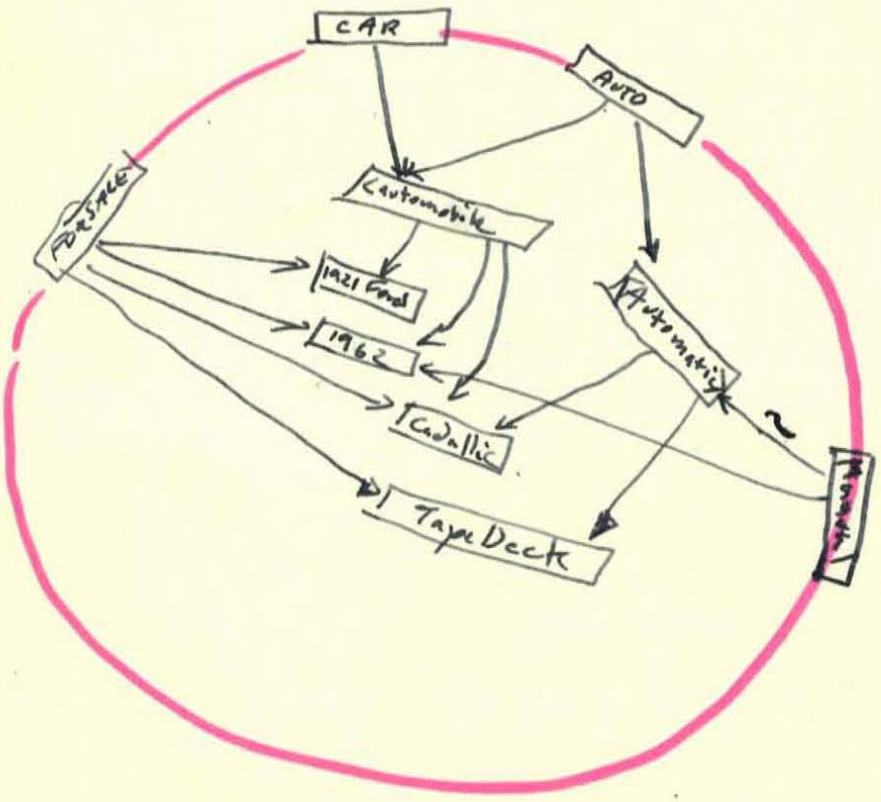
The Use of Indirect Referencing for Community Memory.

This is from some old work philip and I did on the original "database" system for provos in 1978. The original scheme was:

The system is a network of items some of which have an "external" world identity and are called handles. These items can be found via their exterior identity. All other items must be found by tracing the network from the exterior items inward.



Some of the interior items would be data, while others act as place holders for ~~omega~~ concepts. The system can be ~~a~~ enriched by having a variety of link types. The ~~two~~ three most likely such links are.

~~We~~ Opposite, Manual and Automatic in the diagram are opposites.

Similiar Meaning, such as the links between Auto and both Automatic and Automobile

More Specific, a link from ~~a~~ Automatic to Automatic Transmission would be of this type.

There are two kinds of approaches to this kind of problem. The Semantic Network, or knowledge Representation approach of AI opens up the possibility of English Language retrieval and the blending of facts gathered from different sources. ~~Perhaps the two~~ This is both an area of current research and ~~one~~ would create complex, difficult to see as mechanical (see Computer Power & Human Reason behavior.

The other approach is a database oriented one. In this view the handles and links are seen as a means of organizing, synonyms, ~~and~~ antonyms, and words with multiple meanings.

There ~~can~~ seems to be a powerful middle ground. Here the handles are seen as descriptors to do a variety of different indexing schemes:

① ordinary keywords

② keywords with synonyms + antonyms

③ multiple meanings

④ concept based systems of several types including world tree's and topics

⑤ human assisted cross-referencing such as reading lists refined concept indexing, bibliographies, and volunteery censorship.

⑥ automatically created cross-referencing such as cairl's or a more complex factor analysis scheme.

Review of the idea:

(1) separate data base into two kinds of records: data records and index records.
(later we can let the sharp distinction blur, but for the moment leave it)

(2) Index records can carry displayable text, however only index records can be found by any kind of search.

(3) (a) When an index record is found the system extracts the hit list and see's if any items on the next level are also ~~indices~~ indices, If they are the user gets to see them and choose among them.

(5) ~~This~~ Some of these index trees could be automatically resolved by indexing aids.

(4) The outer layer of index records can be found either by a "unique" title or by descriptors.

(5) This mechanism would ~~automatically~~ give us in a single package search by keyword and world tree structures and topics. It can also be used to give semantic grouping. That is each identified idea could have a single index record such as "record cleaner". All records related to ~~vacuum~~ "record cleaners" would hang off of this concept record.

The searching process would than be to find the right collection of concepts rather than the end data.

Hitlists, Comments and Indices

Efron

This is an attempt to unify a number of very useful features within a simple mechanism. The basic idea is to have two hitlists, a primary hitlist and a secondary hitlist. The primary hitlist is actually a stack of hitlists. Most work is done with the primary hitlist, The secondary hitlist is just a place to accumulate items for latter use (such as display, printing or indexing). In a future implementation there may be many secondary hitlists, but one will do for now.

The general flow of work is to do a search, the results go into the primary hitlist; Narrow the search criteria, reducing the primary hitlist; manually select interesting items from the primary hitlist into the secondary hitlist; Look at the comments on a primary item and select a few of them into the secondary list; Go back to the commented item and look at the items it indexs; select a few of them into the secondary list; Go back to the main line and select a few more items. Now exchange the secondary and primary lists and print the primary list.

