

(19) World Intellectual Property
Organization
International Bureau



(43) International Publication Date
27 October 2005 (27.10.2005)

PCT

(10) International Publication Number
WO 2005/101233 A1

- (51) International Patent Classification⁷: **G06F 17/21**
- (21) International Application Number:
PCT/US2005/012700
- (22) International Filing Date: 13 April 2005 (13.04.2005)
- (25) Filing Language: English
- (26) Publication Language: English
- (30) Priority Data:
60/561,607 13 April 2004 (13.04.2004) US
- (71) Applicant (for all designated States except US): **BYTE SIZE SYSTEMS** [US/US]; 433 East Parkway Boulevard, Appleton, WI 54911 (US).
- (72) Inventors; and
- (75) Inventors/Applicants (for US only): **FAY, James, E.** [US/US]; 4 West Coach Road, Boulder, CO 80302 (US). **ALLAM, Scott** [US/US]; 1409 E. Central Avenue, Edgewater, MD 21037 (US). **STAAS, Gary** [US/US]; 2929 Granite Creek Road, Scotts Valley, CA 95066 (US).
- (74) Agent: **KLANN, David, J.**; 433 East Parkway Boulevard, Appleton, WI 54911 (US).
- (81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.
- (84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— with international search report

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: METHOD AND SYSTEM FOR MANIPULATING THREADED ANNOTATIONS

(57) Abstract: A computerized method for manipulating threaded annotations in a collaboration environment is disclosed. The invention provides a graphical interface for viewing, manipulating and communicating the annotations for electronic documents. The invention provides an improved way to share, mark up and discuss electronic documents.

WO 2005/101233 A1

METHOD AND SYSTEM FOR MANIPULATING THREADED ANNOTATIONS

The application claims priority to US provisional patent Application 60/561607 filed 13 April 2005 (13.04.2005).

Background

Collaboration utilizing electronic documents media can be very complex, especially when multiple individuals are involved. And even more so when multiple strings are created, individuals responding to some input and not to others.

Therefore there exists a need in the art to provide an improved way to share, mark up and discuss electronic documents.

Detailed Description

DNA Threaded Note Architecture.

As a broad overview is:

1. Having unique IDs attached to annotations might make our ability to view and sort them much more savvy. Here again though, the display comes secondary and is enhanced because we have assigned a unique ID.
2. The process of how a thread is created by building up its DNA sequence.

These two items are the core technology of the DNA concept.

DNA Concept

1. Much of what we describe with this concept is used to control and manage annotations. I think using the word annotations is too restrictive and greatly undermines the potential to organize information with the DNA concept. I would like to create a term that is all encompassing called an Information Packet. I define an Information Packet as a collection of data (textual, image, sound, etc.) that can be defined as a single unit and has metadata information assigned to it. The units vary in scope and can be as narrow as a single word and as broad as an internet/network file server. Using this definition, a highlight on a PDF is an Information Packet that is narrow in scope. A PDF is an Information Packet that is the unit of a single file. My networked hard drive

is an Information Packet that is a unit of a large collection of files. Each of these information packets has their own unique IDs associated with them.

2. Observing the above example points out a dilemma for assigning Unique IDs. To encompass items with a broad scope would require a large number to assure uniqueness, however to apply a very large number for all items down to items of narrow scope would require substantial and needless file size bloat and processing. Take for example what the phone companies had to do. I may have the exact same phone number as thousands of people across the country. But because they have added the concept of an Area Code, duplicate phone numbers are allowable, although not within the same Area Code. As populations increase they add additional area codes. Therefore, it may be beneficial for us to do something similar, where we don't just have one large number, but several fields of smaller numbers. The smaller scope items may not need all the fields and the larger scope items should use all the fields. How unique does an ID number need to be for a note written by a specific author attached to a specific word, located on a specific page within a specific document? You might be able to get away with a pretty small ID number for that note. But if that note is a free note that does not have the benefit of being subordinate to other Information Packets and it is available to the world through the internet, then that ID number better be very unique. Using several fields of small numbers can allow us to assure uniqueness, while not making small scope items needlessly complex.
3. I would like to add a new "annotation" (or information packet) to the other annotations we have, which are note, highlight, find and bookmark. This annotation would be called category. So a note assigned to the "Personal" category would have the text string "Personal" stored with it. The idea of having a new annotation type of "category" would allow for a more robust data structure. In other words, categories would have their own unique ID and could belong to other categories. This allows us to define a hierarchy structure of categories, rather than purely linear. So a user may opt to create sub-categories to the "Personal" category called "Health", "Family", "Hobbies" etc. Note, bookmark and highlight annotations that belong to these various categories would store the Unique ID of the category rather than the category's

name. The concept of having a category rounds out the capabilities of unique IDs and the DNA concept when listed in context with them.

4. Currently, all annotations have a word displacement assigned to it. This basically means it is attached to a specific portion of a document. Notes that do not have an assigned word displacement would be considered Free Notes, which transfers our software from a Documentation study tool to a full blown note taking tool. For example, I might create a shopping lists: (Bold items are category annotations)

Shopping Lists

Safeway Store

Dairy Section

- 2% Milk
- Cream Cheese {unique ID 1234}
- Cheddar Cheese

Pharmacy Section

- Aspirin
- Cold medicine, etc.

K-Mart Store

etc.

The above items, such as the list item "Cream Cheese" has a unique ID with no parental DNA, but it does belong to the category "Dairy Section". Currently notes that belong in a thread do not have a word displacement, but they do have other Unique IDs in their DNA sequence. This basically sets up a method to reply to a note, "on the whole". BUT, if a threaded note DID contain a word displacement, then this creates a means to have nested annotations or a note within a note. For example, my wife might view my shopping list above and add a reply to note #1234 on the word Cheese. Her threaded note may say, "make sure you do not get the Fat Free, because I am using it for a Cheese Cake for your mom's birthday and Fat Free Cream Cheese does not gel very well in Cheese Cake." The above example may be overly complex because the note #1234 was short and may of not needed a nested note. However, many notes are very long and require note

replies interjected directly in the note. We do this all the time with our e-mail discussions. This method would allow us to create these nested annotations.

Unique DNA for every annotation

Every annotation is uniquely identified by a DNA, creation date pair of numbers. DNA is a randomly generated decimal number between 0 and 1.0.

Complete ancestry in each note

Every note has a list of all its ancestors, each identified by the unique DNA, creation date pair. This is useful for reconstructing a thread as notes are imported.

Construction of note thread

At any given time, no one person may have all the notes in a thread. However, any one who has contributed to the thread, or who has part of it, may export the threaded notes to a file. When these are imported, they fit seamlessly into their threads. That is, the notes fit themselves into the thread in the proper order. This happens regardless of what order the annotation files are imported. If duplicate notes are imported, they are ignored, so no duplicates appear in the thread.

List of users

You can address a note reply to a person or persons or group. When you reply to a note, you're shown a list of all users who have created any of the notes prior to this in the thread or had a reply addressed to them. You can select one of these or type in another name or group.

Quester

As you add threaded notes, Quester is automatically updated with them. A note thread appears as a tree of notes in Quester.

Popup menus

In Quester, you can right-click a threaded Note to get a popup menu.

Right-clicking a threaded note icon in the ReaderLens window also invokes this popup menu.

Here is a description of each menu item.

"Expand thread nodes" and "Collapse thread nodes": Clicking the "Expand thread nodes" menu item expands thread nodes from this node to the end of the thread. Clicking the "Collapse thread nodes" menu item collapses thread nodes from this node to the end of the thread.

"Display thread list": If you click "Display thread list", an annotations list appears listing all the notes in the thread. The annotation list shows the icon, page number, color, author, addressee, creation date and text for each note. This list can be sorted by any of these fields.

"Display thread text": If you click "Display thread text", a window displays with all the text of the entire thread in it, including metadata for each note, highlighting the note clicked on; double clicking note text opens the note. You can save all the text to a file with the "Save text" button. These lists show the notes in order in the thread, though the annotation list may be resorted as usual.

These lists show all the notes in a given thread, not the subthreads.

To get the subthread list, click on one of the notes in the subthread. Each list shows the list all the way up the original personal note that started the thread.

"Delete this annotation" : In Quester, if you click the "Delete this annotation" popup menu item, all annotations are deselected, the one you clicked on is selected so you can tell which one you're deleting, and you get a delete confirmation dialogue. If you confirm, the annotation is deleted, and nothing is selected. If declined, the annotation you clicked on remains the only selected annotation.

"Show detailed annotation information": When you click this menu item, a dialog box appears with nearly all the information in the annotation, in a format similar to the annotation list. It shows an icon for the annotation type, the full note text and the full highlight text. It shows the creation, modification, and import times. It shows the categories. This is the annotation inspector.

Threaded note navigation

A set of buttons in the note editor allows you to navigate the threaded notes tree. The "Previous" and "Next" buttons take you to the previous/next note in the thread. The current note is closed so the screen won't be cluttered. If there are several notes that start at that node in the tree, the "Previous" and "Next" buttons take you to the first note, that is, the oldest or earliest one. The "Next" and "Previous" buttons only appear if there is a next or previous note in the thread. The left and right arrow keys do the same action as the "Previous" and "Next" buttons.

If a note has a sibling in the thread, i.e., they both have the same parent, you can go to the next earlier (older) sibling by clicking "Earlier", if this sibling exists. If there is no earlier sibling, the button isn't shown. Similarly, clicking "Later" takes you to the next latest (younger) sibling, if it exists (and it does exist if the button appears). In the threaded notes tree and in Quester, siblings are ordered by creation date, so the older (earlier) ones are above the younger (later) ones. The up and down arrow keys also work the same as clicking "Earlier" or "Later".

Using these buttons (or arrow keys), you can navigate the entire thread. You know there are siblings by the "Earlier" or "Later" buttons appearing, and these buttons allow you to go to these siblings. The arrow directions match the placement of thread notes in Quester. That is, children and parents are to the left and right of each other, and siblings are above and below each other. The buttons follow this alignment, which makes this scheme reasonably intuitive.

Note thread count

The number of notes in the thread (as numbers in a black rectangle) appears before the note icon in the ReaderLens. This updates as you add or remove notes from the thread. This count shows the number of notes actually in the thread. If any notes are missing in the middle of the thread, the number has an asterisk "*" after it. In the note editor, the number in black in a threaded note is its order in the thread overall, i.e., it's the number of its ancestors plus one. If notes are missing in the middle of the thread, this will be apparent by gaps in the numbers.

