The Processing of Manuscripts

Manfred Thaller

Summary

We are afraid, that this contribution, unlike the others in this volume, may seem to be unduly abstract — or overly technical. Therefore, we would like to summarize our argumentation first, to show more clearly, how the highly technical considerations at the end are the consequence of a substantial, historical argument.

1) Techniques for the processing of digitalized manuscripts exist, which seem to be promising.

2) Experiences have shown, that — unlike in the processing of images in history of art — there is not very much sense in an extended discussion about the quality required: there exist quite clearcut criteria for the quality which is needed in the processing of text. And these criteria call for the processing of fairly high quality images for dealing with manuscripts.

3) Irrespective of large scale projects — like the one at the *Archivo General de Indias* — the application of such techniques to typical medium size collections, like the ones contained in city archives, are financially feasible today.

4) To make the most of such collections of manuscripts within historical research and teaching, a workstation with a bundle of processing techniques is needed, which are, however, not automatically provided by modern image processing packages.

5) To develop such collections of manuscripts from a purely preservational vehicle and a tool for the study of palaeographic properties further into a genuine replacement of previous editorial forms, a much closer connection between the bitmapped image of the manuscript and the transcribed ASCII text would be needed.

1. The problem

Images have in recent years drawn much attention in historical research: their importance for a variety of subdisciplines in the field of cultural and social history has been emphasized quite frequently. Still, while their importance as historical source can scarcely be overestimated, traditional, written sources will never be superseded by them — and historical interpretation, as we know it, is and will be based, first of all, upon the content of written documents, which, for most historical periods, have survived as manuscripts.

Manuscripts, in many respects, share important qualities of images, when data processing is concerned. Many manuscripts have survived in handwriting, which has to be interpreted, rather than simply transliterated character by clearly recognizable character, when their contents shall be converted into a machine readable dataset. Their can be various reasons for that: be it, that they are heavily abbreviated, with abbreviations which are all but clear for many areas, particularly of administrative writing, be it simply that the handwriting is hardly legible — either because written by a calligraphically dissatisfactory person or because of damages having occured over the years.

Handwriting furthermore does not allow a clear separation of content and presentation: in many areas of historical research, particularly medieval diplomatic studies, the interpretation of the form of the letters, as indications, by which author they have been

written, is just as important as the text these letters transmit to our interpretative abilities. The form a letter has in print is usually irrelevant: in older manuscripts most oftenly it is not.

It is scarcely astonishing, therefore, that the advent of image processing techniques, as described in the first contribution of this volume, has very soon lead to their application to manuscript material, focusing most oftenly upon the intensive study of a relatively small amount of material, in some cases also upon the systematic conversion of extremely large corpora of texts. This treatment of manuscript sources by techniques drawn from image processing[1], we will call *manuscript processing* in this paper; a concept which we will develop further, however, to include a number of techniques which are not usually connected with image processing.

What are the reasons why we would like to apply such techniques? The following are most frequently quoted:

- Manuscript material can on the screen be represented in a quality, which is not worse than that of very good photographic reproductions. Indeed, with the exception of very few extremely expensive facsimile reproductions, digital reproductions of manuscript material will usually be better than photographic ones. These representations can not be damaged by careless handling. One of the first purposes of manuscript processing is therefore connected with the preservation of material.
- Digital representations lend themselves easily to the application of various techniques of image enhancement. By appropriate techniques it is relatively easily possible to make manuscripts on the computer screen better readable than in the original. The second reason to start with image processing is therefore the wish to improve the legibility of damaged or from the beginning hardly readable portions of manuscripts.
- Palaeographic research, dealing with the properties of old handwriting, implies a very intensive handling of individual characters, for their systematic comparison accross documents.

2. Experiences and Practical Background

This paper is not a project report; it shall, on the contrary, try to discuss the general possibilities of "manuscript processing" as a field, where we expect great advances in the next few years: nevertheless it has not been written without practical experiences and some of the recommendations implicit in the following pages can be understood only in the context of these experiences. Before proceeding further, we would therefore like to describe the practical background of the author and state explicitly some assumptions drawn from practical work, which underly all the remainder.

All experiences gained are derived from an attempt to implement a general data type "image" into the DBMS *κλειω* which has been developed by the author since quite a number of years now. The experiences from the early stages can be summarized as follows:

[1] We will not repeat any technical details here: a good description of the basic technology, together with examples of early applications, is contained in: Kevin S. Kiernan, "Digital Image Processing and the *Beowulf* Manuscript", *Literary & Linguistic Computing*, 6/1, 1991, pp. 20-27.

Immediate Image Retrieval. This describes the most basic methodology that is, the ability of a software system to display as the result of a query not just the description of an image, but the image itself. To do this within a research environment, a number of practical considerations apply:

- A hierarchical storage administration[2] should be mandatory. This means, that each of the images out of the whole set of 20.000 or 100.000 administered ones can be displayed in the form of a small snapshot without noticeable delay. This is due to tbe fact, that such small snapshots are being kept on the fastest medium within a storage hierarchy, while the bulk of the image data resides on slower media in the background, access to them possibly even implying the manual mounting of discrete storage media, like magnetooptical cartridges.
- *Each* detail of an image has to be available for zooming with an arbitrary size of the zooming step. This is methodologically extremely important. If you provide images and the possibility to look at only four or five predefined details, the editor of a CD-ROM based system considered to be the most important, the editor is still the only one who controls what you are allowed to be interested in within an image. This does not change dramatically the way of your approach to the images, as compared to a printed book. Only when the user has the possibility to control, what details shall be zoomed — preferably zooming with gain of information — access to the material is better than in the printed solution.
- Of course the usual cutting, pasting, mirroring and similar tools are also necessary within the historical working process.

Image Enhancement. Within the experimental system about 30 statistical operations for transforming and filtering image data have been implemented[3]. These methods — as well as the usual false color techniques — can be applied to any image within the database and or/to any segment thereof. The application of these techniques within historical research has usually one or more of the following goals:

- Improving the readability of portions of manuscripts, which became unreadable, because either the writing itself or the material upon whicb it has been written has changed color, resulting in a reduction of contrast.
- The processing of documents, parts of which have been damaged by mould, humidity or similar reasons.
- The processing of items (inscriptions, coins, etc.) where letters which were cut into the surface or are higher than it were partially destroyed by damage to the object; under some circumstances such material can be partially restored.

[2] In this context it has been implemented for the first time, to the best of our knowledge, within a joint project of IBM Japan and the National Museum of Ethnology at Osaka. See Jung-Kook Hong and Sigeharu Sugita: *A Color Image Database for an Ethnology Museum*, in: Heinrich Best et al. (Edd.): *Computers in the Humanities and Social Sciences*, K.G. Saur, 1991, 53-60.

[3] These techniques are currently realized with the help of an image processing library, *Image Assistant*, produced by IBM. At this moment another implementation, using DECImage, a similar package by Digital Equipment, is checking whether our concern for portability has been consistent enough.

Image Binding. Many authors would describe this concept by hinting at "Hypertext" or some of the other "Hyper" concepts. We would like to avoid this, as a matter of intellectual strictness. In data processing there is the concept of nonlinear representations of text: an example of those makes up the bulk of this paper, following below. "Hypertext" is a phrase coined by Theodor Nelson, for a very specific and consistent model of a textual data type defined on the basis of specific tools out of the general realm of nonlinear — or nonsequential — texts[4]. This model has so far never been fully implemented. Application systems like Hypercard[5] are no implementations of Hypertext, but systems to administer subsets of the general concept of nonlinear structures.

We therefore prefer to give a more precise definition of our approach, than just a global — and wrong — reference to Hypertext. "Bound Images" we call the administration of bit mapped data objects, which are nonlinearily related themselves, can at the same time also be parts of an arbitrarily complex network of transcribed information, however. This is done in a way, where each portion of the transcription or description in text form, is explicitly related to an area within the bit mapped object. For the sake of completeness we add, that such areas in our implementation may overlap.

While originally geared to the administration of collections of historical pictorial sources, these tools have been applied in a number of cases to projects involving the handling of manuscript material, including two attempts at teaching such applications in university contexts. These experiments with teaching produced in some ways the most interesting practical illustrations for the requirements which actually have to be fulfilled, before such techniques can be integrated into daily research.

On the one hand, a group of students at the university of Bremen tried to use a digital representation of parts of a prominent source of that city to learn the reading and handling of medieval manuscripts in practical work. At the beginning of the project it was assumed, that the working group would spend some time to choose the most economical way of scanning the source in a resolution which would be most appropriate for the hardware available and employ later the 30 or so techniques for enhancing images offered at the time for improving the legibility of the manuscripts before transcribing them. The equipment available was definitely insufficient: the scanner used has, e.g., been available only for a couple of hours on a run after which a fairly complex process to transport the digital images from a NeXT computer to a PS/2 80, running under the IBM UNIX implementation AIX had to be started, involving the transfer of image files of about 5 — 6 Mbyte via diskette between these computers. The PS/2 80, though equipped with 12 Mbyte memory, was quite slow when handling synchroneously more than two images of more than 2 or 3 Mbyte.

[4] For resons which have to do with the very peculiar funding of this project, it is somewhat difficult to give good bibliographic references. Theodor H. Nelson: *Literary Machines* has since 1981 been published in various versions, usually by the author himself. A good summary, which is also easily available: Theodor H. Nelson: *Managing Immense Storage*, in: Byte 13/1, 1988, 225-238; on the concept see also Janet Fiderio: *A Grand Vision*, in: Byte 13/10, 1988, 237-244; see particularly 238.

[5] This difference is quite oftenly ignored in the HyperCard literature: a particularly bad example in Carol Kaehler: *HyperCard Power*, Addison-Wesley, 1988, 366-367.

Indeed, most of the time of the participants was spent with the overcoming of fairly trivial hardware restrictions.

When it comes to the usefulness of image enhancement, this seminar proved inconclusive at best: the whole process was so cumbersome, that such techniques were basically never really tested. Still, the participants, even under this conditions, found the exercise useful and rewarding. The reason: simply scanning a manuscript difficult to read at 400 dpi and displaying it on a 1280 x 1024 pixel screen produced an implicit magnification, which made the manuscript easier to read on the screen, long before any attempt at image enhancement took place. As a result of this, though restricting themselves to lower resolutions would have been speeding up the processing, the usage of lower resolutions was never considered a serious option by the participants of this course.