Details: Every item contains four critical columns possibly empty -

Commented-by - 1) Row_id's of Comments on this item.
Comments-On - 2) Row_id's of items this item comments on.
Indexes - 3) Row_id's of items this item indexes
Indexed-by 4) Row_id's of items that index this item.

Index Column behaves very much like the Comments column and provides a means by which Gatekeepers, data shepards, and speed freaks can create items which organize other items, such as an item that lists the best puns in the database. To create an index item, one first collects in the secondary list all the items to be indexed. Then one enters the item which is the index, tells why all these items are gathered together, and give it appropriate key words. Then you push the Make Index button which appends all the Row_id's for the items on the secondary list onto the "Indexes" Column.

To see the items which an Index References, just push the See Indexed button. The current primary index will be pushed down and a new Hitlist will be created from the "Indexes" Column of the current item.

Items can index - index items down to any depth. The ~~see in~~ See Indexed Items button will keep taking the user down the tree.

To go back a single level there is a Go Back button, and to get the very first hitlist back there is a Go All The Way Back button.

Comments would work in a similar manner. When viewing an item the user would push the Add Comment button to comment an item. To see the Comments on an item the user would push the See Comments button. Like [redacted] indexes Comments could be nested to any depth. When the See button is pushed the current hitlist is stacked and a hitlist make from the "Commented-By" column. The Go Back and Go All The Way Back buttons would be used to unwind both a stack of comments and a stack of indexes, or a mixture of both.

The basic operations on the secondary hitlist are:

1) Clear the list
2) Exchange the secondary and primary lists
3) Copy an item (the current) from the primary list.
4) Copy the primary list to the secondary list.
5) Make an index of the list
   (All copies add to the list, nothing is destroyed)

The operations on the primary list:

1) Clear the list
2) Add a comment to the current item.
3) for experts only - Add a comment to all items in list
4) Delete an item from the list.
5) Copy the secondary list to the primary list.
6) See the indexed items of the current item.
7) See the comments on the current item.
8) Go Back a level (pop a hitlist)
9) Go All The Way Back (pop the stack)
10) Print the list

All operations (except 3) should have a button.

Note the use of the current item concept needs a good browse mode.

Keywords, Topics and World Trees:

I think keywords should considered the main means of finding information, but it can be very usefully supplamented by a well thought out beiarchy of topics (world tree).

The two major problems with keywords are spelling and picking the right ones. A powerful aid on entering an item is showing as each keyword is entered the number of times that key word already is used and giving the author the opportunity to cancel the use of that keyword. This is basic.

A more difficult aid would be showing synonyms and the frequency of their use when ever a key word was entered.

An even more complex aid would be showing other keywords which are found in association with the entered key word.

On the retrieval side, stem reductions and ~~Index~~ Soundex like codings would increase the number of hits, in an easily implemented manner.

The use of synonyms at this stage would be more complex, but helpful.

The most powerful, but controversial technique is the reduction to a conceptual code based on one or more key words + Something to experiment with later.

Topics,

> Having ~~some~~ users start by entering a topic is very powerful, because it provides an easy mechinism to provide the user with either a special form or a special front-end. The problem is not the impossibility of spanning the universe in a single set of topics, but finding a graceful way of handling items which belong under several topics.

Topics must start with a tree structure in mind.

Topics provide a fast way means into commercial charges for CM. To see the price field or retrieve using it, an item must be classified commercial sale or service etc.

Searching

The "Find leaping green lizards" of the original CM system is far superior to filling out templates for CMish purposes so I suggest we work from the original augmented by a few fields like price and date. - should price & date be a "topics" issue

A large number of tools, like a thesurious would be handy here, but I think it best to start simple.
If topic/world tree information is collected first a simple form like

topic: pets
keywords: leaping green lizards — implicit "AND" — how is the input delimited
price: < 10.50

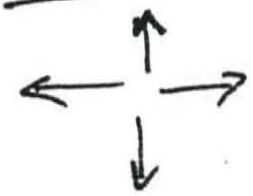
could be displayed.

There should be search AND OR NOT keys

or something similar.

Editing - 1

Motions



Prior Field Prior Screen Prior Item
Next Field Next Screen Next Item

* character motions, scroll the screen through the item, but not onto next item.

Edit Keys

Delete Character
Back space
Delete Line ..

Insert Blank Line (below) no args.

(Insert Toggle) hidden for experimentation (think insert should always be on, in basic editing.

Display Modes

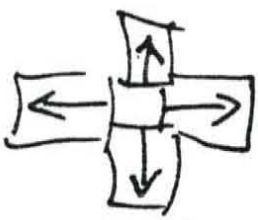
It is a theological mistake to think of the sequitor Entry system when trying work out display modes for CM; they serve different gods.

CM needs a good browse mode and a complete display mode.

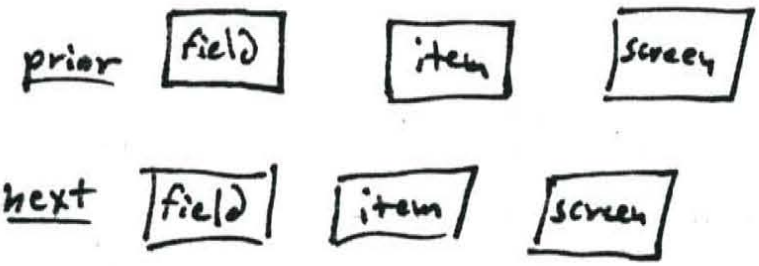
In both modes it is desirable to see the number of comments, The ~~value~~ number of indexed items, and the keywords or "topic / world tree" info.