Exporting notes

Two capabilities facilitate exporting what you want. "Export annotations this session..." exports just the annotations that have either been created or modified since you started the Reader, i.e., this Reader session. "Export annotations since last export..."

exports just the annotations that have been either created or modified since you last exported annotations for this document.

When you export annotations, only the currently filtered annotations are exported. To make sure that a user doesn't accidentally export only the currently filtered ones when he wanted to export all of them, when you export and some annotations are filtered, a dialog warns you before exporting and allows you to cancel the export.

Clicking OK or pressing Return continues the export.

Importing notes

If you import annotations and some of the imported annotations have the same DNA and creation date, but different metadata (modification date or words the annotation covers), you'll get a dialog box informing you that a certain number of the imported annotations may be duplicates. You're asked if you want to check each duplicate. If you say no, then the duplicates are not imported. If you say yes, you'll get a dialog

showing you both the original and the imported annotations--side by side annotation inspectors, as described above.

This allows you to examine information about the annotations to determine which one to keep. You choose whether to keep the original and discard the import--or to use the import, deleting the original--with a button below each inspector. You can't keep both, because then we'd have 2 annotations with the same DNA and creation date and that would be problematic. After you click one of these buttons, you're shown the next annotation pair. A message at the bottom tells you how many annotations you've checked so far out of the total number. When you're done checking all of them, the dialog goes away. You can also cancel at any time, in which case the remaining annotations are not imported and the dialog goes away.

The time an annotation is imported is saved in the annotation. This allows the additional capability: "List annotations imported this session...". It lists all annotations you imported since you started the Reader. This allows a user to quickly determine the new threaded notes, analogous to seeing what new e-mail you have.

Faux delete

Deleted annotations are saved in a file in the user's annotation directory. To see them, use the "List deleted annotations..." function. If you delete an annotation from this list, you have permanently deleted it. Double clicking the icon for the annotation takes you to the location in the document in which the annotation had resided; deleted notes will open. Clicking a "Restore" button restores the selected annotations to the document. When you save (either during the session or at the end), the deleted annotations file is written. The deleted annotations list dialog gets updated as you delete or restore annotations.

Deleting notes

When you delete a note from a thread, the children of that note become the children of the deleted note's parent. If you delete the head of a thread, its children become heads

of their threads. This maintains the thread appropriately. If you restore this deleted note, the thread should look like it did before the deletion.

Hidden notes

You can hide a note by checking a "Hide note" checkbox. If you check this, the note will be hidden--and right away, vanishing from both the ReaderLens and Quester--unless hidden annotations are displayed.

The default is not hidden.

A "Show hidden" checkbox exists on the annotation filter controls dialog. This dialog allows you to set which annotations are displayed. When checked, show annotations that are hidden, such as notes that have the "Hide note" checkbox checked. The annotation inspector shows the hidden attribute. The default setting of this filter is to not show hidden annotations.

Publish mode

Added a new configuration item: "Publish mode". This will only be available with the publishing version of the ByteSize Reader that allows users to publish special annotations, such as expert annotations.

Added a "Publish" check box to the Note editor. The Publish check button only shows up when Publish mode is set on in preferences. If you check "Publish", the annotation becomes a "Publish Annotation", aka, Expert Annotation, that can't be changed or deleted by regular users. I used the terminology "Publish Annotation" to be a little more generic. The idea is to include any content by anyone who chooses to add it to a document with the publishing version of the ByteSize Reader. You can't delete these annotations from Quester or an annotation list unless Publish mode is on.

Associating a file with a Note

Added "File..." menu item to "Edit" menu in note editor. Clicking this displays a dialog displaying the current file name associated with the annotation, if any. Click "Browse" to show a file selection dialog that allows you to specify a file to be associated with this annotation, such as for a publish note (see below). Clicking "Set" will tentatively set this as the annotation's file, which is actually set when you click "OK" in the note editor. The annotation inspector shows the file name.

Types of publish notes

There are 2 types: straight notes and files.

If you open a Publish Note that does not have a file associated with it, then it opens in the Publish Note window, not the regular note editor. The text of the note displays in this window. However, if Publish mode is on, then the note opens in the regular note editor to allow the publisher to modify it.

The Publish Note window is much simpler than the note editor. It does not allow modifying the text or deleting the note--it has few controls. It does have minimize and maximize boxes, however, and shows up in the Windows task bar.

If the note does have a file associated with it, then the file opens in a new Reader window, complete with the ReaderLens, Quester, and the page view--just as if the user opened this file in the regular way. The file must be a PDF file, either vanilla or ByteSized. The file is opened from the "Published Annotations" directory, which is at the same level as the "ByteSize Reader Annotations" directory. It uses only the very last part of the file in determining which file to open. In other words, if you had selected the file "C:\projects\myfile.pdf" using the file dialog to associate this file with an annotation, it would use the file "myfile.pdf" in the "Published Annotations" directory. This allows us to put all the expert annotations in a single place.

The method of distribution and the fact that we are creating a "publishing" system that allows authors and publishers new channels to move the content.

So what is the method of compilation, packaging and distribution? Here are a couple of thoughts on it:

Traditionally distributed content should be defined as structured content such as HTML webpages, XML and SGML, desktop files such as Word Processing files, Spread Sheets and Presentation files (i.e. Power Point), image files, video files and sound files. In addition to this we should also include actual tangible paper in this definition. It seems far fetched to think that traditional paper should be included in our definition because we work on electronic materials. Not so, however. Since PDF is a perfect facsimile of the paper page, it may be possible one day to have a means to create electronic annotations, while viewing a paper document. This can be done if a page were laid on top of a pen based input device, such as a Wacom Tablet.

We provide a system for authors to look at traditionally distributed content and add additional content on top of this material. This additional content is stored externally to the original content. The external portion is crucial since normal annotations and hyperlinks are stored internally to the content. Having it external allows the system to distribute this additional content without having to distribute the base file with it. This also allows for easier subscription models, etc..

We may also include content created from our own system as a platform for the "base" content. That means the ability to add annotations on top of annotations or nested annotations.

The publishing system creates secure content that has publishing controls that are normally reserved for the base content, i.e. Digital Rights management, copy right controls, print controls, editability controls, access rights controls (i.e. limiting the viewing to specific audiences), etc.

Annotations Technology Architecture

Summary

When a note is sent to another person and then responded to, these become "threaded notes."

This is the type of collaboration that is likely... Tom writes a note inside the ByteSize Reader that asks a question. He sends the note to Sally, Bill and Clyde. Sally responds to the note and sends it back to Tom. Bill, because he cannot adequately answer the note himself, forwards the note to Steve and asks him to answer it. Steve writes part of the answer on Monday, finishes the note on Wednesday and sends the answer back to Bill. Bill, without mentioning Steve or forwarding Steve's message, sends his own response to Tom, and writes a side note to himself that he sends to no one. Clyde, who also doesn't know the answer to Tom's question, forwards Tom's message to Lois and Lane. Both Lois and Lane respond to Clyde who forwards their messages, along with his own summary to Tom. Tom gets all these messages and imports them inside the ByteSize Reader to read them. Using the "note" technology we have right now, this would be a mess, at best. In fact, it causes a mess to do replicative sending and receiving of MUIs that don't even contain Notes. Our current technology is flawed even when used with just highlights and bookmarks. The real problem is not just that our current technology is flawed, it is flawed so badly that it causes huge irritation and it paints us into a technology dead end that will cause big problems later on. We need to the way we do annotations, and we need to fix it as part of v1.0 of our software.

There are several objectives for Threaded Notes: 1) an architecture that keeps each note separate and distinct from every other note, but still connected with the correct thread; 2) an architecture that prevents duplication when annotations are imported that contain some of the same annotations that are already in the importers' annotations, 3) an architecture that distinguishes "threaded notes" (TNs) from "personal notes" (PNs); 2) an architecture for TNs that is scaleable, which means that we can grow with it and continue to use it even as we improve it in later iterations of our software; 3) an architecture that allows an unlimited number of TNs, 4) an architecture that accurately keeps track of and displays parent-child relationships between TNs,

even when some links in the chain are missing, 5) an architecture that allows users to delete spurious notes in the middle of a TN chain without messing up or confusing the thread; 6) an architecture that allows users to easily send parts of a complex TN chain to other people; 7) an architecture that allows users to import missing pieces of a TN chain and have those pieces fit to their correct places within the chain; and 8) tools that allow easy display and manipulation of complex notes.

Also addressed are highlights and bookmarks because they too can be inappropriately duplicated through exporting and importing of annotation files.

Separate Folders for Annotations

In a collaborative environment, people need to be able to get at the files that hold annotations that are being both exported and imported. The objective is to put all the annotations in one place and away from everything else.

ByteSize Systems (installed by default in C:\Programs)

ByteSize Reader

ByteSize Reader (with desktop shortcut)

ReaderLens

ByteSizeFiles

DLLs

Images

Lib

libs

Modules

Pmw

Tcl

ByteSizePreferences

ByteSizePreferencesPy

ByteSizeReaderHelp

t1fonts

Log

ByteSize Reader Annotations

Personal Annotations

Clyde Crashcup (example of a user name)

Jane Reddress

Imported Annotations files

Exported Annotations Files

Regrouped Annotations Files

ByteSizer (when ready and purchased separately)

ByteSize EAP (when ready and purchased separately)

ByteSized Documents

(There may be multiple folders inside this folder to organize various types of documents)

ByteSize Expert Annotations

(Because EAs are collections of individual ByteSized documents, each named EA will have a folder that is full of individual EAs and it will have an MUI that links the individual EAs to the document.)

No Duplication of Annotations

Assume that I read a document and add highlights, bookmarks and notes. I export all of these annotations and send them to you. You import my annotations, reply to a single note, export the entire set of annotations and send them back to me. I import the annotations file. I don't want to see duplicates of all of my own annotations, but that's what would happen with our current design.