The second teaching exercise of that kind consisted in a seminar offered by the author at the university of Göttingen. The participants[6] were given a basic "flashy demo" type introduction to the principal techniques of image processing (immediate image retrieval, image enhancement, pattern recognition / comparison plus possibilities of hypertext-like linking of images and connected or underlying texts). After a very short iutroduction into the handling of the tools available, they where asked to choose a source type they considered themselves familiar with and to design an "optimal" system for administering that type of sources on a computer, realizing with the tools at hand so much of such an optimal solution as possible and specifying clearly which technical building blocks were missing. Questions as to the capacities of the hardware were intentionally blocked: while some vague proposals were made to find out, how far decisions like resolution choosen would influence the practicability of the rudimentary prototypes, all participants were told to concentrate on the definition and construction of a tool, which would actually improve the possibilities of historical work with their type of source and ignore all questions of affordability and the like. In comparison to the Bremen exercise the ressources available were relatively ample: four RS 6000's of IBM could be used, each of which was equipped with 16 Mbyte of memory and had access to a network total of roughly 2.6 Gbyte of disk space. Access to scanners was suboptimal, but manageable.

The five participants decided eventually to work on prototypes of three systems for the handling of manuscripts or related materials, based upon the available prototype of an image handling and processing system. In all cases they used sources with which they had become familiar during their regular historical training, most of them in advanced seminars dedicated explicitly to the study of this particular type of source. The resulting systems were dedicated to the following areas: (a) A modernized version of the classical chartulary, where the digital image of a charter, together with it's transcription should be administered by the computer having image enhancement and similar tools available during the palaeographic study of the documents. (b) A system for the administration of historical maps, where indeces of topographical names and similar descriptors should give immediate access to the maps themselves. (c) A system for the handling of political and religious pamphlets from the reformation with a double aim in mind: on the one hand to

[6] 3^{rd} — 5^{th} year students of history.

study the relationships between the argument in the text and the illustration and on the other the way in which illustrations or parts of such became reused in other pamphlets.

Compared to some of the proposals, what might be adequate ressources for employing image related computer techniques in research, the platform we quoted here seemed to be quite substantial: it is quite interesting therefore, that *none* of the systems proposed could have been realized with the available hardware and/or software. We should repeat, that what the students were asked was not "realize a system with the ressources you have here", but rather: "realize a system that really would make it worthwhile to use computers henceforth, when you handle such sources. If such a system requires more ressources than we have here, it certainly is not your fault." In all cases the reasoning behind the requirements, which made it awkward or even impossible to realize what had been asked for was historically absolutely sound. Let us look at a few examples:

- *All* participants opted for the highest available reolution (400 dpi). For charters, it was argued, this was necessary, because any attempt at comparing the palaeographical properties of the texts needed the possibility to get close up looks of individual letters — and some nice examples for shedding light upon erasures (cf. the example in figures 1 and 2 on the facing page) and possible falsifications would only work under such resolutions. For maps it simply was not sensible to accept a resolution, where topographical names inscribed on the map would not be legible. For early modern pamphlets, we might have accepted a slightly lower resolution most easily, were it not, that in that case the chances at comparing parts of illustrations to find out about reusage would have dropped dramatically. As a result of that resolution, the image files contained typically something between 8 and 12 Mbyte per image, which would have been manageable on a 16 Mbyte machine. Unfortunately, however, the whole point of the exercise was the application of various advanced techniques, and that made it neccessary to have more than one version of the image on the screen and therefore in the memory at the same time.
- Commercial systems most frequently are able to display and process one image at a time. If they handle two or three snychroneously, they tend to boast about it. When the software platform we used had been designed for experimental work, it was assumed, therefore, that having 9 — 12 images available for synchroneous processing at any one time would be ample. As it turned out, using the machine for real palaeographical work, with systematic comparisons between huge sets of characters, the snychroneous handling of at the very least twenty, but probably for any but extremely trivial applications at least forty images was considered a conditio sine qua non.
- Particularly in the case of historical maps, but also in the case of charters, the standard size of flatbed scanners (usually the A4 format or its slightly larger American cousins) has to be considered insufficient. While the use of larger flatbeds (A0 would probably go a long way) would theoretically be the easiest solution, it is quite improbable, that such machines become affordable soon: therefore a whole set of possibilities for the simple recombination of images scanned in parts into a complete picture have been asked for.

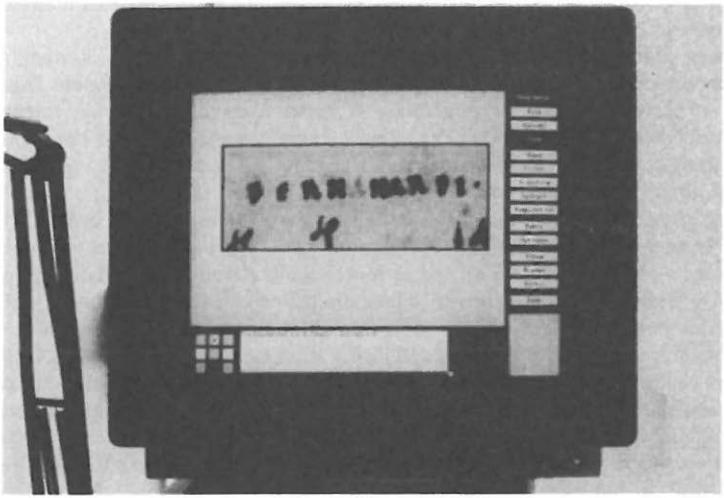


Figure 1: Erasure of 'A' in BERN[A]HARDI before ...

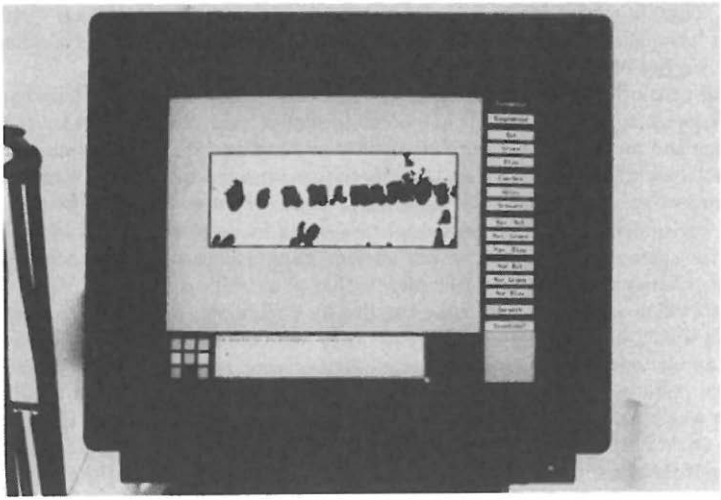


Figure 2: ... and after enhancement of ink shadows.

We have described these experiences at quite some length, because, in our opinion, they illustrate two very important differences between image processing in history, as traditionally described in the area of pictorial sources, and the handling of manuscripts with techniques of image processing.

Processing manuscripts does not usually have to be argued for. A system which displays a manuscript on the screen, which is as well or even better legible than the original, never has to be "sold" to a historian. If we see a manuscript on the screen and can do most of what we could do with the original, there is few doubt, that the technique as such is valuable. If we can improve it beyond the state of the original, the soundness of doing so will go unchallenged.

Bad quality in manuscripts is not tolerated. On the other hand, there is another fundamental difference between manuscript systems and such, which display pictorial sources: when an image — be it a work of art, or as source for the study of the daily life of the middle ages — is displayed on the screen, a long discussion can start, whether the quality with which this image is displayed, will suffice for the purpose of the respective system. Such a discussion can be long, protracted and highly stimulating — it will always, however, be subjective, being totally dependent on the degree in which the individual application claims to make the recourse upon the original unneccessary. In manuscripts, the criteria are cruelly simple: if the piece cannot be read on the screen as well as in the original, the exercise was not worth the effort. (And it seems, that it is relatively easy to use a picture as an illustration, without recognizing details of the relative size of an individual character in a page of manuscript.) If on the screen we can see what the unaided eye can detect on the original, the result is acceptable: but no historian will go for such a solution once she or he has realized, that in principle it is possible to see on the screen much more than in the original. There is a huge difference between a very good image and an image displayed speedily in a low quality; the margin between a well readable manuscript and a collection of irrelevant wiggles on the screen is extremely small.

With this restrictions in mind, always clinging to such quality as has been found appropriate in various tests, we will now look at a possible pilot project and discuss what it might require in time and money. A final word of introductory explanation may be required: in a number of ways we clearly deviate from the experiences reported by the impressive report about the project at the Archivos de Indias later in this volume. These differences are intentional: our understanding of manuscript processing includes the notion of digitalizing sources for conservation. While we see obvious implications of such a technique for archives as institutions, our primary metaphor is that of an edition, however: the processing of corpora which are larger than those handled by traditional editorial techniques, are still relatively small, however, and shall get the best available treatment. Therefore, legibility being an obvious prerequisite, we require, furthermore, sufficient quality to perform palaeographic research on the material prepared. The following descriptions build upon two projects which have been or rather are undertaken with the software platform developed by the author. In one case the surviving material at the former concentration camp of Auschwitz-Birkenau shall be stored in digital form to preserve it beyond the foreseeable dissolution of the originals. In that case the volume of images to be scanned in the near future is somewhere between 50,000 and 100,000 pages. On the other hand our estimates

are influenced by a project during which roughly 2000 photographs of architectural drawings shall be preserved in digital form for a study of the paper upon which these drawings have been sketched.

3. Step one: a digitalized archive

To discuss the possibilities of manuscript processing more systematically after our initial considerations, we will first examine the requirements of the following (so far hypothetical) project:

The City archive of X has large holdings of private charters from the later middle ages. About 20,000 pieces between, say, 1300 and 1500 exist. The collection is being used regularly by the local university for traning in medieval palaeography. During recent years some of the pieces seem to have gotten severly damaged, however. A solution is looked for, which allows to preserve these 20.000 pieces without further damage, makes to the users reproductions available, however, which do not hinder any kind of research. At a later stage, possibilities to edit such material and make it more widely available shall be found.

Almost all archives until very recently, and the vast majority of them even today, would for such a project more or less automatically look for a solution which turns the collection into a set of microfilms or microfiche. We will not try to make a detailed cost-effectiveness comparison between an approach at microfilming and digitalizing such a collection. Indeed, we assume that as of today turning the collection into microfilms would be considerably cheaper, which may change during the next five and will almost certainly change during the next ten years. We would first of all like to prove, however, that the option of digitalizing such collections, while not cheap, is already within an order of magnitude which makes it feasible even today.