There seems little, general to ~~said~~ be said about complete display mode except That cursor motion should consistant with browse mode.

Suggested Motion Buttons



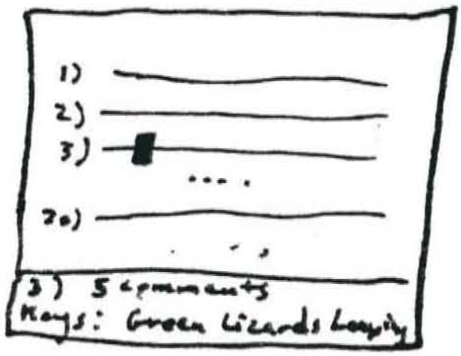
single character motions, without boundaries, hiting edge scrolls screen.



This one is trouble. "what's a field daddie?"

Display Modes <

Browse Mode could have two windows, the main window would display the first line of each item. The other window would display some more info the current item (The one the cursor rest upon), including number of comments, number of indexed items, its keywords, and possibly another line or two. This four or five line window should be maintainable at 1200 baud with a little cleverness.



After a search the ~~ignore~~ items would be immediately displayed in browse mode.

so we need a ~~Flip Mode~~ Flip Display Form button or a two button labelled

Show All

Browse

CM Row

Columns for CM Row—

Author:
Password:
Text:
Extended Text:
Key.words:
Topic:
Commented-By:
Comments-On:
Indexes:
Indexed-By:
* { Date Entered:
  { Last Referenced:
Date Expires:
* { Last Edited:
  { Number of References:
  { Site Entered:

{ Date:
{ Money:

* { possibly
  { # references by site
    : terminal
    : computer accessed

---

* this information could also be captured with a good audit trailer.

Other Problems —

Multi user - is not a significant problem because at first there will be no editing of items and when there is only the first person to claim an item for editing gets it. Administrative operations would lock whole files.

Ownership — Provide a cyphered password column
The password is specified at time of entry and only someone knowing the password could delete or edit the item, unless they were "mighty mouse."

In the future, registered owners would have a single password associated with their name, not item. This is necessary for people to give stewardship of items to others.

Analysis — A complete input stream should be captured from each terminal. later programs can be written to parse it.

Items should record the number of times they have been referenced.

Keywords, Topics and World Trees:

I think keywords should considered the main means of finding information, but it can be very usefully supplimented by a well thought out heiarchy of topics (world tree).

The two major problems with keywords are spelling and picking the right ones. A powerful aid on entering an item is showing as each keyword is entered the number of times that keyword already is used and giving the author the opertunity to cancel the use of that keyword. This is basic.

A more difficult aid would be showing synonyms and the frequency of their use when ever a keyword was entered.

An even more complex aid would be showing other keywords which are found in association with the entered keyword.

On the retrieval side, stem reductions and ~~Soundex~~ Soundex like codings would increase the number of hits, in an easily implemented manner.

The use of synonyms at this stage would be more complex, but helpful.

The most powerful, but controversial technique is the reduction to a conceptual code based on one or more keywords + something to experiment with later.

Topics,

Having ~~some~~ users start by entering a topic is very powerful, because it provides an easy mechinism to provide the user with either a special form or a special front-end. The problem is not the impossibility of spanning the universe in a single set of topics, but finding a graceful way of handling items which belong under several topics.

Topics must start with a tree structure in mind.

Topics provide a fast way means into commercial charges for CM. To see the price field or retrieve using it, an item must be classified commercial sale or service etc.

Hitlists, Comments and Indices

This is an attempt to unify a number of very useful features within a simple mechanism. The basic idea is to have two hitlists, a primary hitlist and a secondary hitlist. The primary hitlist is actually a stack of hitlists. "Most work" is done with the primary hitlist, The secondary hitlist is just a place to accumulate items for latter use (such as display, printing or indexing). In a future implementation there may be many secondary hitlists, but one will do for now.

The general flow of work is to do a search, the results go into the primary hitlist; Narrow the search criteria, reducing the primary hitlist; manually select interesting items from the primary hitlist into the secondary hitlist; Look at the comments on a primary item and select a few of them into the secondary list; Go back to the commented item and look at the items it indexs; select a few of them into the secondary list; Go back to the main line and select a few more items. Now exchange the secondary and primary lists and print the primary list.

Details: Every item contains four critical columns possibly empty —

Commented-by — 1) Row_id's of Comments on this item.
Comments-On — 2) Row_id's of items this item comments on.
Indexes — 3) Row_id's of items this item indexes
Indexed-by 4) Row_id's of items that index this item.

Index Column behaves very much like the Comments column and provides a means by which Gatekeepers, data shepards, and speed freaks can create items which organize other items, such as an item that lists the best puns in the database. To create an index item, one first collects in the secondary list all the items tobe indexed. Then one enters the item which is the index, tells why all these items are gathered together, and give it appropriate key words. Then you push the Make Index button which appends all the Row_id's for the items on the secondary list onto the "Indexes" Column.

To see the items which an Index References, just push the See Indexed button. The current primary index will be pushed down and a new hitlist will be created from the "Indexes" Column of the current item.

Items can index - index items down to any ~~---~~ depth. The ~~see i~~ See Indexed Items button will keep taking the user down the tree.

To go back a single level there is a Go Back button, and to get the very first hitlist back there is a Go All The Way Back button.