The following design changes will prevent duplication of annotations that are a result of multiple exports and imports of annotations files:

- Each annotation, regardless of type, is coded with a random number -- its DNA -- and meta-data ancestry (document, location within document, user name, date, time). If the meta-data excludes "document" and "location within document" it is assumed that the annotation (it can only be a Note because Highlights and Bookmarks have to have document and location ancestry) is free-floating.
- Assume I write a note and then send it to you. After sending it to you, I change the note. When you send a complete set of annotations back to me, the note I get back from you isn't the same note I now have on my computer because I changed it. Which note do I want to keep? I don't know but I do know that I want the opportunity to decide. Therefore, if an annotation is changed, it keeps the same DNA but gets new meta-data ancestry. If during an "import" process the Reader sees that two annotations have the same DNA but different meta-data ancestry, the Reader will alert the user and give choices to keep the old, swap out the old and keep the new, or keep both the old and the new and link them together to show that they are versions of the same.
- When annotations are imported, if two or more annotations are located at the same place in the document, the random numbers and meta-data ancestry are compared. If the comparison shows identical matches, then only one of the annotations kept. If two annotations show the same DNA, regardless of meta-data ancestry, but are located at different places in the document, they will both be displayed.
- Each annotation is uniquely coded with its own piece of DNA. If an imported annotation has the same location code as another annotation in the document (meaning that it would reside at identically the same place in the document), then the DNA and meta-data ancestry of the annotations are compared. If the DNA is different, then the import is allowed to happen and the two annotations will reside at the same place in the document. However, if the DNA and meta-data ancestry is

the same, then the import of that particular annotation is bypassed without notification.

- When an annotation that has the same identification code as another annotation is bypassed for import, the assumption is that the annotation already in the document is the correct one to retain. This assumption would be invalid if we allowed people to change each other's annotations. So, when we disallow changes to other's annotations, it is valid to assume that the annotation already in the document is the correct one to retain and the imposter being imported should be bypassed.

This technique will prevent duplication of notes, highlights and bookmarks in documents when users regularly import annotations from other people.


Threaded Notes Are Separate But Linked

Each note and reply is kept separate. Each note in a thread is linked with a visual thread that makes parent/child (question/response) relationships clear.

Two-Part Icon for Threaded Notes

TNs (threaded note) need to be distinguished from PNs in Quester and in the ReaderLens so that users know which notes are TNs. It would get very messy on-screen to pop up all of the notes inside a TN in either the ReaderLens or Quester. Instead, we will show TNs with an icon and a number that designates the number of individual notes that reside in the TN.

Both PNs and TNs are identified on-screen with the same set of icons we have already developed. Both PNs and TNs are further identified by the same colors that we already use for PNs. TNs are further identified on-screen by an additional icon that is placed immediately to the right of the graphic icon. The additional TN icon will contain a number that corresponds to the number of

notes that reside on the users' computer that are included in the thread. This is an example:  -- an icon for a TN that has eight notes in the string. It is entirely conceivable that the number of notes in a thread could reach into the hundreds and maybe the thousands. This two-part icon will display in the ReaderLens and it will display in the header of the Note.

Quester Displays the Tree Structure of Threaded Notes

Personal notes (PNs) will display in Quester exactly as they display now.

Because the threaded note "tree" can get very complicated, the way to see the entire tree is by opening it in Quester. Quester displays the various levels of threaded notes using the same "+" and "-" signs used to display header levels right now. At the top level, Quester will display a TN tree with the two-part icon, a "+" sign, and the patriarch Note in the tree. Clicking on a "+" sign will open a threaded note (TN) a level and display more "+" signs if there are more levels. As each level of the TN is opened, the display in Quester will show the two-part icon and the number in the icon will show the number of Notes in the particular part of the thread being displayed, along with either a sign to indicate whether more depth is available or not, and the most-parent level Note. Double-clicking on a particular threaded note in Quester causes the PageView and ReaderLens to jump to the place where the TN is located and will cause that particular note to pop up in a note window over the ReaderLens.

Automatic Display of Lineage in Notes

To make it easier to identify and interpret notes, all notes will automatically have thread information placed in a header within the note. The thread information in a PN will be <user name, date, time>. This is the same information that we use in notes now when the user clicks on either "Add thread info" or "reply." The difference is that it will be added automatically rather than only when the user requests it. So, the "Add thread info" option in

our current design will be eliminated. The meta-data lineage that will be added automatically is shown in the boxes below.

It would be incorrect to assume that when a person clicks on “reply” in a note, that he wants to reply to the author of the note. He may, instead, want to reply to the person who sent the note, or a third party. Therefore, our “reply” button needs to be more robust. When “reply” is clicked in a note, a list of users already involved in that particular thread should pop up, and there should be an “other” box that allows a person to type in another user name. Included in the list should be groups of people that the user can define.

The boxes that follow show the automatic lineage displayed in headers in note boxes. Since all threaded notes begin as personal notes, the first header shown is for a personal note. The lineage in a PN is a single line. The second note assumes that I sent my PN to Abe Lincoln. When he creates a reply by clicking on “reply,” the PN is transformed into a TN. The lineage in a TN is three lines: 1) the first line is the author of the new note, 2) the second line shows what the new note is in reply to, and 3) the third line shows to whom the new note is written. The + sign in the “To:” line is a logical operator that separates different people or groups so that the people or groups can be queried separately.

Personal Note:

Author: John A Doe, 7-Jan-04, 11:53 a.m.

Threaded Note:

Author: Abe Lincoln Allam, 8-Jan-03, 10:46 p.m.

Reply based on note from: John A doe, 7-Jan- 04, 11:53 a.m.

To: Stacy Smith + BOD

The header information in notes should all be kept as meta-data that can be accessed for sorting, filtering, importing and exporting. For example, when a user wants to create an annotations file for export to particular people, he can query the meta-data in the “To” header and ask it to build an export file that consists of only replies to the BOD.

The lineage meta-data information in the note headers is NOT in the body of the note and is, therefore not editable. (The “To” names and groups can be edited, but not as text in the note header.)

When a user clicks and drags across text in a note, it will highlight. When the mouse button is released, a modified Doc-Doctor tool will pop up. The options in the Doc-Doctor are: a) Copy text, and b) Copy text with citation. When “Copy text with citation” is selected, all of the information in the triple header is copied along with the highlighted text in the note. When the text is copied into another document, the triple note header information is appended at the end of the note text. Once we have added capability to add annotations to Notes, we can implement the full-featured Doc-Doctor that we use in the ReaderLens.

Display of Notes

Our current architecture is that notes display as icons in the ReaderLens. If the cursor is rolled over the icon, a transient pop-up displays the text of the note, but it goes away if the cursor is moved off the icon. If the icon is clicked, the note displays in a pop up window in the same font that the ReaderLens is using. The pop up window remains visible until closed by the user. In the new architecture, exactly the same thing will happen for PNs.

In the new architecture, threaded notes (TNs) will be treated differently in the ReaderLens. TNs will display in the ReaderLens as icons with an attached

number that corresponds to the number of individual notes in the thread. If the cursor is rolled over the icon, a transient pop-up will display a meta-data list of the individual notes in the threaded note. Dragging the cursor down the list to a particular note in the list and clicking on the note will open that particular note in the ReaderLens using the same font that the ReaderLens is using. If the cursor is moved off the icon or out of the list box, the transient display will go away. Once an individual note is opened in a window, that window will remain open until closed by the user.

The meta-data list of individual notes that pops up when the cursor is rolled over a threaded note icon in the ReaderLens looks like what is in the following table. There is provision to sort the notes according to variables. Ideally, we would set up the sort so that it works over a primary key and then a secondary key. Sorting will only affect the display in the meta-data list – it will not affect how notes display in Quester. This linear display will be unable to show all of the hierarchy. For now, it will have to do. (Take notice of the “To:” field. Ideally, this will hold email addresses with to, cc and bcc options. With these, we will eliminate the need for importing and exporting of MUIs. In v1.0, we do not expect that the email concept will be implemented.)

Sort Notes by:	<input type="radio"/> Author <input type="radio"/> Date written <input type="radio"/> Reply from author <input type="radio"/> Reply from date <input type="radio"/> Written to	
List of notes in this thread		
Author:	Reply based on note from:	To:
Klass, Dan, 2-Jan-04		John A Doe
John A Doe, 4-Jan-	Klaaa, Dan, 2-Jan-	Klass, Dan

04	04	
John A Doe, 4-Jan-04	Klass, Dan, 2-Jan-04	Tom Thumb
John A Doe, 4-Jan-04	Klass, Dan, 2-Jan-04	Personal comment
John A Doe, 4-Jan-04	Klass, Dan, 2-Jan-04	Abe Lincoln
Tom Thumb, 5-Jan-04	John A Doe, 4-Jan-04	John A Doe
Abe Lincoln, 5-Jan-04	John A Doe, 4-Jan-04	Stacy Smith

When a specific note in a Threaded Note is open and being displayed in the ReaderLens, two buttons inside the note let the user open either the parent or the children of that Note. The two buttons are: a) "Open parent of this note," and b) "Open next reply to this note." If there are multiple replies at the same level, then a pop up similar to the one displayed above opens, but displays only the next level of replies to this Note. Clicking on one of the notes in the list will open it. If a parent or child Note is opened, the first Note opened does not close until the user manually closes it. This way, a user can view both questions and replies at the same time.

In our current architecture, the icon associated with a Note displays in the ReaderLens, but does not display in the Note itself. In the new architecture, the two-part icon for the Note will display in the header of the Note. The number part of the icon will display the number of notes, including this note, that are children of this note.

Notes displayed in the ReaderLens may look like those that follow. The first is a simple PN, the second is a TN that has not had Categories added, and the third is a Categorized TN.

Note				
John A Doe, 14-Jan-04,				
Author: 8:02 a.m.				
Dan says the State has no limit on the number of directors a corporation can have				
We need to decide how many directors to have.				
Save	Delete	Hide	Mark as: Follow up needed	Reply
			Categorize this note	

Note				
John A Doe, 4-Jan-04, 8:26				
Author: a.m.				
Reply based on note from: Klass, Dan, 2-Jan-04, 7:36 p.m.				
To: Tom Thumb,				
Dan says the State has no limit on the number of directors a corporation can have				
Save	Delete	Hide	Mark as: Read	Reply
Open "parent" of this note			Categorize this note	
Open next reply to this note				

Note					
John A Doe, 4-Jan-04, 8:26					
Author: a.m.					
Reply based on note from: Klass, Dan, 2-Jan-04, 7:36 p.m.					
To: Tom Thumb, Ron					
Category: Legal					
Category: Directors					
Dan says the State has no limit on the number of directors a corporation can have					
Save	Delete	Hide	Mark as:	Read	Reply
Open "parent" of this note			Categorize this note		
Open next reply to this note					

Notes have a “Mark as:” button at the bottom of the pop up window. When this is clicked, it opens a drop-down or pop-up with these options. The first three options are predefined by us. Create option allows the user to create more “Mark as:” options. The options should have radio buttons so that Notes can belong to more than one class.

