skills to drive people nuts. Ubiquitous peripheral devices, like external drives, modems, and printers added to mystifying (non) behavior of digital

[5] See https://en.wikipedia.org/wiki/Commodore_PET

[6] See https://www.c64-wiki.com/wiki/C64

[7] See https://en.wikipedia.org/wiki/IBM_Personal_Computer

[8] See https://en.wikipedia.org/wiki/WarGames

[9] See https://en.wikipedia.org/wiki/BASIC, https://en.wikipedia.org/wiki/Turbo_Pascal, and https://en.wikipedia.org/wiki/Assembly_language

[10] I do realize this wording may be perceived as unacademic. I suppose "inducing technology related anger and instilling a feeling of utter ineptness" might have done. In the interest of academic precision I chose to keep the original formulation.

technology that was often sure to provoke physical aggression. Because I was one of the geeks that actually had figured out how to tame these weird machines, I was one of those persons that was often called upon to help. I even turned that capacity into a short career at a computer retail firm, which paid for part of my university tuition.

These experiences taught me that the divide between "getting" and "not getting" computers and software is not a divide along the often suspected hard boundary between sciences ("gets computer") versus humanities ("will never get it"). It is also not a divide between young and old. Kids can play games very well, much better than their parents. And they are avid social media users. But they really are not better at all at understanding computers or software (cf. Scott 2013 for more anecdotal evidence). And it is not in any way generational: every generation has the same tiny percentage that naturally takes to how computers operate (or are operated) and the same far larger majority that rather would beat the crap out of the machine most of the time. Of course there are more scientific narratives to back this up (e.g. Kay 1993:81).

Because of this background knowledge and experience I instantly knew that what was said at that reception made no sense. The "problem" was not going to resolve with the influx of a new generation of researchers. Maybe it was the utter positivistic attitude that rang so loudly in these remarks that made me appreciate for the first time an important underlying problem. Deterministic thinking is a form of myopia. Technological positivism locates problems of technology acceptance outside the technology and its creators, creating an illusory positivistic drive through the perceived inherent superiority of the technology. But to understand problems of technology adoption and resistance one needs to accept one's own role as an actor in what is essentially a socio-technical system: you have to understand that you are fully part of the problem. Saying that it is "just a generational thing" is locating the problem well outside yourself and the technology. It is turning a blind eye to what is at least half of a complex interaction. This is what these remarks made me see: if you do not consider yourself as part of the problem as a technology creator, you rob yourself of the ability to mitigate it. Just discarding the problems the humanists had with us and our technology was not going to help us. I had to make those problems exactly my problems.

8.2.10 About Appreciating Differences

It was around this time that we engaged with IBM in exploratory negotiations about a possible cooperation between a computer science department at a university, humanities scholars from the Royal Academy, and engineers and researchers from industry (IBM). Not long before Steve Jobs had more or less said that the humanities were the next big thing (Lehrer 2011), and people were seeking what the implementation of that idea might look like. Sitting at the table at various times during this exploratory phase created an excellent opportunity to witness a gulf of misunderstanding between humanities scholars on one side of the table and computer science and artificial intelligence people on the other side. Computer science people are meticulous, mathematical people. They are interested in the computability of problems and solving the heart of the mathematical problem. While they are very clever and skilled at that, the rationale behind the subtle reasoning with very sparse information that scholars are used to is beyond them. A scholar may produce a well-wrought thirty pages of argument about a single piece of information in a text, tying clues and leads to each other making a case for a certain plausible point of view.[11] In this sense scholars are maybe not so much interested in finding definite solutions to problems, but foremost in creating multiple perspectives and speculative interpretations on "What may

[11] Hans Westgeest's article that links two Middle Dutch documents to the same author because they share a single piece of information found nowhere else (cf. Westgeest 2001:22) still stands out to me as a very good and convincing example of this approach. I take the liberty of adding another tangential note: there is proper cause to systematically designate the way many scholars construct argument as abductive reasoning. Unfortunately in the field the term is not really innate. It describes, however, quite precisely the main angle of attack scholars often use to develop a solid argument where there is hardly any evidentiary material to be found. It consists of using every possible scrap of historical evidence to construct a line of reasoning that is plausible, i.e. not necessarily provable but indeed most likely given the scant knowledge available. Using the term would create a continuum of reasoning spanning the sciences (predominantly inductive, but using other styles too), social sciences (predominantly deductive, but using other styles too), and humanities (predominantly abductive, but using other styles too). Creating a continuous understanding of styles of thinking overarching the scientific domains (cf. Kwa 2011; Crombie 1995) would mitigate the uninformed, dangerous, and damaging two cultures divide (Snow 1998[1959]) that has brought so much unproductive divisive thinking to science and society.

have happened here?" Their method is not looking for universal truths at all. Most accept that truth and fact are situated and historicized social constructs: what counts as truth is determined by Fortuna's favorites, and history is not deterministically moving in a direction of progress.

At the time I would not have been able to describe the situation as in the above because I lacked the analytical vocabulary to relate what was happening to notions such as "analytical", "rationale", "universal truth", "situated". But what I clearly sensed during the IBM episode was that the scholars and the computer science people around the table talked distinctly differently about problems. The hard-nosed scientists would address problems as things that could be divided into smaller problems that individually would have satisfiable and provable solutions. Putting together the solutions of the smaller problems necessarily would lead to either solutions of grander problems or to strategies to attack these problems. In other words: they were always solution-oriented, which seems to be a general trait of most anything computer related (cf. Morozov 2013). The scholars however, would tend to discuss what-if scenarios. They would probe and examine possible views and angles on problems, almost like wine connoisseurs figuring out if the problems were actually palatable. This would lead to a typical unsatisfactory sort of conversation. The humanists would table a specific problem or category of problems in their field and the computer science people would immediately start throwing solution-oriented strategies at it to see how solutions could be made evidentiary and empirical – maybe to the indignation of the humanities scholars who were much more interested in how different arguments with the problem as the topic could be constructed as an intellectual exercise to interpret and understand the humanistic aspect, dilemmas, moral grounds, and ethical considerations of the case in question. They were not interested in the solution-to-the-problem but in the problem-as-a-problem: in its ontological meaning and its epistemological potential; in how it might advance our different understandings of the problem.

