ProofRite: A Paper-Augmented Word Processor

Kevin M. Conroy*, Dave Levin, François Guimbretière
Human-Computer Interaction Lab, University of Maryland,
College Park, A. V. Williams Building, College Park, Maryland 20742, USA

Abstract

Proofreading printed documents is a common step in the writing process, yet no word processing programs offer support for tracking changes made on both paper and digital media. Instead, current systems require users to manually reenter paper-based corrections, a time-consuming and error-prone task. To address this problem, we have developed the first fully distributed Paper Augmented Digital Document (PADD) infrastructure to allow word processors to track paper-based annotations across several common patterns of use. Using this system, we have also developed ProofRite, a word processor that supports digital and physical annotations. With ProofRite, users may annotate document printouts with a digital pen and, upon synchronization, merge the marks captured on paper into their digital document. Marks captured in this way are an integral part of the digital document; they reflow automatically when the surrounding text is modified. In this paper, we demonstrate how to use a PADD-based system to support several common use cases of the writing process, including proofreading, incorporating annotated changes, and integrating feedback from other writers, reviewers, or editors.

Keywords: Human-computer interaction, direct manipulation, proofreading, collaboration, writing process, paper augmented digital documents, annotation, paper-based computing, anoto.

1. Introduction

In *The Myth of the Paperless Office* (Sellen and Harper, 2001), Sellen and Harper observed that, despite the popularity of desktop and tablet computers, paper is still an integral part of the writing process. Many users start the writing process by scribbling ideas, notes, or even entire drafts on paper before creating the first

* Corresponding author. Tel.: 1-301-520-1104.
E-mail addresses: kmconroy@cs.umd.edu (K. Conroy), dml@cs.umd.edu (D. Levin), francois@cs.umd.edu (F. Guimbretière).

digital draft of a document. By making paper an integral part of the writing process, users can benefit from the many affordances of paper documents (Sellen and Harper, 2001). For example, paper documents are easy to navigate by spreading several pages of paper on a table. They are also easy to annotate and accepted in many social contexts, including meetings between co-authors. Unfortunately, modern word processors do not provide tools to easily import handwritten annotations. As a result, users are forced to manually transfer the information gathered on paper back to the computer, a time-consuming and error-prone process.

A wide variety of approaches have been proposed to address this problem. Some system such as FreeStyle (Levine and Ehrlich, 1991) and XLibris (Schilit, et al., 1998) propose emulating paper affordances in the digital medium. These are powerful systems, but they focus on active reading tasks and do not allow the user to change the text of the document. Other systems such as the DigitalDesk (Wellner, 1993) and A-book (Mackay, et al., 2002) propose augmenting paper with digital information provided by a nearby computer. Although they allow user to manipulate content, they require that users be at their desk in order to use the system, thus constraining the mobility user.

More recently, the Paper-Augmented Digital Document (PADD) system proposed a new approach of “cohabitation” (Guimbretiere, 2003). The PADD system allows users to annotate printouts of their documents and incorporates these annotations back into the digital document. In PADD, paper and computer media are considered to be on equal footing: annotations made to either medium are available on the other, and are thus said to cohabitate. PADD sets users free to choose the medium that best fits their current need by allowing them to edit PADD documents on their desktop computer, on paper, or on a tablet PC, in each case knowing that the information they are gathering will always be readily available within the digital world.

In this paper we present the first fully distributed implementation of the PADD infrastructure. Our distributed system allows users to collaborate without having to coordinate their actions. Our implementation also offers a finer control for administrator who can define their own policies for security file access. Our

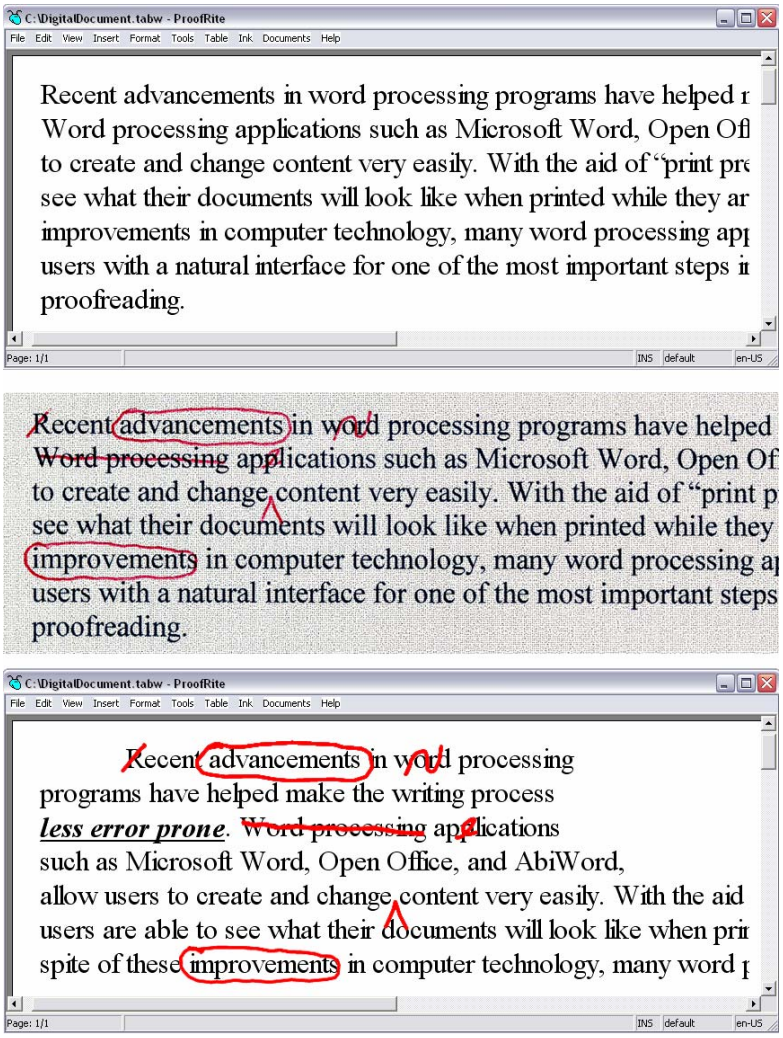

Figure 1 Top: ProofRite document. Middle: The same document printed and annotated on Anoto paper. Bottom: Strokes incorporated and reflowed in ProofRite.

implementation was deploy in several sites, and was the basis for several paper based interfaces including the PapierCraft system (Liao, et al., 2005).

To show how PADD can be used to support proofreading more effectively we implemented ProofRite, an extension of the AbiWord word processor (AbiWord, 2004), which leverage the PADD infrastructure to support paper-digital cohabitation. With ProofRite, users may print their documents and annotate them with a digital pen in the same manner they would with a regular pen. After synchronizing their pen with a computer, the marks captured by the pen can be merged back into the corresponding digital document. The strokes become an

intrinsic part of the ProofRite document as the users' marks will reflow with text around them, similar to the XLibris (Golovchinsky and Denoue, 2002) and Callisto (Bargeron and Moscovich, 2003) reflow systems (Figure 1). With the text and annotations attached, users have the improved freedom to address their notes in the order most convenient for them. ProofRite received very positive feedback during informal demonstrations in our lab and at a demonstration at UIST '04. ProofRite demonstrates the power and usefulness of using a PADD architecture to support cohabitation during the writing process.

2. Previous Work

Various systems have considered the issues surrounding document annotation and proofreading. Some systems digitally augment paper interfaces in an effort to bridge the gap between paper and computers, while others focus on digitally replicating the user experience of annotating on paper.

2.1 Mixed Paper-Digital Systems

Bridging the paper-digital gap has been the focus of a large body of research. To understand how the different systems relate to each other, it is useful to consider three important design dimensions: paper-computer coupling, flexibility, and response time. We classify systems within this design space in

Figure 2. On the horizontal axis is paper-computer coupling: the extent to which the systems couple paper and computers in terms of locale. Systems on the far right side of the horizontal scale require the use of a nearby computer whenever users are working with paper, while system on the left side of the scale can be used in paper-only environments and provide a greater degree of mobility and freedom. The vertical axis represents the level of flexibility users are provided in terms of document content. The topmost systems provide the user with free-form annotation, writing, markup, and drawing capabilities on content created by the users (such as word documents, CAD drawings, and the like). Systems toward the bottom restrict documents to a set of predefined templates or forms. Lastly, the color coding represents the response-time requirements of the system. The white filled ovals indicate that the annotations users make on paper must be processed immediately, while the grey filled ovals indicates a more batch-like process, in which users' strokes can be processed at a later time. In general, tightly coupled

systems require that the system synchronously process strokes as the user makes them

In the upper right quadrant of this design space are systems like DigitalDesk (Wellner, 1993), VideoMosaic (Mackay and Pagani, 1994), Ariel (Mackay, et al., 1995). While using these systems, information captured on paper, such as sketches or data entry, is augmented by computer feedback projected directly onto the paper. For example, a sketch can be copied and pasted to another location, or data entry can be used as input of a digital calculator overlaid on top of the paper documents. Since such systems rely heavily on immediate, synchronous feedback from the computer, they require a tight coupling between computer and paper interactions. These systems are very powerful as they can leverage the flexibility of digital media and offer an extensive array of functionalities. However, they require that paper-based interaction take place near a computer, which limits the systems' availability when compared to pure paper interactions.