Notice the two lines in tan at the bottom of the “Mark as” window. These only apply to TNs and should be grayed-out or hidden for PNs and for the patriarch level of a TN. When either of the two “Parent of a new family of notes” options is selected, a flag is set on this note and the Reader displays it as the Patriarch of a new family. If the “Parent and only member of a new family of notes” option is selected, then only this note is plucked from the TN to begin a new TN. If the “Parent of a new family of notes and bring its children” option is selected, then this note is plucked from the TN to become the patriarch of a

new TN and all of the children of this note in the TN are plucked to come with it. When “Return this note and its family to its original family” is selected, then the note and the children are returned to the original TN. Behind the scenes, the only thing necessary to return a note to its original family is to remove the flag that was attached to it and then repaint the screen. When notes are moved around, all of their original DNA stays with them so it is always possible to rebuild TNs.

Mark as:
Follow up needed
Read
Unfinished (gray if imported)
Create new option
Parent and only member of a new family of notes
Parent of a new family of notes and bring its children
Return this note and its family to its original family

Annotations can be Categorized

When we figured out how to add DNA to a Note and the implications of doing so, Abe Lincoln took it a step further and realized that we could add an additional layer of DNA to “Categorize” annotations of all kinds. Adding Categories to annotations is an incredibly powerful concept.

Notes have a “Categorize this Note” button at the bottom of the window. When clicked, it opens a drop-down or pop-up window like the one that follows. Assume that the user has already created the categories shown. The second to last line in the window gives the user a way to create more categories. The window should grow as necessary to allow the user to create more categories. The last line in the window gives the user a way to delete categories which is a housekeeping function. As the radio buttons show, a note can belong to an unlimited number of different categories at the same time (that’s where the power of the concept comes in). When a Note is given Categories to belong to, these Categories simply add to the DNA of the Note.

Notice that the window that pops up says “Categorize this Annotation” rather than “Categorize this Note.” We want to use this same pop-up to allow users to categorize Highlights and Bookmarks in addition to Notes. Categorization of Bookmarks can be done by adding a “Categorize” button to the Bookmark pop-up. Categorization of Highlights may have to be done by adding a line to Doc-Doctor called “Categorize this annotation.” If we add that line to Doc-Doctor, it would make sense to have it work on all three kinds of annotations.

Categorize this annotation:	
Categories:	
<input type="checkbox"/>	Product development
<input type="checkbox"/>	ByteSize Reader
<input type="checkbox"/>	ByteSizer
<input type="checkbox"/>	Marketing issues
<input type="checkbox"/>	Threaded Notes
<input type="checkbox"/>	Start Screen
<input type="checkbox"/>	Finance & Accounting
<input type="checkbox"/>	Stock valuation
<input type="checkbox"/>	Legal
Create new category	

Delete selected category		
Move up	Move down	

Actions Taken to Export Annotations

Send Only My Replies

Exporting only replies will avoid exporting parent messages of replies and it will avoid sending personal annotations. When we are able to make annotations work like an email system, this process will be obsolete.

“Send Only My Replies” will work like this:

When “Send only my replies” is clicked, a pop-up will display a list of the people to whom I have written replies to annotations in this document. I can select one person from the list. When I click on a person’s name, a second pop-up will allow me to select a date. This date prompts the earliest date of replies to that person. This date check is necessary because I may already have sent previous replies to this same person and I don’t want to duplicate what has been previously sent.

Once the user name and the date have been specified, the software looks for replies written by the current user name, the “reply to” name specified, and the reply date. It then builds a “list all” display. The user can click on any item in the “list all” display and the software will jump to that annotation (just like the “list all” function in “search” works right now). The user can click on and delete from the “list all” table any particular reply that he does not want sent (deleting from the “list all” table deletes the annotation from the “list all” table, but does not delete the annotation from the document. The user can also click on “add to list” and then add to the list by clicking on any annotation displayed in the document.

Once the user is happy with the “list all” table, he can click on “Create export file” which will bring up the usual file name and location boxes. The default file name

should be: "document name – user name – reply to name - date." (I think "date" should be part of the file name because I have noticed that many people have their computers configured not to display file dates.) The default location should be c:\my documents\ByteSize\files to export.

Regrouping Annotations

An obvious and powerful extension of Categorizing annotations is regrouping them according to those Categories. Let's say you have a document that contains a large number of annotations. These annotations are a combination of both PNs, TNs, Highlights and Bookmarks. You go through these annotations and Categorize the Notes, Highlights and Bookmarks according to your own design. Now that you have categorized the annotations, you want to group them according to Categories.

An example may help... Assume you are the campaign manager for a presidential candidate named Nowar, and that you are reading through the ByteSized text of a speech by his rival candidate named Prowar. As you read, you annotate the document, looking for points around which to develop a political strategy to defeat Prowar. You categorize some of your annotations and develop three categories called Foreign Policy, Economy and Education. Upon additional reflection, you add another some additional Categories called Middle East, South America, Illegal Drugs, and Military Spending. Now, you would like to look at the annotations when they are grouped according to some of your Categories. To do that, we will invoke a new command under the Annotations menu called "Regroup."



When clicked, the Regroup pop-up displays the various Categories you've created with radio buttons in front of each Category. The process of regrouping is simple: click on the radio buttons to specify which Categories to display


The "Regroup Annotations" function is invoked from the "Annotations Menu." When it is invoked, the pop-up looks like what follows:

Regroup annotations	
Select categories to group together:	
<input type="checkbox"/> Economy <input type="checkbox"/> Education <input type="checkbox"/> Foreign policy <input type="checkbox"/> Middle East <input type="checkbox"/> South America <input type="checkbox"/> Illegal Drugs <input type="checkbox"/> Military Spending	
Display only annotations that fit in all categories selected	
<input type="checkbox"/> Display annotations that fit in any categories selected	
Regroup and display	

The "Regroup annotations" function allows two choices for displaying results. The first operates with "and" logic which means that to be selected for display, the annotations must fit in all of the selected categories. The second operates with "or" logic which means that to be selected for display, the annotations only have to fit in one of the selected categories.

Once the annotations are regrouped, they are displayed in a window that looks much like a combination of other windows we already have. The annotations are displayed like they are in Quester. The order of the annotations is changed like they are in the Citations Manager. Following is an example of a "Regrouped display of annotations."

Regrouped Display of Annotations	
Name of this regrouping:	Analysis of Prowar's record on drugs
	"I will keep illegal drugs off our streets."
	Prowar voted to reduce antidrug aid to Columbia by 37%

 Prowar's promise is inconsistent with his vote.		
Move up	Move down	Display only text of annotations
Save this regrouped display	Unhide hidden annotations	Hide highlighted annotation in this display

Whenever annotations are categorized and regrouped, there will be particular annotations that are irrelevant to the purpose of the regrouping. The “Hide highlighted annotation in this display” button allows users to hide spurious annotations. The “Unhide hidden annotations” button allows users to display annotations they previously chose to hide.

Annotations have three types of information: DNA (which we never show to users), meta-data (which users will sometimes want to see, and sometimes not), and content (which users will always want to see). The “Display only text of annotations” button causes only two bits of information to display about each annotation: the type (Highlight, Bookmark, Note icon) and the content. The alternative button is “Display text and ancestry of annotations.” When “Display text and ancestry of annotations” is clicked, the content and the meta-data of the annotation is displayed. Alternatively, because there will be so much meta-data attached to annotations, it might keep things cleaner if meta-data is only displayed as a pop-up tip when the cursor is placed on the annotation’s icon.

When “Save this regrouped display” is selected, a file-save box pops up that allows the user to give a name to this regrouping of annotations (which is then displayed in the second line of the window), but it does not allow the user to select a location to save the named regrouping. Regrouped displays of annotations are always saved in the folder shown in the section of this report called “Separate Folders for

Annotations.” What is saved in the regrouped display of annotations is the specific DNA identifiers for the various annotations in the order created by the user, and a flag to show whether the particular annotation is hidden or displayed. It is not necessary to save the actual text of the annotations in the regrouped display because the text is saved with the annotation rather than the regrouping.

Display of saved regrouped annotations is accessed by an item under the Annotations menu called “Display regrouped annotations.” When this is invoked, a named list of regrouped annotations is displayed. The user can then click on the name she wants and then click “open.”

Filtering of Threaded Notes

In the pop-up box for Threaded Note Filters display only the lead note in each thread along with a number that corresponds to the number of notes in the thread. Beside each lead note, put a radio button that gives the user a simple way to either display or not display the entire thread. For now, those are the only choices. It would be nice for all individual annotations to have this radio button too so they can be individually filtered in or out regardless of what the macro-filtering option does to them. One may be able to filter globally and then, go through the list of hidden annotations and selectively reveal individual annotations.

Changing Annotations from Others

An important part of this concept is that users must be prevented or at least cautioned against changing other people’s annotations. In the preferred embodiment, users will be prevented from changing other people’s notes. In a less preferred embodiment, a changed annotation will create a new annotation with new DNA that will reside at substantially the same parent/child level in the TN hierarchy, or off to the side of the hierarchy.

In the preferred embodiment, when a user clicks inside a note to put the cursor inside a note, the software will trap all keystrokes while the cursor is inside the note. The

trap will compare the "author" of the note to the current user name. If the two names are not the same, the following note will pop up on the screen (the reader will appreciate that the meaning of this message can be accomplished with many different but similar words):

"You can delete or hide this annotation, but you cannot change it because you did not create it."

Not allowing changes to annotations applies only to "changes" and only to "Notes." It is preferred to allow users to "delete" any kind of annotation created by other users, "hide" any kind of annotation created by others, "filter" any kind of annotation created by others, and "regroup" any kind of annotation created by others. We will allow users to change their own annotations, and we will allow users to copy-and-paste text or graphic from any annotation. We will allow users to create new Notes that are either personal or threads that string from Notes created by others.

What about changing Highlights and Bookmarks from others? The easiest way to deal with this is to force users to create new Highlights and Bookmarks rather than allow them to change others' Highlights and Bookmarks. (If we allow changing of others' Highlights and Bookmarks, we need to modify the DNA along with the meta-data ancestry. That gets complicated. Easier to just delete the old Highlight or Bookmark and create a new one that is your own.) Therefore, trying to change Highlights and Bookmarks that you didn't create should be trapped and should cause the warning message to display.

Faux-Delete

Imported annotations create clutter in a document and some are not worth keeping. While we will not allow people to change other people's annotations, we will allow them to faux-delete selected annotations from other people. Faux-delete means that the annotation is moved to a faux-delete file so it is out of the way, but can be retrieved if it is later determined that the annotation was actually important. Faux-delete is a new concept for us, but not new to computer users who understand that

anything placed in the trash can is really only faux-deleted. Faux-delete and can be very important as a CYA tool.