I have Charles van den Heuvel (historian of science at the Huygens Institute for the History of the Netherlands), and Sally Wyatt (Professor of Digital Cultures at Maastricht University, who was at that time also affiliated with the Virtual Knowledge Studio) to thank for bringing to my attention some

of the essential differences that I had not grasped until then. Charles at some point talked about "ephemeral and heterogeneous data", which sounded esoteric to me. Data for me fit in categorial, discrete, or continuous variables: measures of size, distance, density, time, place, person, and so forth. It took me a while before I understood that humanities researchers did not think at all about data in only such constrained computable terms. Humanities data are both more particular and less constrained. They are more particular in that they are situated, i.e. located in place and time, bound to historical context, and possibly – actually likely – subjective. They are less constrained in that sometimes the precision of a particular data point may be less relevant, so that in some cases "ca. 409", "5th century", and "13 October 409" are equally valid.[12] It is therefore fair to say that humanists mostly do not work with data-as-data, but that they combine, examine, and interpret information, which may be defined as data-plus-context (Thaller 2018). Where physicists might be inclined to subdivide measurements into as accurate as possible times, locations, persons, temperatures etc., the humanist is often interested in a very particular heterogeneous combination of data where sheer availability may be more decisive than exactitude. Measurement of basic data itself in the humanities is tricky and difficult because it is an interpretative process, often historical data is a best guess based on information written down in a different context centuries before. But for the humanist these are even "merely" preparatory moves, after which follows the more important move of interpreting the data as a complex of information.

Without understanding this difference humanities reasoning may be taken by more hard-nosed scientists as a form of improper or imprecise empiricism, whereas it is really a matter of two fundamentally different understandings of what data and information are. Confusing one for the other will lead to misunderstanding each other's methods and aims in research. Computer scientists and engineers discarding this difference may fail to see how methods cannot simply be exported from the computer science domain into a humanities domain. Trying to do so will fail with frustration on both sides. Human-

[12] Which is computing wise a bit of a conundrum because it is unclear and highly context dependent when a measure becomes positively invalid. Does "ca. 409" mean that 400 is still included? And 550? Humanities computing therefore cannot be as straightforward as simple arithmetic.

ists are not primarily interested in stringent solutions to specific problems, and they are not especially looking for patterns that allow them to predict future outcomes. That approach is decidedly hard-sciences empiricist. Johanna Drucker (2011) has justifiably pointed out that importing such methods unamended into the humanities would mean supplanting a refined humanistic method with a naive scientism.

Sally Wyatt succinctly captured the above when in one of the IBM exploratory meetings she said: "Humanists do not solve problems, they create perspectives." That thought would later become the recurring motif in the white paper that would eventually result from the involvement with IBM and other partners (Millen et al. 2013). It would also be an eyeopener for me.

8.2.11 About Taking Textual Scholarship Seriously

Slowly thus, remark by remark, experience by experience, my perspective shifted. I began to appreciate that paradigm shifts, in the way Kuhn (Kuhn 2012[1962]) described them, and deterministic diffusion of innovation do not happen just because we want them to happen. Looking back I have also no reason to believe the scholarly editors felt anything like an epistemological crisis. But they surely felt threatened. If not by management then certainly by us, engineers, who must have come across as the leading edge of a technology that was pushed upon textual scholarship. The discussion on the side of the self proclaimed innovators was fueled by terms like "digital", "computational", "revolution", "augment", "improve", "resistance", "support vector machine", "software", "good practices", "programming", "critical mass", "new generation". Almost none of it must have signaled that we were also interested in a discourse, in a dialectic between digital and non-digital textual scholarship. We also failed to adopt sufficiently the concepts and terminology that the textual scholars used, which might have created enough trust and cooperation for what we were trying to do. If a productive epistemological trading zone had been established we should have noticed a far greater influx of scholarly terminology into our vocabulary rather than the other way round (cf. Galison 2010:39). Most of all we did not grasp sufficiently

the essential difference between a scientistic approach and a pluriform perspective view on text and textual scholarship. To solve this the dynamic had to change – as Anne Beaulieu had indicated – from technological push to mutually constructive dialectic. And both we and the textual scholars would have to become far more reflective about our own motivations, aims, methods, and techniques.

A final nudge that changed my attitude from programmatic innovator to analytical thinker was given by Paul Wouters whom Douwe Zeldenrust and I had asked about supervising PhD work that we intended to undertake based on our experiences in the Alfalab project. Paul, unsure of what I intended to work on, suggested I write a "would be" introduction to my dissertation to create some ground for mutual understanding. After reading my draft Paul smiled and put it plain and simple: "Content wise that seems fine to me. But your angle should be a bit more cogitative. A little less programmatic, a little more analytic." Sometimes eyeopeners come in really helpful obvious shapes. Basically Paul told me to try to understand more and contend less. I still had to increase my analytical stance a few notches.

So since that time, I sought to understand. Textual scholarship foremost. I returned to some basic readings in textual scholarship (e.g. Mathijsen 2003; Greetham 1994), reread works that are on the intersection of digital methods and textual scholarship (McGann 2001; Buzzetti 2002; McCarty 2005; Landow 2006, etc.). I talked to researchers in the field that did interdisciplinary work. Susan Schreibman, for instance, Fotis Jannidis, Barbara Bordalejo, Peter Robinson, Tara Andrews, Dino Buzzetti, and many more). These conversations were important to understand how those scholars understood the interaction between textual scholarship and the domain of computing. And often they also served to dot my i-s and cross my t-s. I remember, for instance, a conversation between me and Susan Schreibman over a lunch in Amsterdam where she kindly prompted me to keep a clear distinction between textual and literary criticism.

This renewed immersion in textual scholarship made it clear to me that methods of textual scholarship can only be understood as reciprocal. If the aim is to establish a text philologically or to analyze it in a literary sense, understanding and interpretation of the text are a necessity. But interpretation is

by definition something that involves an interpreter, a reader. And because the interpreter, the text, and its creator are all situated, interpretation cannot be other than intersubjective. This period also made clear to me that if I was to understand textual scholarship on any fundamental level, to be able to understand how digital techniques and methods might be useful in any fundamental way for textual scholarship, I needed to understand interpretation on a more fundamental level.

8.2.12 About Interpretation

I think it is important to note that I discussed my work on this course towards understanding textual scholarship often with my friends and colleagues in the then software engineering department of the Huygens Institute, mostly with Ronald (the same as mentioned before, who by that time had become a lead developer) and with developer and project manager Gertjan Filarski. In their projects, like me, they had experienced mostly what appeared to them as resistance from textual scholars in the institute. The tone of our conversations would be quite sarcastic, to the point of cynicism. Nevertheless they, and sometimes other software developing colleagues, due to their role and experience were formidable sparring partners in reflecting on the interaction between developers and scholars. Gertjan was later to become head of the department of digital infrastructure when several humanities institutes clustered in Amsterdam after a few more years. During the preceding years he grew into an outspoken opponent of anything to do with textual scholarship that skewed towards a pluriform or relativistic view on method or subject. We have had several disputes about what he perceived as an obsolete method – which is to say, conventional textual scholarship in general – and what I had started to regard as a different style of scientific thinking. Surprisingly then maybe, I have to thank him for turning me back to the topic of hermeneutics which I had left behind me a long time ago, around about 1996 when I had to peruse it in more general terms for my master's education in Dutch literature and linguistics. I had found it a topic or method that was hard to understand and, to be fair, a bit esoteric.