Comments would work in a similar manner. When viewing an item the user would push the Add Comment button to comment an item. To see the Comments on an item the user would push the See Comments button. Like ~~[redacted]~~ indexes Comments could be nested to any depth. When the See button is pushed the current hitlist is stacked and a hitlist make from the "Commented-By" column. The Go Back and Go All The Way Back buttons would be used to unwind both a stack of comments and a stack of indexes, or a mixture of both.

The basic operations on the secondary hitlist are:

1) Clear the list
2) Exchange the secondary and primary lists
3) Copy an (The current) item from the primary list.
4) Copy the primary list to the secondary list.
5) Make an index of the list
   (All copies add to the list, nothing is destroyed)

The operations on the primary list:

1) Clear the list
2) Add a comment to the current item.
3) for experts only - Add a comment to all items in list
4) Delete an item from the list.
5) Copy the secondary list to the primary list.
6) See the indexed items of the current item.
7) See the comments on the current item.
8) Go Back a level (pop a hitlist)
9) Go All The Way Back (pop the stack)
10) Print the list

All operations (except 3) should have a button.

Note the use of the current item concept

Searching

The "Find leaping green lizards" of the original CM system is far superior to filling out templates for CM-ish purposes so I suggest we work from the original augmented by a few fields like price and date.

A large number of tools, like a thesurious would be handy here, but I think it best to start simple.
If topic/world tree information is collected first a simple form like

topic: pets
keywords: leaping green lizards
price: < 10.50

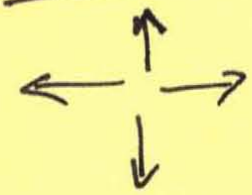
could be displayed.

There should be search keys
AND
OR
NOT

or something similar.

Editing - (

Motions



Prior Field Prior Screen Prior Item
Next Field Next Screen Next Item

* character motions, scroll the screen through the item, but not onto next item.

Edit Keys

Delete Character
Back space
Delete Line

Insert Blank Line (below) no trgs.

(Insert Toggle) hidden for experimentation (think insert should always be on, in basic editing.

Display Modes

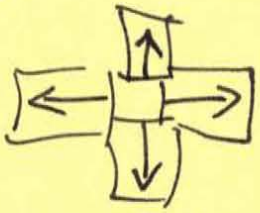
It is a theological mistake to think of the Sequitor Entry system when trying work out display modes for CM; they serve different gods.

CM needs a good browse mode and a complete display mode.

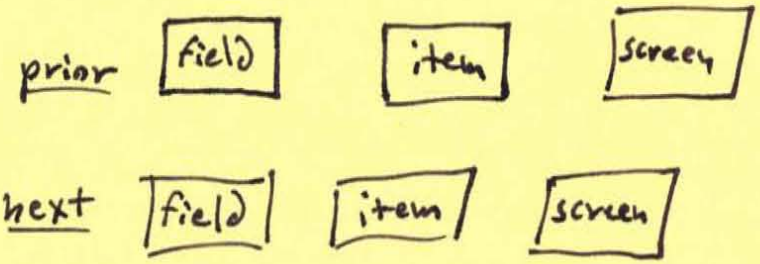
In both modes it is desirable to see the number of comments, The ~~index~~ number of indexed items, and the keywords or "topic / world tree" info.

There seems little general to ~~said~~ be said about complete display mode except that cursor motion should consistant with browse mode.

Suggested Motion Buttons



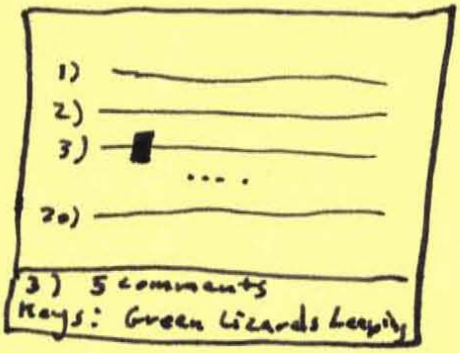
single character motions, without boundaries, hiting edge scrolls screen



This one is trouble. "what's a field daddie"?

Display Modes-2

Browse Mode could have two windows, the main window would display the first line of each item. The other window would display some more info the current item (The one the cursor rest upon), including number of comments, number of indexed items, its keywords, and possibly another line or two. This four or five line window should be maintainable at 1200 baud with a little cleverness.



After a search the items would be immediately displayed in browse mode.

so we need a ~~Flip Mode~~ Flip Display Form button or a two button labelled Show All Browse

Other Problems —

Multi user - is not a significant problem because at first there will be no editing of items and when there is only the first person to claim an item for editing gets it. Administrative operations would lock whole files.

Ownership — Provide a cyphered password ~~field~~ column
The password is specified at time of entry and only someone knowing the password could delete or edit the item, unless they were "mighty mouse."

In the future registered owners would have a single password associated with their name, not item. This is necessary for people to give stewardship of items to others.

Analysis — A complete input stream should be captured from each terminal. later programs can be written to parse it.

Items should record the number of times they have been referenced.

CM Row

Columns for CM Row —

Author:
Password:
Text:
Extended Text:
Key words:
Topic:
Commented-By:
Comments-On:
Indexes:
Indexed-By:
* { Date Entered:
  { Last Referenced:
Date Expires:
* { Last Edited:
  { Number of References:
  { Site Entered:

Date:
Money:

* { possibly
  { # references by site

---

* this information could also be captured with a good audit trailer.

Another User-Computer Dialog for Community Memory

Phil Kohn

Community Memory Project

Given the functionality defined by the Yet Another Design Document by M. Moore, a new approach to the person-computer dialog is described. Difficulties that novice users tend to have are outlined and a set of principles for the interface design are developed. These principles motivate and justify the new design. Implementation is not discussed in this document.