In the lower right quadrant of the design space are systems such as the A-book (Mackay, et al., 2002), PaperLink (Arai, et al., 1997), Intelligent paper (Dymetman

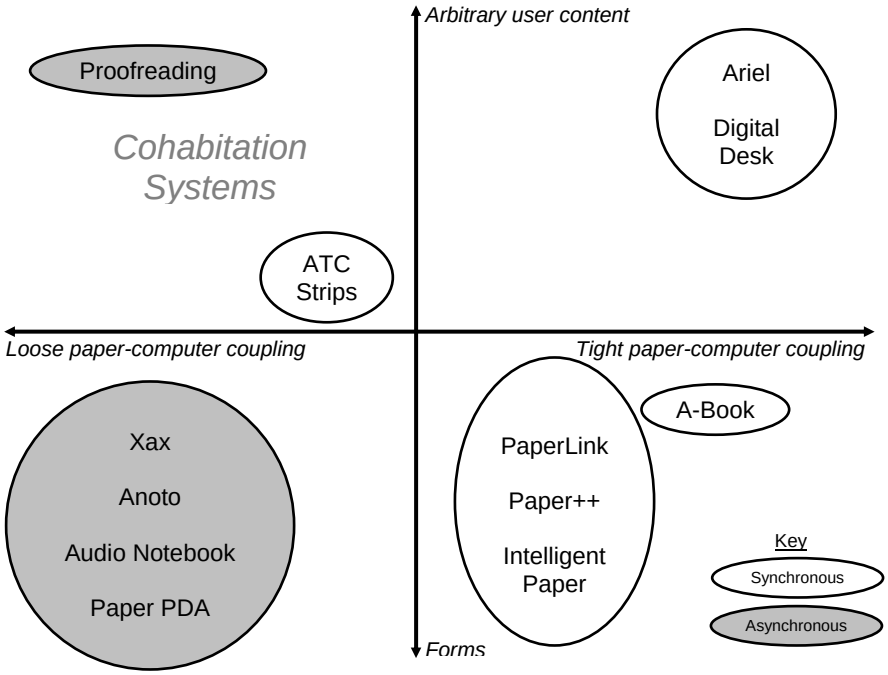

Figure 2 Design space of mixed paper-digital systems.

and Copperman, 1998) and Paper++ (Signer, 2005). These designs use computers as an auxiliary source of information to complement paper. A user of Intelligent paper (Dymetman and Copperman, 1998), for instance, can click on a link printed on paper to play a movie or gain access to more detailed data related to the topic at hand. Though more limited in the kind of feedback they can provide, systems like these are also more flexible with regard to the setting in which they can be used because of a more relaxed paper-computer coupling. These systems can also accommodate a slight delay between the time the information is captured on paper and the feedback is provided on the computer, but they still require the users' strokes to be processed in nearly synchronous fashion. Even if one takes into account the trend of miniaturization of computer hardware, the requirement for a nearby computer terminal can nevertheless restrict the possible patterns of use and be a burden for users when compared with traditional paper usage. For example, users might prefer to use paper in casual situations such as proofreading a paper in a café or while riding public transportation. In such a situation, the availability of a nearby computer might not be desired or even guaranteed.

In the lower left quadrant of the design space in

Figure **2** are systems such as XaX (Johnson, et al., 1993), the Paper PDA (Heiner, et al., 1999), and Anoto (Anoto, 2002) which focus on decoupling the paper and computer interactions. Most of these systems focus on form-filling applications and therefore constrain user inputs to a specific format. For example, the main focus of the Paper PDA is data entry and management of a pre-printed, paper-based calendar application, while the XaX system let users scan data into a database using a specially designed paper form. Each of these systems allows users to interact on paper without a nearby computer, and thus offers many of the affordances of paper: portability, widely varying size, ease of annotation, and so on. These systems also differ in the way they capture information on paper. For example, Paper PDA and XaX are based on scanning, which presents the drawback of requiring an additional step and often limits the quality of the recognition process as the timing used to make each stroke is unknown to the system. The Anoto system (Anoto, 2002) utilizes a digital pen technology which captures strokes as they are performed. This simplifies the capture process and provides timing information, but requires the use of a special pen and papers. It is important

to note that the various techniques presented by each system are fairly independent of the capture technology used. Since most user interactions are performed in the paper world without a connection to a nearby computer, these systems assume an asynchronous, batch-like processing model. We also included the Audio Notebook (Stifelman, et al., 2001) in this quadrant as this system focuses on capturing information and offers limited capabilities toward the manipulation of digital content; though it captures information on blank pages, it does not let users manipulate printed content.

The upper left quadrant of

Figure **2**, which represents cohabitation, has yet to be explored extensively. In this portion of the design space, the coupling between paper and computer is weak (each can be used without the other), yet users might manipulate their own digitally generated content based on the annotations made on printouts. Cohabitation systems were made possible by the recent introduction of digital pen-and-paper systems such as Anoto (Anoto, 2002). Anoto offers a powerful new primitive to system designers: when users write on special paper with a digital pen, their annotations are stored digitally in the pen itself. The pen may then be synchronized (e.g. by plugging it into a USB cradle), thereby uploading the digital strokes to the user's computer. In the background, systems like the PADD infrastructure can capture this information and performs all of the storage and communication that allows users to seamlessly move their work between paper and computer. Cohabitation systems are not limited to batch interaction. For example, a real-time cohabitation system could be used to manage the paper flight strips used by air traffic controllers (Sellen and Harper, 2001); information entered directly on the paper flight strips could be streamed to ATC computers and used to warn controllers of possible conflicts or to propagate their decisions into the system.

2.2 Fully Digital Systems

As device miniaturization and the increased quality of displays have made it possible to build a high resolution computer in a notepad-sized form factor, it is now possible to consider a fully digital system as a substitute for paper. Several systems have explored the issues of fully digital document manipulation, including FreeStyle (Levine and Ehrlich, 1991) and XLibris (Schilit, et al., 1998). The

XLibris system was developed for a domain similar to our work's: a tablet class computer with proofreading and active reading use cases in mind. XLibris allows users to digitally annotate a document while reading and provides several tools to structure the collected notes.

The initial XLibris system only supported documents with a fixed layout, but of course an important aspect of digital document is that they can be dynamically laid out to adapt to users preferences (e.g., when changing the font). After Brush, et al. (Brush, et al., 2001) studied user expectations for digital reflow of annotations, Golovchinsky and Denoue (2002) introduced an extension of Xlibris supporting reflow for active reading tasks on static documents. Shortly thereafter, Bargeron and Moscovich (2003) proposed the Callisto framework for reflowing annotations made to static web pages in a custom version of Internet Explorer. XLibris and Callisto allow users to annotate documents digitally and reposition these annotations as the layout changes, but neither system permits the user to change the document's contents, a significant limitation for the writing process.

Although these systems are very powerful, they suffer from the limited screen real estate of standard tablet PCs, making it difficult to perform tasks that the affordances of paper easily permit, such as spreading the pages of a document out on a table so that the user can see the entire document at once. ProofRite addresses this problem directly by allowing users to capture their annotations on tablet PC and on printouts. After a user annotates a printout and synchronizes their pen, ProofRite inserts their annotations in the corresponding digital documents precisely as if they had been entered digitally on a tablet. Annotations may then reflow with the text even if the original text structure is modified. As our work focuses more on the issue of paper-digital document interaction, we did not pursue an exhaustive evaluation of the more complex reflow issues that previous systems address, but we do show how to perform reflow while allowing users to change the document's content and structure.

2.2.1 Microsoft Word

Mainstream applications are also beginning to take advantage of tablet PCs' features. For example, Microsoft Word 2003 (Microsoft, 2003) introduced a limited interface for making digital ink annotations on tablet PCs. When a user annotates a given piece of text and then alters the text, the annotations only reflow

vertically, not horizontally, making it difficult at times for the user to know with what text their annotations are associated.