The basic cost would of course consist of the computer needed: after the cautioning words we have given about the file sizes to expect, we have to envisage a work station which is powerful enough to process images of ten — twenty Mbytes without getting stuck. Typically, that would be a UNIX workstation with about 32 Mbyte of central memory or more. At current rates, this would translate into roughly US $ 30,000, taking academic accounts already into consideration. That price should cover a local disk of a few hundred Mbytes, which, given the sizes of the images quoted, would be definitely to small to begin with. At the current stage of technological development, we would therefore propose to add a 1Gbyte magnetic disk and a rewritable magnetooptical disk drive with two times 250 or two times 500 Mbyte per disk. Together this would add a little bit less than US $ 10,000 to our configuration, with each Mbyte of magnetooptical medium, used for the long term storage of the scanned material costing about 50 US cents. (With a clear downward tendency.) To that we would have to add a black and white scanner; for practical reasons it usually turns out to be efficient, to have a PC with a modest hard disk, which is fully dedicated as scanner server. Scanner plus PC plus all necessary connections between UNIX system and scanning unit should be available for US $ 5,000. If we assume, that we have charters, which (as private charters) are relatively small and, as a rule, fit unto the flatbed scanner, experience in the various projects quoted, but also in separated attempts at scanning, proposes two things: (a) we will not be able to get high quality reproductions at much below 10 Mbyte per charter of raw data and (b) considering the

various copying operations from scanner PC to optical background storage, we will not be able to keep up the scanning operation with a speed much higher than 5 charters / hour. This means, that the media cost for our project will be roughly at five US $ a charter and roughly 100,000 US $ for the whole archive. For scanning the 20,000 charters we will need 4,000 hours, or, assuming 200 effective 8 hour work days a year, roughly 2.5 years.

145,000 US $ and 2.5 person years, to convert all late medieval private charters of a city archive into digital representation. This is a lot of money; when we compare it with other steps to be taken for the preservation of endangered cultural property today, it looses some of the frightening magnitude it may have at first look, however. What is even more important: as we have noticed, more than two thirds of the expected expenditure for equipment have been set aside for storage media.[7] This, however, is precisely the area of development, where prices are currently going down most rapidly. That is, we have every reason to assume, that, what we propose here will become cheaper by about 25 % every year. (An increase in scanning speed is likely, but much harder to predict than the price development.)

While we have in many ways discussed a worst case figure now, we have on the other hand left a few potential sources of cost unmentioned. So we have assumed so far, that the archive would have no objections to the private charters being pressed firmly upon the flatbed scanner to have them absolutely level for scanning. This could be replaced by a strategy, by which a movable scanner is mounted on a suitable mechanism over the original which than is pressed down by a glass plate or another such device as used in professional photography of documents. Such a solution would probably add between ten and twenty thousand US $ to our bill for equipment — we will not discuss it here, however, as that is just the kind of cost which we also would incur in any microfilming operation at almost exactly the same rates.

So, that's the price of a digital archive. What advantage will we gain over a microfilming operation? Of course we could now quote the various possibilities of computer supported techniques of image processing: image enhancement; the possibility of improved handling of parts of the image for palaeographic or similar studies; and so on. More important for an immediate comparison, however, are a few advantages which are built right into the logic of digital image representation:

- Photographs deteriorate steadily, in an unpredictable step by step or rather shade by shade manner. Digital images can be compared bit by bit — it is clearly possible to say, whether they have the same quality as they had some years ago or not.
- When microfilms are copied after some time, the copy will in some degree be worse than the original; copying them will usually involve quite a bit of manual handling. Digital images can be copied by simple copy commands, the copy containing provably the same information as the original.

[7] This calculation may be open to doubt, as we have given figures for the uncompressed images; we will not go into that in detail, but propose, that we continue to use this figure, using the media cost which is not actually consumed after suitable compression for the addition of redundancy in our storage scheme for increased security.

- Storage media constantly becoming cheaper each successive copy will be more compact than the previous one.

We would like to take some pains at this stage to explain, why we have discussed this example here at such length and with those many financial details. Not, *because we think, that* at this moment *digitalizing manuscript material is already a financially convincing alternative to microfilming, when we speak about preservation "only", but to prove,* as *far as calculations can do that, that even this would be possible already today within the archival organisations of the more affluent European cities, without all too shatteringly expensive requirements.*

4. Using digitalized archives

We have in the last section concerned ourselves only with the question, how far the digitalization of substantial, but not overly huge corpora of manuscripts would be economically feasible. One could let it rest at that and simply predict, that, the figures being as they are presented, digitalizing manuscripts will eventually replace microfilming as a precaution to preserve valuable material. We would, however, like to expand this argument further.

What we described so far, is only the first step in a process. We propose, that it should be seen as the initial step in the following *idealtypische* procedure:

Step 1: A set of manuscript sources is scanned with the kind of resolution described. According to the preceding technical discussion, we can assume that the administration of 10.000 - 20.000 pages of manuscript on PC's of the upper or workstations of the lower range will be fairly easily possible within two or three years from now, manageable within the range of the typical two- to three-years project, dear to the heart of very many funding agencies.

Step 2: The documents which in that way are preserved permanently in photographic quality, are loaded into a data base: this data base contains at the beginning just a series of document identifications (basically archival numbers) and the scanned images of these documents.

Step 3: When working with such sources, the historian first of all transcribes the manuscripts literally. For this purpose the following tools are at his command:

- By a system of graphically displayed reference coordinates, the historian can bind individual transcribed phrases (or important abbreviations, or individual letters significant for the specific scribe) to the transcription of these portions of the text.
- Is a reading difficult, it is possible
  - to improve it by the means of the image enhancement techniques discussed.
  - but also to get a set of comparable portions of the manuscript transcribed so far. Either by asking for the graphical representation of cases where the possible readings have been transcribed at earlier stages of the working process or asking for cases where such transcriptions are within a given degree of similarity to the graphical form to be transcribed now.

Step 4: As soon as a continuous transcription exists, or parallel to it's creation, tools exist to insert into the text symbolic markup, specifying e.g. persons mentioned or topographical entities referenced. While in the design plans of our project a more general

notion of markup is employed, *specifically emphasizing that there are situations, where the graphical representation of symbols may be important*, SGML (*Standard Generalized Markup Language*[8]) would be a good example for a fairly general markup language which could be employed for such purposes. While we would like to stress, that we consider the question of how such markup schemes should be constructed for other than printing purposes to be anything but closed, one should point to the efforts of the *Text Encoding Initiative*[9] to define common rules for the applications of such markup schemes in specific areas of the Humanities.

In our context such markup is instrumental in converting the transcribed text into a component of a structured data base, into which each portion of the transcription is supposed to be parsed immediately after markup has been completed.

This, actually, would leave the notion of "digitalized images replace microfilms as safety copies" far behind. Indeed, what we describe here, is a potentially new technique for the dissemination of manuscript materials, being radically different from the notion of the classical printed edition. The differences are:

A clasical printed edition, which we will call a *static edition* henceforth, has two clearly identifiable and completely separate stages. In stage one, an editor collects material, and tries to prepare it for printing. As long as that stage lasts, the material is useless for the community at large: it is not possible to distribute the edition before it is fairly complete. On the contrary, in the process we described, which we call part of the concept of a *dynamical edition*, it would, sufficient standardisation of the storage media being assumed, be possible at any stage of the work to distribute it in precisely that stage.

As a kind of corrollary to the previous statement: a static edition is finished at a very precise moment. After that, changing and improving it becomes next to impossible. A dynamic one is never finished — it will certainly be rather difficult to ensure, that such a dynamic representation is not worsened by additional changes (I assume everybody is aware of the great concern that very problem is creating for Theodore Brunner of the Thesaurus Linguae Graecae[10]), but just because of this problem, we should not overlook the great potential of continually improving editions.

The connection of a photographic reproduction with a transcription of the text furthermore opens up completely new uses for an edition: in our introductory example of section 4 we mentioned, that part of the reason for preserving the charters was their constant use in the training of historians at the local university. A base of numerous charters in high quality reproductions, partially transcribed, obviously would open up completely new possibilities for teaching.

---

[8] Charles Goldfarb: *The SGML Handbook*, Oxford University Press, 1991; cf. Lou Burnard: *What is SGML and how does it help?*, in: Daniel Greenstein (Ed.): *Modelling Historical Data*, Scripta Mercaturae Verlag, 1991 (= *Halbgraue Reihe zur historischen Fachinformatik*), 65-79.

[9] C. Michael Sperberg-McQueen and Lou Burnard (Edd.): *Guidelines for the Encoding and Interchange of Machine-Readable Text*, Chicago and Oxford, Draft Version 1.0, 1990.

[10] Cf. Theodore Brunner, *Director's Letter*, in *Thesaurus Linguae Graecae Newsletter*, # 15, July 1989, pp. 1-2.

Finally: can you imagine a printed, static edition of the quoted roughly 20,000 late medieval private charters of a city archive?

Let us dwell a little bit more upon this choice between "static" versus "dynamic" administration of "edited" source material: one might wonder, whether this is not partially behind the controversy about the necessity to describe images in a controlled or an open vocabulary as well. A project which makes images available via a controlled vocabulary is still very much like a printed edition: the work group responsible for the initial description is doing the "right" thing; the user of the images later is very much in a passive role, "consuming" the correctly prepared material. It may be, that this model can be defended in history of the art, where after all the number of items is *relatively* small. If the new forms of history, focusing upon the strata in society below the small elites, which are documented in fairly small corpora, shall ever be as closely connected to the content of the sources, as traditional historiography has been, we will have to strive for types of edition, which make material available, *before* a huge editorial staff has invested heavily into each individual page.