May 4, 1983

Another User-Computer Dialog for Community Memory

Phil Kohn

Community Memory Project

Given the functionality defined by the Yet Another Design Document by M. Moore, a new approach to the person-computer dialog is described. Difficulties that novice users tend to have are outlined and a set of principles for the interface design are developed. These principles motivate and justify the new design. Implementation is not discussed in this document.

May 4, 1983

Another User-Computer Dialog for Community Memory

Phil Kohn

Community Memory Project

Given the functionality defined by the Yet Another Design Document by M. Moore, a new approach to the person-computer dialog is described. Difficulties that novice users tend to have are outlined and a set of principles for the interface design are developed. These principles motivate and justify the new design. Implementation is not discussed in this document.

May 4, 1983

Another User-Computer Dialog for Community Memory

Phil Kohn

Community Memory Project

1. Why we have to do it right the first time

1.1. First impressions are important

Because of the limited amount of direct experience most people have with computers, their first session with CM will probably result in judgements being made about whether or not CM and computers generally, are a welcome part of their lives. If they find their first exposure to be frustrating (e.g. they speak Spanish and CM doesn't), they will be unlikely to try it again (in addition, many of their friends may never try it when they hear the review). In order to build up a base of users, the most critical time is the first months.

1.2. Consistency and Compatiblity

Users often do not see learning to use the computer as an end in itself. Unlike most "computer people", they find the unknown potential of dialog with a new computer to be more scary than intriguing. They would like to use the computer as a tool. The mark of a good tool is its transparency: the degree to which you can use it without thinking about the details of how the desired result in ones mind becomes translated into actions. (A good tool becomes an extension of the ultimately transparent tool: our own body and mind.) Tool users see the learning process as an investment. For these people, it is important not to have to spend any more time learning than is absolutely needed. In particular, having to learn another user interface at a later time or at a different location will be seen as extremely inconsiderate (esp. since they are paying for the computer time). Even with professional programmers there is an inertia to changing modes of behavior that have become second nature (for example it is taking longer for me to learn VI then it did to learn EMACS because of the reflexes built up by years of EMACS experience).

This line of thought leads me to believe:

1 If we have multiple user interfaces (and I'm not sure that we need to at first), they should all be available as options at each terminal. This will have the added benefit of being a much more controlled experiment since the subjects will be the same for different interfaces Also, the usage pattern of the interfaces will tell a lot about what people think about them.

2. The user dialog design should be close enough on the first version to the final interface that non-compatible changes can be keep to a minimum.

The moral of this (and I have learned this through multiple painful experiences) is that every hour spend toward being satisfied with the design pays for itself many times over in the degree of success one has in attracting and keeping users as well as the time spent coding and debugging (coding errors and user interface flaws). I have seen this happen in the industry where competition causes intense time pressure to produce working software; as I understand it, we are not competing with any other group, and in fact we are well ahead of our time

2. Principles of User Interface Design

Computers and people often seem to accent each others stupidity. The wisest saying I've heard regarding the design of user interfaces is: "Always imagine your grandmother trying to use it" (This came from the interface designer of Wang's very popular word processor). In our case it is less of a joke, we hope that grandmothers will use our system!

It is often hard to empathize with the novice user; it involves more than the (already difficult) task of shedding knowledge that we take for granted about how computers work. One must also understand the emotions that can turn off a potential user.

These include:

1. Fear of doing something wrong. Even people who don't know anything else about computers know that computers are powerful and expensive and easily fallible.

2. Overstimulation. Too many new things happening at once. What is the machine doing? A whole screen full of letters and no idea of what is immediately relevant and what is just extra information. There are words and messages that the user doesn't understand.

3 Uncertainty. Literally hundreds of buttons and no idea of what is expected by the machine.

4. Tension about being misunderstood by the machine. Some buttons have been pushed, the machine seems to have accepted them, but did it really understand ? Is it doing the right thing or something horrible or maybe it just died.

5. Frustration is the real killer. All the user seems to be able to get are error messages When the machine finally does do something. it isn't what was expected or desired.

Here are some design principles, mostly drawn from the literature, mostly common sense.

2.1 The user should know what is expected of him/her.

This includes knowing at every point in the dialog, all and only the valid keypresses available (except advanced features). Buttons that seem available but only produce error messages can be frustrating. A correlary is that all hard special function keys should be valid at most points in the dialog.

2.2 Information about what each option would do must be immediately available

Users often want the chance to test the water before jumping in. This helps to reduce their uncertainty and allows them to feel secure about experimenting. In this context immediately has two meanings: with a minimum of keypresses (optimally one should do) and a minimum of waiting time (help should be a local function).

2.3. Immediate acknowledgement/feedback about each keypress.

The user should be constantly informed of what is happening in a consistent way (e.g. in the same place on the screen). The immediacy is important because it lets the user know when the computer is ready and when it is working. (Most novices will not type ahead.) I have seen many novices that forget to hit return. They sit there thinking that the computer is busy working because they expect a pause without feedback after returning

2.4 Users should be expected to do a minimum of typing.

It is sometimes hard to remember that the majority

of people don't know how to type and find it to be a frustrating process. It would be a nice pie-in-the-sky to have a spelling checker. More practical would be a keyword lookup table hashed by the sound (soundex code) of the first part of the word.

2.5 Good messages

Jargon is insidious. Messages should use words that everyone knows. A quick poll of some of my non-computer oriented friends shows that none of them knew what a "keyword" was, nor were their intuitive guesses very accurate. After I explained it, they suggested that "index word" was more obvious by analogy to a book's index. Though this would break with standard database jargon (and would no doubt be hard for us to get used to using) our non-computer oriented users might find it easier to comprehend and remember.

