2. Collaborative writing

2.1 Why the interest in collaborative writing software?

Two trends have provided the impetus for the recent interest in the development of computer support for collaborative writing: collaborative writing is becoming increasingly common both in the workplace and in the classroom (Duin, 1991); and writers are relying more and more on computer technology to support their writing practices (Dorner, 1992).

Although published work would seem to indicate that most writing is produced by a single author, a belief reflected in much of our talk about writing, surveys indicate that collaborative writing occurs far more frequently than most people think it does—in many workplaces, in diverse occupations, involving a wide range of writing activities (Ede & Lunsford, 1990; Couture & Rymer, 1991; Beck, 1993). For example, Ede and Lunsford (1990) found that 87 percent of their survey respondents, approximately 700 members from seven large professional organizations in the United States, reported writing as members of a group or team at some time on the job. Some of these people collaborated on almost every document they produced in the workplace. Collaborative documents ran the gamut of length and formality from memos to published books.

Collaborative writing has also been gaining in popularity as a method of instruction at all levels of the educational system over the last ten years (Forman, 1992) for a variety of reasons. As well as being considered socially beneficial, collaborative writing and other collaborative activities may benefit students cognitively because they encourage students to reason overtly about their mental activities (Brown & Campione, 1990). Further, although schools primarily view intelligence as an individual's possession, cognitive research on *distributed cognition* (see section 2.2.2) is expanding the school's notion of intelligence, where it resides and how to foster it (e.g., Brown & Campione, 1990; Scardamalia & Bereiter, 1994). Finally, collaborative writing experience appears to be valuable preparation for future employment, given the above findings.

2.2 Problems with computer-supported collaborative writing tools

As reliance on technology grows in the workplace and in the schools, it becomes even more important for system designers to be aware of occupational and educational practices so that they can provide appropriate support to help people work and learn more effectively. Despite the general enthusiasm for the word processor, other writing tools, such as grammar checkers, are not widely used by writers (Rimmershaw, 1992). Collaborative writing tools are particularly underused. So, although the impact of computer technology on writing has been significant, it has not been as influential and helpful as it promises to be, especially for collaborative writers. The lack of success of many computer writing tools among word processor converts indicates that there are some serious design flaws in much of the writing software. I will review some research that suggests where the main problems lie in the currently available collaborative writing software.

Attempts to provide collaborative writers with computer support began more than twenty years ago and development has received a marked increase in attention over the last ten years, but synchronous, remote on-line collaborative writing remains a rarity. Investigations of writing in the workplace indicate that collaborative writers use the computer primarily for: email communication; sharing texts on-line; sharing templates, style sheets etc.; and retrieving information from databases (Van Pelt & Gillam, 1991). When authors do use available collaborative technology to write together, it is usually out of interest in exploring new media, rather than because they believe that the system will aid their collaborative writing (Newman & Newman, 1992). The low uptake of collaborative writing software points to an obvious need to rethink the development of collaborative writing tools to ensure that the next generation of software will be embraced by joint authors.

Why has collaborative writing software been so pointedly ignored by collaborative writers? Given the size and diversity of the collaborative writing population, and individual writers' increasing reliance on the word processor, the reasons appear to be associated with the software itself, rather than with the users. Despite the recent proliferation of group software, most attempts to use it outside of experimental situations fail because collaborative writing is not well supported by current computer-based aids (Duin, 1991). Many of these writing aids embody assumptions about collaborative writing that do not necessarily reflect the reality of the collaborative experience, forcing writers into some practices that are at worst harmful, at best useless (Rimmershaw, 1992), demand too much effort from the user (Sharples, Plowman & Goodlet, 1993) and often have unexpected side effects (Beck, 1992).

Duin's (1991) review of collaborative writing software indicates that much of it has primarily focused on providing on-line access to other writers and their writing, rather than providing writing tools. In Posner's (1991) review of six computer-supported writing systems (Aspects, GROVE, ForComment, PREP, Quilt and ShrEdit), she observed that none of the systems comprehensively supported collaborative writing, and that almost none of them provided what she considered "good" support for thirteen requirements for collaborative writing that she identified. Beck (1992) criticized three existing collaborative writing software systems (ShrEdit, PREP, and CoAuthor) on the grounds that they limit the user to a single way of writing collaboratively. Sharples et al. (1993) found that two existing software systems designed to help collaborative writers (The Coordinator and Quilt) require too much effort on the user's part, and are not well integrated with computer practices. Despite the fact that the main thrust of most current CSCW is to provide more flexible systems to support different approaches to collaborative writing (e.g., Neuwirth, Kaufer, Chandhok, & Morris, 1994), Sharples et al. (1993) discovered in their evaluation of four systems (ShrEdit, StorySpace, PREP, and MUCH) that few of the systems are meeting their stated goals, and they question the issues these systems address.

Although the central issue in studying computers and writing is finding the relationship between the writing process and technology designed to support it (Holt, 1992), these software reviews indicate that few collaborative writing software designers appear to have a solid understanding of this relationship. There are four significant and interrelated problems that have hindered them in this respect: the lack of a common definition of collaborative writing; the need for a model of collaborative writing; the paucity of research on collaborative writing; and the difficulty of testing collaborative writing software. I will discuss each one, with reference to its impact on the development of collaborative writing software.

2.2.1 A matter of definition

There is no single, commonly accepted definition of collaborative writing (Harris, 1994). Proposed definitions are often vague, because collaborative writing is difficult to describe since its boundaries are fuzzy. In order to get as much information as they can, researchers who conduct surveys of collaborative writing usually choose to make their definition broad, thus encouraging respondents to discuss approaches to writing which might not be perceived as methods of collaborative writing by all people. For this reason, Ede and Lunsford (1990) used the following definition in their survey about writing in the workplace: "any writing done in collaboration with one or more persons" (p. 15).

Rimmershaw (1992), on the other hand, generated her all-encompassing (and poetic) definition of collaborative writing *after* conducting her interviews of academic collaborative writers, because she noticed how widely their personal definitions varied:

any piece of writing,
published or unpublished,
ascribed or anonymous,
to which more than one person has contributed,
whether or not they grasped a pen,
tapped a keyboard,
or shuffled a mouse. (p. 16)

Many researchers believe it is important to make a distinction between at least two types of collaborative writing. Van Pelt and Gillam (1991), for example, distinguish between *team-work collaboration* in which an author receives input from a team, but retains authority over the document, and *shared-document collaboration* in which authors share responsibility for all important decisions made during the writing of the document. Shared-document collaboration is closer to what most people think of when they refer to collaborative writing, and is rarer than team work. Team-work collaboration encompasses almost all writing, or at least most published writing. Although other researchers use different terms to describe these two types of collaborative writing, the distinction made is similar (e.g., Couture & Rymer, 1991).