We mentioned above, however, that actual palaeographical work asks for additional tools, which are hardly available in current software. To integrate such palaeographic capabilities, at least the following tools would have to be available within the standard working environment:

- The capability to handle a large number of high quality images of individual characters synchroneously on the computer: "high number" probably translates into something like thirty, or more precisely "sets of thirty", where a "set" can speedily be exchanged against another one, meaning, that something like two- or threehundred subimages have to be available at any one time for *rapid* swapping into the current work area.
- The capability to superimpose images above each other. To do this, however in a way, which allows:
  - To abstract a character, say "f" to a basic shape, which is described as a set of graphically displayed relationships between various parts of the "ideal" letter. (Say the ratio between height and length of middle horizontal stroke; the angle between middle stroke and vertical line; the relative length of a bottom horizontal stroke, if existing; and so on.)
  - Two or more of these "abstracted" letters shall than be superimposed, by operations which allow symbolic modifications of one of them. (Say "superimpose "f1" and "f2" after scaling "f2" to the height of "f1" and rotating "f1" into full vertical.)
  - The whole operation would have to be controlled by a mixture of interactive graphic commands ("pick the 'center' of the letter") and symbolic operations (like the ones described before).

5. Are Manuscripts Images?

We started by saying, that "manuscript processing" can be preliminarily defined as "the application of image processing tools" to manuscripts. We changed this statement slightly, however, by specifying later, that in at least one of the current focal points of discussion, there is a great difference between images and manuscripts: The "neccessary quality" of an image displayed on the screen, is very much open to debate, pending the decision, whether you consider an image just a preliminary illustration of a data base which shall provide access to a collection of such images stored in more conventional ways or as an object of study right on the screen. A manuscript is either as readable on the screen as in the original or it is not; if not, the exercise is futile. A manuscript on the screen can either be improved against the original, or it can not; if not, the solution is severly restricted.

The second major methodological conflict about images seems currently, whether it is more useful to describe them in a closed universe of a predetermined terminology, or to allow arbitrary, natural language descriptions in a not standardized vocabulary. Again, in the case of manuscripts the decision is more easy: A "description" of the manuscript, which contains a transcription has the possibility of eventually replacing a printed edition; any other has not.

We would not doubt, that there is still a lot to be done beyond transcribing a manuscript: in the case of "describing" a charter, e.g., we could construct a similar discussion, whether it is more useful to "describe" the shape of a Chrismon in controlled or uncontrolled vocabulary — but somehow this conflict seems to be relatively minor, given the major decision made before. Incidentially, this single example is just one instance of a question lurking in the background of the whole concept of an "edition", where a photographic reproduction is available all the time: If an editor describes the writing habits of the scribe of a charter, it would be obviously useful to integrate into the running text small snapshots of appropriate characters and allow the user to expand, while reading the description, these snapshots into a section of the original manuscript giving the context of the individual character. That is, for the formulation of "descriptions" of manuscripts we actually would need a major new concept in dataprocessing, being an "enhanced string", which actually would allow a mixture of ASCII characters and arbitrary portions of bitmaps to be administered as a fully integrated data type — that is not just being displayed in an appropriate way, but searched for, sorted and administered in powerful data bases.

Here, we feel, a radical change of style is indicated. While so far, we have discussed what would be needed to make a technology useful for a historian under certain circumstances, now we have to discuss, what technical tools have to be forged for such a purpose. We therefore separate the discussion of this point into an appendix, which is addressed to the technologically highly sophisticated reader, presenting a bluprint for a technical solution for these problems.

Appendix*

6. Design Proposal for a Nonlinear Data Type "Text".

6.1. What makes a text "historical"?

Speaking on the most general level, we consider a text to be "historical", when it describes a situation, where we do neither know for sure, what the situation has been "in reality", nor according to which rules it has been converted into a written report about reality. On an intuitive level this is exemplified by cases, where two people with the same graphic representation of their names are mentioned in a set of documents, which possibly could be two cases of the same "real" individual being caught acting, which, however could also be homographic symbols for two completely different biological entities. At a more sublime level, a change in the color of the ink a given person uses in an official correspondence of the 19^{th} century could be an indication of the original supply of ink having dried up; or of a considerable rise of the author within the bureaucratic ranks. Let us just emphasize for non historians, that the second example is all but artificial: indeed the different colors of comments to drafts for diplomatic documents are in the 19^{th} century quite often the only identifying mark, which diplomatic agent added which opinion.

What these introductory examples should demonstrate, is, that the text — the computer interpretable representation of a written document — forms in historical research an intermediate layer between two other layers of information. On the one extreme we have abstract factual knowledge about the various entities described in a text, which allows the interpretation of it; on the other there are purely graphical characteristics of the written document, which may carry meaning, but need not do so.

That the second problem is a genuine markup problem is probably obvious: if we use a computer to prepare diplomatic drafts of the 19^{th} century for printing, we obviously need a way to describe a portion of the document as being "written with blue pencil". Which, at the time of the first transcription is exactly what it says, a literal description of a graphic property, though during the process of research it may well acquire a more abstract connotation, like `author=M. Simpson`. This could of course be interpreted as such properties being eminently fitted to abstract rules for markup, because at the time of producing the markup we have not yet the faintest idea what the final representation in print, if any, of the specific graphic property is to be. Quite besides that at least I cannot very well see, how it should become possible to propose a finite list of such potentially significant graphical properties, there is a more basic problem. We all the time have now been speaking about graphical properties which *may* represent some meaning. Which is another way of saying, that this graphic property is purely accidental. To bring it to a point: almost all the examples given in the discussions on standardization during the last few years dealt with how to tag a structure which is clearly understood and where the graphic representation is accidental. Historical work deals with structures in a text which we want to *discover*, where the graphics we see may be all the clues we ever might get.

* The following definition is — with minor modicfications also contained in the author's paper *Historical Information Science: Is there such a Thing? New Comments on an Old Idea*. This paper has been presented in the fall of 1991 at the Academia dei Lincei in Roma; it is to appear soon in a volume edited by Tito Orlandi.

The real difficulty behind this might be a somewhat imprecise definition of what a "text" is to begin with. To me it seems, that in almost all the contributions to relevant discussions, a "text" is seen as either the starting point for research or the result of research: either what you get from a colleague to make your linguistic, stylometric or whatsoever analysis from or what you are going to deliver to the printer and, potentially, at the same time to another colleague. In the understanding of this document a text needs to be a considerably more dynamic kind of thing, the formally treatable representation of the current assumptions of a researcher about what his documents actually contain.

This on the one hand means, that we have to provide facilities to mark up graphical attributes which *may* acquire substantial meaning; on the other there have to be provisions for a link between a text and a set of assumptions about portions of it. To go back to our initial example: when we provide for marking a portion of a text as representing the "name of a person", we will also have to provide for a link to some background data base, which contains descriptions of "real" persons, being represented in the text by all kinds of conflicting graphic representations. State of the art data bases in history actually carry this a step further, by providing separate links between the graphical variation of a name to an algorithm, which is supposedly able to filterout the "accidental" orthographic variation of the name, before it is being linked to factual knowledge about the person this name is a tag for.

So, a historical text is in this document considered to be an representation of assumptions about some historical reality, containing on the one hand descriptions of graphical properties, which may require interpretation, at the other linkages to representations of knowledge, which are connected to a specified portion of the text.

This simple model has, however, to be extended into two directions. A "historical" text is in our opinion something which has come to us under a consistent set of circumstances: our interpretation of colored annotations can obviously be valid only within a corpus of materials which came into existence within one bureaucratic unit. Similarly the language of a medieval chronicle can be analyzed, at least in the first step, only within one copy of that chronicle, though it may have been transmitted, with minor variations, in a whole family of texts. At the same time, however, the reality described by the process of formulating a political document can very often be understood only, if two parallel sets of comments upon some drafts, by different branches of the bureaucracy, are interpreted synchroneously; and the "story" told by a medieval chronicle can only be analyzed, if it is seen as complete as possible: though sometimes no single text exists, which contains all the parts of it. By first approximation this means, we need a mechanism, to administer an integrated document as an entity, which consists of several layers of traditions, each consisting of some "text" — i.e. a collection of words — which can only be interpreted in the context of some assumptions about the rules applicable to it. As, obviously, for some portions of the fictitious "true chronicle of x" conflictingly tradited texts will exist, this leads directly to the requirement of a text representation which allows a given portion of text to have more than one equally valid form. We have, therefore, also to provide for a mechanism, which allows a dynamic handling of variants, which enables software, to treat one coherent representation of a text on a computer, as if it would just consist of one manuscript as well as if it would consist of the logical sum of two or more manuscripts. The computer representation of a machine

readable text should therefore in our opinion not only make it possible to handle variants, but to treat all streams of tradition combined into a "text" as potentially equal.

Finally, we have to define the relationship between a "text" as a running representation of a tradited document and a "text" as converted into a database according to some abstract model. In our opinion these two representations should be seen as very close to each other, allowing the database to inspect the natural language context out of which its entities of attributes have been derived, and on the other hand allowing the user of the machine readable text to jump from one portion of it to another portion which, irrespective of the language used, deals with the same abstract concept. More pragmatically: if you enter into a historical database a query like "When did the monastery of St.X receive more than five solidi from a single tenant?" we want to see at least the unstructured description of the relevant entries in the administrative records, if not the scanned image of the respective page, and when we encounter in our running text a peculiarly verbose eulogy about a given benefactor, we would like to be able, to get all other sources related to that benefactor, be the in the same source or not.

We are quite aware, that the requirements we just described can be met only in part by existing software: we think however, that there is small, if any, sense to concentrate completely upon the task of how to code data in such a way, that the can be handled by present day software. As a matter of fact tagging a text exhaustively and completely, seems to create such an additional overhead, that I see spurious chances at best, that any historian could be convinced to enter all needed tags by hand. So what we define here is certainly no markup, which normally will and shall be entered by a historian: to pretend otherwise would in my opinion be nothing but fictitious. The sense of a standardized text representation at least in historical research — and decidedly there — can only be to create a means for the communication between software systems, not between human historians. As such, however, we need a medium that does take account of things to come and is broad enough to give software designers some reason, why they should invest into implementing components, which support such recommendations as an exchange format. To do so we need some foresight; which is why we start from a non-existing system.

6.2. A "historical text engine"

To allow us to do all the things specified in a coherent computing environment, we would first of all like to sketch how such an environment should behave.

We asume on the following pages, that all texts are treated as "information strings". A running text consists simply of a collection of linearily ordered strings of this type; a data base or knowledge representation consists of texts which are connected in a non-linear way. As every linear structure can be described as a trivial case of a non-linear one, running texts, (factual) data bases, full text bases, knowledge bases and, as we will see, collections of bit-mapped data objects are all to be considered as specific realizations of a general representation of information. To make that possible, we assume further, that all the necessary string handling operations are taken care of by a "text engine" which relies on other software components to be provided with correct "information strings" irrespective how they are administered. We will see, however, that links to other "information strings" can be part of any of them.