Error messages should always explain corrective actions that the user can take.

2.6. Defaults

It should almost always be possible for a user to avoid dealing with an aspect of the system. The system should provide and display reasonable values for fields that are not entered

2.7. Attention should be drawn to relevant parts of screen.

This can be done very effectively with reverse video and blinking. All the information on the screen should be relevant to the purpose and functioning of that screen.

2.8. More destruction should require more keystrokes

The user should have to try hard to do something destructive to him/her self. There should not be any way for a user to disrupt the structure or functioning of the whole system.

2.9. Consistent analogy/model.

People can understand complex things through analogy to something known. For example, a library model of CM might include looking at searching as a process of indexing by titles or subjects or authors, etc.

3. Changes in functionality from Yet Another Design Document

3.1. Explicit title line

Using the first line of a file as a title can be problematic for even the expert (note the number of CM messages with blank or uncommunicative first lines). The advantages of an explicit title are:

1. Messages can be referred to directly by title (assuming uniqueness).

2. The title can be used as a cheap way of getting the default keywords out of the user. The title can be scanned and all non-content words removed. The remaining words can be the default keywords Another advantage of this is that searchs by title can allow one to scan the closest hits in the case where there is no direct hit

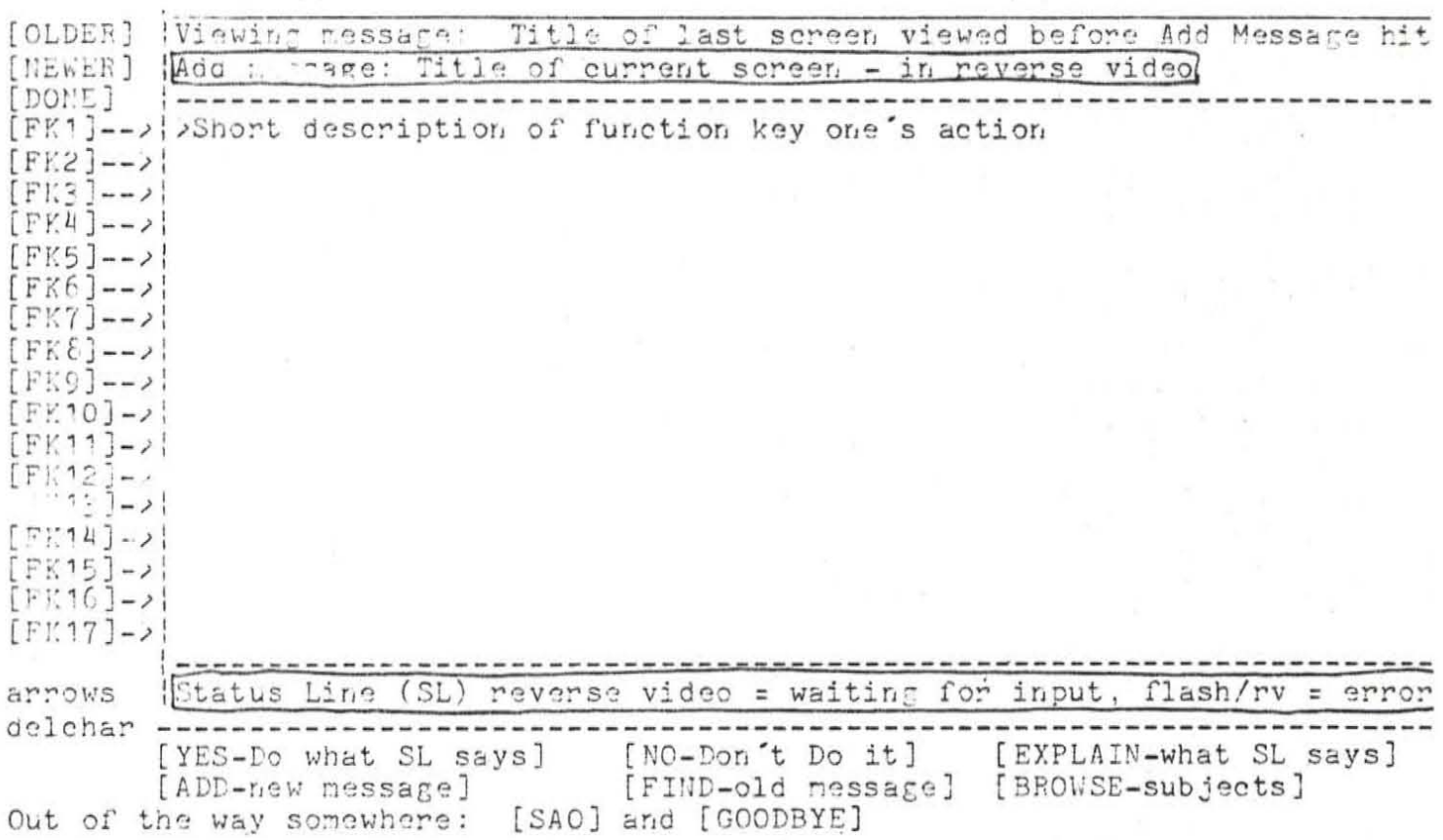
3.2 Name, Address and Phone

Since one of the goals of the system is to get people together, it seems important to have this information. It should, of course, be totally optional. The name can also be used as a default keyword during message adding. This would allow people to select messages by author.

This information, along with a brief introduction (i.e. who am I and what am I into) can be put in a message with keywords including: first and last names, "hello", "hi", "who", "where", "address", "phone", etc.