Acknowledging the fact that most writers do not work entirely alone emphasizes the ubiquity of collaborative writing, despite the fact that it is often not accredited in the final product, thus revealing a practice that is "hidden in plain sight" (Ede & Lunsford 1990). However, there are two risks Ede and Lunsford (1990) identify that may result from defining collaborative writing in a broad way. First, the definition may fail to distinguish writing from other intellectual activities, because it is difficult to pinpoint where working together ends and writing together begins. For software developers of computer support for collaborative writing, however, distinguishing writing from other intellectual activities might not be an important concern, but rather, what is reasonable and possible to support will drive development. Second, a broad definition may conflate individual writing and collaborative writing because the boundary between writing alone and writing collaboratively is not clear. However, a good collaborative writing tool should support both individual and group writing, for two reasons: first, such systems will allow writers to write alone or with others, without having to switch tools; second, Posner (1991) has identified the "single-writer strategy" (see section 2.3) as one of the ways in which collaborative groups write.

Interestingly, although the definition of collaborative writing forms a significant part of the discussion in the survey and literary research on collaborative writing practices, one never encounters an explicit definition of it in the collaborative writing software research. The issue of definition is not irrelevant for software designers, however. Although a broad definition should have a positive, rather than a negative, influence on software development, the lack of a common definition of collaborative writing may be affecting collaborative writing systems in detrimental ways. First, some designers only look at one aspect of collaborative writing, rather than at the whole picture. Beck (1992) found that although collaborative writing is a complex task that can be carried out in a myriad of ways, most current collaborative writing software developers have implemented a single view of what collaborative writing is and what constitutes support for collaborative writing, thus restricting users unnecessarily. Beck believes that a broader view of what factors might affect collaborative writing will lead to a better understanding of what kinds of computer aid joint authors could benefit from. Second, erroneous conclusions about what types of support collaborative writers need may be drawn from incorrect assumptions about what is meant by collaborative writing. Couture and Rymer (1991) point out that the loose way in which the term *collaborative writing* is used may be causing some researchers to overestimate the amount of shared-document collaboration that is actually being done in the workplace. In their survey of 400 professionals from a wide range of organizations, they found that although 78 percent of respondents said they *sometimes* or more often get feedback on their writing, only 24 percent contributed to a multi-authored document *sometimes*, *often*, or *very often*. Overestimation of shared-document collaboration seems to have been the major contributing factor to Sharples et al.'s (1993) finding that some of the features that are popularly offered by current software are not in high demand by collaborative writers. These features, such as simultaneous editing capabilities, are mainly ones associated with shared-document collaboration. Although an ideal system will support all collaborative writing practices, Sharples et al. (1993) recommend working from a principle of supporting common existing practices, to ensure a useful tool for collaborative writers.

If software designers do not pay attention to how collaborative writing is defined, they may make assumptions about what collaborative writing practices need to be supported on the basis of their own, possibly narrow, views of what collaborative writing is. To avoid producing useless or underused tools, designers need to be aware of what collaborative writers, researchers, and they themselves mean when they use the term *collaborative writing*.

2.2.2 Towards a sociocognitive model of writing

The second contributing factor to the problems surrounding collaborative writing systems is the absence of a comprehensive *model* of collaborative writing. Sharples (1991) believes that many of the problems associated with collaborative writing systems may largely be due to the fact that none of these systems is founded on a well-formulated model of the writing process, collaborative or otherwise. As Sharples and Pemberton (1992) point out, a writing tool is constrained by the assumptions and limitations of the model, whether explicit or implicit, on which it is based. Although a common model of collaborative writing has yet to be established, cognitive research on writing provides a foundation for such a model.

The scientific study of writing is little more than twenty years old. Although there has been a great deal of research interest since the early 1970's (Hayes & Flower, 1987), writing research is still relatively new. Beginning in the 1970's, writing researchers began to challenge commonly held assumptions about writing by examining the processes writers actually went through in their attempts to translate their thoughts into writing. There was a major shift in writing research away from the analysis of the quality of the written product, towards an understanding of the process of writing, and the relationship between this process and the written product (Cochran-Smith, 1991). Previously, writing models were based on the end product—the text. Two such product-based models that dominated the "folk theory" of writing have been largely debunked by cognitive research: the muse-inspired writing of the professional artist-writer, and the stage model of writing taught by pedagogues according to the cookbook method (Flower & Hayes, 1980). In contrast, cognitive researchers today view writing as a dynamic process. Flower and Hayes (1980) describe writers in the act as thinkers on "full-time cognitive overload" (p. 33), juggling knowledge, language, and rhetorical constraints as they try to achieve their writing goals.

Flower and Hayes's (1980) research on writing has been very influential, despite several limitations (see Hartley, 1991). They investigated the cognitive processes of writing using a technique called *protocol analysis*. Protocol analysis involves asking and prompting subjects to externalize their thoughts while they are performing a cognitive task, in this case, writing. The data that is collected is the verbatim report of what the subjects said (including pauses, etc.), along with the text and notes they wrote during the experiment. The transcript itself is the protocol. The protocol and writing are then examined to find out what kinds of cognitive processes appear to have been used by the writer, and a model of these processes is inferred. The writing model that Hayes and Flower (1987) developed from their work has been widely used by researchers and educators.

Flower and Hayes's research suggests that there are several critical features of the writing process: writing is goal-directed; these goals are hierarchically organized; and writers accomplish their goals using three major processes. In their 1987 paper, they refer to these processes as *planning*, *sentence generation*, and *revision*.[4] Planning involves the retrieval and shaping of knowledge to fit the textual and audience constraints. Sentence generation is what we often think of as writing—putting ideas on paper in coherent sentences. During revision, the writer evaluates and attempts to improve the draft. These processes are similar to the three components (pre-writing, writing, and re-writing) of the stage model, but what distinguishes their model of writing is the recognition that these processes are interleaved—sometimes iteratively, sometimes recursively. The interaction of these processes provides some explanation for the variety of strategies used by writers, as well as for certain problems writers encounter as they go about their task.