The method to faux-delete selected annotations is the same as what we use now: click on "delete" in the note or bookmark pop-up window, click on a highlight in the ReaderLens and click on "erase annotation" in the Doc-Doctor, or click on an annotation in Quester and then press the "delete" key. (Notice that this method of erasing a Highlight is not the same as what we do now. Having to trace over a highlight to erase it is annoying.)

Faux-delete will not appear to be any different on the surface than what we do now. But, deleted annotations will move to a "deleted annotations file" that has a user preference setting. The user preference setting will allow users to decide how much information to keep in the "deleted annotations file." When the file gets full, the oldest annotation in it will be discarded to make room for the newest annotation. Each document will have its own "deleted annotations file" and the preference setting to decide how many annotations to retain can be set uniquely for each file. The default setting should be 50.

To retrieve deleted annotations from the deleted annotations file, the user will click on a menu item in "Annotations" that says "Retrieve a deleted annotation." Retrieving deleted annotations is a one-at-a-time operation. A pop-up window will display deleted annotations for that particular document. The deleted annotations will display in a list with fields for all of the available meta-data. When a particular annotation is highlighted, a button on the bottom of the window called "preview" will show the full text of the annotation. Clicking on "Retrieve" will retrieve the specific annotation, place it in the document, jump the Quester, PageView and ReaderLens to that location, and highlight the retrieved annotation in Quester.

Faux-delete should apply to Expert Annotations too. When we get to designing the mechanisms for EAs, we'll need to keep faux-delete in mind.

Coding of Annotations

Therefore, we need a thread system that works when pieces of the system are missing.

What is disclosed scales infinitely, is simple to implement, does not require a radical change, and is unique. The model we developed is based on the concept of how DNA is modified when immunity to a virus is developed, and then how immunity is passed on to children. Children keep their immunity, and hence their link to their ancestors, even if their parents, siblings and children are “erased” from the family tree.

In our new system, each and every annotation is coded with DNA to make it unique. Just as the laws of physics prevent two objects that are identical from occupying the same space at the same time (see the movie Time Cop with Jean Claude Van Damme), we want to prevent duplication of annotations in the same document on the same computer at the same time because it leads to messiness and confusion. (Our current system violates this principle.)

Our current assumption is that each individual annotation is attached to particular words on a particular page of a particular document. Therefore, each particular annotation only has to be kept separate from other annotations attached to the same set of words on the same page of the same document. Going forward, we want a system that is more flexible than that. We want to be able to set up Notes that are associated with other annotations (threaded notes) and even Notes that are free standing. (A free-standing Note is like an email program, which is just what we may end up creating.) Therefore, the DNA for each particular Note needs to have enough complexity to make unique and unlikely to be duplicated by random chance. I’ve chosen to use five-character DNA in the examples that follow, but five characters are probably not enough. Doubling that number to ten characters allows a billion possibilities. Or, using hexadecimal to code DNA may be more efficient.

Highlights, Personal Notes and Bookmarks will contain only a single “strand” of DNA: a unique random code number. Each individual note in a Threaded Note (TN) will contain its own unique “strand” of DNA and it will also contain all of the DNA strands that show its lineage. Any method as know in the art may be used to create

random code numbers, how many characters each code number should have, and the combination of numbers and letters in each code is contemplated. It would seem that five characters is probably enough and seven characters is certainly enough, however any number may be utilized. These bits of DNA are, of course, not visible to users.

When a Personal Note is transformed to a Threaded Note through the act of clicking on one of the various forms of reply buttons inside the note, a new note is started. The new note contains all of the DNA (code number) of the previous note AND it is given its own piece of DNA. Further, each piece of DNA is kept separate and each piece of DNA is kept in sequence. The following examples illustrate how the system works.

PN from John A Doe that starts the TN. This note is sent to Abe Lincoln and Gary.

Level	Code
1	91827

Abe Lincoln's reply is sent only to John A Doe

Level	Code
1	91827
2	65748

John A Doe's reply sent only to Abe Lincoln

Level	Code
1	91827
2	65748
3	01592

Abe Lincoln's reply sent to both John A Doe and Gary

Level	Code
1	91827
2	65748
3	01592
4	40174

Gary's reply to John A Doe's original note, which is sent only to John A Doe

Level	Code
1	91827
2	09812

With only these five notes in the thread, we can construct a picture of what will show up on John A Doe's, Abe Lincoln's and Gary's computers.

John A Doe's computer will display this thread:

Level	Code
1	91827

Level	Code
1	91827
2	65748

Level	Code
1	91827
2	65748
3	01592

Level	Code

1	91827
2	65748
3	01592
4	40174

Level	Code
1	91827
2	09812

Abe Lincoln's computer will display this thread:

Level	Code
1	91827

Level	Code
1	91827
2	65748

Level	Code
1	91827
2	65748
3	01592

Level	Code
1	91827
2	65748
3	01592
4	40174

Gary's computer will display this tread:

Level	Code
1	91827

Level	Code
1	91827
2	65748
3	01592
4	40174

Level	Code
1	91827
2	09812

It is not difficult to see from this simple example that the system works, is infinitely scaleable no matter how many notes are involved, no matter how many participants are involved, no matter whether everyone has every note in every thread or not, and no matter how convoluted the sending and receiving of notes is.

Here's an interesting twist that creates a ton of new power... Take a Note that is deep within the tree of a complex Threaded Note and add a flag to it. While the Note is unflagged, it fits in the Threaded Note hierarchy as we've already defined. But, let's say the user wants to split the Threaded Note apart and use this specific Note to start a new Thread. No problem. When the user splits the Thread apart at this Note, we add a flag to this particular Note and retain all of its unique DNA. The flag tells the Reader to take this Note and display it as a new patriarch Note. Beneath it are all of its children. Because all of the Notes retain all of their DNA, the original Threaded Note is easy reconstructed if the flag is removed upon the users' request.

The examples shown only a single dimension of DNA for each note, but DNA is not confined to a single dimension. When Annotations are Categorized, it adds another dimension to the DNA. A Categorized Note may have DNA that looks like this:

Level	Code
1	91827
2	65748
3	01592
4	40174
South America	
Drugs	
Prowar	

Title: Software that enables complete tracking of threaded notes, dynamically adding expert annotations to documents, and a mechanism to determine first-occurrences of text strings in documents.

Problem: There is a need to determine the first-occurrence of text strings in electronic documents because the first-occurrence is usually where terms and phrases are defined.

Many documents, especially non-fiction written by industry, the military and government, academia and various institutions use language (acronyms, words, phrases) that is not common to the general public. In fact, this language is often obtuse even to those who use it, and particularly to people who are new to the organization. When a person reads a document without starting at the beginning, it is very likely that that person will come across language that is not understood. However, it is general practice for arcane language, particularly acronyms, to be defined when first used in a document.

Search engines typically find the “next” occurrence of a particular text string or “all” occurrences of a text string. This is done with the thought that the objective is to “replace” the string with a different string.

This object is not to aid in “editing” a document. Instead, it is done with the object of adding the reader in “understanding” the document.

Solution: Software that finds the first-occurrence of selected text strings in documents.

In an embodiment, the user “clicks and drags” over the selected language. Upon releasing the click, a menu window pops up on screen that allows the user to select from a number of different tools. One of those tools is “Find first occurrence.” (The reader will understand that the actual wording of the menu choices can be infinitely varied while meaning the same thing.) When selected, the software makes note of where the beginning of the document is and finds the first occurrence of the specified text string.

Another embodiment is to select the text string and then click a tool-bar button that invokes the “find first occurrence” function.

The reader will appreciate that the order of these two events is irrelevant: a) “selecting” the text string, and b) invoking the “find” function to locate the first occurrence of the text string.

The “find first occurrence” concept is most effectively linked with a traditional “find next occurrence” function because not all documents define terms with the first occurrence.

An important part of the “find first occurrence” concept is that the function keeps track of the location in the document where the function was invoked. When the user has found the definition of the selected text string, the user can immediately return to the location in the document where the function was invoked. Ending the function and returning to the location in the document where the function was invoked can be accomplished in a variety of equivalent ways. These methods include but are not

limited to pressing the escape key, or clicking on a labeled button. The reader will appreciate that the labeling of this button is not material to its function.

Problem: There is a need for “expert annotations” in electronic documents that can be dynamically and remotely updated.

The concept of “expert annotations” (EAs) is well established in “paper documents,” but not with “electronic documents.” With paper, a set of EAs is published and distributed. To change the EAs, a new paper is printed and distributed. With electronic documents, EAs that embedded in and part of the document suffer the same problem – updating the EAs requires distributing a new document.

Solution: EAs that are separate from but linked to the electronic document by means of a separate linking file makes it easy to dynamically and remotely update EAs, and maintain a history of the updates.

EAs can be either separate, stand-alone files, or they can be combined in a single file. In the case of stand-alone files, the linking file contains location information to correctly position the EAs within the base document, and EA file names to attach the right EA to the right place in the base document. In the case where the EAs are combined into a single file, the information to link the EAs to the correct location inside the base document can be part of the EA file, or it can be a separate file.

An advantage to this EA system is that to update specific EAs involves only those EAs rather than all of the EAs and the base document. A second advantage is that obsolete EAs do not have to be removed, they can simply have their linking information modified. A third advantage is that a running history of EA modifications can be maintained by keeping all of the EA updates with their linking information.

Problem: There is a need for a robust system to “thread” related notes together.

In email systems, people typically respond to notes by copying the note into an email reply, and then embedding their own notes within the previous note, and so on and so

on. After several iterations involving several people, the email is so lengthy as to be cumbersome. When this happens, people typically start a new thread which breaks the logical link to the previous thread and leaves behind all of the information in the previous thread. The thing that makes this system cumbersome is that an entire single email needs to be read and reread just to see a few words of reply.

The reader will appreciate that this problem is common to a wide range of electronic communication systems that involve comments, replies and counter replies.

A related problem is that when annotations are embedded in a document, sharing annotations requires sending the entire document back and forth. This works when only two parties are involved, but breaks down when three or more parties are involved because someone's annotations are frequently missing. When annotations are not part of the document and can be transmitted back and forth without sending the base document with them, the problem is that annotations get duplicated over and over when they are imported unless participants are careful enough to only send previously unsend annotations.

Solution: An electronic system of comment-reply-counter-reply whereby each note and reply is separate and distinct, and where each note and reply maintains a complete "DNA" log of its ancestry so that the thread can be dissected and reassembled accurately even if pieces of the thread are missing.