To understand interpretation one cannot evade the topic of hermeneutics, which is the theory of text interpretation and which historically goes back towards ancient text interpretation and biblical exegeses. There is a clear hermeneutic tradition from Aristotle's *Poetica* via Augustinian interpretation of religious texts, Schleiermacher and Dilthey's Romanticism, Wimsatt and Beardsley's rejection of authorial intent, postmodern philosophers like Derrida and Foucault, semioticians like Eco, and so on. Understanding this tradition as a philosophical foundation of textual scholarship became the topic of two chapters of this dissertation (i.e. chapter 2 "Screwmeneutics and Hermenumericals: the Computationality of Hermeneutics" and chapter 6 "Author, Editor, Engineer – Code & the Rewriting of Authorship in Scholarly Editing"). Tracing this tradition became a reflective argument in itself to accept the subjective nature of all interpretation. Stephen Ramsay (2011c) argues on hermeneutics that humanists always have been in the business of constructing plausible histories from subjectively-selected facts, from information much contended by authorities, educated opinions, and scant evidence. I also rooted my understanding of the tradition of hermeneutics in the traces of histories written by yet others. On comparison my interpretation seems not incommensurable with the interpretations of others, but it is still my interpretation and understanding. Or, with Korzybski, it is not the territory but a map. Thus tracing the tradition of hermeneutics resulted in new understanding about a multitude of possible perspectives on heterogeneous and non-neutral text, the situated pluriform meaning of signs, the subjectivity of any form of interpretation, the endless semiosis of semantics. Seeing how all of that connected to the methods of scholarly editing increasingly made it harder for me to view such editing as a scientific approach that could easily be turned into a uniform process.

8.2.13 About a Brave New Model

And yet, integrating through unifying is exactly what we set out to do in the institute time after time. We tried a unifying approach with eLaborate. Then we tried again with Alfalab. And after that, newly-appointed director Lex Heerma van Voss and I set out once again to harmonize the different

approaches to editing that were applied by the textual scholars in the institute. Henk Wals moved position shortly after the deliverables of the Alfalab project had been launched, becoming director of the International Institute for Social History (IISG). He left behind a legacy in the form of a new institute merged from the Huygens Institute and the Institute for History of the Netherlands (ING). In manifest obviousness the new institute had been named The Huygens Institute for the History of the Netherlands (Huygens ING). Less obvious was how the various methods of scholarly editing applied in both institutes could or should be harmonized. From a managerial perspective the merger seemed sensible. Both institutes have a strong tradition in editing and publishing historical sources. Both also maintain smaller but equally excellent analytical strands of research. The managerial challenge that Henk faced was to enlarge the analytical line of work while maintaining quality and output of the editorial strands of work.

Henk Wals has always been a strong advocate for digital methods and techniques as a way of scaling humanities research, of having new research questions emerge, and to be able to answer humanities research questions that until now seemed infeasible to address given the amount of data needed or the sheer complexity of data. As recounted above, eLaborate was to be a pivotal tool in scaling the production of scholarly editions, both by ensuring scholarly editions would be henceforth digitally born (reducing the need for the painfully slow production process of very expensive print publications and retro-digitization) and by converging the various methods of scholarly editing used in the institute towards one harmonized method "recognizable as a Huygens ING method" (Wals et al. 2012:23–24, transl. from "een herkenbare Huygens ING-aanpak").

After some ten years, continuous substantial personnel effort, and large financial investments – easily summing to over half a million Euro, though probably much more – several sites in the eLaborate platform testified to the various projects undertaken to produce digital scholarly editions with added content from analytical research on the historical texts of those editions (cf. e.g. "Published" n.d.). However, when Henk Wals moved and Lex Heerma van Voss became director, we had not succeeded in scaling the scholarly editing process. Tremendous effort had been put in digitally remediating the work of scholarly editors and textual scholars, but essentially without

changes to the scholarly method. The results were scholarly editions digitally remediated to look as much as possible as their print counterparts (cf. also chapters 3 and 4) – and most of the times editors would only grudgingly accept the non-print result. In sum this meant that the remediated editions required – in *addition* to the expertise and effort of textual scholars – the knowledge and effort of software engineers, web developers, and graphic designers to translate the scholarly work to the Web – which arguably rendered them as expensive if not more expensive than print publications. Really scaling this process to achieve a shorter turnaround and a greater efficiency for edited materials still required a new model for the process of editing.

When Henk left the institute Lex Heerma van Voss and I took this as our challenge. Henk had already come up with a model for added scientific value. He presented it as a pyramid (see figure 8.3), which likely was inspired by Ackoff's "data-to-wisdom" pyramid (Ackoff 1989). The base of the pyramid was constituted by the physical documents from heritage institutions like libraries, archives, museums, etc. that cared for them in their collections. The scholarly editors of Huygens ING became involved in the layer above that. They created scientific resources based on the documents available through the services of the layer of "GLAM" institutions (i.e. galleries, libraries, archives, and museums). It was there where the impact of eLaborate was expected to be most valuable. Finally, the top of the pyramid consisted of analytical and synthesis oriented research by the institute's leading scholars on historical topics of science, politics, literature, culture, and so forth.

Over the course of various discussions Karina, Lex, and I adapted the model by adding and refining layers, creating a rather box-like model in which layers were stacked and where each layer added its own specific scientific value. After a popular Indonesian dessert – a kind of densely layered cake – Lex called it the "spekkoek" model. The idea was that fully-enriched digital scholarly resources would emerge at the top of a stack of process layers. Each layer is connected to a specific data processing or scholarly task that builds on the results of the underlying layer.

In the case of scholarly editions one can imagine the physical archives and libraries as a first layer. Digitization is a second layer. Transcription a third.

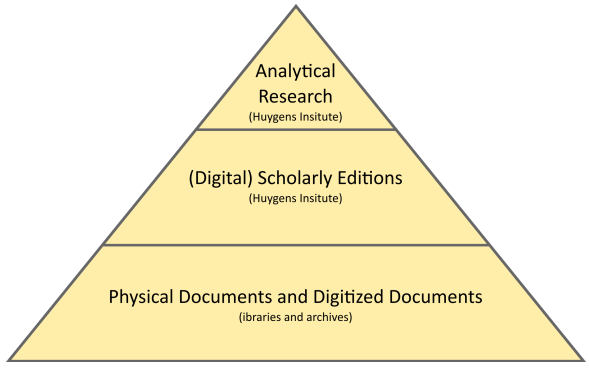

Figure 8.3: Pyramid of added scholarly value, after an idea by Henk Wals.

The key is that every layer is "thin" in that it is connected to one specific and definable task. This decouples the layers from each other so that to execute its task on a resource a layer only needs the finalized input of another layer, while it is not dependent on any information or tasks in another layer for any of its internal processing. Decoupling ensures that the process of an individual layer can be evolved to a maximum efficiency.