In addition to leveraging tablet PC capabilities, Microsoft Word also offers a widely-used feature called track changes. This feature allows users to make changes to the text (and insert comments), providing other authors or reviewers with the ability to review, accept, and/or reject those changes. Version management is not the current focus of our research, and this feature was not available in Abiword when this work was done. We believe that our current implementation strategy will make it easy for our system to take advantage of this feature once it become widely available in Abiword.

3. Motivating Use Cases

Our work was inspired by the work of Sellen and Harper (2001), which shows that paper and computers often coexist in the workspace of knowledge workers. They report that for the typical knowledge worker, "paper is a key part of [the writing] process alongside the computer" (ibid., p. 53) as it supports authoring, review, planning, and various communication needs. In their study, they report that paper was used in conjunction with electronic tools for document modification or editing 89% of the time. Further, within those few companies for which a paperless office is a reality, paper is still used extensively as a transient medium. For instance, Sellen and Harper (2001) report on DanTech, a "paperless" office which uses paper when it is beneficial to do so, such as annotating engineering drawings during review meetings. After the meeting, DanTech's workers quickly transfer the information to the digital world (e.g., by typing up the notes) and the paper is immediately recycled.

Paper and computers complement one another well, so it is not surprising that they have become coexistent. Paper is lightweight, easy to annotate, and comes in wide variety of sizes. Paper documents can easily be rearranged, shared, or flipped-through. Computers excel where paper has drawbacks: paper is difficult and expensive to distribute, large amounts of paper can become very heavy and costly to archive, searching through many printouts for single words can be a burdensome task, and workers generally prefer typing large amounts of text rather than writing by hand (Sellen and Harper, 2001).

These observations were at the core of the design of the original PADD infrastructure (Guimbretiere, 2003), as well as our extensions to it. Our goal is to provide users with a paper-computer cohabitation system making it easy to use paper as a transient medium when paper's affordances are desired. By providing a means to import annotations captured on paper into digital documents, systems like ProofRite that make use of the PADD infrastructure allow their users to seamlessly integrate knowledge captured in the paper world into the digital realm.

To demonstrate the type of usage patterns ProofRite supports, let us consider how Alice and Bob, two knowledge workers from two different universities, might use ProofRite to edit a paper on which they are collaborating. Alice and Bob have been working on a new paper for a while now, and they decide to meet together to discuss Alice's latest version of the draft. Before the meeting, Alice uses ProofRite to print two copies of the current draft that she is planning to review with Bob. When she prints the document, a copy of the digital document is automatically uploaded to the PADD database for later use. During the meeting, Alice presents her changes to Bob and uses her digital pen to annotate her document with notes. Bob also take notes on his copy and agrees to review the document in more detail later that day. Once at her computer, Alice synchronizes her digital pen. The document she used in the meeting automatically appears on the screen downloads her annotations. Later that day, once Bob has finished reviewing the paper and has uploaded his annotations to the PADD database, Alice can download his annotations in the digital document in a different color. With the paper-based annotations imported to her digital document, Alice addresses each annotation in turn. As modifications are made, annotations on other parts of the text reflow, allowing Alice to address each of the marks in the order that is most convenient for her. After all of the annotations have been addressed, Alice is ready to start a new version of the document in the writing process.

To support such activities, our system is divided into two major components. The first component is an implementation of a fully distributed, network-based PADD infrastructure, which provides support for establishing and maintaining the link between digital documents and their printouts. The second part, the ProofRite word processor, is an extension of AbiWord (AbiWord, 2004) that interacts with the PADD infrastructure to fetch the strokes made on paper and then merge them

into the digital documents, as shown in Figure 3. Upon merging, the strokes become part of the document's structure and reflow as the document changes. In the following sections, we present each component of our system in greater detail.

4. The PADD Infrastructure

The PADD infrastructure was proposed in (Guimbretiere, 2003) as a means of establishing and maintaining the relationship between digital documents, their paper printouts, and the annotations collected on the printouts. Our system offers significant improvements over the original prototype. Built on a distributed architecture, our design allows multiple users to annotate multiple printouts of a document without requiring access to the computer which originally printed them. Also, our version offers a simple API to allow client applications to both print a document and retrieve the annotations made on printouts of that document. Finally, the new architecture provides support for easier setup and configuration. These changes are a significant improvement over Guimbretière's implementation (2003) because they allow distributed users to collaborate on multiple documents and versions of these documents at once. Designing an infrastructure that allows users to have access not only to their documents but also to all of their collaborators' annotations presents several technical challenges.

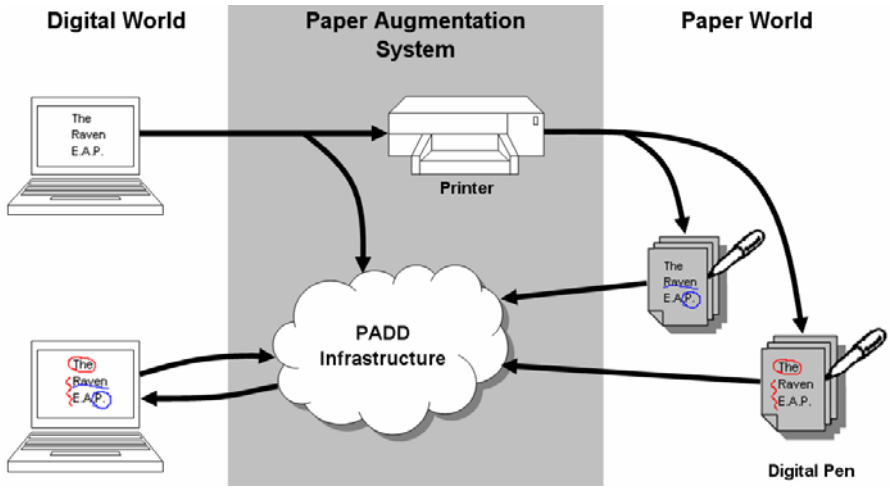

Figure 3 System architecture of ProofRite

4.1 Architecture Overview

We show an overview of the PADD infrastructure's design in Figure 3 and Figure 4. When a user prints a document, the ProofRite application contacts a printer server to determine the unique Anoto page IDs for each piece of paper on which the document will be printed. Using these IDs, ProofRite queries the PADD directory service to retrieve the PADD database managing that page ID range. Once the database managing the page ID range has been found, ProofRite establishes a link between the digital document and the printed paper by uploading a copy of the document to the file repository. After the file has been uploaded, the printout proceeds and the user may begin annotating the printout using their digital pen. Once the user synchronizes their pen with a computer, the stroke collectors will use the PADD directory services to upload the annotations made by the user to the PADD database. A client-side service then downloads a copy of the document from the PADD database and imports annotations from the PADD server, allowing the user to continue the writing process. We now present each component of the architecture in more details.

4.2 PADD Directory Service

The directory service establishes the correspondence between a printout on a given

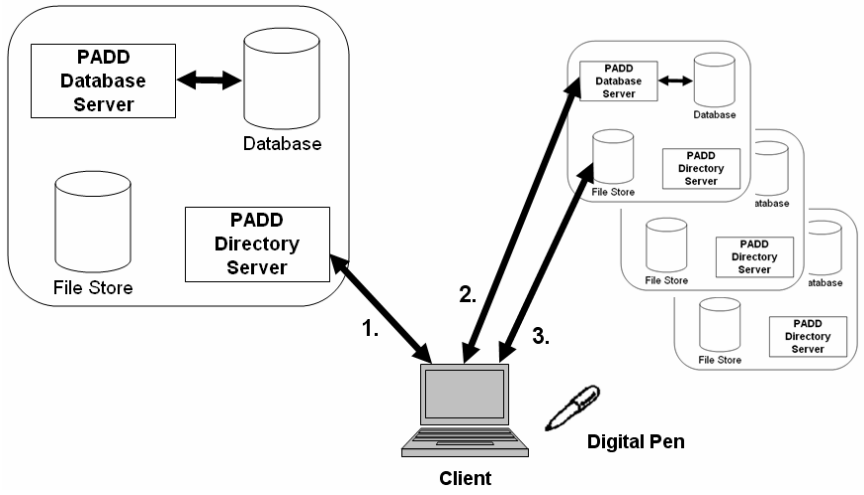

Figure 4 The communication between PADD client and infrastructure. The client (1) contacts the local directory server to determine the appropriate PADD Database, (2) contacts the PADD Database server, and (3) uploads or downloads files from the File Repository.

page and the database that manages that page. The lookup procedure requires that clients know only about the his or her local directory service, similar to other lookup protocols, DNS being perhaps the most well-known example (Mockapetris, 1987).