The design described has several key advantages: 1) the design keeps each note separate and distinct from every other note, but still connected with the correct thread; 2) the design prevents duplication when annotations are imported that contain some of the same annotations that are already in the importers' annotations, 3) the design distinguishes "threaded notes" (TNs) from "personal notes" (PNs); 4) the design allows an unlimited number of TNs, 5) the design accurately keeps track of and displays parent-child relationships between TNs, even when some links in the chain are missing, 6) the design allows users to delete spurious notes in the middle of a TN chain without messing up or confusing the thread; 7) the design allows users to easily send parts of a complex TN chain to other people and know that replies will still fit

seamlessly with the structure of the complex chain; 8) the design allows users to import missing pieces of a TN chain and have those pieces fit to their correct places within the chain; and 9) the design contains tools that allow easy display and manipulation of complex notes.

The user will appreciate that the design described herein can be applied to more than just "notes." The design can also be applied to highlights, bookmarks and any other kind of freestanding or linked information that is passed back and forth.

In order for ByteSize Systems to give our users a pleasurable and improved reading experience, we have known from the beginning that we must give our users various tools to duplicate, even improve, their current reading experience as compared to a hardcopy book. Readers currently take notes, use bookmarks and highlight text in their hard copy books. They can also flip pages, look in the Table of Contents, scan pages visually, always know what page they are on, photo copy pages, look in the index (if one exist), switch from book to book easily and quickly, and can even tear out pages if they choose.

Tools contemplated:

1. Highlight Text (using a variety of colors and have edit features like eliminating highlights, changing colors, or filtering by color, printing highlighted material)
2. Make Personal Notes (as well as export and import them to/from others)
3. Bookmark Pages (with capabilities to label them with a Subject line)
4. Change Font Sizes
5. Copy and Paste Text
6. Copy and Paste Text with Citation (and change the form of the citation)
7. Search (for words or phrases) (and display results in a variety of ways)
8. Look at Quester (Table of Contents) and via double click automatically go to a page
9. Print Pages
10. Import and Export Annotations
11. Skim (extract only the first sentence of each paragraph)

12. History Stack (automatically go backwards or forwards to previous pages examined)
13. Expert Annotations (Locked down, unchangeable by the user, upgrades can be imported)
14. Make Threaded Notes (as well as export and import them to/from others)
15. Navigate the document via Quester, via Page Up/Page Down buttons, via Previous Page button or Left Arrow, via Page Forward button or Right Arrow, Go to Page Tool, End and Home buttons, and Search List.
16. Know what page they are on and how many pages are in the document via Page Window.
17. Change the size, shape, and layout of windows on their computer screen.
18. Can open and close any window via buttons on the Tool Bar.
19. Can open and close multiple books or files easily and quickly.
20. Utilize HELP at any time while Reading to understand the tools and ByteSize software.

All of the tools available for ByteSize users are listed because some tools might be used often and some tools rarely or never used, depending on the user and the material read.

As we have worked to develop the software, we have found that some of the tools that are frequently used should be easy to find and use to increase the productivity and improved reading experience for the user. To increase the productivity of the user, we must find ways to eliminate key strokes or mouse clicks. Thus the idea for the PUMT was created.











How the Doc Doctor Works

In general when the cursor is in the ReaderLens and the user left clicks, the Doc Doctor is triggered and pops up on the screen and the cursor marks a spot in the text where some action will be taken by the user. As now, the Doc Doctor stays open and allows the user to select a tool or task from within the Doc Doctor.

The following changes are recommended from what we currently have:

1. The Highlight, Note, Bookmark and Text Select tools will be taken off the Tool Bar.
2. An icon appears to the left of each tool that appears in the Doc Doctor.
3. The Doc Doctor pops up below the line of text (or above the line of text if the line of text is within 2.0 inches of the bottom of the ReaderLens window) where the cursor is placed. (Currently our Doc Doctor box partially covers the line of text where the cursor was left.) (Abe Lincoln suggest that the top corner of the pop up box be minus 15 pixels in the x and y direction.)
4. The CANCEL command is removed from the Doc Doctor. The Doc Doctor is canceled or removed just by left clicking the cursor outside of the Doc Doctor box.
5. The user will have the preference setting to only have icons appear in the PUMT once they know what the tools are and the associated icons. The advantage of only having the icons appear is that the Doc Doctor would be smaller and not cover as much text.
6. We want the box around the Doc Doctor to be more noticeable or distinct. We would like to increase the weight of the border. Currently there are two light gray lines around the top and left sides of the box. The right side and bottom has one gray and one black line.
7. Use search feature to find the "first occurrence" of a given word or phrase (since most authors give the full meaning of an acronym the first time it is used) then our users could quickly find the meaning of an acronym. They could then use the History Stack button to go back to where they were reading/working previously. Additional tools to add include "Find Previous Occurrence" and "Find Next Occurrence". These quick tools give the user a quicker method for finding text in the document. The advantage is of putting these "search" items in the Doc Doctor since we already have a powerful search function are as follows: 1) SPEED because accessing and using "search" the other way requires a whole bunch of key strokes, 2) SIMPLICITY because this method of searching doesn't require any thinking at all, and 3) UNIQUENESS because nobody has a search function that works like this and it gives us sizzle.

The following Table will give an approximate example of what the Doc Doctor may look like.

ByteSize Doc-Doctor™	
	Add Highlight
	Add Note
	Add Bookmark
	Copy Text with Citation
	Copy Text
	Copy Graphic
	Find First Occurrence
	Find Previous Occurrence
	Find Next Occurrence
	Erase Annotation

Secondary Doc Doctor concepts

1. The user will have the ability to add, change, or delete any tools that appear in his/her Doc Doctor. We will have a Tool Manager that allows the user to set preferences for the tools that s/he uses most. If someone never highlights, then why would they want the highlight tool in their toolbox?
2. The user can choose to use their mouse or keystrokes to choose the tool. Because we know some people prefer to use keystrokes to mouse clicks we will give the option to set their preference in the Tool Manager. This feature would work as follows: Each command would have an associated keystroke. For example, Highlight, might be Control H. By hitting Control H, it would be the same as left clicking and then clicking on the Add Highlight command. Once I hit Control H, the highlighter would be activated.
3. The ability to immediately capture text and send it to a friend via two clicks of my mouse? Just like Carpenters have various tools that they take to certain jobs, we

might have various tool kits for certain professions. A graphic Artist needs tools that an Architect would not, and vice versa.

4. The user always knows exactly where the cursor is. The top left side of the Doc Doctor box is extended the height of the font in the line of text where the cursor was placed. (The bottom right side of the Doc Doctor box if the text selected is at the bottom of the page and the box has to pop up vs. down) This extended line will indicate to the user where the cursor was left. Another variation of this idea is to include the positioning of the Doc Doctor in the IMAGE BROWSER (IB) Manager. Once we have an IB Manager we can set the location and size of windows that pop up. The IB gives users the option to never have a window open over the ReaderLens so that the user can always see the text s/he is working with.

Actions Taken to Imported Annotations

There are numerous things that a person will want to do with imported annotations:

Read

Threaded notes can easily get too complex to display by clicking on the icon in the ReaderLens. Threaded notes are only linear when they are between two people. When threaded notes involve questions and answers from many people, the linear thread is replaced by complex branches like those of an old oak tree.

When comments flow back and forth freely, it is probable that people will get a lot of them. It is also probable that people will lose track of which imported notes they have read and which they have not read. It will be very handy for us to give people a way to mark notes as read or unread.

I think it is imperative that we find a way to identify notes as "read" or "unread." When notes are imported, they should all be identified as "unread." Further, we need to give users a handy way to find "unread" notes without going through the document page by page and note by note to find them. Once notes are read, we need to give users a way to further identify the notes as "read" or "follow up needed." One way to do this is by creating additional sort and filter fields in our annotations manager.

Read Later or Ignore

Everybody has and should have the right to do this, and to know when it has been done. Hence, the need to be able to mark notes as “read” once they have been read. We should design this function so that we can add to it with multiple types of flags, some that we predefine and some that are user definable.

Reply

Our current note box has a “reply” button on the bottom border. A PN is transformed to a TN when “reply” is clicked. In our current architecture, clicking “reply” inside a note adds two carriage returns to the bottom of the note, adds thread info (user name, date, time), and then allows the use to type a reply that simply adds text to the original note. This architecture fails as soon as a recipient imports it because it creates duplicate notes. It also fails for the same reason that multiple people adding to the same shared email message fails – it gets hopelessly messy and defeats meta-data classification.

In the new architecture, clicking “reply” begins a new note that is a child of the parent. In the new architecture, a user can click “reply” to any note within a complex TN and a new note will be created that is a child of the note to which he replied. To be clear, “reply” causes a new note box to pop up. The header in the new note box (previously explained) will explain the relationship of the new “child” note to the “parent” note.

There will be times when a person starts a reply, but doesn't finish it. We need a way to mark replies as “unfinished” so they are not mistakenly sent prematurely. The button in the note window to mark notes as unfinished will say “Mark note as UNFINISHED.” Unfinished notes will NEVER be included in MUIs that are exported. An additional meta-data sort field needs to be added to annotation filters called “Unfinished.” Clicking the Unfinished button again will clear the Unfinished marker.

Forward

There are some annotations that a user will want to forward to someone else without making a direct response.

To forward an imported note to another person, click in the "To:" field in the header and use the pop-up list of names to enter a name. Selected names will be added to the names already in the field.

Filter and Sort by Icon

Our filter function works on a variety of fields right now. To this, we need to add filtering and sorting by icons.

Respond to Someone Other than the Author of a Note

A note from Bob may give me what I need to respond to Clyde. The third field in the header of a note allows the author of a note to determine to whom the note is written. Eventually, we will want to build in a "permissions" function to allow users to control distributions of certain sensitive annotations.

Personal Comment

Some imported annotations will cause a user to want to write a "personal comment" rather than a "reply." A "reply" is something that one intends to send, either to the author or to someone else. A "personal comment" to an annotation needs to retain thread information, but it is not something that a user intends to send to anyone else. This distinction is important when viewed in the context of what the "send replies" function does.

To create a personal comment Note to an existing Note, click on Reply to open a new Note in the thread. If the "To:" field is left blank, the new Note is a personal comment because it is not sent to anyone.

In essence, a "Personal comment" becomes a personal note (PN) within a threaded note (TN), but it is also different because it requires the context of the parent note rather than just the document content to make sense. For this reason, "Personal comment" should be added to the meta-data list and should become a sortable and filterable field.