Another influential model, particularly in the educational domain, is Bereiter and Scardamalia's (1987), which addresses one of the limitations of the Flower and Hayes model—it accounts for the difference between novice and expert writers. Rather than proposing a single writing model, they suggest that expert writers actually have access to an additional composing process. Their *knowledge telling* model, which is similar to Flower and Hayes's model, accounts for a process that relies mainly on skills gained in everyday interaction, whereas their *knowledge transforming* model accounts for a more studied ability that involves deliberate control over parts of the writing process that are not attended to in knowledge telling. While novice writers rely only on knowledge telling, expert writers have access to both writing processes.

For researchers interested in collaborative writing, however, there is a serious limitation to these cognitive models: they describe only the single writer writing alone (Hartley, 1991). There exist, as yet, no widely accepted models of collaborative writing, cognitive or otherwise, although attempts to fill this gap are being made (e.g., Rose, 1994). My experience leads me to believe (and this is implicit in much of the computer-supported collaborative writing literature), that the single-writer model applies equally well to multiple writers in many respects. People tend to bring their methods of doing things as individuals to their work as part of a group. When people write together, however, the cognitive processes of writing cannot be considered apart from the social processes of collaboration (Sharples et al., 1993). The importance of this point is highlighted by the evolving perception in cognitive psychology that cognitions are situated and distributed, and that social and other situational factors do not simply affect cognition, but should be treated as cognitions (Salomon, 1993). In this constructivist point of view, intelligence is regarded as something which is distributed among the participants and the technologies they use to accomplish a given task in a particular environment. Since the distributed cognitive abilities may be greater than the sum of the individual parts, the system must be examined as a whole, rather than simply looking at components in isolation (Salomon, 1993). In addition to the primary interest in how cognitions interact, one focus of much of this research is on developing enabling technologies that can be embedded within a social context to facilitate collaborative work that is aimed at advancing knowledge (e.g., Scardamalia and Bereiter's (1994) *knowledge-building society*; Brown and Campione's (1990) *community of learners*). Certainly, such a goal is (or should be) shared by developers of collaborative writing software.

Although no well-defined model of distributed cognition has yet been developed, cognitive models provide software developers with important information about the mental processes involved in writing, and research on distributed cognition presents a broader view of how people think. Using these theoretical underpinnings as a starting point will make clearer the assumptions and limitations of the tools developed, allowing better evaluation and subsequent development.

---

[4] In their earlier work, they called these processes *planning*, *translation*, and *reviewing* (Flower and Hayes, 1980). I do not know why they chose to change their terminology.

2.2.3 Collaborative writing research

Beck (1992) attributes problems with collaborative writing software primarily to the general lack of research on collaborative writing. The main reason for this lack is the relative recency of research interest on collaborative writing. Although collaborative writing is currently the subject of much attention in a wide variety of disciplines, research on this topic is even newer than cognitive writing research; it was not until the period of 1982–1987 that academics began to seriously investigate collaborative writing (Batschelet, Karis & Trzyna, 1991). I will briefly summarize the current state of collaborative writing research, then suggest the next step in the research agenda.

Although the research interest is new, writing together is not a recent phenomenon (Sharples, 1993). Surveys on collaborative writing (e.g., Beck, 1993; Couture & Rymer, 1991; Ede & Lunsford, 1990) have actually brought to light (and thereby given credibility to) a widespread practice that has been largely ignored, not only by researchers, but also by society in general. Interviews (e.g., Rimmershaw, 1992; Posner, 1991; Ede & Lunsford, 1990) have provided further insight into the complex ways in which collaborative writers carry out their writing tasks. However, although this body of research has contributed information about general trends involved in collaborative writing practices, the fact that collaborative writing research is relatively new and small, and that the object of investigation is immense, has led to more questions being raised than answered about collaborative writing. Now that the general nature of collaborative writing has been explored, a different type of research is required to find out how the writing process is adapted and negotiated by collaborative writers in order to successfully write together: the case study. Case studies supply more specific information about practices than is generally gained in surveys and interviews. Since social interactions among collaborative writers and available technological support will affect the writing process, studying collaborative writing groups at work is the best way to discover their actual practices. Unlike writing alone, the very nature of collaborative writing forces writers to externalize their writing processes to a greater degree as they communicate about their project, thus providing the researcher with explicit information on the writing process (Plowman, 1993). Also, case studies may reveal details about collaborative writing that surveys and interviews miss because the respondents do not realize the importance of them, and therefore, fail to report them.

There are several drawbacks to conducting case studies. They are both time-consuming and labour-intensive (Sharples et al., 1993). In addition, a large number of studies is required to ensure that a full range of collaborative writing practices is explored, because the context and the purpose of the writing, as well as the make-up of the group, will have a significant impact on the strategies used. However, case studies promise to provide more detailed information about collaborative writing practices which software designers can better incorporate into collaborative writing systems.

2.2.4 Testing collaborative writing software

Duin (1991) suggests that software developers have underestimated the difficulty of evaluating collaborative writing software. Baecker, Nastos, Posner and Mawby (1993), Sharples (1992), and Duin (1991) all agree that *user-centred iterative design* is the best way to test collaborative writing software. *User-centred* implies the prototype is based on behavioural research. The iterative design consists of an iterative cycle of design, implementation, and evaluation by collaborative writers, which is repeated until a satisfactory product is produced (Baecker et al., 1993).

Reference to Baecker et al.'s (1993) work demonstrates the efficacy of the user-centred iterative design process in eliminating some of the problems associated with collaborative writing software design. After reviewing the usability studies at each stage in the development of their system, SASSE, Baecker et al. were forced to seriously reevaluate the types of computer assistance they intended to develop. Support for some practices that they initially thought were important were dropped, whereas work on supporting previously ignored practices shaped their later design.

There are several drawbacks to iterative design, however. It is costly and time-consuming. The evaluation phase is often not possible to conduct in the workplace, forcing developers to rely on experimental rather than actual writing situations (Duin, 1991). Finally, due to the immense number of variables associated with writing, using software, and especially group interaction, Duin (1991) believes that a good product can never be guaranteed, even with extensive iterative testing.

2.2.5 Conclusion

The problem of defining collaborative writing, the need for a sociocognitive model of how people write together, the lack of research on collaborative writing and the difficulty of evaluating collaborative writing software are interrelated issues that have contributed to the poor design of most current collaborative writing software. Much work remains to be done before these issues are resolved and comprehensive technology which is embedded in a broader social context can be designed to support collaborative writers.

Such work is beyond the scope of this thesis, however. Instead, I will focus on how software developers can proceed with the design of collaborative writing software given recent research in information technology and what we now know about collaborative writing. In particular, I am interested in scaffolding tools for the facilitation of practices and the alleviation of difficulties that occur in collaborative situations.