In the original PADD system, the directory service (called the Paper Look-up Service) was only called at stroke collection time after the pen had been synchronized, but in our system, the directory service is used at the onset of each query to the PADD infrastructure. In doing so, we simplify the user interaction, as users need only know a single local directory service in order to perform any PADD operation. Bootstrapping a computer to access the PADD infrastructure is therefore as simple as knowing the address of its local directory service. A canonical server name such as "padd.*domain-name*" could also be used, making the process even simpler.

4.3 Storing Documents and Annotations

When a document is printed, the page IDs of the printed paper and a copy of the document file are stored in the PADD infrastructure. Additional calibration information (portrait/landscape, scaling, etc.) can also be stored at this time. This information is needed during stroke collection to establish the correspondence between the strokes captured on paper and the corresponding digital material. In the original description of the PADD system (Guimbretiere, 2003), both a snapshot of the document and the calibration information were stored in a single database. Storing the document and calibration information together in this way is restrictive for a distributed environment; users may want different file storage policies for their documents, including access control, encryption, and so on.

To be flexible to users' varying policies, we have found it best to decouple data and meta-data storage. Our system only stores calibration information in a *PADD Database* and uses *file repositories* to store document snapshots. By decoupling the two, we provide users with the freedom to address their storage requirements as they see fit.

4.3.1 PADD Database

The PADD database (PADD DB) stores the association between the paper or physical page IDs and the corresponding pages of a digital document printed on

them. To maintain the link between the physical and digital realms, the PADD DB stores, for each document, the file repository retaining the digital version of the document and, for each printout, the calibration information for this printout (portrait, landscape, scaling, etc.). The database also lists the file repositories storing the strokes made on a given physical page. Intuitively, the PADD DBs are the gatekeepers for document access. To store or retrieve a digital document given a physical printout of that document, users' requests must first go to the PADD database responsible for the printout's page address. This PADD DB can first check the users' permissions before returning the locations of the document and its annotations. Compared with the original PADD implementation (Guimbretiere, 2003), our new design yields a significant increase in communication complexity, but in doing so offers full support for complex patterns of collaboration between users. Because PADD DBs in our design are run on remotely accessible servers, multiple users may make multiple annotations to the same PADD document without having to explicitly share the underlying document. This modification is vitally important to support users distributed in remote locations who each print a separate physical copy of the document but want to share annotations with each other through a single digital copy.

4.3.2 File Repository

Our file repositories hold a snapshot of documents before they are printed as well as captured annotations, stored in *strokes files*, in a place that remote users can access. As explained above, both of these are necessary for performing strokes processing. Improving on Guimbretière's design (2003), the use of file repositories assures that user-specific file storage policies and mechanisms are supported by coupling file repositories with PADD Databases in terms of policy, but decoupling them in terms of mechanism. In other words, PADD DBs are aware of file repositories and know what each repository stores, but are unaware of the specifics of the protocols. This way, each user can choose their own methods of storing files – FTP servers, peer-to-peer file systems, etc. – without requiring any infrastructure changes. Allowing for such flexibility is beneficial in multiple ways. First, deployment is simplified, as the system may make use of existing file repositories, assuming their clients are aware of the protocol. This also means that each user can select the level of security with which they are most comfortable (SFTP versus

FTP, for example) without any infrastructure changes. Security is discussed further in Section 6.1.3.

4.4 Digital Pen and Paper System

Like the original PADD system (Guimbretiere, 2003), our system uses the Anoto digital paper technology (Anoto, 2002). The Anoto system is based on printing a fine pattern of dots on top of normal paper. To the naked eye, this pattern looks like a light grey background, but when observed by the camera on a digital pen such as the Logitech io Digital Pen (Logitech, 2005), it presents an encoded pattern that the pen's camera interprets to determine the specific page ID of the piece of paper that the user is annotating as well as the position on the page. In a sense, the Anoto pattern presents a coordinate system that uniquely identifies an annotation's position on any piece of paper in the system. The pattern space managed by Anoto is very large (more than 2^{48}, or over 280 trillion, pages) which makes a one-to-one correspondence between printed pages and page IDs reasonable for the foreseeable future. For more information on how the Anoto system ensures unique page IDs, see (Anoto, 2002).

There are no inherent traits of the Anoto system that are prevalent in our infrastructure, ensuring that other systems may be easily incorporated in the future. Non-commercially available systems include DataGlyphs (Hecht, 1994), MEMO-PEN (Nabeshima, et al., 1995), and the Microsoft magic pen (Huang, 2004).

4.5 Printing

Like the original PADD system, our system uses a slightly modified[1] HP 5550 and HP 6980 inkjet printers to print user content on top of the pre-printed Anoto paper. This solution is simple and efficient as it only requires users to use Anoto paper in lieu of their normal paper to start using the system. Unfortunately, it also presents the drawback that simple user error while reloading the printer with new paper might cause the PADD database and the printing process to become unsynchronized. We expect that this problem can be easily solved either by introducing a sensor inside the printer to read the page ID used at print time, or by

[1] We removed the black ink cartridge to force the printer to go into "reserve mode." In that mode, the Cyan, Magenta and Yellow colors are used to simulate a black ink which is invisible to the infrared pen camera.

simply having the printer print the Anoto pattern and the users' content at the same time. The latter solution would also simplify calibration.

When multiple users access the same PADD printer, a potential race condition arises as each users' computer is not certain which page ID their documents will be printed on as they only have access to their local page ID history This could be easily remedied by an in-printer sensor or by having the printer print the patterns, as above. It would also be possible to write a simple printer driver that contacts a centralized computer connected to the printer to obtain the page IDs on which it will be printing. This way, the computer from which the user is printing can reliably obtain the proper information to upload to the PADD DB even when multiple print jobs are simultaneously submitted.

4.6 Collecting and Processing Strokes

Once users are done annotating their printouts, they need to synchronize their pen to transfer their strokes to the PADD system. To offer more flexibility, our implementation establishes a clear separation between stroke collection and stroke processing. Strokes are collected by simple *strokes collectors*, which upload the information captured on paper to the PADD infrastructure so that it can be processed by *strokes processors*. We decided to split the original stroke collector into a separate processor and collector because, in multi-user environments, it is often the case that the computer that collects a given collection of strokes is not the same computer that is incorporating them into a document. In the earlier example, Alice may not want to allow Bob to see her annotations, so his strokes collector need only upload them to the PADD DB so that Alice's strokes processor may access them. This simple upload functionality is handled automatically by the strokes collector when it detects that a user has synchronized their pen; it simply observes the pen's metadata (specifically, the page IDs) to determine which PADD DB to contact.

Decoupling strokes collecting and processing also allows processing to be delayed until some point in the future. This proves useful when, for example, Alice does not want the strokes to be processed until her entire team of collaborators has synchronized their annotations.

4.6.1 Strokes Processors

Strokes processors are responsible for merging digital strokes captured on a document printout with the corresponding digital document. The result can be as simple as overlaying an image of the paper-made annotations (as in an Acrobat plug-in (Guimbretiere, 2003)) or as complex as inferring meaning from the users' strokes such as automatically processing proofreading marks. Strokes processors can be invoked by users, perhaps in a word processor, or by some other exogenous event, such as a daily batch process. When invoked, a strokes processor first retrieves all of the annotations made on one or more printouts of the target document. This is done by first using the directory service to find which databases are responsible for managing the document and its strokes, then querying theses databases (as well as the corresponding file stores) to retrieve the strokes files and calibration information.

Once the strokes processor has all of the associated annotations as well as the digital document itself, the strokes processor then performs the corresponding processing. Since the processing is most likely document type-specific (e.g., merging strokes into the digital document), we anticipate many strokes processors to be created for many different document types (e.g., PDF and DOC). We have implemented two strokes processors: an Acrobat plug-in for PDFs and a processor for AbiWord (as described in the following section) for ABI files. As we demonstrate in the following section, by placing more functionality in a strokes processor, the PADD infrastructure can be extended to support usage patterns as complex as those in proofreading with distributed collaborators.

This architectural extension opens the possibility of the development of a diverse ecosystem of strokes processors, thus allowing future developers to focus on creating new ways to user information collected from annotations without having to reinvent a distributed environment to support multiple users. We envision strokes processors as being the predominant means of extending the PADD system to include more advanced functionality.

5. ProofRite: A PADD Word Processor

As found by Sellen and Harper in their study of IMF workers (Sellen and Harper, 2001), the creation and editing of digital documents still relies heavily on paper.

During the document life cycle, users alternate between editing the digital version of the document on their computer, printing the document for convenient review, and merging the paper-based changes back into the digital version of the document. This manual process is error-prone and is often frustrating for users. As each annotation is addressed, the structure of the document might change drastically, making it more difficult for users to establish a correspondence between the information presented on the printout and the digital document on the screen. This problem is made even more difficult if users have to merge the comments from several printouts.