As a whole the model strongly resembles the Unix pipe architecture, popular in the IT world, where any process or transformation on data is independent of any other (Ritchie 1984:1581–1591). All processes function as a black box with input and output specifications only. To minimize and if possible eradicate any dependencies between layers the task in each layer needs to be as specific as possible. The model as a whole is strongly process-oriented rather than product-oriented. Thus, for instance, digitization is primarily seen as a generic serviceable task, and not so much as a specific action in the creation of a unique scholarly edition.

In this model the very first layer – atop any of the ones where we presume the physical collection work to be situated – only takes care of primary digitization, meaning that print and manuscript material are digitally photographed or scanned at high resolution and are stored in a repository with minimally sufficient technical metadata and minimally-sufficient bibliographic metadata. Technical metadata will usually already be embedded in an image file.

Although a file format like TIFF also allows for limited but in most cases sufficient inclusion of arbitrary metadata as well, bibliographic metadata will usually be stored through some other digital structure (database, document store, XML files, and so forth) for more convenient handling. Some stable URI identification scheme will allow unambiguous identification of an image and its metadata. Creating these objects is a clearly defined task with scholarly value, requiring a number of high-end technical and bibliographic skills that can be executed as an independent task (cf. e.g. Hughes 2004, in particular part 2, chapter 8). On top of this first digitization layer, a layer for primary transcription is stacked. The only input this layer requires is the output of the previous layer: a digital image and its bibliographic metadata. It is the specialized task of this next layer to create initial transcriptions of the digitized images. To do so OCR (Optical Character Recognition) can be applied, which will work for modern sources to a high degree of precision but will usually produce worse and worse results the older the historical resource is that an image is derived from.[13] The aim of this initial transcription layer is not to provide a transcription that is completely trustworthy from a scholarly perspective, but to provide quick access to the textual and pictorial content of the digital images. These initial transcriptions have no authoritative scholarly status of any kind. The rationale for this layer is that not all research needs high-end quality transcriptions, and that often even poor OCR is enough for text mining systems to create some high-level overview of what is available in a particular set of resources, or to isolate documents that might be connected to particular topics. The idea here is thus to preempt the very long turnaround of creating the highest quality scholarly transcriptions and editions that require much more time for meticulous and authorized scholarly editing. While not denying the value of such editing the initial transcription layer provides early open access to initial – likely imprecise but for some scholarly tasks sufficient – digital transcriptions of document images. Although mechanized transcriptions might not carry a scholarly authoritative stamp of quality, they do have scholarly value that is added in isolation in this particular layer.

[13] It should be noted that the technology of HTR (Handwritten Text Recognition) as applied, for instance, in Transkribus (https://transkribus.eu/Transkribus/) and crowdsourcing solutions as applied, for instance, in Vele Handen (https://velehanden.nl/) are working miracles to mitigate these limitations.

The general approach will be clear by now: the "spekkoek" model adds layer upon layer, where each layer is associated with a specific scholarly task that adds scholarly value and quality to the historical resource. On top of the raw transcription layer one can imagine a layer in the stack where rough and mechanized transcriptions would be improved by human transcribers. These might be experts knowledgeable about specific script in the case of ancient or medieval documents, in other cases transcribing could be crowd-sourced to volunteers. ELaborate was envisioned to play a major role in this layer. The layer above that would be where predominantly the scholarly editors of Huygens ING would be involved. They would correct transcriptions based on their expert paleographic, historical-linguistic and subject knowledge. In a next layer they would add annotations. And so forth. In theory the topmost layer, through a kind of bubbling-up process, should see the emergence of high-end scholarly-validated authoritative editions of historical sources.

8.2.14 About Another Failed Dialogue

Implementing this model as a solid and shared methodology in the Huygens Institute never succeeded. But it was in fact never a set aim. Sketching out the model was foremost a first attempt at making explicit, comparable, and discussable the large number of "idiolectic" methods that were rooted in formal scholarly training as much as in individual experience and tacit knowledge. We carefully refrained from initiating a discussion with a premeditated opinion that the methods in use were invalid with regard to digital media or that they were in any scholarly understanding flawed. The intention was to seek consensus for a model by inviting scholarly editors to an institute-wide discussion. Preliminary work for the discussion was to be done by a working group that would generate an inventory of editing methods in use in the institute. From this common ground a suggestion would arise about how to amend the existing methods with a digital counterpart. The goal was to find true consensus on an inclusive digitally-remediated method. Lex assigned me the task of tabling the topic of a shared digital method and I intended it to be a true remediation (Bolter and Grusin 2000:60; Karlsson and Malm 2004:13). That is: not just a medium shift but also a renegotiation of

the method with regard to the differences between the media involved. Lex would probably not put it in STS terms like this, but I am convinced that his sincere intention was also to work towards consensus. Even though we realized that some darlings of methodology might be hurt in the process, we were convinced that enough advantages and advances in the digital method would make the collateral damage bearable, even worthwhile for all scholars and scholarly editors involved.

The discussion went nowhere however. The discussion got desperately stuck in endless reiterations of the same arguments. And it appeared to be impossible to find even the simplest thing in various methodologies that looked sufficiently similar to have the scholars agree that it could be defined as a generalizable task that might be digitally remediated. In all, seven preparatory meetings were organized. But failing to find some common thread and being deadlocked in repeated argument, the initiative petered out and was eventually abandoned.

Several external reasons might be put forward to explain why this happened. No formal resources have ever been appointed to the initiative, for instance, although a methodological discussion was bolted down even in the official research program of the Huygens ING (Wals et al. 2012:23–26). There was no budget, time-wise nor funding-wise, that allowed participants to account hours spent on work for the discussion group. Participation was voluntary and time and effort spent would be silently covered by departments' budgets. It was expected that the intellectual exercise and obvious relevance of the methodological discussion for the future praxis of digital textual scholarship at the Huygens ING would be sufficient incentive for senior scholars to participate. And in fact initial participation indeed involved a group of highly visible and experienced researchers of which a large contingent also thought digital methods were something to be seriously considered and examined as a means of practice for the future work at the institute. But the informality of it all ensured that any intellectual effort for the working group became at best an afterthought for scholars that were without exception continuously being overburdened by deadlines on overly long to-do lists and being flooded by urgent interventions in whichever project they were working on.

8.2.15 About Incommensurabilities

However, even if lack of formal project space and support might have accelerated the demise of this initiative, I do not believe these were major causes. More important were intrinsic methodological incommensurabilities. It was clear from the repetitious discussion that even though all textual scholars were working on texts and documents, each and everyone's methodology only coincided on a very abstract level. For instance, Selection of sources is an important step for every scholarly editor. But it matters what the selection pertains too. It matters if you are involved with the edition of twentieth century novels or with medieval scholarship.