2.3 A taxonomy of collaborative writing practices

Given the wide variety of collaborative writing practices, a systematic way to implement support for these practices is required. Posner (1991) has developed a taxonomy of collaborative writing practices that is based on the research of others as well as on her own interviews of ten people who described a total of twenty-two collaborative writing experiences in detail. Although not deeply rooted in formal theory, it is a useful guide for software designers for two reasons. First, it provides a vocabulary to describe what people do when they write collaboratively. As Lunsford and Ede (1986) point out, a vocabulary is important because “what we do not have a name for, we simply do not recognize.” (p. 74). Second, it presents the various ways in which collaborative authors write, in a concise way, making clear some of the implications these practices have for collaborative writing software design. I will outline Posner’s taxonomy, then discuss some implications for collaborative writing software design.

Posner identified four roles which participants in a collaborative writing group may take: *writer*, *consultant*, *editor* and *reviewer*. Although the functions of these roles are generally agreed upon, the kinds of changes to the document that are permitted within these roles is often defined by the group. For example, the editor’s role may range from being allowed to only correct typographical, spelling, and syntax errors, to being permitted to rework the document extensively.

Posner distinguished six collaborative writing activities:

- *brainstorming* (generating ideas)
- *researching* (gathering information)
- *planning* (defining and dividing the work)
- *writing* (transforming the ideas into text)
- *editing* (making changes to the text)
- *reviewing* (commenting on the document).

She found that these different activities occurred in various combinations and sequences during a collaborative writing project. This observation is similar to Hayes and Flower's (1987) description of the iterative and recursive nature of individual writing.

Posner classified four different document control methods used by collaborative writers: *centralized*, *relay*, *independent*, and *shared*. Centralized control involves one person being responsible for the document, while others play a more peripheral role. Centralized control can be maintained throughout the document production, or can occur in the final stages, when one person takes what has been written and integrates it. Relay control indicates that document control is passed from writer to writer during the document development. Independent control is the partitioning of a document among the writers, each of whom control a different portion. Shared control means that two or more people jointly control the document throughout the writing process.

Finally, Posner recognized four collaborative writing strategies: *single writer*, *scribe*, *separate writers* and *joint writing*. The single-writer strategy is similar to the individual writer's strategy. Only one person writes, while other members of the group provide support for activities other than the actual writing. The scribe strategy involves the group discussing the work, with one person in charge of recording the writing which evolves. The separate-writer strategy entails dividing the document into sections, so the writers can work on different portions of the document in parallel. In the joint-writing strategy, two or more writers compose the text together. Any of these strategies may be augmented by the assistance of one or more consultants, who may make stylistic recommendations, structural suggestions, editorial comments and/or contribute domain-specific expertise.

Since there are many methods that collaborative writers use to accomplish their tasks, any collaborative writing system would ideally support a range of writing practices, allowing users to select their preferred strategy for each collaborative project undertaken. Reference to Posner's taxonomy illustrates specific kinds of assistance that become necessary when collaborative writers work together in certain ways. Furthermore, Posner noted that during the writing of a document, the roles, control method, and/or writing strategy may change. Therefore, it is also important that computer software allows a group the flexibility to redefine their mode of writing at any time.

2.4 Support for the collaborative writer

Flower and Hayes (1980) have identified various types of constraints that writers must contend with in their attempts to produce a written document. Word processors and other types of computer writing technology, such as spelling checkers and idea planners, have primarily been developed to lessen the effects these constraints have on writers. The better the support that technology can supply writers with, by allowing them to off-load difficult or error-prone cognitive burdens, the more writers' minds are freed to concentrate on the writing process. Ultimately, this will lead to both better writing and more satisfied writers. When writers collaborate, some of these constraints are distributed (e.g., background knowledge), but at the same time new constraints are added (e.g., need for consensus). In addition, certain problems associated with writing become more difficult to handle because of the collaborative situation. There are also new practices that people adopt when they write together. Developers of collaborative writing software could help collaborative writers by providing support for these new practices and problems. I will discuss ten requirements for a collaborative writing system that would allow writers to choose their preferred practices, and provide assistance for problems associated with collaborative writing: communication, synchronous writing, roles, collaborator identity, annotation, version control, software and hardware compatibility, a global perspective, format consistency, and style consistency.

2.4.1 Communication

The most obvious way in which collaborative writing differs from individual writing is in the need for members of a group to communicate with one another. Optimally, collaborative writing involves face-to-face meetings supplemented by a range of communication channels, spoken and written, synchronous and asynchronous, so that collaborative writers can choose the most suitable channel for particular tasks within the writing process (Kraut, Galegher, Fish & Chalfonte, 1992; Sharples et al., 1993). Since the ultimate goal of collaborative writing software is to break down the physical distance between potential collaborative writers, it should not only support the collaborative task, but also offer appropriate means of communication for accomplishing the task. Ideally, this goal involves identifying and providing support for communicating in ways not currently easy, or even possible, in addition to facilitating customary forms of communication (Pea, 1994; Scardamalia & Bereiter, 1994).

Computer support for collaborative writing is crucial for writers who are separated by distance, yet wish to use either the scribe or joint-writer strategy of writing. Both of these strategies require the participants to write together, in contrast to the single-writer and separate-writers strategies. However, all potential and actual collaborative writers could benefit from a range of communication tools. Kraut, Egido and Galegher's (1990) study of the effects of proximity on group work found that maintaining social cohesion through frequent, high-quality, low-cost communication was an essential factor in both initiating and sustaining collaborative relationships. They point out the need for communication tools which facilitate both planned and unplanned, synchronous and asynchronous contacts among collaborators, be it of writing or any other group activity. By high-quality, they mean that any information which needs to be communicated can be transferred quickly and accurately. By low-cost, they mean the low behavioural cost to the users; the communication should not require a planned effort to use it. For planned interaction, collaborators should be able to choose the most appropriate means of communication. Any restrictions on avenues of communication will limit the collaboration. Given the rich referential field of face-to-face interaction (e.g., facial expressions, gestures to physical objects, external representations, etc.), a highly interactive multimedia environment is imperative (Pea, 1994).

2.4.2 Synchronous writing

Collaborative writing systems should allow both synchronous and asynchronous writing. Posner (1991) recommends synchronous writing support for jointly authored documents because her interviews revealed that collaborators did sometimes work on a section of a document simultaneously, despite lack of support for such a strategy. Synchronous writing capabilities also support asynchronous, distributed writing. If writers share control of the document, there may be times when their writing occurs simultaneously although they do not intend this, particularly if there are many writers, or if a deadline is near. Thus, software which does not support synchronous writing may restrict asynchronous writing by preventing access to a document already in use by a collaborator.