In any implementation of the following concepts, a "text engine" could therefore be only realized in close connection with other dedicated software systems, which take care of administering the relationships between various information strings. These do not form part of the present considerations. As the definition of the various items of information to be handled requires references to them sometimes, we will, however, just shortly define the three most important tools of that type.

In our concept we did stress the similarity between a running text and a structured data base; indeed we will later see, that we are also considering cases, where one collection of information strings can alternatively but synchroneously be interpreted as a running text and as a data base. To make that possible, we assume that besides the text engine, which we cover here the following exist.

A *text administrator*. This is a very primitive program, which does not very much more, than performing I/O on strictly sequentially stored collections of information strings. A text processing system in our concept would use such a text administrator to save and load information strings from background media, which than are processed with the help of tools from the text engine. Whenever we use the term "textprocessing" in the remainder of this paper, we refer to software, which performs typical tasks of current day textprocessing, including primitive full text retrieval applications, by using the services of both, text administrator and text engine.

A *data base engine*. This is a family of software tools, which are responsible to administer non linear collections of information strings. These software components are responsible for the handling of all problems resulting from the adaptation of current retrieval concepts to handle context sensitivity of queries and uncertainty or ambiguity of structural relationships.

A *knowledge engine*. This is a family of software tools, which are responsible for the administration of all such conversion and transformation processes, which are built upon knowledge as are based upon dictionary-like structures or complex sets of rules. As all information is supposed to be evaluated dynamically, these software components in turn use components of the text engine, when the need to handle information strings arises.

These "information strings" which are treated by our assumed text engine in the environment shared with these other major modules, consist of linked lists of "uninterpreted items", which exist in an "interpretative environment". Whensoever an information string is handed to the text engine, it is guarantueed, that the later is supplied with a full copy of an interpretative environment.

While more precise definitions of interpretative environment and uninterpreted items will be given shortly, it makes their respective roles probably easier to understand, when we describe them somewhat intuitively first. As a first approximation, we could consider the interpretative environment as a table of mappings of abstract font commands into concrete printer operations. So when we look at the text engine operation "prepare output on a specific printer" the printing of the string starts with all such applicable parameters regarding printing and spacing, as can be derived from the interpretative environment handed over with the string to be printed. After this, the uninterpreted items are inspected and item by item converted into such strings and/or printer commands as represent their output

form in the light of the current interpretative environment. While this is a description of a current day printing process, in our opinion it should get further: the "font" of a text not only being relevant, when it is being printed, but also, when a string in font "A" is compared to a string in font "B".

A very important consequence of this separation between uninterpreted items and interpretative environment has however to be clarified already now. As mentioned initially, we deal not immediately with the question of markup. We assume, however, that the internal representation of a historical texts indeed needs some features, which are inherently like a symbolic markup: i.e., some information about how the text shall be processed, which is interpreted only, when the text is being processed. This produces a subtle difficulty, when we are speaking about non-linear structures of text, where individual parts of the text shall be accessible. As we will see further, the interpretation of character i of a text may depend on some information, that is contained between character $i-100$ and $i-90$ of that text. So, if we want to interpret the i^{th} character correctly, we would have to know, that information relevant to that character occurs before it in the string. Therefore we assume, that a "string of information", as we define it, is always administered so, that it is only accessed at a point, where it can be guarantueed, that all information necessary for it's interpretation is available. More formally, we speak of *entrance points* into a collection of strings, where a complete copy of the interpretative environment for the following character is available. All characters between two entrance points can only be correctly interpreted, when the text engine reads and interprets first all parts of the string of information, which are situated between the nearest entrance point and the character in question.

The importance of this concept can scarcely be overestimated. Indeed, the need to provide a sufficiently but not unnecissarily large number of entrance points, is the main reason, why we distinguish so sharply between a strictly sequential and linear text administrator, a strictly non linear data base engine, which, however, can assume, that from each of its items a path to the nearest applicable entrance point is defined, and a knowledge engine, which handles dictionaries of relatively small information strings, each of which has its own entrance point, as the can be accessed completely at random.

6.2.1 Types of uninterpreted items

Strings of uninterpreted items are made up of five different classes of items:

- *Basic items.*
- *String qualities.*
- *String links.*
- *String variants.*
- *Embedded structures.*

The role of these classes of constituents are in turn:

6.2.1.1 Basic items.

These items carry the actual information derived from a historical source. In the most trivial case, the consist of simple character codes. All such items, however, are considered to have logically the same rank. That is, a small bit map (e.g. for a non-deciphered language like the Indus hieroghlyphs or a non-textual symbol, like a water mark in paper) or a plain ASCII character can both form distinct items of a "string" in our sense. This assumes, that the text engine contains tools, which can sort and compare *all* types of basic items.

The following types of basic items are defined:

- *Simple characters.*
- *Character tokens.*
- *Bit mapped tokens.*
- *Pictures.*

6.2.1.1.1 Simple characters.

Simple characters are described by a sequence of n bytes per character, n defaulting to one in most text engines. It is assumed in this paper, that characters which represent letters, have one case only. For reasons which are given further below, it is assumed that case is just another string quality which does not justify a special treatment. Each simple character is represented by a numeric value, which indexes a table that contains a variable amount of information about the character. That information consists of:

- Sorting position of the character within the table.
- 'Binding' ofthe character. By this property we define its behavior in conjunction with neighboring items to it's left and right within the same string.

6.2.1.1.2 Character tokens.

A character token is represented by a traditional — henceforth called primitive — string of simple characters; in most real-world application starting with a common escape character. While being represented by a primitive string, the are conceptually, however, just the same as simple characters: the degree of similarity between two text tokens — as, e.g, expressed by a table of sorting values — is therefore completely independent of the string representation of the tokens. As the two primitive characters "a" and "A" may or may not be considered identical, independent of the code values assigned to them, in a historical text the two text tokens "`\chrismon`" and "`\cross`" may or may not be considered to be identical or similar; there is, however, no inherent relationship created by both tokens starting with the primitive string "`\c`".

6.2.1.1.3 Bit mapped tokens.

Bit mapped tokens are tokens, for which all is valid, what has been said about the properties of text tokens. Bit mapped tokens do not consist of a sequence of primitive characters, however, but of a sequence of the form: `escape_character-length-bitmap`. A further difference is, that their similarity is defined not by a tabular listing of their relationships, but the decision rules for the compariosn of the bitmaps themselves.

6.2.1.1.4 Pictures.

Intuitively pictures are obviously the same as bitmaps: indeed, their internal representation is assumed to follow the same rules, as just given in the preceding section. While a bit mapped token is assumed to be an atomic item of information, a picture is assumed to be a possibly structured entity, which may occur as part of a text, will more often be connected to it, however, by the mechanism describe in section 3.1.3.5 for text links.

6.2.1.2 String qualities.

As mentioned initially, each uninterpreted information string exists in an interpretative environment. This is defined by a number of assumptions, which are true for the first information of the string. The information string contains, besides the basic items discussed sofar, which carry the "real" information, indications for a change in any of these assumptions. This implies for the text engine, that all of its constituents are guarantueed to start the processing of a string only at well defined starting points, all operations defined on the strings parsing allong them. While this may seem to be a distraction, we would like to emphasize it here, as otherwise the concept of string quality cannot be understood.

Every string of information exists in an environment which defines its

- modes,
- *style*,
- color,
- *size* and
- *view*.

It should be noted here, that these names have been choosen for intuitive plausibility, as have the examples below. The flexibility of the concepts, however, is to be derived from the abstract definitions given.

6.2.1.2.1 String modes.

String modes define the absence or the presence of a set of attributes. That is, a given item of information can have an attribute or can miss it. It is not possible, to have a mode in a certain degree. Every string of information inherits from its interpretative environment a set of default modes. If a certain mode is not defined in the environment, it is assumed to be absent.

The most intuitive example of a string mode is the case of a character. As we defined before, that simple characters are assumed to be caseless, it would completely depend on the interpretative environment, whether the string

`this is a string`

would be interpreted by the text engine as upper or lower case. By interpretation we mean in this and all following examples, the behaviour of *all* components: an "upper

case" string would be printed as upper case (if possible on the output device) but its being uppercase would also influence comparison operations (see below).

At any point in a string a mode can be activated or deactivated:

Mode: +case this is a string

would always result in an uppercase string, irrespective of the assumptions of the interpretative environment,

Mode: -case this is a string

always in a lowercase one. The interpretation of

Mode: +case t*Mode: -case* his is a string

is clear.

As each mode, which is not explicitly defined in the interpretative environment, is assumed to be absent, their number is arbitrary and has not to be known by the text engine. Modes which are encountered in an information string, for which the text engine has no explicit instructions are therefore completely ignored. In the case of

Mode: +case t*Mode: -case* his is a *Mode: +german*
Mode: +case z*Mode: -case* eichenkette*Mode: +german*

the mode "german" would in most search operations be ignored; in full text or data base applications it could, however, be used as a selection criterion irrespective of the structure which defines the relationship between this information string and all others in the currently administered data; in Anglosaxon text processing applications it could be interpreted as underlining.

6.2.1.2.2 String style.

While any item in a string of information can at the same time have an arbitrarily large number of modes, it alway has precisely one style. Statistically speaking, the style of an item is handled on a strictly nominal level: there are no assumptions about any relationship between two different styles expressed in the internal representation of styles.

The most intuitive example for the style of a text would be font information, as in the example

this is a *Style: german* zeichenkette*Style: basic*

Please note, that this is not exactly the same than the previous example: while in the previous one, "z" could acquire the mode case, without loosing the mode german, no part of Zeichenkette could acquire the style gothic without loosing the style german.

6.2.1.2.3 String color.

The quality of color is similar to that of style, by its values being mutually exclusive. Its intuitive interpretation is probably obvious and the introductory remarks of this paper show a potential application. For a systematic interpretation, however, it is much more important, that this quality is supposed to represent statistically an ordinal level. That is, it is assumed to be represented internally by ordinal numbers, which allow expressions of similarity. (A similarity to the implementation of the enum concept in the 'C' programming language is intentional.) The two strings:

Color: dark blue George Smith

and

Color: light blue George Smith

would in most interpretative environments therefore be assumed to be closer to each other, than the strings:

Color: dark blue George Smith

and

Color: light red George Smith

It would, however, *not* be possible, to express the difference in the degree of similarity between the two pairs of names. (This is a statistical statement and a definition of the concept of color, not a statement about artistic and/or biological perception of colors.)