To illustrate how the PADD system can help the proofreading process, we have developed ProofRite, an extension of AbiWord (2004) which collects and re-flow proofreading annotations made printouts of digital document. ProofRite is the first system to allow users to annotate printed documents and import their annotations directly into their digital documents.

5.1 Capturing Annotations

When a user prints a document from the ProofRite word processor, the document is sent to the PADD infrastructure and is printed on paper. The user is then free to annotate the document using any supported digital pen. Once a user has annotated a printed document, synchronizing the digital pen will cause the corresponding

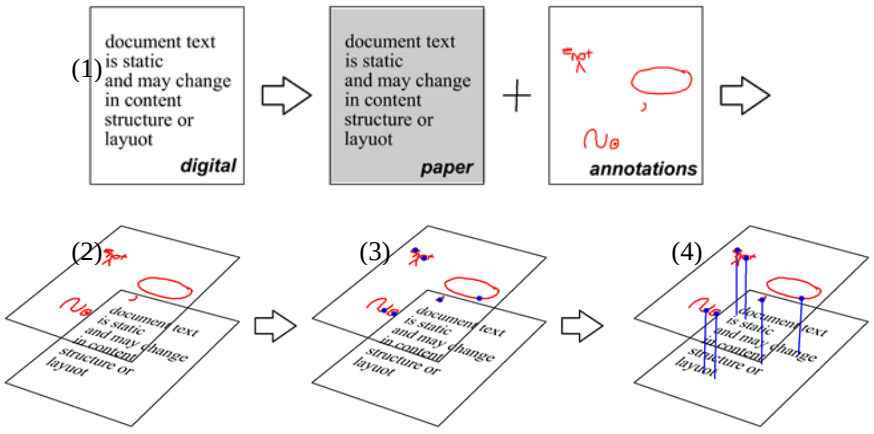

Figure 5 (1) Marks made on paper can be (2) thought of as a separate layer. To synchronize this layer with the digital document, (3) we identify the semantic insertion point of each mark and (4) insert an anchor at that point, creating a multivalent layer of annotations.

document to be opened on the user's computer. Upon user request, ProofRite will then import the annotations made on paper and insert them into the document.

The first step in this process is to translate strokes from paper-space to digital-space, taking into account the page size, scaling, and orientation of the printout. These pieces of information can be retrieved from the PADD infrastructure, which automatically captures them at print time. With this information, ProofRite constructs an affine transform to transform each point of each stroke from the Anoto-paper coordinate system to our editor-specific coordinate system. This step simplifies the anchoring process and makes it easier to deal with annotations from different origins.

To support the increased usage of tablet PCs, ProofRite also allows users to annotate their documents directly on the screen using the stylus. Recovering stroke data from the tablet PC stylus is less complex, as the stroke data is reported in screen coordinates and a simple transformation converts it into the editor-specific coordinate system.

5.2 Associating Annotations with Text

Once the annotation is converted to the editor-specific coordinate system, the next step is to associate each annotation with its corresponding document item (e.g., word, letter, or paragraph). This step guarantees that as the text reflows, each annotation will remain with its meaningful context. This is particularly important for proofreading annotations, as their meaning is dependent upon the location at which they are displayed.

As a simplifying approach, we decided to focus on a subset of ANSI standard proofreading marks (ANSI, 1981) taught in grammar school and used by many professional writers and editors. In particular, we optimized our system to handle the markings shown in Figure 6. Although these strokes do not fully cover the wide range of potential annotations, we believe that this set strikes a good balance between system complexity and usefulness to demonstrate the feasibility of our approach and elicit user feedback.

5.2.1 Anchoring Proofreading Annotations

As shown in Figure 6, we have observed that many single-stroke proofreading annotations have a visually distinct, abrupt change in direction. This change in

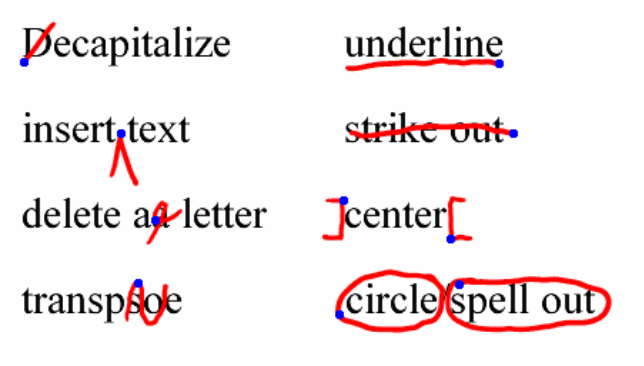

Figure 6 Common proofreading marks and their calculated change in direction (blue dots).

direction denotes the text that is most relevant to the annotation. For instance, the chevron or "insert text" mark changes direction precisely where the text is to be inserted. Based on this observation we used a simple sliding window algorithm to filter out natural tremor and detect abrupt changes in direction. This heuristic correctly identifies the change in direction for single-stroke markings and additionally identifies an anchor point for straight-line annotations that lack a natural change in direction. In Figure 6, we present several examples of the anchor points ProofRite determines for common proofreading annotations.

Once our heuristic has computed the most likely anchor point, we use this location to identify the underlying word in the document layout and insert an intra-document anchor, as proposed by Phelps and Wilensky's (1997), at this location in the document. Initial tests showed that our heuristic works very well for in-line proofreading marks. For instance, if a user crosses out a word, goes to the beginning of the line, and indents it several times, types new text, and then creates a new paragraph, our heuristic correctly preserves the meaning of the cross out mark. Other supported marks include insert, cross- or strike-out, punctuation, transpose, letter deletion, and de-capitalize.

5.2.2 Storing Annotation Data

Previous systems, such as XLibris (Golovchinsky and Denoue, 2002) and Callisto (Bargeron and Moscovich, 2003), have stored annotations made to a document in a separate file or database. This design allows a system to support document annotation on a file that a user may not have permission to modify. The focus of our project, however, is to explore the writing process, thus, we assume that the

users of the ProofRite system have permission to modify the documents that they are annotating. As a result, we store our digital annotations directly into the document, thereby ensuring users that their strokes and documents are mutually available. To store an annotation, we first insert a unique stroke identifier at the contextual anchor point as described above. We then store the corresponding stroke information as a serialized stroke/ink objects in one large block in a data segment at the end of document. We found that storing the strokes inside the document made our system integrate more seamlessly with the AbiWord architecture and make standard functionality such as copy and paste annotations relatively simple to implement.

5.3 Reflowing Annotations

The fundamental challenge of digital reflow is to determine the correct position at which a stroke should be rendered in order to preserve the semantic meaning of the mark. Our solution relies on expanding the word processor's layout engine to recognize the intra-document anchors inserted in the document and use these anchors to reflow the annotations.

When the word processor's layout engine updates the position of our intra-document anchors, we notify the multivalent layer of ink to move the stroke as well, using an affine transformation to calculate the change in position of the anchor. The multivalent layer then refreshes its display and shows the annotation at the new location. This allows the document to be modified in an arbitrarily complex manner while still preserving the meaning of associated annotations. By updating the multivalent layer based on the layout changes that have occurred

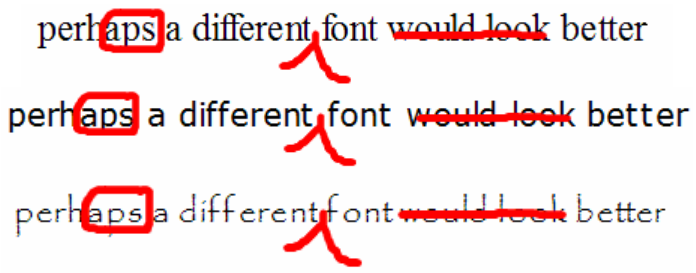

Figure 7 Top: Times New Roman Font, Size 12.Middle: Verdana Font, Size 10. Bottom: Papyrus Font, Size 10. Note that the current implementation does not scale annotations when font size changes, but does scale annotations to account for changes in users' zoom level.