2.4.3 Roles

Posner (1991) identified four different roles which members of the collaborative writing team might play (see section 2.3, above). To support these roles, she suggests that writing software should make the roles explicit. For example, writers would have unrestricted access to the document, consultants would have read-only access and reviewers would have read and comment access. Depending on how much leeway the writers allowed them, editors might have full access, or be restricted to commenting. Such clearly defined roles would protect the document from revision by unwanted sources. Sharples et al. (1993) raise the concern, however, that explicit roles may be too rigid, since roles may be assigned prematurely, before the collaborative writing process has really begun, or roles may change as the writing develops. Flexibility in reassigning roles would avoid these potential problems.

2.4.4 Collaborator identity

Preservation within the text of the identity of the writer(s) of each piece would provide collaborators with important information, particularly when using relay or shared document control. Knowing who wrote what may assist writers in resolving questions or disputes that may arise with regard to content.

2.4.5 Annotation

Consultants, editors and reviewers, who work independently of the writer, would be better supported if they could insert editing marks on the electronic document and have access to an on-line comment section for same-page annotation, particularly if they are not permitted to alter the text in any permanent way. Representational editing tools would allow them to perform the traditional type of editing and annotation done on proofs, but would eliminate the need for transferring a hard copy back and forth, and ease integration of the accepted changes into the document. The tools would allow writers to see the original text and suggested revisions closely linked, thus reducing the ambiguity of the suggested changes. Such tools would also offer an additional mode of communication for all of the writers involved in the document production.

2.4.6 Version control

As writers make changes to the document without consulting their collaborators, the difficulty of merging these revisions and/or recovering deleted material may quickly escalate into a serious problem. Version control would help avoid wasted time and lost text engendered by distributed writing. Two collaborative writing practices for which version control is essential are the joint-writing strategy and the relay-control method (Posner, 1991).

2.4.7 Software and hardware compatibility

Incompatible software or hardware may have a deleterious effect on collaborative writers' practices. Rimmershaw's (1992) interviews of collaborative writers revealed that incompatible word processors affected decisions which were made about document control, writing strategies, and writing roles. Writers are sometimes forced to write collaboratively in ways they would prefer not to. The development of distributed computer-supported collaborative writing systems would eliminate incompatibility problems, as all users would have access to the same document.

2.4.8 A global perspective

Although word processors have facilitated writing in many ways, there are some drawbacks to the technology. One problem that is gaining attention is the difficulty of achieving a global perspective of the text (Severinson Eklundh, 1992). Particularly when producing long documents, writers often require access to overviews to guide their writing. In a longitudinal survey of seventy writers, which Severinson Eklundh and a colleague conducted (cited in Severinson Eklundh, 1992), approximately three-quarters of the respondents said that they had difficulty getting a global view of the text when using a word processor. Although it can also be difficult to gain an overview when using paper copy, there are two factors which exacerbate this problem on the word processor (Severinson Eklundh, 1992). First, the standard word processor generally allows a restricted view of the text—not even a full page can be viewed at one time, let alone several. Second, due to the scrolling, the user has no spatial information on which to rely when navigating through the text. These factors affect the ongoing evaluation of the text as a whole. Although writers using a word processor tend to revise more than when writing on paper, this increase is due mainly to additional local, low-level revision, whereas they tend to postpone global revision until later in the writing process. This leads to less coherent text, especially as the text grows longer. These problems are compounded when the text is a group effort—particularly when the joint writing strategy is used, and the document control method is relay or shared. Extra working time and frustration may be avoided by providing collaborative writers with access to a global perspective of the joint document as it develops. Severinson Eklundh (1992) makes several suggestions as to how designers of computer-based writing systems may help users access a global view of the evolving text: provide more than one type of overview of the document; make overviews active, thus allowing users to move text and move around in text using the overview; allow writers to move easily from one representation to another.

2.4.9 Format consistency

Inconsistency of format might not confuse meaning, but it may distract readers, thereby interfering with comprehension. It might also diminish the credibility of a document, its author and its publisher; inconsistency of format suggests that the document was produced in haste or with a lack of care (Farkas, 1985). Maintaining consistent spelling, punctuation, font style, and layout (particularly in figures and tables) throughout a document can be time-consuming and requires extra attention. English is notorious for its multiple spellings, such as *centre/center* and *yogurt/yoghurt/yoghourt*. There are also some punctuation rules which have alternative forms (e.g., *Charles'/Charles's*). As well, within a document of any length, there may be font, face, or size changes to indicate headings, subheadings, etc. Although many people have preferred spellings and punctuation, and font regularities are relatively easy to check by observation, within-document consistency control is useful—particularly if the document is large, or if the writer is a poor speller who spells erratically throughout the document, causing the spelling checker to recommend first one, then an alternative spelling. When the document becomes a joint effort, especially when using independent or relay document control, or the separate-writing strategy, such consistencies will become even more difficult to maintain. Within-document consistency software could be designed to flag variations in spelling, punctuation etc. conventions to ensure the document is consistent throughout.

2.4.10 Stylistic consistency

In addition to bringing their preferred styles of working to the collaborative writing process, individuals also bring their own styles of writing, which they have developed during their previous writing experiences. By writing *style*, I mean a writer's linguistic choices that are characteristic of the individual, a group, a genre, a historical time period, a communicative goal, a register, and/or a rhetorical stance. These choices may be conscious (e.g., consideration for the subject matter, the audience, the purpose, etc.) or unconscious (e.g., dialect). Despite the influence of genres and group styles, writers generally develop individual styles within the group norms. Indeed, many writers cultivate distinctive styles, as can easily be seen by comparing two authors—even, contemporaries from the same country with similar educational backgrounds writing in the same domain. Ede and Lunsford (1990) found that: "Of the disadvantages (of collaborative writing) cited, perhaps the most often mentioned involved what one engineer called 'the tough task of making a common single style from numerous styles.' " (p. 60). Attempts to merge styles may result in time-consuming revision, frustration and interpersonal conflict, and they are not always successful.

Aside from the fact that the writers may be required to produce stylistically consistent documents, why is consistency important? There are several reasons, two of which are analogous with the problems that format inconsistency may create. First, a variety of writing styles within one document may be distracting to readers. Second, as with format inconsistencies, style shifts may lead readers to believe that the document was written in a careless or hurried manner (Farkas, 1985). Finally, shifts in style are a cognitive burden to readers, since they force readers to change their expectations (Enkvist, 1964). Reading comprehension may be impaired by the additional load, particularly if the reader is unfamiliar with the subject matter, or is not a fluent reader.