6.2.1.2.4 String size.

String size, too, has a pretty obvious application. It is similar to the concept of color, allows additionally, however, to express a difference in the degree of similarity between two strings of information being compared. In the three fragments:

Charter a: *Size:* 20pt \chrismon *Size:* 10pt In nomine individuę trinitatis ...

Charter b: *Size:* 30pt \chrismon *Size:* 10pt In nomine individuę trinitatis ...

Charter c: *Size:* 40pt \chrismon *Size:* 20pt In nomine individuę trinitatis ...

the chrismon in a is more similar to the one in charter b therefore, than to the one in charter c; the proportion between the sizes of chrismon and main body of script, however, is identical between charters a and c, while both are dissimilar to b in precisely the same degree.

6.2.1.2.5 String views.

The qualities fo far — with the possible exception of color — may be seen as an attempt to define classical typesetting attributes in a sufficiently systematic way to allow their interpretation on an intermediate level between typographical representation and conceptual understanding. We recapitulate: the color of a note in a diplomatic document may ultimately acquire some meaningful, abstract interpretation; at the beginning of an editorial process, however, it will be exactly what it looks like: proof that Mr. X used a blue pencil.

The concept of *string view*, on the other hand, has been introduced to handle phenomena, which often occur in manuscripts, have, however, no generally accepted typographical conventions assigned to them.

Typical examples would be portions of a text, which are legible and obviously part of the original manuscript, but which later have been crossed out, additions being added at the same time, or manuscripts, which have been written by a number of scribes, some of which can be identified, while others can not clearly be distinguished. Obviously all these

properties of a manuscript could in principle be covered by the string qualities given so far.

The tools provided so far, did always assume, however, that each of the qualities would exist alone: a set of binary qualities, exactly one nominal quality, exactly one ordinal and exactly one which allows comparisons of degrees of similarity. To generalize this model, we introduce the concept of a *text view*, which is defined as *View*: Type, Name, n. As it's first argument, it accepts any of the previously identified text qualities, i.e., `mode`, `style`, `color` or `size`. It introduces a named text quality, which has the properties discussed so far. So our previous notations could be seen as shorthand for a more general text view notation. The following equivalences would hold:

Mode: n == *View*: `mode, default,` n
Style: n == *View*: `style, default,` n
Color: n == *View*: `color, default,` n
Size: n == *View*: `size, default,` n

The difference between these two definitions is, however, more important, when it comes to actually implementing such a model. We assume, that a text engine optimizes the four *default views* with regard to speed of processing of individual information string. This means, that when one of the default views is encountered in an information string, it will be taken care of by an extremely quick operation. When an explicitly named view is encountered, however, the text engine is allowed to reorganize the interpretative environment to allow for it. (All comparison operations have to allow for `size` sensitivity without loss of efficiency; a comparison which has to allow for five independent sensitivities for `views` of type `size`, however, is allowed to be significantly less efficient than a comparison that handles just the default `size view`).

This differentiation — and more so the space it is assigned — may be a reflection about the author's background in actual program development: we consider this differentiation to be extremely important, however, as, on the other hand we assume, that historical texts can be handled correctly only, if the number of `views` allowed is unlimited.

6.2.1.3 String links.

While we consider string qualities to be a more systematic description of classical textual properties, string links define the conditions for assembling individual information strings into larger objects, like texts or data bases. Basically we consider it necessary to embed into a string reference points, from which it is possible to branch to other strings. The intuitive example for this would be a footnote.

As we mentioned initially, we consider a text not so much to be something which primarily has to be printed, but as a representation of the current knowledge about some historical phenomenon. All such points, where it shall be possible to branch from a given point of reference within a text to somewhere else, are therefore meaningful only as being connected to specific operations of the assumed text engine.

These operations are:

- *branches*,
- *text references*,
- *data base references*,
- *knowledge references* and
- *bitmap references*.

6.2.1.3.1 Branches

A branch is the most simple string link. It consists of a pair of addresses, connecting an arbitrary point within a string of information with the entrance point into another string. The intuitive example for it is a note in textprocessing. Branches pointing from an arbitrary point of an information string to the entrance point into another string, we will call *exceptions* and denote with the symbol [Name →]; branches from the entrance point of a string to an arbitrary point of another information string, we will call reference and symbolize by [← Name].

In the case of a footnote, these elements would be used as follows:

```
... in recent publications [footnotes →] this point
is usually not discussed any more ...

[← footnotes] Cf. John Smith: ...
```

The text engine resolves the arrows in these string links as follows:

- exceptions are plain pointers to the beginning of another string of information, allowing the interpretative environment to be initialized the standard way.
- references are similar pointers to the arbitrary point from which the exception did branch away. They can, however, only be traversed, if this point in the collection of strings has been reached via a previous refernence from the corresponding exception. In such cases the text engine stacks a copy of the state the interpretative environment has been in, when the exception was activated. If the reference is reached by any other navigational operation within the collection of strings in question, it is not possible to follow it to the spot of the exception.

6.2.1.3.2 Text References

Text references allow it to bracket a specific portion of text and logically to assemble all such portions into a specific collection of texts. An intuitive example within text processing would be the creation of registers.

Text references consist of pairs of the form

```
[Name ⇒] some string [⇐ Name]
```

which we shall discuss as "forward reference", "reference string" and "backward reference" respectively.

A *forward reference* consists of

- a mark pointing to the end of the reference string,
- a pointer to the next forward reference in the collection of strings with the same name and
- a pointer to the nearest entrance point into the information string containing the next forward reference in front of it.

It enables a text engine therefore, to navigate from one text reference immediately to the next; does not remove the necessity, however, to interpret the portion of the information string in front of the respective forward references to bring the interpretative environment into the state it has to be, when the reference string shall be interpreted correctly.

A *backward reference* contains the same information, does so with respect to the preciding text reference in the collection of information strings in question, however.

6.2.1.3.3 Data Base References

Data base references have no precise equivalent in traditional applications. A specific data base pointer [↑ x] is defined by

- the data base which shall be referenced,
- a procedure specifying for the data base engine in question, how the reference shall be converted into an information string.

When control returns to the text engine, a modifiable copy of the information string created in this way is brought to its disposal and for purposes of textprocessing integrated transparently into the "text" proper. Obvious, but trivial examples of usage would be the dynamic treatment of bibliographical and/or biographical information.

6.2.1.3.4 Knowledge References

Knowledge references are denoted by identical bracketing reference symbols of the form

[↓ KB] `object text` [↓ KB]

Object text refers to an information — like complex chronologies, historical currencies and similar, as described in previous papers by this author — which needs to be transformed, before submitted to specific classes of treatments. (Think once more of sorting or comparisons.)

The *knowledge* referenced is assumed to consist of a formal definition of the format a object text has to have to be processed plus a collection of suitable dictionaries to apply specific transformation rules.

A specific [↓ x] consists of

- a specification of the knowledge (base) to be accessed plus
- a set of operations of the text engine, which trigger the transformation of the object text.

For all such operations, the text engine does not process the object text itself, but the result of the conversion.

6.2.1.3.5 Bitmap References.

A bitmap reference consists of a pair of identical bracketing symbols of the form

[⇑ x] `interpretable equivalent` [⇑ x]

It consists of

- a reference to a bit image, which either is administered as a "picture" as defined above or as a completely independent file and
- a set of coordinates within the bitmap.

The text engine references the bit image, when suitable display units are available, uses the interpretable equivalent, however, when functions are being called, which require operations like search or comparison.

6.2.1.4 String variants.

String variants have an obvious application: the administration and processing of the apparatus criticus of a text. The following model for the integration of variance into a general text representation model sees this only as a starting point, however. We rather assume as application the creation of dynamic text representations, i.e. of text representations, which allow software to treat all variances of a text as "equal". This could be envisaged by display modules, which allow the user to switch from variant α to variant β by hitting a function key, the displayed text in all cases where variants exist in both stages following the readings of manuscript α or β.

It is assumed that — unless indicated otherwise by the interpretative environment — an information starts with a part that is present in all witnesses for a given text. This assumption is changed, as soon as in the processing of the information string a *block border* is encountered. Block borders, which can be nested, limit parts of an information string, for which substitutable variant readings exist.

Generally a text with variants would therefore be represented as

`text present in all witnesses` [Block ≫] `variant readings` [≪ Block]
`text present in all witnesses`

The *variant readings* consist of blocks of the form

[*Variant*: Name (→,⇒)] `text of reading` [(⇐,←) Name :*Variant*]

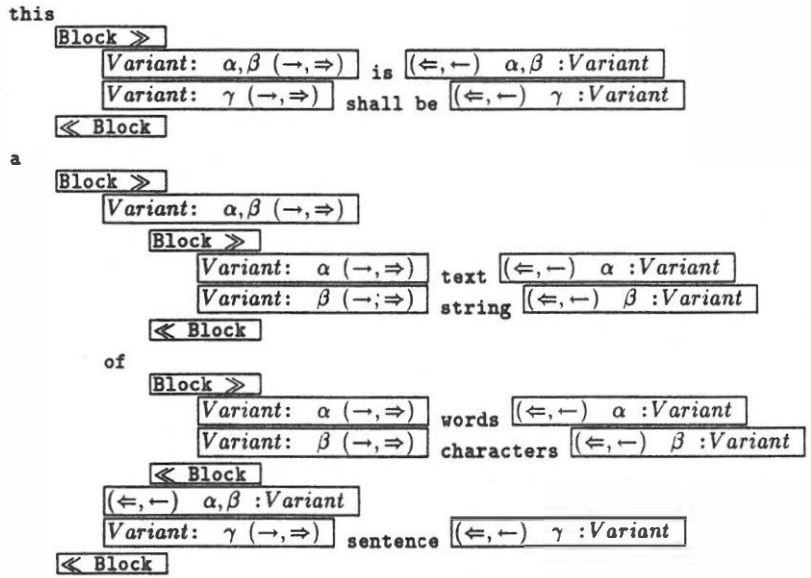
where nameset specifies in which witnesses a given reading is present.

The (→,⇒) and (⇐,←) symbols indicate, that all variant readings are linked as a list, where each atom has a pointer to the next variant reading, but at the same time also a pointer to the end of the respective block to speed up processing.