Marita Mathijsen explains that determining who actually wrote a particular text obviously influences whether it will become part of the editors selection or not (Mathijsen 2003; also cf. Greetham 1994 again) . But this can play out dramatically differently on a practical level. In the case of a highly successful twentieth century dead Dutch white male novelist like, for instance, Willem Fredrik Hermans, the "enormously rich personal archive" and "fascinating [...] correspondence between [the author] and his publishers, his fellow writers, his literary friends, and his enemies" provide a plethora of information about the authorial process (cf. Kegel 2016). This creates its own problems of bias and subjectivity, but it at least allows for some argument grounded in abundant paratext on what to select for a scholarly edition and what to exclude. In other, usually more historical cases even determining who wrote what is difficult. Various anonymously written manuscripts (so called "witnesses") containing a particular text must be compared to establish what should be selected and edited (Mathijsen 2003:123–124). In such cases textual scholarship may become as much bibliographic archaeology as editing the text (cf. Westgeest 2001 for a particular interesting case).

Thus what looks commensurate on an abstract level, i.e. selection, means vastly different scholarly processes on a practical level. More importantly the historical case violates the assumption about neatly separable scholarly tasks. The genealogical provenance work requires highly skilled and specialized knowledge of both the historical text and genealogical methods – consider stemmatology for instance – which is only present in a handful of experts. But in the layer model the resulting knowledge (cf. Westgeest 2001

again) also has consequences for metadata in other layers, e.g. the layer of collection formation and the layer of digital imaging. This does not necessarily invalidate the model itself, but does show its limited encapsulation of the actual praxis and dynamics of textual scholarship..

Thus even if two scholars agree that what both do at some point in the editorial process can be called, for instance, "transcription", the actual practice of "transcribing" may be two rather incommensurable activities: the one being executed in some archive with pen and paper next to a box full of scraps of documents, the other being carried out behind two computer screens, one with a word processor open and the other depicting a digitized microfilm. The very nature and properties of the documents and the texts that they contain differentiate what "transcription" means. For one scholar it means knowing Middle Dutch grammar, dialectic variation, and lexicography by heart, and being an expert paleographer of medieval Dutch manuscripts. For another it means developing intimate knowledge of the handwriting in the diaries and letters of a modern author, and amassing sufficient contemporary contextual knowledge on topics that the author wrote about, to be able in the end to translate idiolect phrasings written in a peculiar, hermetic, personal stenographic system.

8.2.16 About a Digital Tool Breaking Down

A series of discussion meetings can hardly be called a key moment. Key process seems more suitable as a label. This process in any case made me experience something I had never witnessed: failure of abstraction.

Abstraction is one of software engineering's most powerful tools. It is a method of modeling by which many particulars can be molded from an abstracted super class. Imagine, for instance, one is developing a drawing program that can create circles, rectangles, and triangles. One can construct three distinct objects, each of these having lines of code to position the specific shape, to draw it, scale, fill, store, delete it, and so forth. But many of these actions are not specific for either shape. This is most easy to see for e.g. positioning, deleting, and storing: the information whether we are talking about a circle, a rectangle, or a triangle is irrelevant to the position of the

centre point, the deleting of the object's digital information, or the storage thereof on a hard disk. From an engineering point of view it is therefore convenient and efficient to create a super class, e.g. one called "Shape", that has functions with specific lines of code for positioning, deleting, and storing. All specific shapes will then be created as "offspring" objects from this super class, and they will inherit the functions for deletion, positioning etc. Only particulars unique to the specific shape then need to be expressed as lines of code in the object associated with the specific shape. Thus Shape-Triangle will have different code in a function called "draw" than Shape-Rectangle, but they will share the exact same lines of code for the function "delete". One can dispute endlessly whether abstraction is fundamentally an inductive or deductive approach. Most likely an inductive or deductive propensity is dependent on individual choices by programmer. When coding myself I tend to think of my theorizing about the world as deduction and about my practice of coding as induction.

The salient point of this with regard to the methodological discussion on textual scholarship was that this method failed utterly when trying to create abstractions for the activities of textual scholars. It is rather straightforward to define commonly-understood abstract tasks in textual scholarship: selection of sources, reading sources, transcription, annotation, and so forth. However, once one delves below that level one is immediately confronted with a garden of forking paths that is different for every combination of scholar and source, as the example of transcription above illustrated.

Failure of a tool is an excellent opportunity for learning (McCarty 2005:41–43). This failure of abstraction and the process of experiencing it up close in these meetings was key in understanding how hermeneutic the practice of textual scholars actually is. It is hermeneutics to its deepest core, it finest veins, its smallest act. If hermeneutics was turtles, it would be turtles all the way down. The scholar that develops an intimate knowledge of the handwriting in the diaries and letters of a modern author, does so not just to be able to read the handwriting for transcription. Through the very experience of learning to read and then reading the manuscript the scholar develops a sense for the author's style of writing, and for the content, topics, events recounted in the text, and so forth. The point of which is not just to be able to create the transcription. The reading experience also create affordances

for later annotation of the text and for analytical reasoning on the author's developing thoughts.

8.2.17 About the Integrated Nature of Textual Scholarship

The toughest argument against supporting hermeneutics digitally is thus hermeneutics itself: it requires intimate engagement with a text or document in all possible aspects. It requires experiencing its very materiality, the feel for how the sign on the page was formed, how words were strung together. It requires an abundance of contextual knowledge, annotating without end, reading and rereading. Only this experience results in a foundational knowledge of all the aspects of a text from which a viable process of interpretation can start. The back and forth between conceptually charting the meaning of a text on a high level and the minute inspection of a comma that could change that meaning: this is the hermeneutic method.

It is exactly this intimate engagement with all aspects of a text as a whole that one forgoes if one "topples" this process and tries to pull it apart into a series of different tasks, each embedded in some isolated technical layer attributed with a number of dedicated skills. The decoupling of scholarly tasks that would be required to realize the digital scaling of the academic work would also eradicate the hermeneutic aspect of that work. With hermeneutics you cannot have your cake and eat it. In managerial terms the hermeneutic process is an integrated vertical process. It requires from a task owner a specific skillset that is dictated by a specific resource. The process generates new knowledge about a text by progressing through each layer of the "spekkoek" model. This process is not straightforward progressive but iterates many times between layers (tasks). Eventually the process yields a sufficient reservoir of knowledge to produce a scholarly-viable interpretation of a text.