Eventually, to be really powerful, we would like to give users the ability to create Notes that can be accessed from inside the ByteSize Reader, but they are free-floating and unattached to specific documents.

Save

Saving of imported annotations should be prompted immediately after importing with this message.

Save imported annotations now or wait until later? Now Later

Delete

Imported annotations create clutter in a document and some are not worth keeping. While we will not allow people to change other people's annotations, we will allow them to faux-delete selected annotations from other people. Faux-delete means that the annotation is moved to a faux-delete file so it is out of the way, but can be retrieved if it is later determined that the Note was actually important to retain. Faux-delete is a new function. The method to delete selected annotations is the same as what we use now: click on "delete" in the note or bookmark pop-up window, click on a highlight in the ReaderLens and click on "erase annotation" in the Doc-Doctor, or click on an annotation in Quester and then press the "delete" key. (Notice that this

method of erasing a Highlight is not the same as what we do now. Having to trace over a highlight to erase it is annoying.)

Change

As we are now, any user can change any annotation on his computer, regardless of who created the annotation. The new system will not allow changing annotations created by another user. This will be done by comparing the "user name" in preferences with the meta-data "user name" attached to annotations.

Not allowing changes to annotations applies only to "changes." We will allow users to "delete" annotations created by other users, we will allow users to change their own annotations, we will allow users to copy-and-paste from any annotation, and we will allow users to create new annotations.

Identification numbers of annotations will be compared to prevent duplication.

In the system described in this proposal, each annotation is uniquely coded. If an imported annotation has the same location code as another annotation in the document (meaning that it would reside at identically the same place in the document), then the unique identification codes of the annotations are compared. If the two identification codes are different, then the import is allowed to happen and the two annotations will reside at the same place in the document. However, if the two identification codes are the same, then the import of that particular annotation is bypassed without notification.

When an annotation that has the same identification code as another annotation is bypassed for import, the assumption is that the annotation already in the document is the correct one to retain. This assumption would be invalid if we allowed people to change each other's annotations. So, when we disallow changes to other's annotations, it is valid to assume that the annotation already in the document is the correct one to retain and the imposter being imported should be bypassed.

This technique will prevent duplication of notes, highlights and bookmarks in documents when users regularly import annotations from other people.

Preview Without Importing Annotations

The way our software currently works is this: You send me an MUI with annotations. In a "threaded world," most of the annotations in your MUI will be PNs or TNs. To see the notes in your MUI, I open the corresponding document and import your MUI. This is a document that I use a lot, so it already has dozens of annotations in it. In fact, it already has a bunch of annotations from you that you previously sent to me, and from other people who sent MUIs to me. This new MUI from you that I just imported has TNs that you wrote and it has notes from other people because you had to involve other people to answer some of my questions. Without looking at every annotation in the document, how do I see just the new stuff that you sent to me? I could filter to see only annotations from you, but that will include old notes from you that I've already read, and it will miss notes from others that you included in the MUI you sent. This is messy, complicated, time consuming, frustrating and subject to error. Further, what if I try to do some housekeeping as I read and decide to delete non-meaningful notes from your MUI as I read them, and then later realize that what I thought was meaningless turned out to be meaningful? Oops.

"Preview without Importing Annotations" is a new feature that works like this: You send me an MUI with annotations that are mostly notes. I click on "Preview without Importing Annotations." A pop-up asks which MUIs I want to preview. I put check marks by the ones I want to preview and click "ok" to open. All of the annotations currently open (mine) are temporarily hidden. The MUIs that I checked open in the document. The display of Quester is temporarily changed so that it only displays the outline at the root level which means that all of the annotations in the new MUIs that I checked are visible without opening any more levels of Quester to hunt for them. I can read the new annotations by scrolling through Quester. I can do several things besides reading while in this preview mode:

Double-Click on Annotations in Quester to See Them in Context

Double-click on an annotation in Quester, and PageView and ReaderLens jump to that place (just like they do now). Annotations display in the ReaderLens with icons, note content pops up when the cursor rolls over the icon, and a note box opens if the icon is clicked on.

Remove without erasing specific annotations

There will be times when I preview a set of annotations from you that I see a particular note that, while perhaps funny and entertaining, is not something I want to actually import and keep. However, I probably also do not want to permanently delete it. The answer is to remove it from the preview without changing the MUI that it came from. When I "Quit Preview and Import," only the visible annotations will import and be saved. But, if I ever want to go back and see all of the annotations you sent, all I have to do is go back to the unchanged MUI file you sent to me.

Quit Preview

When I click on Preview in the menu, the pop-up offers two options: 1) "Quit preview without importing," which dumps me from the preview mode, closes the MUIs that I was looking at without importing them into my own MUI, reveals the annotations I was looking at before I activated the preview mode, puts Quester back to the way it was opened before I activated the preview mode, and returns PageView and ReaderLens to where they were before I activated the preview mode. 2) "Quit preview and import" dumps me from the preview mode, imports the visible annotations and merges them with my annotations, saves the new merged MUI to disk after confirming, reveals the newly imported annotations along with my own annotations (in the current filter configuration), puts Quester back to the way it was opened before I activated the preview mode, and returns PageView and ReaderLens to where they were before I activated preview mode.

There are many style issues that could be addressed in a “Style Manager” (SM) for the ByteSize Reader. However, to keep the concept simple for users, and to define an architecture we can get implemented quickly, this first iteration of a “Style Manager” is pretty simple. This first iteration is something that can be improved in successive generations.

There are five primary objectives for the first SM: 1) a greater degree of differentiation between header elements, 2) more white space in the ReaderLens to make reading easier for young readers and people with cognitive disabilities, and to make the ReaderLens a better presentation device, 3) more selection of layout and formatting styles so that people set up the ReaderLens in a style that is most comfortable for them, 4) more control over extracting from the PageView “exactly and only” what is to be displayed in the ReaderLens and then more control over minute styling in the ReaderLens to make the ByteSize Reader a “Dynamic Presentation Device,” and 5) greater reading comfort, pleasure, comprehension and speed.

Objective #1 – Differentiation Between Header Elements

This first iteration of the SM uses two techniques to increase differentiation between header elements:

- Allowing the user to set the number of “points” between header levels. The default is “2-points” between header levels, but the SM allows users to set the points between header levels at an integer between one and five.
- Allowing the user to put flags in front of headers. When “on” this setting puts triangle symbols to the left of headers. It puts one triangle in front of a Header1, two in front of a Header2, three in front of a Header3, and so on.

Objective #2 – More White Space

The “white space” objective speaks for itself and for this first iteration of the SM is easily accomplished in with two settings: 1) allowing users to insert blank lines between elements, and 2) allowing users to single space, 1.5 space or double space content in the ReaderLens. The tradeoff regarding blank lines between elements is that more white space (blank lines) improves readability, but also decreases the amount of content that can be displayed in the ReaderLens. The ReaderLens default may be zero blank lines between elements to maximize the amount of content that can be displayed. In general, the higher a person’s “reading level” the less white space is required for a document to be readable. Given that most people read at an 8th grade level or less, I think we need to give people the option of adding more white space. Children just learning to read need a lot of white space.

Objective #3 – Select Layout Style

The “layout style” takes a bit of explanation. The printed material I read follows three fundamentally different layout styles (I made up the names for these three styles. With a little thought, we can probably come up with better names. On the other hand, names may not be terribly important because users can simply select one, see what it looks like, and then select a different style to see what it looks like. I think they will quickly figure out which style works best for their needs.):

Book Style

In “book style,” the style that the ByteSize ReaderLens uses now, all paragraphs are weighted the same. It is possible to tell one paragraph from the next by the fact that the first line is indented five spaces.

Outline Style

In “outline style,” the style more common to business writing, first lines of paragraphs are not indented. Instead, paragraphs are arranged in “outline” format with children paragraphs indented five spaces from parent paragraphs (our current indent is three

spaces). Children headers are indented below parent headers. Paragraphs and headers are separated by one blank line.

Legal Style

A third layout style is what lawyers use: headers are centered, first lines of paragraphs are deeply indented all the way to the center of the page, and all lines are numbered with the numbers to the left in a separate column so that text does not flow beneath the line numbers. Line numbers should correspond to lines in the PageView rather than lines in the ReaderLens since ReaderLens lines change dynamically as a function of font and window size. There is one blank line between elements.

In this first iteration, I could be happy if we let users “check” which of the three styles they want to see and then we let defaults take over from there.

Objective #4 – Dynamic Presentation Tools

ByteSize can be a very effective presentation tool because it can display information well and adapts dynamically to the presentation. PowerPoint gives the user a high degree of control over content selection, white space, styling and special effects, but it would be extremely awkward to use PowerPoint to dynamically adapt to changing content on the fly during the middle of a presentation. The ByteSize Reader is already well suited to dynamically finding and adapting to changing content on the fly, but it does not give the user a high degree of control over styling or displaying “exactly and only” selected content. The ReaderLens could become more like PowerPoint if the user had more control over white space, more control to select exactly and only the content to be displayed in the ReaderLens, more control to select and display noncontiguous bits of content, and more control to style content that is extracted to the ReaderLens.

“The ByteSize Reader as a Dynamic Presentation Device” is a whole new concept. The concept of using the ByteSize Reader as a presentation device is simple, but powerful:

- Give users tools to extract “exactly and only” selected content from the PageView for display in the ReaderLens. What does “exactly and only” mean? It means being able to extract a few characters, a few words, a sentence, a paragraph, or a graphic, and only that much.

Another way to get to “exactly and only” what you want to display in the ReaderLens is to extract more than you want from PageView and then “erase” what you don’t want displayed. This needs to be part of the tool set too.

Give users tools to extract “noncontiguous” content from PageView into the ReaderLens. “Noncontiguous” means things that don’t sit next to each other. Right now, our Reader extracts in a purely linear fashion. That is, one paragraph after another. As a presentation tool, we need to give users the ability to extract and display “exactly and only” what they want from one section of a document, do the same from other sections of the document (or even from another document), and have them all display in the ReaderLens at the same time.

Give users tools to change the order of content that is displayed in the ReaderLens. As a dynamic presentation device, presenters may want to change the order of what is displayed in the ReaderLens. I imagine two different tools to change the order of what is displayed:

Highlight-drag-drop. This is the conventional way to change the order of elements. Click and drag over content to select it, click and drag it to a new location, and drop it in the new location by releasing the mouse. This method works, but it is cumbersome if you want to reorder a lot of elements quickly during the middle of a presentation.