The mechanism is probably best explained by two examples. Lets first look at the situation where manuscript α gives `this is a text of words`, while manuscript β reads `this is a string of characters`. In this case we assume a representation which as:

`this is a`
[Block ≫]
[*Variant*: α (→,⇒)] `text` [(⇐,←) α :*Variant*]
[*Variant*: β (→,⇒)] `string` [(⇐,←) β :*Variant*]
[≪ Block]
`of`
[Block ≫]
[*Variant*: α (→,⇒)] `words` [(⇐,←) α :*Variant*]
[*Variant*: β (→,⇒)] `characters` [(⇐,←) β :*Variant*]
[≪ Block]

When, to show nesting, we add manuscript γ which reads `this shall be a sentence`, we get:



As this may look somewhat complicated: let me remind the gentle reader, that we are describing here a structure, which internally shall be able to handle something. Entering markup into a text, which in turn is parsed to provide the required pointers, would be one way to achieve this.

6.2.1.5 Embedded structures.

While the preceding parts of this paper form presumably a general model for the representation of historical texts, this section may be more specific to the activities of the author. It deals with the problems from representation, where a data base, which follows a network model, is integrated into a text. The idea is, that data base queries are processed, which are translated via structural information contained in a data dictionary into such navigational processes as are required to select various items of information for processing. Unlike in traditional data bases, that information is, however, not "extracted" from a data base. Instead of extracted information from natural text, we assume, that the text as a whole is represented on the machine, various pointers — similar to the classes of such, which we described sofar — being "hidden" within the running text, which define that a given part of it is not just a series of characters, but at the same time the content of a structurally defined field of a data base.

The idea is, that we have a text, like the sentence *On the 19th of March 1764, John Winslow and George Bilmington appeared before this court to claim ....* which by software based upon our ideal text engine can be handled for typical purposes of textprocessing, other software based upon this engine can at the same time, however, treat terms contained within it — like *John Winslow* — as the value of the attribute **name** of an incidence of the

entity `person`. Applications would be tasks like: *Select all court records, where people who have been born more than fifty miles from a given city of reference appear as interested party; select out of the running text all portions which are marked as "quotations"; compute a set of stylometric indices for the vocabulary used in such quotations.*

As already mentioned, we do not claim in this section, that our considerations here are general. We assume, that such models of the combination of structured representations of information and the covering text containing that information, can only be realized with data structures that allow inconsistent information to be handled and avoid normalization procedures. As such models are few, we have so far only considered the problem of "hiding" $\kappa\lambda\epsilon\iota\omega$ data structures within the text. In the case of that system, the data are structured into a network which consists of entities[11] called *groups* which among themselves can be linked by arbitrarily many relationships. Groups have an arbitrary number of attributes, called *elements*, which conceptually consist of variable length arrays allowing an arbitrary number of values, called *entries*, for each attribute.

To hide a network like that within a collection of information strings, as we described them so far, we see two possibilities:

On the one hand, the structural network of a data base could simply exist parallely to a text as we described it here, all entries of the network consisting of pointers to the relevant portion of the text, together with length information. (Obviously these pointers would have to consist of the nearest entrance point of the text plus an offset to the relevant portion.) The advantage of such a construction is obvious: existing database software could be taken over more or less unchanged and all textual functions needed, are contained per definition in a text engine, as we defined it so far. In our opinion this otherwise obvious model has, however, two major shortcomings: obviously updating the text could easily corrupt all the addresses contained in the entries of the data base. serious, as this problem is, we could handle it conceptually, by allowing for ⇑ DB links with a slightly modified definition, which would have to be inserted into the text at each point, which is the target of a data base referencing it. While this update problem is certainly tricky, it could eventually be mastered by the techniques hinted at.

Conceptually much more severe is a problem, which is introduced, when we consider even the most simple concept of data types. We already have introduced in section 3.1.3.4 above the concept of a *knowledge pointer* (⇓ KB). We did so, because of the necessity to treat "special kinds of text" in a special way. It is not sensible to sort calendar dates in alphanumeric fashion; and converting temporal notation related to the medieval saints into a notation that can meaningfully be sorted is no completely trivial task. This, however, is a typical data base problem. Of course a number of solutions could exist: one would be to include the functional equivalence of a ⇓ KB into the data dictionary for this field

[11] We discuss here the problem of hiding data structures within a general text representation. $\kappa\lambda\epsilon\iota\omega$'s groups are not entities in the sense of conventional DBMSs: to clarify, that they are not, we call them groups; as a precise definition of the differences would go beyond the scope of this paper, we do not give it however, but mention that envisaging them as entities will not be totally wrong. The same is the case for "elements" and "entries" introduced below.

— which means a type of redundancy which is dangerous. Another solution would be, to let the [↓ KB] point to the data dictionary, rather than to the relevant knowledge bases to begin with. This could mean, that we need different implementations for a [↓ KB] that occurs in a text without a hidden structured data base than the one used in texts with such an underlying structure; a bad solution. Avoiding that, however would bring us into the situation, where texts without an underlying dummy structure would not be allowed; not a bad solution, but an outright impossible one.

To avoid these situations, we propose therefore to discuss data models, which are per definition built into "natural texts". This would seem unusual from the point of view of other disciplines, but in history — or rather in the source-oriented model of historical dataprocessing — data bases, as this author has pointed out elsewhere, are always not so much collections of information, which define their own reality, but an attempt to structure information that has been tradited in a coherent form, i.e., usually as a text.

In the case of *κλειω*, we consider this possible, by allowing the text engine to administer three additional classes of constitutents of a string of information: *group pointers*, *element pointers* and *entry pointers* (denoted as [*Group*: Name ∞], [*Element*: Name ∞] and [*Entry*: Name ∞] respectively). They are defined as a generalization of the concept of a text reference introduced in section 3.1.3.2. The ∞ symbol replacing the ⇒ / ⇐ component of the [Name ⇒] / [⇐ Name] notation is assumed to indicate, that this generalization allows for an unlimited number of references starting from each embedded structural symbol, while our textual references provided always for precisely on physical reference.

6.2.2 The Interpretative Environment

The interpretative environment, which we have known intuitively so far as a kind of refinement to the concept of a printer driver consists of two independent components.

There exists a table which describes for each information string to be processed all the text qualities which it has, when any basic item is being encountered by any component of the text engine. This part of the environment has to be complete: that is the reason, why we introduced the concept of an entrance point into an information string. This part of the interpretative environment is loaded whensosever a particular information string is presented to the text engine.

The second part of the interpretative environment describes not an information string, but the status of the text engine itself: it consists of applicability information for each possible textual quality and optionally a mapping of that quality to an input convention or an output property. This concept can best be interpreted, if we look at the idea of case sensitivity in text operations. Case sensitivity would be modeled in our concept as a state of the interpretative environment, where case is an applicable mode (and characters are during output mapped to different representations).

6.2.2.1 Applicability of Modes

The applicability of a mode consists of a statement, if the status of this mode shall be checked, when a basic item is being encountered by the text engine. If during printing the mode **german** is applicable, a mapping of characters with this mode into another font (or

underlining) will be used; if it is not applicable, such mapping will be ignored. *Mutatis mutandis* this is also the case, when comparsions are performed.

6.2.2.2 Applicability of Style.

The applicability of style consists of a statement, if it shall be checked when a basic item is being encountered by the text engine. For applicability style could be interpreted like a mode: if it is applicable, any difference in style will make two basic items unequal; there is no way to define a degree of difference.

6.2.2.3 Applicability of Color.

The applicability of color defines whether optional I/O mappings shall be used, and a range, within which otherwise identical basic items have to be considered equal with regard to color. This range can be given as a pair of absolute numeric values, or as a table which specifies for each `color` in one information string, which `colors` in the other are acceptable.

6.2.2.4 Applicability of Size.

The applicability of size defines whether optional I/O mappings shall be used, and a range, within which otherwise identical basic items have to be considered equal with regard to size. This range can be given as a pair of absolute numeric values, or as percentage of the larger of two basic items, within which a smaller one is still to be considered equal.

6.2.2.5 Applicability of Views.

As views are a more general form of the other textual qualities, their applicability is identical to that of the default modes of the four classes given above. The implementation considerations given in 3.1.2.5 define the difference between the applicability of any named view of the four types and the four default views of each type.

6.2.3 The Text Engine

What kind of activities our hypothetical text engine is performing, was shown intuitively more or less during the preceding sections. Obviously it is related to the production of output; obviously it is also performing comparison operations: our example of a "german sensitive" comparison as an equivalent to the optional "case sensitivity" of current text processing software implied so much. Though the definition of such a text engine is certainly not part of a discussion of text representation, we would like to include a brief, but somewhat more systematic, definition than given so far. The reason for this is, that we assume, that recent discussions on text representation had a tendency to concentrate a bit to much on printing. We would therefore like to describe operations, we think necessary to make use of all the qualities described in a more general way, i.e., influencing any kind of operation the type of text we describe here has to undergo.

Whensoever we define in the following sections an ability the "text engine shall have", this is a abbreviated expression therefore for the following reasoning: "We assume that processing historical data of requires a specific ability. If historical data are to be processed by more than one software system, we therefore need a standard for the encoding of the property calling for this ability."

6.2.3.1 Texts Handled.

The text engine shall be able to handle texts, which are mixtures of byte coded and bit mapped items. All items a text consists of has to be processable by all components in question. Which class an item has, has to be transparent for any applications programmer using tools provided by the text engine.

6.2.3.2 Import / Export.

The text engine shall provide tools to import and export strings. This means, it shall be able to convert its own internal representation of an information string into a form that clearly distinguishes between different classes of items, specifically between byte coded and bit mapped ones, so software components, which do not have the ability to handle both, can extract those portions of a text, which they can handle. This export format, which is described as *external text format* has to be transferable on standard communication links.

6.2.3.3 Comparison and Sorting.

The text engine shall have as part of its interface functions which are able to compare and sort sets of information strings, fully controlled by an interpretative environment as described above.

6.2.3.4 Searching.

The text engine shall be able to use any informations string, which specifically includes such as contain bit mapped items, as a search key in the administration of large string collections, like dictionaries.

6.2.3.5 I/O.

The text engine shall be able to convert its internal representation into a form which represents its various classes of items also on output devices, which do not provide means for the most obvious kind of presentation. It also shall contain parsing functions, which convert formats provided by input devices or software supporting sophisticated forms of inpnt into a common internal presentation.