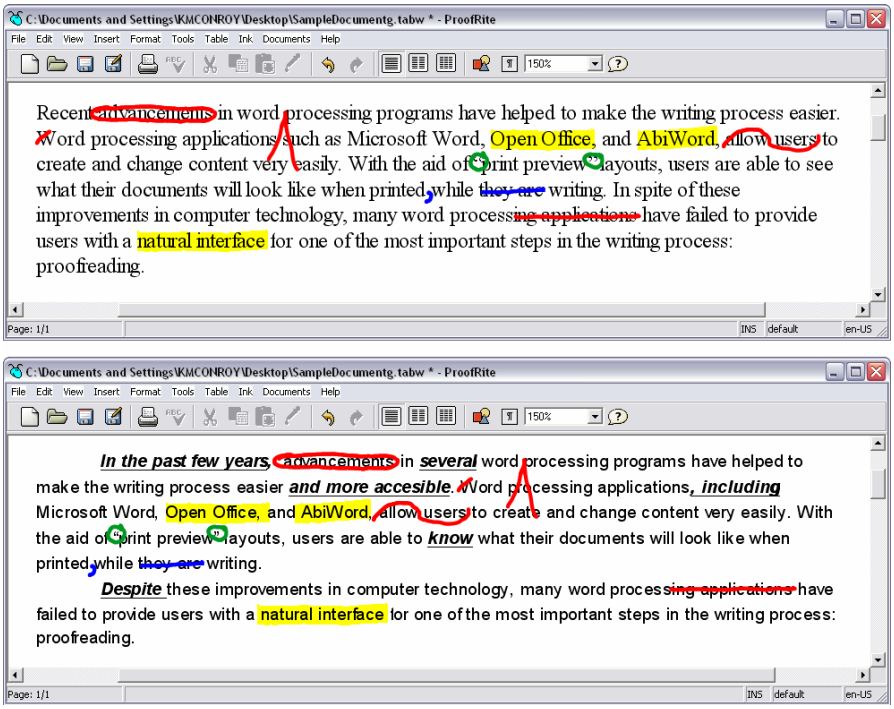

Figure 8 Above: A document with digital annotations. Below: The same document after the user has added text (in underlined-bold-italics for emphasis), deleted some text, created a new paragraph, and changed the font from Times New Roman, size 12 to Arial, size 10.

since the anchor was previously handled, it is possible to reflow in a document with non-static text. For example, Figure 7 demonstrates how various markings reflow as the user changes to several different font faces. Figure 8 demonstrates how annotations reflow when the user changes font faces, styles, and makes structural changes to the document. In each figure, note how annotations reflow with the text to preserve meaning.

5.4 Splitting Annotations

As the document is reflowed, annotations created by users might need to be reformatted to accommodate the new text structure. For example, if a stroke spans several words on the same line and the reflow causes these words to appear on several lines, the stroke mark needs to be reformatted so that the mark does not overextend into the margin or covers the wrong words. To address this issue, we used the solution presented by XLibris (Golovchinsky and Denoue, 2002). When an annotation spans several words, we split each annotation into several smaller annotations so that each partial stroke only spans a single word. This ensures that if

the original text reflows over several lines that the associated annotation can reflow with the corresponding text correctly. Our current system only considers word boundaries (separated by white space), but it could be easily extended to the case of hyphenation boundaries if the paragraph style required it. While our system supports reflow of annotations across several lines, our multi-line reflow is not as advanced as the Callisto (Bargeron and Moscovich, 2003) system. In particular, we do not reformat “circle” markings to show multiple circles on multiple lines of text.

5.5 Rendering Annotations

Natural annotations are rich visual objects that contain a wealth of information. To preserve this, we rendered strokes using anti-aliased, Bezier curve models in the Microsoft Tablet PC API that can accurately model the look-and-feel of human handwriting. This ensures that the digitized annotations collected through the PADD infrastructure seem natural to the user. However, rendering such a stroke can often be processing-intensive and may be prohibitively slow to update the entire multivalent layer of each with each keystroke.

To ensure system responsiveness, ProofRite, like many word processors, uses caching and clipping to only re-render portions of the document that have changed significantly. If a paragraph of text is moved down a line but no changes have been made, then the cached version of the paragraph may be translated down rather than re-rendering the underlying document elements. When this occurs, it is necessary to either translate a cached version of the ink or to redraw the corresponding portion of ink. This helps ensure that the system is able to render a large amount of handwriting on the screen without hindering the users’ ability to interact with the system.

5.6 Multiple Users

To support multiple users, ProofRite permits users to import annotations from the PADD system from a variety of sources. Currently, we render annotations in the ink color of the pen in which it was made. It would be easy to extend the system to filter annotations based on the source of the annotation and to change the color of digital annotations based on the user that made the annotation. It would also be easy to allow users to filter annotations based on various metadata criteria, such as

user/author, date, priority, or other data provided by strokes processors, such as the semantic meaning of the annotation. Thus, it would be possible to have a color-coded view of changes meeting a certain criteria, such as a supervisor's high priority comments or a co-authors notes on formatting.

5.7 Informal User Feedback

We have demonstrated our system to many different classes of users, including children, adults, researchers, students, professionals, during open house events at our lab and during a demonstration session at the UIST'04 conference. Overall the system has been well received. The users have found both the Acrobat plug-in and ProofRite application easy to use. Most users understood how readily use the system without any instructions other than at a high level, such as, "Feel free to make any annotations you wish to the document and then simply plug the pen in." Further, upon seeing a demonstration, many users elaborated on how their personal work patterns could benefit from such functionality. For instance, an elementary school teacher noted that she could use ProofRite to grade homework and automatically have students' grades and corrections available to them as soon as she plugged in her pen. Many users quickly realized how the ability to incorporate multiple users' strokes into a single digital document could facilitate their meetings in which each participant's annotations are usually manually collated. Such a reaction seems to imply that we have been successful in designing a system which closely reflects current patterns of use and will be able to augment current user activities with only minimal changes. This feedback is also clearly in accordance with the direction suggested in [Sellen and Harper, 2001], in which it is made clear that in today's digitally rich environment, a system bridging the gap between the digital and paper world will be well received by users.

6. Discussion and Future Work

Through developing ProofRite we have observed several ways to expand support for more complex use cases via changes to both the PADD infrastructure and the interaction with digital annotations. In this section we discuss these insights, as well as specific proposals for future work.

6.1 PADD Infrastructure

Most of the extensions we have made to the PADD infrastructure have been to increase flexibility in the architecture to address users' needs. At first glance, it might seem that this added complexity might impede easy deployment; however, there are many lessons we can take from the Internet to mitigate this. While it is quite difficult for an individual to establish a web site from scratch, internet service providers (ISPs) provide their users with resources, such as Internet access, storage space for their content, and an always-on infrastructure that make it very simple for users to have an Internet presence. We believe that a parallel structure can be developed for the PADD system. In our case PADD Service Providers (PSPs) will administer the PADD infrastructure for users. They will be responsible for maintaining the directory services, storing documents and annotation on highly accessible and reliable servers, and probably providing strokes processing capabilities, as well. Our system is designed in such a way that, when users purchase PADD-compliant paper, the store from which they purchase it could also be their PSP. In this manner, any necessary page ID distribution remains transparent to the users.

6.1.1 Deploying the Infrastructure

Currently, there are two active PSPs: one in our lab at the University of Maryland, and one in the HCI lab at the University of California, San Diego. We have found deployment to be simple. In fact, we often start temporary PSPs for testing purposes. Our PADD infrastructure is therefore an easy, viable tool for others' future research such as ethnographic studies. We discuss this point further in Section 6.1.6.

For future, wide-scale deployment, paper distribution is a necessary consideration. Similar to IPV6 (http://www.ipv6.org/), it is important to disseminate page IDs to PSPs in such a way as to simplify handling page ranges. In the ideal case, each PSP would have a contiguous range of page addresses meaning that very little state (the first and last page addresses) would be needed by the directory service to route requests. Otherwise, each PSP would need state linear in the number of noncontiguous page ranges they hold as in the case of IPV6.

6.1.2 Extending the Page ID Space

The Anoto paper system assigns a unique page address to each physical sheet. Each address has four 12 bit fields – bookcase, shelf, book, and page – yielding more than 2^{48} sheets of uniquely addressable paper (Anoto, 2002, Phelps and Wilensky, 2000), a number that, although large, is reachable in a matter of years if globally deployed given current page usage trends (Sellen and Harper, 2001). Therefore, over time, it is likely that page addresses will begin to collide. This may prove to be a problem in the long run, as we designed our PADD infrastructure with the underlying assumption of unique, system-wide IDs. Again, looking at the methods currently used in the Internet provides useful insights.