One way to avoid the problem of merging styles would be to impose a particular style on the writing before the collaboration begins. However, there are three reasons why it is difficult for collaborative writers to produce a stylistically consistent document from the beginning, unless the single-writer strategy is used.[5] First, many people are loath to abandon a preferred writing style. It is not simply obstinacy on a writer's part to cling to a personal writing style, however; one of the five strategies writers use to reduce the cognitive burden of writing is to draw on a routine or well-learned procedure (Flower & Hayes, 1980). Changing one's well-developed writing style will make the writing task more difficult, since more attention will have to be allocated to an aspect of writing that is normally under less conscious control. Second, many writers do not want to impose stylistic restrictions on others. Writing together often involves conflicts about content and procedure, which can be time-consuming and stressful to resolve. Allowing each collaborator to write in his or her own writing style avoids an additional source of potential conflict among the group members. Third, people tend to have difficulty describing style, so even in cases when they want to impose a specific style, they may not be able to adequately provide their writing partners with the information needed to write in that style.

Once a document is written, there are two reasons why integrating different writing styles is potentially difficult for collaborative writers, even if the document is given to one person at the end to merge the various parts (centralized control). First, many people are poor at consciously recognizing inconsistent style. They might be dissatisfied with the document, yet not know why. Second, even when people recognize that the document does not have a single style, they are often unable to articulate the specific stylistic inconsistencies they have noticed.

For these reasons, unless centralized control is maintained throughout the document production and the single-writer strategy of collaborative writing is used, writing a document with a single voice may be very difficult. Collaborative writing systems that could help writers merge their styles would be especially useful for writers of documents which require a consistent style not only within a document, but from document to document (e.g., documents that are part of a series; documents produced by a large corporation).

2.5 Existing collaborative writing systems

Many of the existing collaborative writing systems provide support for only one or two specific aspects of collaborative writing. For example, Grove is a joint outliner (Posner, 1991); CRUISER provides visual and audio channels that allow informal interactions between potential collaborators (Duin, 1991). However, some current systems do furnish collaborative writers with more comprehensive support (e.g., Aspects). Nevertheless, since collaborative writing involves a wide range of practices, such systems do not necessarily support the same aspects of collaborative writing. Even when the practices that are facilitated and the problems that are alleviated are the same, they are often supported in different ways, to different extents. Therefore, it is difficult to make direct comparisons between systems. To present an overview of which of the above requirements are being addressed, and how, in current computer-supported collaborative writing systems, I will describe, in some detail, three current systems that exemplify various approaches to designing collaborative writing software, focusing on the type of assistance provided for the issues identified above.

2.5.1 Aspects

Aspects is a commercial collaborative editor developed by Group Technologies Inc. (now called GroupLogic) that supports basic word processing. It also has drawing and painting tools, and graphics can be added to text documents. It is available for both Macintosh, and IBM and compatible systems. Users can choose to work off-line, over a network or point-to-point via a modem or serial cable, allowing both synchronous and asynchronous editing. Users create conferences to jointly edit documents that were created either with Aspects's writing application, or another word processor. Up to sixteen people can join each conference. The creator of the conference can define

[5] Even if a document is written by one person, there is the potential for stylistic inconsistency, particularly if the document is large and is written over a long period of time.

others' roles and access rights to any documents in the conference in two ways. First, they can control who joins each conference, or allow free access. Second, they can specify whether others are allowed to edit, and whether a turn-taking protocol is enforced, allowing only one editor to revise the document at any one time. Communication among users is facilitated in two ways: each member of the conference can select a unique telepointer to use to indicate to their collaborators any part of the document they need to refer to, and a "chat box" is available for users to exchange notes about their work. The document is continually updated to ensure that all users have the same version. Simultaneous editing of the same section can cause inconsistencies, but users are warned when this problem occurs and the divergent copies are saved and renamed. Collaborators can choose to share their view of the document with some or all users, or they can unlink the view, allowing independent work. To find out who else is currently editing, collaborators can use the Who is Viewing command. Aspects does not provide any support for format consistency, a global perspective or style merging. Collaborators are only identified when using communication channels; no information is maintained to indicate who wrote which parts.

2.5.2 MILO

MILO is a system developed at the Dundee Institute of Technology, implemented in the X-Windows system (Jones, 1993). It supports distributed asynchronous writing of structured documents that may contain both text and graphics. Exchange of documents is both hardware and system independent. The document is stored as one or more notes, each of which contains a text field, a text editor, and a graphics editor. The notes can be structured before, during, or after writing using an idea processor, which allows both a linear and a tree-like global view of the document. Notes that function as annotations can also be created. A history of the document, including who created each note and when, is continually updated. There is destructive semi-automated version merging to keep documents up-to-date. Co-authors can communicate through email. Additional tools include a search facility and a spelling checker. Future work on the integration of annotations, and provision of two additional kinds of text structure view is planned. MILO does not provide means for: enforcing format consistency; specifying users' roles; synchronous writing; or ensuring stylistic consistency.

2.5.3 SASSE

SASSE (Synchronous Asynchronous Structured Shared Editor) is a prototype collaborative writing system that was developed at the University of Toronto (Baecker et al., 1993). It supports both synchronous and asynchronous editing of documents from workstations linked over local or wide-area networks. Import and export facilities allow the use of other file formats. Text is colour-coded by author, allowing writers to easily find out who wrote what, although this information is lost if the file is exported. The names of all collaborators, their colours, and whether they are currently active can be displayed when desired. An active outline editor allows users to see and manipulate the structure of the text. A gestalt view of the entire document provides a second type of global document view for users. The gestalt view also includes information about where collaborators are working in the text, as do colour-coded scrollbars. For more detailed information about a collaborator's activities, there is an observation view that lets writers see exactly what a selected collaborator is doing. To prevent writers from simultaneously working on the same section of a document, there is a locking mechanism, indicated by a padlock icon, that only allows one person to change selected text and text which has just been typed. There is a simple version-control mechanism that shows which parts of the document were changed, and by whom. A telepointer allows collaborators viewing the same part of the document the ability to select or refer to particular text without locking it. There is an annotation mechanism for exchanging notes about the document. SASSE also supports brainstorming. SASSE does not, however, support different roles for collaborators, format consistency or style merging.