The task-oriented horizontal model assumes that scholarly knowledge can be added independently in each layer. But I found that this is only true to a very limited extent. The first layer of digitization requires technical skills, bibliographic knowledge, and possibly scholarly knowledge on book history and paleography. This might be done in an isolated layer. Also the next

level, OCR'ing the text, might still be done in isolation. However, progressing one more level up towards transcription, the correcting of the OCR'd pages requires expert scholarly knowledge of the language, script, and probably also content of the text. This kind of scholarly expertise is not widely available, and it is rather likely that the paleographic knowledge needed in the primary digitization layer for a specific document is tacit knowledge of a person that will also be needed for the transcription in the third layer. Moving upward through the layered model increasingly more knowledge of the underlying layers is needed to create valid interpretations of the text at hand. An argument might be made that once a corrected text is available – a so-called diplomatic transcript – annotation could be done independently of the creation of the diplomatic text. However, in practice the creation of the diplomatic transcript is part of a process that generates a large amount of knowledge about what could and should be annotated to ensure a sufficient understanding of the text. Vice versa the questions that result from trying to transcribe the text drives contextualizing research that overlaps considerably with the annotation task. Moreover, even if the creation of a so called critical text might be completely decoupled from the basic transcription, the work in this layer is still strongly coupled to a higher-level interpretation of the text, because the annotations are the evidentiary material that a textual scholar gathers to produce a scholarly-viable interpretation of the text.

Thus hermeneutics and the praxis of textual scholarship are deeply intertwined. That is: every scholarly task related to a certain document and a certain text is a contribution to the practical part of hermeneutics, which is interpretation. This praxis – and thus hermeneutics – is very much embodied in textual scholars with years of practice and with actual work on a specific historical resource. The uniqueness and heterogeneity of historical documentary and textual sources combines with a uniquely-experienced body enacting hermeneutics that together result in a unique method applied to edit a unique text. Although that method can be understood in the abstract and general terms of scholarly tasks, these tasks are not neatly progressive in time nor is their actualization identical each time when the tasks are enacted. Rather they are imbricated, and they iterate and adapt continuously until such time as an edition can be said to be "finished".

This intertwined and highly resource-attuned nature of hermeneutics is, I

claim, the direct cause of the futility of our attempt to remodel the editing process as a stack of unproblematically unrelated layers associated with individual scholarly tasks. For a long time we sought the cause with the textual scholars – perceiving their attitude as resistance to change. The discussion meetings were in part a result of the Huygens Institute merging with the Institute for the History of the Netherlands. The Huygens Institute had a strong history in text criticism and literary editing, while the Institute for the History of the Netherlands had its roots in a tradition of documentary editing. The discussion group was therefore also aimed at harmonizing these different approaches. It appeared from the ongoing discussions however, that the scholars were dead set on pointing out the incommensurability of the two, as it seemed that either species of textual scholar was only able to talk about the differences and specificity of their scholarly practices. The merger of the institutes thus caused the discussion group to be about a threefold methodological merger: the merger of two different approaches to scholarly editing and the merger of those with a digital approach to editing. For a long time this reinforced my thinking that the complexity of the process and in part a resistance from the scholars to digital techniques in general were to blame for the lack of progress in the discussion and the impossibility to harmonize methods.

From a higher-level managerial perspective it is indeed hard to see where the differences are between documentary editing and literary editing or textual criticism. All editors themselves agree that there are similar tasks in all scholarly editing processes. How then can it be so hard to harmonize these task a little? The differences are apparent in the individual praxis of textual scholars, but they are hard to spot for those who seek harmonization and efficiency on an organizational level where only the labels of the abstract tasks are visible. Thus what we failed to see is that each scholarly editing project below the level of abstract tasks indeed explodes in a melee of specific subtasks that are highly attuned to the material at hand.

8.2.18 About the Real Problem

Suppose however that things had been different. Suppose that the discussion group at the level of merging two distinct traditions of scholarly editing would have found complete agreement and harmony – supposedly because the traditions were somehow indeed commensurable. Would we in that case have been able to successfully digitally remediate this methodology according to a model of layered and independent tasks? Again with the power of hindsight: no, we would not have been able to do so.

Recall that management had two goals in mind when it suggested to create a "digital edition machine". The first was scaling up the production of scholarly editions, or at the very least minimizing the turnaround for (digitized) scholarly resources becoming available for further study by researchers other than the editor. At the same time it wanted scholarly editions to become more accessible. "Accessible" in this case could mean two things. Either editions from the institute would need to be more affordable than the forbiddingly expensive print editions – that were so expensive due to the scholarly effort needed, which combined with the low number of copies printed and sold, as they were often targeting small numbers of specialists. Or it would mean making editions digitally accessible in open access. Digital scholarly editions were seen as technology that would actually serve both these purposes: affordable and accessible editions.

Digital technology is excellent at scaling. Once a suitable model has been found for the data or a process it is a formidable tool to reproduce the modeled task or product in staggering quantities with dazzling speed. It can do so where either data or process are uniform or generalizable enough so that it can repeat identical operations on similar data under the same constraints. The trouble with textual scholarship is that this never happens, neither at the level of the word, nor at the level of the text. The hermeneutics of textual scholarship is deeply intertwined with the heterogeneity of its data. Determining what some word in a text means is dependent on the language of the word, the knowledge of the linguistic context of that word, the cultural context of the text, and often also on what is on the page: could these written or typed glyphs actually signal another word, maybe? Reading, transcription, and interpretation are thus interdependent tasks. If one was nevertheless to

isolate each task in a layer, assuming it would therefore somehow become scalable, you would find that the task would need a different combination of language, reading, and cultural expertise each time. In such a process there is nothing to be abstracted, nothing to be generalized. Would one nevertheless take to abstraction, then one would eradicate the hermeneutic aspect of the process, or in other words the point of the scholarship involved (i.e. interpretation). A real risk, had we pushed forward, would have been that the constraints of the chosen technology would have become leading. Because the technology can only scale a uniform process based on uniform patterns in data, the actual scholarly process or the scholarly data gets adapted in such ways that the technology will be able to scale it. This is a particularly pernicious corollary of the promise of scale that is characteristic of software engineering – the technology does not scale the actual process or data of scholarship, but an abstracted form that is tractable for the technology.

The conclusion must be that current software engineering approaches and current general-purpose computer languages are not adequate to scale scholarly editing without losing some of its hermeneutic essence. What is possible is supporting scholarly editing in a computer-supported collaborative work fashion, which is by and large as far as eLaborate got. It facilitates a rather convenient way to produce digital-born scholarly editions by shortcutting the tedious task of marking up a digital transcription in the form of TEI-XML while keeping the ability to annotate text. But eLaborate does not scale the core intellectual and skilled process of scholarly editing. It merely enforces an abstracting away from the more precise hermeneutic tool that TEI-XML might be. By doing so it might shorten the turnaround of a digital scholarly edition becoming available online, though I think this has not been convincingly shown in practice. But even if it did, it did so mostly by enforcing a particular uniformity of look and feel of the editions. A uniform typography that lessened the ability of scholarly editors to put the look and feel of their editions in service of a scholarly grounded interpretation of the text (cf. Andrews and Van Zundert 2018).