Click-to-add-number-and-sort. This is a new way to change the order of items displayed in the ReaderLens that is much faster than the highlight-drag-drop method. When this method is activated, the user clicks in front of displayed elements in the order that he wants them displayed, and a series of little numbers pop up on the screen in front of the elements. For example, if I clicked on the third element first, a #1

would pop up in front of it. When the user has clicked all the elements he wants to reorder, he then clicks on "reorder" and the elements sort according the numbers and reorder themselves on the screen. To make this method work well, I would put an icon in front of every element when this tool is activated. Then, when the user clicks inside the icon to establish order, the icons would change to include numbers. That way, the user could easily distinguish elements and know what has been ordered and what hasn't.

Give the user a high level of control over how extracted content is styled in the ReaderLens. This styling will include changing color, changing fonts, highlighting, underlining, displaying as text that reflows or graphic that does not (giving the user the choice between displaying with "reflow" or "literally" is particularly important for lawyers), adding graphic symbols such as boxes and arrows and drawing tools to draw attention to particular words or parts of a graphic. The user should be able to control these effects on a "character" basis rather than just on a global basis. Advanced styling would even include the ability to move and reorder what is displayed in the ReaderLens.

Give the user tools to add "in-line" notes. Right now, our note tool creates notes in boxes. When these notes are closed, they display as icons. When they are open, they display in boxes that sit on top of and cover up the text and graphics displayed in the ReaderLens. We need to give users a tool to display notes "in-line" with the content in the ReaderLens so that it doesn't cover anything. "In-line" means that the note text will simply flow across the screen as if it were extracted text, but it will be styled in a way to distinguish it from extracted text. The user should have control to display these in-line notes so that they can either display in-line, in boxes like we display notes now, or as icons that show location but do not display content.

Why is this a powerful idea and who would use it? "Expert presentations" just like the one we did to architects in Texas. Continuing education is something most professionals have to do. Giving presenters a "dynamic presentation device" opens up new possibilities and would give ByteSize huge exposure. Lawyers could use "ByteSize Presenter Tools" to display discovery documents that are used as evidence.

Teachers and professors could use ByteSize Presenter Tools to display and teach from a textbook in class. This last idea is, potentially, the biggest. One of the problems I keep running into when discussing ByteSize with students and professors is that none of them particularly want to have a bunch of notebook computers open and running during class, but students need to be able to “follow along” in the book during class. The ByteSize Presenter Tools allow students to follow along without having their own computers up and running during class.

Objective #5 – Greater Reading Comfort, Pleasure, Comprehension and Speed

Reading for pleasure is what a person does on a chilly and rainy afternoon curled up in a window box with a good book, a cup of hot chocolate in hand, a golden retriever at foot and a cheery fire glowing softly in the background. The type of content that will be ByteSized may be diametrically opposed to achieving this level of comfort and pleasure, but it is still our goal.

On a different scale, but just as important to achieving “satisfaction” from reading, are comprehension and speed. No matter how comfortable and pleasant the reading experience is; if comprehension doesn’t happen, frustration and disappointment are the result because the time and effort were wasted. When “learning” is the objective, “time” is always an issue because learning faster is always better.

I think the tools built into this first version of the SM just begin to scratch the surface with regard to “greater reading comfort, pleasure, comprehension and speed.” As we learn from the people who read ByteSized material, we will develop new and better tools to make reading better for them. For that reason, I see the SM as a feature that will continue to evolve.

This first iteration of the SM has tools that I think will make reading better: the “layout styles,” the number of blank lines between elements, line spacing, font point size difference between headers, font selection, triangles to distinguish header levels, and rotation of sentence colors.

Style Manager Controls

Following is the proposed layout and functionality for this first iteration of the SM. The idea with “B” is that it is an “override.” That is, the default of A-1 is zero blank lines between elements, but the user can override that to include up to five blank lines if desired. The default for A-2 and A-3 is one blank line between elements, but the user can override that by typing something else into B. “C” through “L” are also overrides. The default for A-1 is to indent five spaces, the default for A-2 and A-3 is to indent zero spaces. We need to save people from themselves by building in the following boundary conditions: If A-1 is selected, then the minimum for C is 3 spaces. If A-2 or A-3 is selected and B is set to zero, then the minimum for C is 3 spaces.

ReaderLens Style Manager		
	Style Item	Style Choices
A	Layout Style 1: book (default) 2: outline 3: legal	1
B	Number of blank lines between elements (0-5)	0
C	Number of spaces to indent first line of paragraphs (0-5)	5
D	Line spacing (single, 1.5, double) (default is single)	Single
E	Scale graphic elements (5-200, default is 50)	50
F	Nominal number of words to display (1-500, default is 150)	150
G	Font point size difference between header levels (1-5, default is 2)	2
H	Font to use for body text (default is Times)	T
I	Font to use for headers (default is Arial)	H

J	Add ◀◀ to distinguish headers in ReaderLens (default is "on")	<input checked="" type="checkbox"/>
K	Eliminate end-of-line hyphens (default is "on")	<input checked="" type="checkbox"/>
L	Rotate sentence colors (default is "off")	<input checked="" type="checkbox"/>

Further details on displaying information is contained in U.S. Patent Application 10/691,927 filed 22 October 2003 (22.10.2003), the disclosure of which is incorporated by reference.

Claims

We claim:

1. A method for viewing electronic information comprising the steps of: displaying in a first window a physical page from an electronic document containing information from a predefined page format, wherein the electronic document comprises representations of at least one physical page, and a visual reference emphasizing information on the at least one physical page, extracting the information emphasized by the visual reference on the at least one physical page, presenting the extracted information in a second window and presenting a navigation tool in a third window, attaching a threaded note including DNA.
2. The method of claim 1 wherein the physical page is represented in an electronic page view.
3. The method of claim 1 wherein the physical page is represented as an icon including a thumbnail of the physical page.
4. The method of claim 1 wherein the first and second window are the same window.
5. The method of claim 1 wherein the second window is an enhanced interactive window including a thumbnail image of a physical page, a graphic image of a physical page, text, free flowing text, icons, hyperlinks, menus, and control elements.
6. The method of claim 1 wherein the navigation tool presents an extraction of content and annotations.
7. The method of claim 6 wherein the step of selecting further comprises the step of enclosing the annotation with a box.
8. The method of claim 1 wherein the second window may be placed in various positions relative to the first window, including on top of the first window, adjacent to the first window, and partially covering the first window.

9. The method of claim 1 wherein the second window further comprises a control panel for managing the extracted information.
10. The method of claim 1 wherein the extracted information may be viewed simultaneously in a multiple of enhanced interactive windows.
11. A computer system for viewing electronic information, the electronic information comprising textual and graphic data of electronic books and documents, the system comprising: an information manager for manipulating graphic images of physical pages and annotations bounding electronic information from the physical pages, an enhanced interactive window for displaying the electronic information, and a navigation tool for manipulating the electronic information in a third window, attaching a threaded note including DNA.
12. The system of claim 11 wherein the enhanced interactive window comprises at least one window that displays extracted information from the physical pages.
13. The system of claim 11 wherein the enhanced interactive window includes a thumbnail image of a physical page, a graphic image of a physical page, text, free flowing text, icons, hyperlinks, menus, and control elements.
14. The system of claim 11 wherein the information manager further comprises a database for storing annotations, extracted electronic information, and the relationships in the system.
15. The system of claim 11 wherein the database includes a structure tree for storing relationship information associating electronic information with extracted electronic information.
16. The system of claim 11 wherein a visual reference emphasizing the electronic information is displayed on the physical pages.
17. The system of claim 11 wherein the annotations and the electronic information are

stored in separate files.

18. The system of claim 11 further comprising a scroll bar based on logical increments.

19. The system of claim 11 further comprising an acronyms manager, a citation manager, an icons manager, or style manager.

20. A computer system for viewing electronic information, the system comprising: a visual display for displaying an enhanced interactive window and a navigation window based upon electronic information in a page description format; a storage medium for storing and retrieving information related to the electronic information in the page description format; a computer processor coupled to the visual display and to the storage medium for accessing and processing information stored in the storage medium to provide a display of the enhanced interactive window and the navigation window; an input means coupled to the computer processor for entering information related to the electronic information in the page description format; a software portion for creating a plurality of relationships between the enhanced interactive window, the navigation window, and the electronic information in the page description format; and a software portion for providing a graphical user interface for navigation and display and a method to encode a threaded note including DNA.

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US05/12700

A. CLASSIFICATION OF SUBJECT MATTER
 IPC(7) : G06F 17/21
 US CL : 715/512
 According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED
 Minimum documentation searched (classification system followed by classification symbols)
 U.S. : 715/512

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2003/0070139 A1 (MARSHALL et al) 10 April 2003 (10.04.2003), see entire publication	1-20
X	US 6,389,434 B1 (RIVETTE et al) 14 May 2002 (14.05.2002), see entire patent	1-20
X	US 6,687,878 B1 (EINTRACHT et al) 03 February 2004 (03.02.2004), see entire patent	1-20
A	US 6,105,055 A (PIZANO et al.) 15 August 2000 (15.08.2000)	1-20
A	US 6,484,156 B1 (GUPTA et al) 19 November 2002 (19.11.2002)	1-20
A	US 2003/0061028 A1 (DEY et al) 27 March 2003 (27.03.2003)	1-20
A	US 2003/0023679 A1 (JOHNSON et al) 30 January 2003 (30.01.2003)	1-20
A	US 2002/0010707 A1 (CHANG et al) 24 January 2002 (24.01.2002)	1-20
A	US 6,279,014 B1 (SCHILIT et al) 21 August 2001 (21.08.2001)	1-20

Further documents are listed in the continuation of Box C. See patent family annex.

* Special categories of cited documents:		
"A" document defining the general state of the art which is not considered to be of particular relevance	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"B" earlier application or patent published on or after the international filing date	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"O" document referring to an oral disclosure, use, exhibition or other means	"&"	document member of the same patent family
"P" document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search 11 July 2005 (11.07.2005)	Date of mailing of the international search report 22 JUL 2005
----------------------------------------------------------------------------------------	-------------------------------------------------------------------

Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US Commissioner for Patents P.O. Box 1450 Alexandria, Virginia 22313-1450 Facsimile No. (703) 305-3230	Authorized officer 741 Doug Hutton Telephone No. 571-272-4137
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US05/12700

C. (Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 6,571,234 B1 (KNIGHT et al) 27 May 2003 (27.05.2003)	1-20