These brief descriptions indicate the limitations of most of the current collaborative writing software in meeting writers' needs. They also demonstrate the difficulty of comparing existing collaborative writing systems to one another, since the goals of the systems vary widely. Support for the identified issues is being provided to some extent in at least one of these collaborative writing systems, except for format consistency and style consistency. In fact, there are no references to the problem of maintaining consistency throughout a document in the literature concerning collaborative writing systems.

2.6 Other computational approaches to consistency

2.6.1 Consistency of format

Consistency of format is not an issue that has been addressed in the computer-supported collaborative writing literature. However, the difficulty of maintaining a consistent format in long, complex, multi-authored documents has been recognized in the field of technical writing, particularly by large, multinational corporations (Schreurs & Adriaens, 1992). This concern with ensuring document consistency has led to the development of *in-house style* guidelines: a set of predefined rules governing a wide variety of document constraints on issues ranging from layout to style. In addition to enforcing consistency, another aim of in-house style rules is to create documents that are easily understood (by both native and non-native speakers) and easily translated (by humans or machines). Since the manuals required to outline all the necessary rules are long and complex, software has been developed to help technical writers with the difficult task of conforming to the desired standards (e.g., Hoard, Wojcik & Holzhauser, 1992; Schreurs & Adriaens, 1992; Dale & Douglas, 1992).

Companies have tended to protect their in-house style guidelines, so little of this work is available for public use (Schreurs & Adriaens, 1992), which may be one reason that this type of software does not seem to have had an impact on collaborative writing systems. Although the focus of the in-house style software is to impose a particular company's rules, the method used to maintain a consistent format could be adapted for collaborative writing software, allowing collaborators to impose their own rules to ensure internal consistency. Indeed, Dale and Douglas (1992) found that the mechanisms underlying the system they designed to apply in-house style rules had more general applicability.

The maintenance of format consistency appears to be the type of problem that computers are well able to handle, since many features of format are well defined and straightforward to detect. A component to check format consistency could be easily incorporated into a collaborative writing system, allowing writers to run their document through format-checking software, much as they use a spelling checker, when they are in the final stages of writing. Fairly dumb systems could be designed to accept user input specifying preferred conventions, which would work well for issues such as punctuation and typeface consistency. Higher level text aspects that require consistency would require a more intelligent system, such as Dale and Douglas's (1992) "language sensitive" copy editor. By incorporating some knowledge about natural language, they have been able to produce text tools that are more robust than current natural language systems, but smarter than current commercial tools, which rely on simple string searches.

2.6.2 Consistency of writing style

Consistency of writing style has also not been addressed in the computer-supported collaborative writing literature. Unlike format consistency, however, the maintenance of stylistic consistency is much more difficult to handle computationally. In fact, there has been little study, even informally let alone computationally, of style in the qualitative, quotidian sense that I use it here.[6] Style is

[6] In the next chapter, I will review the study of style.

difficult to study because it operates at many linguistic levels, and is intertwined with the realization of the propositional content of a text. However, there are also specific reasons why the scope of software designed to help writers with their style has been limited.

First, much of the research on style is subjective, and defies computational formulation (Ryan, DiMarco & Hirst, 1992). For example, although with the appropriate reading ability and background knowledge readers could recognize irony in text, how does one formally describe what it means for a text to be ironic? Second, certain aspects of style rely on a large amount of world knowledge (e.g., pun), that is presently computationally unmanageable. Third, many features of style that are well defined cannot be detected by computers of today (e.g., inversion). Stylistic features appear in a wide range of linguistic levels, many of which require sophisticated language processing to be recognized. Until robust natural language processing is a reality, only certain types of stylistic features will be computationally tractable. Finally, we do not know how to compute relevant indices of stylistic evaluation, even for simple aspects of style, such as sentence length (Sanford & Moxey, 1989).

Despite these constraints, there are three related areas of software development that claim to have tackled stylistic issues: style checkers, stylistic instruction, and in-house style checkers. There has also been some investigation of style in the natural language processing literature. I will briefly summarize some recent work done in these areas, first providing an overview of the current state of commercial style checkers, then discussing some prototypes, including research done by computational linguists, to illustrate the difficulties involved in designing software to analyze stylistic aspects of language.

Style checkers The broad usage of the word "style" is reflected to some extent in the multiple meanings of *style checker* or *analyzer*. "Style checker" is a term often used interchangeably with *writer's aid*, thus denoting any tool that aids in the writing process, including such tools as spelling checkers. It is also an alternative name for *grammar checker*, conflating the enforcement of both syntactic rules, the violation of which is simply wrong (e.g., subject-verb agreement), and prescriptive stylistic rules, which should be interpreted as guidelines for writers to follow at their own discretion (e.g., avoid the use of passives). In my discussion of current style checkers, I will focus on the detection of features that would be considered stylistic according to my definition of style (see section 2.4.10).

Style checkers are designed to recognize undesirable stylistic features and help writers eliminate them from the text. They are primarily intended for use by business people (Bolt, 1993), although there is at least one style checker that allows the user to choose from five standard sets of stylistic rules (Lancashire, 1991). Despite some claims to the contrary, there is no actual stylistic analysis done by currently available style checkers. Rather than embodying a semantic theory of style, these programs use simple heuristics that involve little or no linguistic processing to impose principles of "good writing style" as prescribed in style guides, such as Strunk and White's (1959) (Dale & Douglas, 1992). Instead of analyzing a complete sentence, only parts of sentences are analyzed, relying mainly on part-of-speech information stored in the checker's lexicon. The lexicon is also used to identify jargon, weak modifiers, colloquialisms etc., (Bolt, 1993). A simple string search is used to detect the use of many "forbidden" features, such as nominalizations or hackneyed phrases. Feedback to the writer may be given in the form of summary statistics (e.g., average sentence length), advice (e.g., *use the active, rather than the passive voice*) or an alternative form (e.g., consider *proceed* instead of *precede*). The best known, as well as one of the oldest, of these systems is the **UNIX Writer's Workbench**. Since these systems have no real knowledge of syntax, semantics, or pragmatics, many errors are missed (e.g., systems cannot detect the subject-verb agreement error in: **The dog groomed by the children of my friends are…*, Dale & Douglas, 1992). Also, correct forms are often flagged as possible errors. For example, many systems erroneously detect the passive voice in a sentence such as: *They were tired.* Even worse, errors that are purportedly governed by the rules in the programs are not always detected (e.g., *He writing a book is surprising.*). After reviewing six well-known commercial style checkers, Bolt (1993) concludes that although the performances of these programs vary, all of them perform so poorly that they are of questionable value to writers, especially those who are writing in a second language, since they may not have the skills to separate false flags from true errors or to detect errors which the programs miss.