Many, myself included, have claimed that the key to battling this problem is in increasing the digital literacy of textual scholars (cf. e.g. Ramsay 2011a, Smith 2012, Van Zundert 2018). Chapters 5 and 6 of this book also speak to this idea. That is: teach scholars to code so that GUI based solutions like

eLaborate that necessarily enforce a certain uniform model of scholarly editing and edition onto the scholar are no longer needed. Instead textual scholars, empowered to wield general-purpose computer languages, would in that case code and publish their own digital scholarly editions without the intervention of GUI based tools. And although I remain a strong advocate for the idea that all humanities scholars should at least acquire fluency in one general-purpose computer language, I do not think this would fundamentally solve the problem. Because, as Annamaria Carusi pointed out so hyperbolically, current general-purpose computer languages are almost without exception so-called first-order languages. Although it is rather crudely put and not entirely correct, it is also not an exaggeration to say that such languages only support boolean (binary) reasoning. At the very least their make-up invites a developer to tackle any problem as a boolean operation, even if it is quite possible to implement more subtle forms of reasoning with general-purpose computer languages.

A rhetoric of scale and speed is prevalent in mainstream software engineering which is driven by a plain vanilla type of application of first-order computer language in combination with a logic of process automation. This is a type of software engineering that is rooted in a limited binary apprehension of the capabilities of first-order logic for data modeling and analysis, where processes and data typically are modeled after the question of whether stringent, i.e. boolean, comparisons can be made. The basic rhetoric structure of this type of coding is "if X equals Y then Z will be done", where "equals" is defined in absolute terms: either the condition is fully met or completely violated, e.g. "if year is larger than 1066 then label 'post Norman Conquest' ". This discrete logic breaks down on hermeneutic requirements. An inference such as "if genre is Bildungsroman then label 'literary' " is not supported because humanistic categories are rarely absolute but rather more often have unclear, intersubjective, and overlapping boundaries.

It is the boolean nature of general-purpose computer languages that causes uninspired software to be rather demanding with respect to the uniformity of data and process if it is to scale data transformation or analysis, because in its most basic form boolean logic and process automation do not deal well with imprecision, uncertainty, heterogeneity of data, and ambiguity of information. The uniforming nature of current general-purpose computer lan-

guages proliferates through much of the reasoning supported or expressed by it, simply because it is de facto the easiest way to program reasoning in these languages, and it is much harder to do anything else. Arguably the majority of industrially-produced software falls within this category because mass production favors uniform processes and products. Such industry software is meanwhile a dominant shaping power in society (cf. e.g. Berry 2014) and might well be seen as a pernicious form of Foucault's disciplining powers (Foucault 1995[1975]).

There are actually ways to circumvent an uninspired application of computer languages, even with the currently most-used general-purpose programming languages (such as Java, Python, Ruby, C#, JavaScript, etc.). There are, for instance, mathematical approaches to deal with uncertainty in the form of probabilistic models (e.g. Bayesian models and networks). These models, well known in the field of natural language processing (Cohen 2016), could well be applied to more hermeneutic reasoning regarding historical textual resources. Knowledge at the fronts and fringes of mathematics and logic bare promise of forms of logic that are better equipped to deal with incomplete, imprecise, and ambiguous information. Modal logic (Pratt 1980; Mastop 2011), possible worlds models (Fagin et al. 1995), and grey information theory (Lin, Chen, and Liu 2004) have all forwarded possible solutions to deal with data and information in ways that relate far closer to hermeneutics than first-order logic proper. All of these, however, remain in the realm of speculative research and none have produced generally available usable software libraries yet. But these advances do show that current programming paradigms need not be the be all and end all of the application of computation in textual scholarship, as has been noted by some scholars (Thaller 2018; Van Zundert 2018).

As for eLaborate and the layered model… We were trying to model and scale the scholarly process of applying hermeneutics, but we did so by computationally providing a digital mimesis of a few rather mundane scholarly tasks and concepts – such as text, transcription, and annotation. By doing so we were enacting a mere feat of cargo cult: we mimicked a few externally visual traces of the hermeneutic work of which the essence is rather a cognitive process in the bodies of textual scholars. Given the severe technical limitations of our computational tools, we would not even come close to representing

or modeling human interpretation. One could object that this was never the goal, and that realizing CSCW was indeed achieved. That is a fallacy: the very intention was to scale scholarly textual editing. We – all those involved – never truly realized that this implied scaling the cognitive process of hermeneutics. Attaining that implied objective was futile from the start, given the sticks and rocks we have for tools.

If we are indeed to strive for a computational hermeneutics that will require a far more intimate and deeply involved engagement with quite fundamental forms of computing, logic, and mathematics, both on the part of computer science mutatis mutandis software engineering and on the part of textual scholarship. It will require a sincere shift of minds within these two interacting domains, and the courage to do so mutually and respectfully. Engineers should dare to admit that current software tools – and even the computer languages in which they are written(!) – are severely limited with regard to modeling and reasoning about complex intersubjective information, and subsequently that true computational hermeneutics requires vastly improved computational tools. Textual scholars on the other hand need to start seeing text as an interesting and imprecise form of computational data that yet should be able to be modeled as digital and mathematical information, and that may be edited in different ways than just as a remediation of print text on a digital screen (cf. again Thaller 2018; Van Zundert 2018, as well as chapter 4 of this dissertation). In all of this eLaborate was a failed attempt, doomed from its very beginnings. But failed attempts are part of a path to achievement. Regarding the multidimensional modeling of text the work by e.g. Ronald Dekker, Elli Bleeker, Bram Buitendijk, Astrid Kulsdom, and David Birnbaum (Haentjens Dekker et al. 2018) as well as others are meaningful steps on this journey. But there is a long way ahead of us. Even more ambitious computer science research will still be needed to render hermeneutics somewhat computable.

8.3 Conclusion

I came into textual scholarship with a rather hard-nosed "can do" mentality, reasoning that re-establishing textual scholarship on a digital and computa-

tional footing should not be much of a bother. This I found after so many years of experience with textual scholarship: the scientific value of textual scholarship is in its hermeneutic analysis of texts and this hermeneutics cannot be adequately expressed or enacted by the current digital technology we call code.

Three key remarks, I feel, remain to be noted. Not because they contributed that much to my findings per se, but because they provided independent confirmation of those findings coming from non-suspect sources. The first is a remark made by Carl Posy when I attended a dinner he organized at his house for the people that had lectured at a workshop at the Hebrew University of Jerusalem. (Consult Posy 2013 for an example of his work on the philosophy of mathematics.) We were discussing various topics surrounding digital humanities and textual analysis, and at some point Carl remarked: "It is not so much about computation as it is about computability." This remark signified for me that there is good reason to relate the domain of scholarship to computation. In the same sense that Anne Beaulieu wanted to examine the rather unquestioned dynamic between software engineering and computer science on the one hand and humanities on the other, Carl Posy invited me to consider that maybe the humanities hold rather more interesting challenges for computer science than just as an application ground for what computer science can do today. The questions that scholarship can put to computer science are hard and much more interesting than scholars tend to think: they highlight the still limited abilities to reason computationally about humanistic information.