A well-known problem with the design of the Internet is the poor distribution of the first IP addresses. Without tools such as Network Address Translation (NAT) (Egevang, 1994), it is likely that there would be a shortage of IP addresses.[2] Conceptually, NAT hides private addresses behind public addresses, and thus private addresses may be duplicated at many different private networks. This is applicable in the paper world by allowing separate entities to have their own (private) address ranges but requiring the use of "public pages" (pages with public addresses) when collaborating with other institutions. Like NAT, this approach is appealing and requires minimal changes to the system, but it also presents some difficulties that are unique to paper. First, users will need to be aware at print time

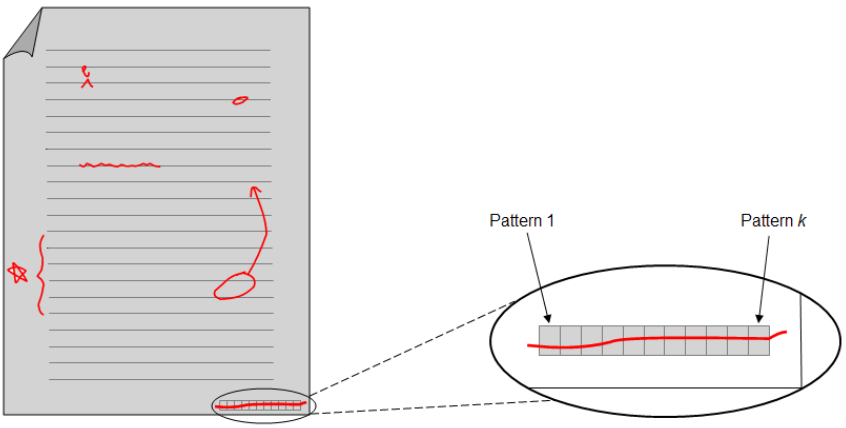

Figure 9 An example key-based method of extending the paper address space.

[2] In fact, NAT is considered one of the fundamental reasons that the transition to IPv6 has not been immediately necessary.

if the document needs to use the public or private part of the address space. Also, if a pen is synchronized in the wrong private domain, collisions could occur (equivalently, one cannot obtain a private IP address in one network and expect to use it in another).

To address this problem we envision printing onto each piece of paper a 'key' similar to the Fly Pen system data strip (LeapFrog, 2005): a small strip consisting of several paper patterns, exemplified in Figure 9. In case of a potential conflict, the user would simply need to mark a line across the key before synchronizing their pen. The strokes collector would use the key information, along with the paper's ID itself, to distinguish between other pages with the same ID. Each of the k patterns in the key can be chosen independently, that is, "with replacement," yielding $(2^{48})^k$ unique keys. In effect, this would increase the number of unique page IDs to $(2^{48})^{k+1}$ (the additional factor of 2^{48} comes from the paper's ID itself). Since most printed documents have some amount of unused space at the margins, there is likely to be sufficient room for such a key. Depending of the pen technology used, the users might use this feature before synchronization or in a more interactive way if their pen can interact with a nearby computer through a wireless connection (such as the Bluetooth feature available on several Anoto pens). However, the problem with this key-based approach is that it detracts away from the seamless transitions (just plug in the pen) between paper and computers that we aim to achieve. Of course, the precise extent to which users are willing to provide additional input cannot be determined without further studies.

6.1.3 Security

As with any multi-user environment, security must be considered. When using the PADD infrastructure, users may want to restrict access to their documents and/or their annotations. Though we cannot anticipate every security need of future users, we believe our distributed design is flexible enough to handle many of them. For example, our infrastructure allows for specifying access permissions with as fine as a per-stroke granularity. Also, a strokes collector could be created that allows anything written in "blue" to be readable by all, but annotations made in "red" pen to only be accessible to the user who wrote them. In other words, users may assign specific security policies on a pen basis (using a "private" pen and a "public" pen). Additionally, access permissions could be calculated by a strokes processor. Using

this approach, users could, for example, draw a circle around text they want to be made visible to other users. One avenue of future work is to explore the trade-offs between the flexibility of security specifications and the ease of paper-based input.

6.1.4 The PADD Infrastructure as a Publish-Subscribe System

While developing our new PADD infrastructure, it became clear that the system can be viewed as the set of publishers (printer drivers, stroke captures modules, stroke processing modules) and subscribers (stroke processing modules and clients such as ProofRite). As a result, we believe that the system can be made more flexible by providing a publish-subscribe infrastructure. Many instances of such systems have been proposed, including the Internet Indirection Infrastructure (*i*3) (Stoica, et al., 2002). Since our design already has a two-tiered structure (i.e., the PSPs and PADD clients), adapting it to other systems will be relatively easy. This extension will make it easy to implement off-line services, support complex pipelining of strokes processing, and implement complex user notification patterns. For instance, an off-line stroke processor could automatically be invoked to begin merging the annotations into the document as soon as they are collected. This approach will significantly reduce the time it takes to open newly annotated documents. Allowing for batch proceeding also offers the possibility to have humans involved in the stroke processing process to increase accuracy, for example. The natural next step in that direction is to allow for several processing steps to be "piped" together, but our system does not yet support such a feature.

6.1.5 Disconnected Operation

By distributing the original infrastructure presented by Guimbretière (2003), we have introduced the constraint that clients be connected to the infrastructure while printing or gathering strokes. While this represents a typical office situation, many potential uses of paper involve being off-line, such as being on a train or airplane. To support users in such situations, we intend to explore how the PADD system can be extended to support off-line mode, in which we cache all operations locally to the user's machine until a network connection becomes available. For instance, stroke processors can cache strokes synchronized during offline operation and batch process them once attached to a network. Disconnected client-side support is an area of future work.

6.1.6 Paper as an Indexing Mechanism

It has been shown that users can glean an efficient semantic structure from the physical distribution of their documents, be it from a file cabinet or even an untidy workspace (Sellen and Harper, 2001). With the PADD infrastructure, users need only a single page of a physical document to have access to the entire digital document. Therefore, users can benefit from the semantic structure of their physical data in the digital world (Ishii and Ullmer, 1997), while still having the conveniences of their digital documents. As noted in the study of the Chocolate-Manufacturing Company (Sellen and Harper, 2001), this can help to improve the utility of document management systems by permitting users to store documents in a shared filing cabinet and easily open the digital version of the document to create new documents such as work orders and contracts from physical access to archived records. In essence, a user's file cabinet can become their file system.

6.1.7 PADD as an Ethnographic Tool

Currently, to study paper-based interactions, ethnographers must survey or observe users, as was done by Sellen and Harper (2001). This data collection method is cumbersome for users and might be imprecise. By its very design the PADD infrastructure is able to record all the interactions performed on paper as well as capture how paper interactions are interleaved with computer-based interactions. With PADD, ethnographers can gather a more precise picture of the differences between paper and digital document interaction. We are therefore considering developing new interfaces to our infrastructure that will make it easy for ethnographers to perform analyses on the patterns that users exhibit while interacting with the PADD system.

6.2 Digital Annotations

Like XLibris (Golovchinsky and Denoue, 2002, Schilit, et al., 1998) and Callisto (Bargeron and Moscovich, 2003), ProofRite was designed to reflow marks with the text that surround them. While our algorithm was sufficient for the purpose of integrating with the PADD infrastructure, it is still not robust enough for everyday life. Here, we discuss what we believe to be some of the most important requirements of any proofreading tool similar to ProofRite.

6.2.1 ProcessingProofreading Marks

Our system only considers single stroke marks with semantics that only pertain to a single line. Many proofreading marks, however, consist of several strokes and can span large portions of the document, such as handwritten text. Other marks, such as circling a group of words, require that the circling mark be resized, or more drastically altered, as the text moves. Both XLibris (Golovchinsky and Denoue, 2002, Schilit, et al., 1998) and Callisto (Bargeron and Moscovich, 2003) have started to explore these questions (particularly with circled text), and we intend to adapt their techniques to our system in the future. One promising direction of research will be to take into account underlying text attributes to help resolve possible ambiguities in the marking. For example a valid capitalization mark can only be valid on top of a lower case letter. We believe that such an approach will significantly improve the recognition accuracy of our system.

6.2.2 Margin Notes and Drawings

From our initial tests, it is clear that a practical proofreading tool must smoothly handle both margin notes and drawings. Many annotations made in margins capture more complex ideas such as comments, suggestions, or notes about the document's content. A successful system will need to address both the segmentation of each annotation on the page and also the display of the annotation on the digital documents. While several systems such as XLibris (Golovchinsky and Denoue, 2002) have started to explore the former problem, annotations visualization in presence of reflow is still an open problem. This is especially true for annotations captured on paper, where users have a great deal of freedom with respect to the orientation of their hand writing. Furthermore, it is sometimes the case that annotation can overflow on top of the printed text or from one page to another. Our current system set aside these important problems, as our main focus was to investigate the document life cycle, but we intend to explore this problem in the future.