4. Front of CM: screen and button layout

First a diagram of the layout, then a discussion of what things do and how they interact. Abbreviations enclosed in brackets or boxes are buttons. On the real thing they would be labeled with a descriptive phrase shown in the table below. In fact the basic documentation for CM (i.e what parts of the screen mean, what hard buttons do, etc.) should be printed on the front panel of the machine. Smaller fonts could be used for nonessential information.



[OLDER] Show titles of earlier screens: scroll back in history of screens seen by user

[NEWER] Show titles of later screens: scroll forward in history of screens

[DONE] Done with screen: remove screen from history list. go back to previous screen (the one shown at the top of the screen)

[FKn] Soft keys. Hit these to choose an option from the screen. On side of screen to allow longer descriptions and more keys.

[YES] Does the function described on the status line or answers yes to questions displayed there.

[NO] Aborts function (even if in progress, e.g. break) on status line.

[EXPLAIN] Explains request, question, function or error message currently on status line.

[SAO] Show Advanced Options: causes more advanced soft keys to become visible. Pushing it again makes them go away (this should be mentioned in status line).

[ADD] Start a new "Add a new Message" screen

[FIND] Start a new "Find old messages" screen

[BROWSE] Start a new "Browse subjects" (world tree) screen

[GOODBYE] Start a new "Goodbye" screen.

5. Some screen design issues

5.1. Density of buttons on the side

Since there is one button per line of display the buttons will have to be small. But, as will be explained below these side buttons serve as a pointing device (they could be replaced with a mouse, a joystick, a touch panel or a slide pot). In other words, pushing a side button will only cause the corresponding label to light up (reverse video) and a longer description of the key to appear at the bottom; no action takes place until the YES button is hit (which would be a large, heavy-duty button positioned to be hit by the other hand for efficient users). Because there is feedback, the buttons need not be precisely lined up with the corresponding line of the display. Also the buttons could be arranged in two out-of-phase columns.

5.2. Area of screen available for text

At first I was very concerned about maximizing the area available for text display. Then I realized that programming requires much more context to be available at once on the screen than reading text. For example, I find a 24x80 display barely large enough for programming (66x80 is more like it), yet I am quite satisfied to read a ticker-tape display that only shows five words or less of context. (This says something about the different cognitive processes involved in program understanding and text understanding.)

6. Telling CM to do something

To tell the computer to do something requires two steps: selection/preview and confirmation. This allows the user to know exactly what CM will do before it actually does it. The user can feel more secure about experimenting with buttons they have never used before (at first this includes all of them). It also

eliminates the element of gambling and surprise that can freak-out the novice.

When a choice is to be made, there is always a default (most often used) option selected before the user does anything. The operation that is selected is always described in a one-line format on the status line. Since the machine is waiting for the user to do something the status line is in reverse video to draw attention to the decision at hand. If the selected function is a soft key then its label will also be in reverse video. The status line will usually say something of the form. "Hit YES to ...".

If the user hits YES. the status line announces the operation in progress (not reverse video). If the user hits EXPLAIN, a longer explanation of the function is displayed at the bottom of the screen. If the user hits NO, the status line gives a short explanation of how to select another function (EXPLAIN can be hit here for the full details of the selection process). The user can also simply select another function (in several ways described below), the status line will explain the new function; if the function is a soft key, it becomes highlighted. Thus the user can look at the one line and longer description of as many functions as desired at any point, with a single keypress.

Selection can be made by one of the following methods:

1. Hitting a soft or hard function key.

2. Using the cursor up and down arrows to change which soft function is selected.

3. Typing in the capitalized letters on the function's (hard or soft) label (as shown on the display or cabinet). At each point in typing, CM's best guess is displayed on the status line. Incorrect typing results in a polite message similar to hitting NO. Hitting RETURN after a typed command is the same as hitting the YES button. Since most commands can be typed as one or two letters, this method of interaction is as efficient as many systems geared toward professionals.

7. Modeless interaction and the Screen History

The concept of modeless interaction does not mean that there is no context (e.g. a menu tree or a half written message). It means that at each point the user has the option of doing another function without losing their current context. In this sense their terminal is

not preempted; they can always choose to interrupt what they are doing and come back later (e.g. stopping in the middle of Adding to View another message). The Xerox Smalltalk system has proven that this can be done intuitively with a window system and a mouse, by using the analogy of separate windows (pieces of paper) for separate processes in progress Very young children have learned Smalltalk.

Since the screen is what the user sees, it seems intuitive to imagine that new screens don't distroy old ones, but are simply added to a list of all the screens seen by the user. All types of screens are on this list including FIND, ADD, BROWSE and VIEW. The novice need not know about this; yet it should be modelled simply enough that novices will grasp it easily. Showing the title of the previous screen above the title of the current screen (the current one is distinguished by reverse video), reminds the user of what was seen last and also reinforces the model of a history of screens.

it is difficult to assume session boundaries

The BACK TO OLDER SCREEN button allows the top two lines of the display to scroll through the titles of screens in the order they were seen. Only the current screen's title is in reverse video. The current screen is not changed by this. This allows the user to view where s/he has been without affecting things. After hitting this button, the status line will display: "Hit YES to go back to screen: <title of screen>". Hitting YES causes the selected screen to be spliced out of the list of screens and added to the end. This prevents screens that are viewed many times from cluttering up the list. Adding to the end of the list keeps the most recent history accurate; it also allows screens that are used most recently to be at the end of the list where they can be accessed most easily.

Hitting another (hard or soft) key other than OLDER, NEWER, DONE or YES will cause the top two lines to revert to showing the current and previous screen titles. Starting a new screen always causes the screen to be added to the end of the history list.