The second one is related to reading Willard McCarty's "Humanities Computing", where he argues that the moment we learn something is the moment our digital technology fails and errors out (McCarty 2005:41–43). This wisdom goes back to Heidegger obviously (cf. Heidegger 2010:334–348 [1953]) and was not essentially new to me. However, it did put me in a state of mind where I started to consider computer languages as a failure instead of as a solution, certainly with respect to textual scholarship. And this proved a more fruitful and productive line of thought than simply regarding computation, computer languages, and digital data as an answer to anything.

The third one is the most recent and came in the form of a blog post by Manfred Thaller (2018). Thaller lucidly describes a number of problems that render current mainstream computer languages inept for any hermeneutic purpose, and that the challenge is to come up with a solution for this.

These three remarks combine into what I find a formidable challenge. If it can be argued, as I did here, that current mainstream computer languages do not support hermeneutics because they lack a hermeneutic nature, then why do we allow that to result in a push on textual scholarship to comply with the uniform nature of first-order logic? Would it not be far more interesting and challenging, in a fully interdisciplinary understanding, to devise an analytical tool in the form of a computer language that is actually hermeneutic in nature from the bottom up? (Cf. also Van Zundert 2018.) It is not impossible, and it is certainly less boring than transforming the analysis of the human textual record into something uniformly bland.

A few loose ends remain to be tied up. What does this all mean, and how analytical is this chapter? To start with the latter: is this writing just some opinionated essay, or is it genuine research? It is analytical in the sense that I kept to Anderson's principles for analytical autoethnography (Anderson 2006) as outlined above. I am a complete member of the social world being studied. If anything this autoethnography is a testament to the deeper-level reflectivity that Anderson points to: I recorded to the best of my abilities the influence I had on methodologcial development in textual scholarship in my academic context and the reciprocal effects of that work on my thinking. I think that qualifies as "self-conscious introspection guided by a desire to better understand both self and others through examining one's actions and perceptions in reference to and dialogue with those of others." Anderson's third requirement is that an autoethnographer should be visible in his ethnography, and this requirement is why at times this chapter takes on a colloquial style. Having a colloquial style seems to be a stock criticism of peer reviewers of many of my articles anyway. I agree: sometimes I do use a colloquial tone. But it is not because I do not know how to wield the so-called distanced and disinterested academic style. My colloquialism is on purpose, especially in this autoethnography: I do not want to give the illusion that my argument is more widely applicable than it is by applying a misplaced academic generalizing style of writing. In this case all I have as evidence is

my personal experience and notes – subjective material by definition, but nevertheless valuable to learn from. I have tried to give voice to the insights of others, by focusing on key remarks, and I clearly demonstrated how my beliefs and insights changed through the perspectives and words of others. I maintained an analytical stance by supporting my narrative of experience with ample references from the literature. I also kept an analytical attitude by taking distance from this material in time, by returning to it and rewriting it several times over a period of two years. I witnessed how all versions became indeed less self-absorbed, less indulgent, more honest, and more interested in seeking some truth. The text stopped being about where I wanted to go at some point in time. Instead it became to reflect my genuine interest in where my topic came from and how I got to think about it the way I do now.

The main conclusion I finally draw from all this is that technological determinism blinds and that curing that blindness is pivotal if we are to see the real methodological challenge that lies at the intersection of textual scholarship and computer science. In the case I describe there was a strong belief in innovators in academia that some technology was methodologically better than the methods and techniques that existed. This believe was conveniently in accord with certain managerial needs. I bracket the managerial needs as they are not that interesting in my opinion and largely irrelevant to the main conclusion of my argument. It is this particular blindness of highly-trained, skilled, and clever professional researchers that intrigues me. We had not much more going for the innovations we argued than that the technology was new and digital. It promised scale and speed, but we had no evidence in any particular academic use case to back that promise up. This attitude could be called mild technological myopia still. In the end, confronted with a stubborn hermeneutic methodological tradition, both textual scholars and engineers learned many valuable lessons I think, even if it meant not achieving certain hoped for innovations. This mild myopia is not something I am all too worried about. It creates enough friction of its own in pipedream projects to make forced interdisciplinary projects to implode, be it sometimes at staggering economical and human labor costs. The blindness I truly worry about lies in an unwillingness to investigate respectfully and with sincere interest another person's ideas and perspective. It was only in eventually digging myself a way into "the textual scholar's mind" that I, as

an engineer, could start to see the exact properties of a mismatch in methodology. Seeing that, I could acknowledge the invalidity and pretense of an easy process-oriented digital reshaping of scholarly editing. And it opened up an intellectual space that allowed me to recognize the – in my opinion – far more interesting challenge that textual scholarship can table for computer science: developing a formal logic and language that fits hermeneutics rather than shoehorning hermeneutics into current first-order logic languages and thereby losing it all.

One last concern I want to address. I have meanwhile moved my perspective so far to the philosophical side of the spectrum that I am at risk of blaming the engineers and computer scientists for said blindness. But I should stress that I found scholars at least as blind, albeit in a slightly different way. There is anxiety among scholars that they are not being taken seriously by software engineers. Many scholars feel intimidated by shiny and hip information technology. But this intimidation, aggravated by a false modesty about the value of some twenty centuries of scholarship methodology, is inhibiting in detrimental ways. There is a shared task here and no easy sitting back. Textual scholars should raise their voices to ensure that the process of softwarization of text technology progresses responsibly from the perspective of scholarship. In recounting my experiences, it sounds to me like the voice of the textual scholars is heard but that it is somehow also not very present. In chapter 6 of this dissertation I point to the shared accountability and responsibility that both scholars and technology innovators have in methodological innovation. However, it appears to me that often the scholars are kept out of the wind in ethnographic work in DH. Antonijević' work, for instance, at places reads rather apologetically to me where it pertains to the role of scholars in methodological innovation (Antonijević 2015). But if engineers indeed have done little to open their minds to the core principles that underpin hermeneutics, so have the scholars – certainly in my case. They have been utterly unable to voice their methodology and concerns in ways that convey the relevant properties to engineers comprehensibly – often because the underlying assumption is that engineers will not or will not want to understand. Textual scholars should realize that their aims and work bear formidable challenge for computer science engineers, but that it is also their responsibility to develop and expose these ideas in ways that make the

challenge clear. Putting that responsibility in the hands of engineers alone is capitulating to a technological deterministic mindset that will render textual scholarship irrelevant when more and more text becomes part of a digital medium. The proper form of digital hermeneutics requires scholars to care about computational literacy and to invest in learning what computational literacy really might do for them. The best way forward is a dialogue, not a monologue as some suggest (Bordalejo 2018; Robinson 2013c).