Stylistic instruction A subset of style checkers are those that are designed to improve users' understanding of writing style, as well as to improve their written style (e.g., McGowan, 1992; Payette & Hirst, 1992). Stylistic instruction involves teaching writers the principles and conventions of written language beyond good grammar. Violations of stylistic rules are identified, explanations for the rules are given, and improvements are suggested. Ideally, instruction involves informing the user not only of the correct application of stylistic features, but also of the effects that these devices tend to have on the reader. Although regular style checkers often provide explanations for the rules they espouse, stylistic instruction software provides more detailed stylistic information. Since currently available stylistic instruction programs do not differ significantly from regular style checkers, two prototype stylistic instruction programs will be presented as examples of how style checkers could be augmented to help writers improve their style.

Since these style checkers are intended primarily for students, particularly those who are learning a second language, designers of stylistic instruction software should be aware of two important issues, if they are to provide effective instruction. First, the explanations that are given to the users should be appropriate to their needs; otherwise, users might be overwhelmed or frustrated by unsuitable feedback. An example of how this objective may be accomplished is given in McGowan's (1992) work on a project called **McRuskin**. This style checker focuses on providing explanations of stylistic rules (only one of which has actually been implemented), that vary according to the user's stated level of knowledge about the rule (low, average, or high), and the context of the document (audience, purpose, subject, use, and author's ability). In addition, there is a seven-layer discourse module that allows the user to seek further clarification if the explanation is not understood.

Second, the software should ideally help users develop a flexible understanding of stylistic rules, so that they also learn how and when to apply the rules in different writing situations. Payette and Hirst's (1992) prototype, **STASEL**, has, in addition to a syntactic style analyzer, a goal-directed style analyzer, which enables it to judge sentences according to the apparent stylistic goals of the writer. Rules for the stylistic goals are implemented according to the analysis of syntactic correlates. If the sentence is judged to have met the goal of clarity (the only one which has been incorporated), the user is informed, and the stylistic rule that was used in the analysis is provided. If not, stylistic problems are pointed out, and remedial feedback is given. The structural analysis is presented along with the diagnosis, allowing the user to connect the system's comments with the actual sentence elements.

Most currently available stylistic instruction software, however, has neither a user model nor a text model. Therefore, instruction is generally restricted to a single explanation of each normative feature of style. Also, the software is as limited as current style checkers in terms of the level of stylistic analysis that is performed.

In-house style checkers Another subset of style checkers is in-house style checkers. As well as handling format consistency (mentioned above), they often also purport to deal with stylistic consistency. An in-house writing style is usually developed by defining a *controlled* or *simplified* grammar, which is a subset of a grammar of some natural language. Current prototypes are promising because, due to the imposed syntactic restrictions, more robust parsing is possible, thus allowing the analysis of more complex syntactic structures than is possible in general style checkers (e.g., Hoard et al., 1992; Schreurs & Adriaens, 1992). However, although using predefined rules makes many aspects of writing consistency easier to control (e.g., standard verb tenses), there is a potential cost involved: the restrictive nature of controlled grammars may result in writing that does not sound natural (Hoard et al., 1992). There is also the cost of developing a controlled grammar that allows adequate expressivity—a time-consuming and open-ended task.

One of the Writer's Workbench programs, **prose**, handles the issue of in-house stylistic consistency in a different way. As well as performing stylistic analysis, **prose** allows a writer to compare a document (ideally, at least 2000 words long) to one of three pre-defined standards developed from selected technical and training documents. The document is checked for readability, variation of sentence type, average sentence length, the use of passives and the use of nominalizations. If the measures are more than one standard deviation from the mean of the standard to which the document is being compared, the writer is alerted to this fact. Users can also define their own standard, using the **mkstand** program. Statistics for the five variables are derived from one to seventy-five documents (ideally, twenty or more) supplied by the user, each of which must be at least 1900 words or 90 sentences long. New documents can then be compared to this standard (MacDonald, 1983). This approach is easier and less time-consuming than developing a controlled grammar, but there are several drawbacks. First, a body of already-consistent texts is needed to bootstrap the style standard. Second, fewer stylistic variables (out of an already-limited number) can be controlled. Finally, Gringrich (1983) found that even experienced writers were often unsure of how to apply the advice given by **prose** to their writing.

Natural language processing Computational linguists have explored many aspects of language, including style (e.g., Hovy, 1990; DiMarco & Hirst, 1993). Work on generation and understanding of style is concerned with the choices writers make and how these choices affect (or are intended to affect) readers. Ensuring that these effects are not lost in machine translation is also a prime motivation for research in this area.

The implementations summarized above focus mainly on syntactic stylistic choice, which has also been explored in natural language processing. DiMarco and Hirst (1993) have developed a stylistic grammar that codifies syntactic stylistic knowledge. They began by classifying commonly used stylistic terms into one of three abstract properties associated with sentence style: balance, dominance and position. Next, they defined abstract elements of style that are related to these properties. These stylistic elements were then correlated with syntactic elements. Finally, they defined formal rules to link the abstract elements with specific stylistic goals, such as clarity. Building on this grammar, they have developed a stylistic parser that analyzes text according to these rules, correlating observed syntactic patterns with the writer's possible stylistic goals on three dimensions: clarity/obscurity, staticness/dynamism and concreteness/abstraction. This work demonstrates a way to create a formal representation for the vague notion of style, which may eventually be used to create style checkers that use more than simplistic heuristics.

Aside from this research, however, style has not received much attention from computational linguists (DiMarco & Hirst, 1993). The work that has been done remains largely exploratory, and is therefore not currently robust nor comprehensive enough for use in commercial products.

2.7 Conclusion

Despite my criticism of existing software for its failure to provide comprehensive support for collaborative writing, cognitive research has not yet produced a detailed-enough model to enable the development of such support. Therefore, it is difficult to create a collaborative writing environment from first principles. Consequently, I take an empirical approach to investigating a component of collaborative writing that has been identified in the literature, but has not yet been addressed in software design: stylistic consistency. Creating a consistent style from more than one distinct writing style is a cognitively difficult task that may well be supported by a tool to off-load some of the cognitive burden. I do not propose that such a tool should stand alone, however, but rather that it be offered as only one of a suite of tools within a collaborative writing environment.

How to design such a tool remains a question. It is clear from the descriptions in section 2.6.2 that the current computational handling of style is limited, and will remain so for some time. However, there is another area of computational research which has proven useful in stylistic investigation: *stylostatistics*.