Halbgraue Reihe
zur Historischen Fachinformatik

Herausgegeben von
Manfred Thaller
Max-Planck-Institut für Geschichte

Serie A: Historische Quellenkunden

Band 14

Erscheint gleichzeitig als:

MEDIUM AEVUM QUOTIDIANUM

HERAUSGEGEBEN VON GERHARD JARITZ

26

Manfred Thaller (Ed.)

Images and Manuscripts in Historical Computing

Max-Planck-Institut für Geschichte
In Kommission bei
SCRIPTA MERCATURAE VERLAG

St. Katharinen, 1992

Printed in Germany
Druck: Konrad Pachnicke, Göttingen
Umschlaggestaltung: Basta Werbeagentur, Göttingen
ISBN: 3-928134-53-1

Table of Contents

Introduction

Manfred Thaller

This book is the product of a workshop held at the International University Institute in Firenze on November 15th, 1991. The intention of that workshop has been to bring together people from as many different approaches to "image processing" as possible. The reason for this "collecting" approach to the subject was a feeling, that while image processing in many ways has been the "hottest" topic in Humanities computing in recent years, it may be the least well defined. It seems also much harder to say in this area, what is specifically important to historians, than to other people. In that situation it was felt, that a forum would be helpful, which could sort out what of the various approaches can be useful in historical research.

To solve this task, the present volume has been produced: in many ways, it reflects the discussions which actually have been going on less, than the two companion volumes on the workshops at Glasgow and Tromsø do. This is intentional. On the one hand, the participants at the workshop in Firenze did strongly feel the need to have projects represented in the volume, which were not actually present at the workshop. On the other, the discussions for quite some time were engaged in clarifying what the *methodological* issues were. That is: what actually are the topics for scholarly discussion beyond the description of individual projects, when it comes to the processing of images in historical research?

The situation in the area is made difficult, because some of the underlying assumptions are connected with vigorous research groups, who use fora of scholarly debate, which are only slightly overlapping; so, what is tacitly assumed to hold true in one group of research projects may be considered so obviously wrong in another one, that it scarcely deserves explicit refutation.

We hope, that we have been successful in bringing some of these hidden differences in opinion out into the open. We consider this extremely important, because only that clarification allows for a fair evaluation of projects which may have started from different sets of assumption. So important, indeed, that we would like to catalogue here some of the basic differences of opinion which exist between image processing projects. The reader will rediscover them in many of the contributions; as editor I think however, that sumarizing them at the beginning may make the contributions — which, of course, have been striving for impartiality — more easily recognizable as parts of one coherent debate.

Three basic differences in opinion seem to exist today:

(1) Is image processing a genuine and independent field of computer based research in the Humanities, or is it an auxiliary tool? Many projects assume tacitly — and some do so quite outspokenly — that images on the computer act as illustrations to more conventional applications. To retrieval systems, as illustrations in catalogues and the like. Projects of this type tend to point out, that with currently easily available equipment and currently clearly understood data processing technologies, the analysis of images, which can quite easily be handled as illustrations today, is still costly and of uncertain promise. Which is the reason why they assume, that such analytical approaches, if at all, should be undertaken

as side effects of projects only, which focus upon the relatively simple administration of images. Their opponents think, in a nutshell, that while experiments may be needed, their overall outcome is so promising, that even the more simple techniques of today should be implemented only, if they can later be made useful for the advanced techniques now only partially feasible.

(2) Connected to this is another conflict, which might be the most constant one in Humanities data processing during the last decades, is particularly decisive, however, when it comes to image processing. Shall we concentrate on levels of sopbistication, which are available for many on today's equipment or shall we try to make use of the most sophisticated tools today, trusting that they will become available to an increasingly large number of projects in the future? This specific battle has been fought since the earliest years of Humanities computing, and this editor has found himself on both sides at different stages. A "right" answer does not exist: the debate in image processing is probably one of the best occassions to understand mutually, that both positions are full of merit. It is pointless to take permanently restrictions into consideration, which obviously will cease to exist a few years from now. It discredits all of us, if computing in history always promises results only on next years equipment and does not deliver here and now. Maybe, that is indeed one of the more important tasks of the *Association for History and Computing*: to provide a link between both worlds, lending vision to those of us burdened down by the next funding deadline and disciplining the loftier projects by the question of when something will be affordable for all of us.

(3) The third major underlying difference is inherently connected to the previous ones. An image as such is beautiful, but not very useful, before it is connected to a description. Shall such descriptions be arbitrary, formulated in the traditionally clouded language of a historian, perfectly unsuitable for any sophisticated technique of retrieval, maybe not even unambigously understandable to a fellow historian? Or shall they follow a predefined catalogue of narrow criteria, using a carefully controlled vocabulary, for both of which it is somewhat unclear how they will remain relevant for future research questions which have not been asked so far? — All the contributors to this volume have been much to polite to phrase their opinions in this way: scarcely any of them does not have a strong one with regard to this problem.

More questions than answers. "Image processing", whether applied to images proper or to digitalized manuscripts, seems indeed to be an area, where many methodological questions remain open. Besides that, interestingly, it seems to be one of the most consequential ones: a project like the digitalization of the *Archivo General de Indias* will continue to influence the conditions of historical work for decades in the next century. There are not only many open questions, it is worthwhile and neccessary to discuss them.

While everybody seems to have encountered image processing in one form or the other already, precise knowledge about it seems to be relatively scarce. The volume starts, therefore, with a general introduction into the field by J. v.d. Berg, H. Brandhorst and P. v. Huisstede. While most of the following contributions have been written to be as self supporting as possible, this introduction attempts to give all readers, particularly those

with only a vague notion of the techniques concerned, a common ground upon which the more specialized discussions may build.

The contributions that follow have been written to introduce specific areas, where handling of images is useful and can be integrated into a larger context. All authors have been asked in this part to clearly state their own opinion, to produce clearcut statements about their methodological position in the discussions described above. Originally, four contributions were planned: the first one, discussing whether the more advanced techniques of image processing can change the way in which images are analysed and handled by art historians, could unfortunately not be included in this volume due to printing deadlines: we hope to present it as part of follow up volumes or in one of the next issues of *History and Computing*.

The paper of M. Thaller argues that scanning and presenting corpora of manuscripts on a work station can (a) save the originals, (b) introduce new methods for palaeographic training into university teaching, (c) provide tools for reading damaged manuscripts, the comparison of hand writing and general palaeographic studies. He further proposes to build upon that a new understanding of editorial work. A fairly long technical discussion of the mechanisms needed to link images and transcriptions of manuscripts in a wider context follows.

F. Colson and W. Hall discuss the role of images in teaching systems in university education. They do so by a detailed description of the mechanism by which images are integrated into Microcosm / HiDES teaching packages. Their considerations include the treatment of moving images; furthermore they enquire about relationships between image and text in typical stages in the dialogue between a teaching package and a user.

W. Hall and F. Colson argue in the final contribution to this part the general case of open systems, exemplifying their argument with a discussion of the various degrees in which control about the choices a user has is ascertained in the ways in which navigation is supported in a hyper-text oriented system containing images. In a nutshell the difference between "open" and closed systems can be understood as the following: in an "open system" the user can dynamically develop further the behaviour of an image-based or image-related system. On the contrary in static "editions" the editor has absolute control, the user none.

Following these general description of approaches, in the third part, several international projects are presented, which describe in detail the decisions taken in implementing "real" image processing based applications, some of them of almost frigthening magnitude. The contributors of this part were asked to provide a different kind of introduction to the subject than those to the previous two: all of them should discuss a relatively small topic, which, however, should be discussed with much greater detail than the relatively broad overviews of the first two parts.

All the contributions growing out of the workshop came from projects, which had among their aims the immediate applicability of the tools developed within the next 12 - 24 months. As a result they are focusing on corpora not much beyond 20.000 (color) and 100.000 (b/w) images, which are supposed to be stored in resolutions manageable within $\leq$ 5 MB / image (color) and $\leq$ 0.5 MB / image (b/w). The participants of the workshop felt strongly, that this view should be augmented by a description of the rationale behind

the creation of a large scale project for the systematic conversion of a complete archive. The resulting paper, by P. González, describes the considerations which lead to the design of the *Archivo General de Indias* project and the experiences gained during the completed stages. That description is enhanced by a discussion of the strategies selected to make the raw bitmaps accessible via suitable descriptions / transcriptions / keywords. A critical appraisal, which decisions would be made differently after the developments in hardware technology in recent years, augments the value of the description.

The participants of the workshop felt furthermore strongly, that their view described above should be augmented by a description of the techniques used for the handling of images in extremely high resolution. A. Hamber's contribution, dealing with the *Vasari project*, gives a very thorough introduction into the technical problems encountered in handling images of extremely high quality and also explains the economic rationale behind an approach to start on purpose with the highest quality of images available today on prototypical hardware.

As these huge projects both were related to institutions which traditionally collect source material for historical studies, it seemed wise to include also a view on the role images would play in the data archives which traditionally have been of much importance in the considerations of the AHC. E.S. Ore discusses what implications this type of machine readable material should have for the infrastructure of institutions specifically dedicated to Humanities computing.

Image systems which deal with the archiving of pictorial material and manuscript systems have so far generally fairly "shallow" descriptions. At least in art history, moreover, the rely quite frequently on pre-defined terminologies. G. Jaritz and B. Schuh describe how far and why historical research needs a different approach to grasp as much of the internal structure and the content of an image as possible.

Last not least R. Rowland, who acted as host of the workshop at Firenze, describes the considerations which currently prepare the creation of another largescale archival database, to contain large amounts of material from the archives of the inquisition in Portugal. His contribution tries to explore the way in which the more recent developments of image processing can be embedded in the general services required for an archival system.

This series of workshop reports shall attempt to provide a broader basis for thorough discussions of current methodological questions. Their main virtue shall be, that it is produced sufficiently quick to become available, before developments in this field of extremely quick development make them obsolete. We hope we have reached that goal: the editor has to apologize, however, that due to the necessity to bring this volume out in time, proofreading has by neccessity be not as intensive as it should have been. To which another shortcoming is added: none of the persons engaged in the final production of this volume is a native speaker of English; so while we hope to have kept to the standards of what might be described as "International" or "Continental" English, the native speakers among the readers can only be asked for their tolerance.

Göttingen, August 1992