6.2.3 Auto-Correction

During our initial tests of ProofRite, we observed that it would be very useful to automatically apply annotated corrections to the digital document. This would allow users to make either paper-based or Tablet PC based annotations and apply

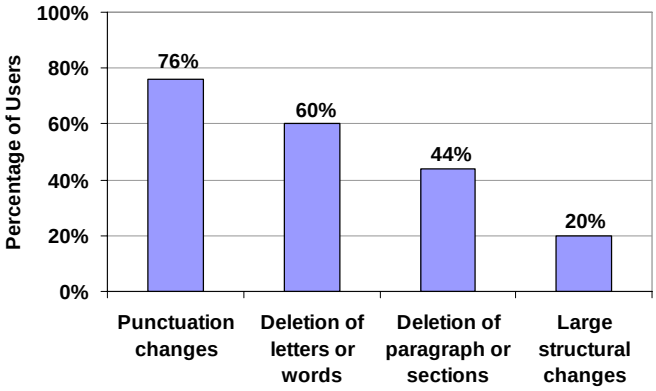

Figure 10 The types of auto-corrections users would allow.

the changes automatically. Beyond the obvious technical difficulty of auto-correction, such as a reliable recognition and anchoring, we believe that user acceptance of such a system will depend on their level of comfort with the auto-correction process. For example, while users might allow automatic processing of punctuation marks, they might be uncomfortable with large scale changes.

To better understand this problem we conducted an informal survey during a demonstration of our system at the UIST'04 conference. During the demonstration, we asked users what types of auto-corrections they would allow their word processor to make. The types of auto-corrections that users would allow are listed in Figure 10 in increasing scope, from small (punctuation changes), to medium (deletion of paragraphs), to large (large structural changes). It is evident that users are more willing to allow for smaller auto-corrections than large. The trend seems to indicate that users would want their word processor to correct the simple, straightforward, and perhaps tedious changes while users would prefer to make larger, more complex, and potentially creative or stylistic changes on their own. We contend that once presented with reliable correction mechanisms, users' willingness will increase. This is a very interesting area for future work.

6.2.4 Version Management

Yet another crucial stroke processing function is the incorporation of strokes in version management tools. From our knowledge worker example in Section 3, if Alice incorporates her strokes and begins editing the document before Bob uploads his annotations, then it may not be possible to incorporate Bob's comments into the

same document. A naïve solution would be to require Alice to not begin editing until all of her collaborators have uploaded all of their annotations. Users are more likely to want to start editing and to then want to be able to incorporate strokes made on printouts of older versions into the up-to-date version.

In the current version of ProofRite, the strokes are processed against the version of the digital document current at printing time. Since our implementation is compatible with the underlying word processor framework, it should be easy to extend it to take advantage of word processor-based version management systems.

7. Conclusion

In this paper, we have addressed the gap that exists between the usage of paper and computers with respect to a common paper-computer activity: proofreading. We reported our design of a fully distributed Paper Augmented Digital Documents infrastructure and introduced the ProofRite word processor. ProofRite allows users to annotate a document printout on paper and later incorporate paper-based annotations into their digital documents. Annotations captured on paper are given the benefits of digital reflow, allowing users to continue the writing process without having to lose or manually enter annotations between revisions. Furthermore, we described how annotations made by multiple users to multiple copies of a PADD-compliant document can be integrated into a single digital copy. Our system received very positive feedback during informal demonstrations in our lab and during a demonstration at the UIST'04 conference. We believe that our system will provide a transparent bridge between the paper and digital world and can also be used as an effective ethnographic tool to further understand how people integrate paper in their everyday life.

8. Acknowledgements

The authors wish to thank Dom Lachowicz, Marc Maurer, and the rest of the AbiWord development team for technical assistance; Jim Hollan and Ron Stanonik for their assistance in deploying PADD at UCSD; Samantha Conroy for comments on drafts of this paper; and Chunyuan Liao for developing the PADD printer driver software.

9. References

Abiword (2004), http://www.abiword.com, http://www.abiword.com.

Anoto (2002), Development Guide for Service Enabled by Anoto Functionality, http://www.anoto.com.

Ansi (1981), *American national standard proof corrections*. 1981: American National Standards Institute.

Arai, T., D. Aust and S. E. Hudson (1997), PaperLink: a technique for hyperlinking from real paper to electronic content. *Proceedings of CHI'97*, pp. 327 - 334.

Bargeron, D. and T. Moscovich (2003), Reflowing digital ink annotations. *Proceedings of CHI'03*, pp. 385 - 393.

Brush, A. J. B., D. Bargeron, A. Gupta and J. J. Cadiz (2001), Robust annotation positioning in digital documents. *Proceedings of CHI*, pp. 285 - 292.

Dymetman, M. and M. Copperman (1998), Intelligent Paper. *Proceedings of EP'98*, pp. 392 - 406.

Egevang, K. A. F., P. (1994), The IP Network Address Translator (NAT), RFC 1631, May 1994.

Golovchinsky, G. and L. Denoue (2002), Moving markup: repositioning freeform annotations. *Proceedings of UIST'02*, pp. 21 - 30.

Guimbretiere, F. (2003), Paper Augmented Digital Documents. *Proceedings of UIST'03*, pp. 51 - 60.

Hecht, D. L. (1994), Embedded Data Glyph Technology for Hardcopy Digital Documents. *Proceedings of SPIE Color Hard Copy and Graphic Arts III*, pp. 341 - 352.

Heiner, J. M., S. E. Hudson and K. Tanaka (1999), Linking and messaging from real paper in the Paper PDA. *Proceedings of UIST'99*, pp. 179 - 186.

Huang, G. (2004), Microsoft's Magic Pen. MIT Technology Review, May 2004.

Ishii, H. and B. Ullmer (1997), Tangible bits: towards seamless interfaces between people, bits and atoms. *Proceedings of CHI'97*, pp. 234-241.

Johnson, W., H. Jellinek, J. Leigh Klotz, R. Rao and S. K. Card (1993), Bridging the paper and electronic worlds: the paper user interface. *Proceedings of CHI'93*, pp. 507 - 512.

Leapfrog (2005), Fly Pen, http://www.leapfrog.com.

Levine, S. R. and S. F. Ehrlich (1991), The Freestyle System: A Design Perspective, in *Human-Machine Interactive Systems*, Klinger, A., Editor. 1991. pp. 3-21.

Liao, C., F. Guimbretière and K. Hinckley (2005), PapierCraft: a command system for interactive paper. *Proceedings of UIST'05*, pp. 241 - 244.

Logitech (2005), IO digital pen. (http://www.logitech.com).

Mackay, W. and D. Pagani (1994), Video mosaic: laying out time in a physical space. *Proceedings of MM'94*, pp. 165 - 172.

Mackay, W. E., D. S. Pagani, L. Faber, B. Inwood, P. Launiainen, L. Brenta and V. Pouzol (1995), Ariel: augmenting paper engineering drawings. *Proceedings of CHI'95*, pp. 421 - 422.

Mackay, W. E., G. Pothier, C. Letondal, K. Bøegh and H. E. Sørensen (2002), The missing link: augmenting biology laboratory notebooks. *Proceedings of UIST'02*, pp. 41 - 50.

Microsoft (2003), http://www.microsoft.com/word.

Mockapetris, P. (1987), Domain Names - Concepts and Facilities, RFC 1034, ISI, November 1987.

Nabeshima, S., S. Yamamoto, K. Agusa and T. Taguchi (1995), MEMO-PEN: a new input device. *Proceedings of CHI'95*, pp. 256 - 257.

Phelps, T. and R. Wilensky (2000), Robust intra-document locations. Computer Networks, 2000. **33**(1-6): pp 105 - 118.

Phelps, T. A. and R. Wilensky (1997), Multivalent Annotations. *Proceedings of ECDL'97*, pp. 287 - 303.

Schilit, B. N., G. Golovchinsky and M. N. Price (1998), Beyond paper: supporting active reading with free form digital ink annotations. *Proceedings of CHI'98*, pp. 249 - 256.

Sellen, A. J. and R. H. R. Harper (2001), *The Myth of the Paperless Office*. 1st ed. 2001: MIT press.

Signer, B. (2005), Fundamental Concepts for Interactive Paper and Cross-Media Information Spaces (ETH No. 16218), PhD thesis, Swiss Federal Institute of Technology, Zurich. 2005

Stifelman, L., B. Arons and C. Schmandt (2001), The audio notebook: paper and pen interaction with structured speech. *Proceedings of CHI'01*, pp. 182 - 189.

Stoica, I., D. Adkins, S. Zhaung, S. Shenker and S. Surana (2002), Internet indirection infrastructure. *Proceedings of SIGCOMM'02*, pp. 73-86.

Wellner, P. (1993), Interacting with paper on the DigitalDesk. Communications of the ACM, 1993. **36**(7): pp 87 - 96.