The GO TO NEWER SCREEN button scrolls back down to more recent screens. If at the end of the list, give an error message on the status line. Again, hitting YES will make the screen title above the dashed line (where the current title normally is) become the current and last screen on the history list.

The DONE SCREEN button removes the selected screen (the one above the dotted line) from the list. If the current screen is being viewed (the default situation), the DONE button has the effect of finishing the screen

and going back to the previously seen screen (a pop). One can use CM without this button, and its function is easy to grasp: "take me back to what I was seeing last".

More advanced features (not discussed further in this document) could use the last two screens as an argument. For example, a merge index command might merge the last two screen's indexes or hitlists This is quite general since any two screens (old or new) can be made the current and previous screen (and both titles are displayed on top) without effecting anything else In this way binary operators could be implemented that would cover everything that an algebraic notation can do.

8. Screens

8.1. WELCOME to community memory

This is the first screen that the user interacts with Standard stuff like how to get help, how to give suggestions and a disclaimer should be printed on the front of the machine.

```
[OLDER]  ┌──────────────────────────────────────────────────────────────────
[NEWER]  │**** Welcome to Community Memory ****
[DONE]   │------------------------------------------------------------------
[FK1]--->│>I feel uncomfortable with computers
[FK2]--->│ I am a new user of Community Memory
[FK3]--->│ Change my name from:              J.  Doe
[FK4]--->│ Change my address from:           Somewhere
[FK5]--->│ Change my phone number from:      000-000-0000
[FK6]--->│ Change my self description (below)
[FK7]--->│------------------------------------------------------------------
[FK8]--->│ I am a very boring person who hasn't bothered to set my real
[FK9]--->│ name, phone number and address.
[FK10]-->│
[FK11]-->│
[FK12]-->│
[FK13]-->│
[FK14]-->│
[FK15]-->│
[FK16]-->│
[FK17]-->│
         │------------------------------------------------------------------
         │Push YES to find out more about computers and Community Memory
         ├──────────────────────────────────────────────────────────────────
         [YES-Do what SL says]   [NO-Don't Do it]   [EXPLAIN-what SL says]
```

8.1.1. I feel uncomfortable with computers

If this is selected, we try to do our best to reassure and amuse the user. A series of simple games will get the user to push buttons and notice where things are and what they do. There might also be an information menu so that the user can find out more about selected aspects of computers in general. After this the user is feed into the "I am a new user..." sequence

8.1.2. I am a new user of Community Memory

This will give the user a series of games to learn the basic editing functions. Each should have a simple goal that demonstrates that the user understands one aspect of the system; once the goal is achieved the game is over. This makes the games extremely fast for the more advanced new user. (e.g. use the arrow keys to get the cursor to a highlighted goal on the screen) There should also be a short walk through example of each screen, explaining its purpose , its function keys. etc.

8.1.3. Change Name, Address and Phone number

Name, address and phone are totally optional. If a name or pseudonym is entered, CM would check its memory for the name (a message to this effect will appear on the status line). If the name is found, the address and phone number will appear on the screen. If it is a new user, the status line will read: "Push YES to enter a slightly different name, this one is in use" (flash/highlight). EXPLAIN at this point will recommend using a middle initial, etc. If the name is not found and it is not a new user, say: "Your name not known, push YES to become a new user".

A more advanced option might allow passwording your name to prevent forgery of messages. This password could also be used as a default for protecting messages added for later editing. At a future date, regular users might be billed by mail; in this case the password is needed to protect the user against being billed for usage by someone else.

8.1.4. Change self description

This puts the user into the text editor on the lower part of the screen. Hitting any function key will exit the editor.

8.2. FIND old messages

The default screen shows the most recently used (or most recently added, or most often read?) titles for immediate access.

```
[OLDER] |Previous Screen Function: Previous Screen Title or Keywords
[NEWER] |FIND: Title or Keywords
[DONE]  |-------------------------------------------------------------
[FK1]-->|>Select if keywords:
[FK2]-->| Select if price between $0.00 and $9999999.99
[FK3]-->| View: [1] Title of most recently used (or most popular) message
[FK4]-->| View: [2] Title of next most recently used/added message
[FK5]-->| View  [3] Title of next most recently used/added message
[FK6]-->| View: [4] Title of next most recently used/added message
[FK7]-->| View: [5] Title of next most recently used/added message
[FK8]-->| View: [6] Title of next most recently used/added message
[FK9]-->| View: [7] Title of next most recently used/added message
[FK10]->| View: [8] Title of next most recently used/added message
[FK11]->| View: [9] Title of next most recently used/added message
[FK12]->| View: [10] Title of next most recently used/added message
[FK13]->| View: [11] Title of next most recently used/added message
[FK14]->| View: [12] Title of next most recently used/added message
[FK15]->| View: [13] Title of next most recently used/added message
[FK16]->| View: [14] Title of next most recently used/added message
[FK17]->| View: [15] Title of next most recently used/added message
[FK18]->|---See titles off screen ([16] through [10238])-------------
        | Push YES to narrow down these messages by subject or by title
        ---------------------------------------------------------------
        [YES-Do what SL says]   [NO-Don't Do it]   [EXPLAIN-what SL says]
```

8.2.1. Select keywords

Hitting this will bring up the status line: "Type in a title or subject names separated by spaces, then hit RETURN". The line to the right of the title/subject prompt will be cleared and set to reverse video. The cursor will be placed at the beginning of the box. If the cursor hits the right margin, the line scrolls left. Hitting return adds the keywords to the screens title and causes the status line: "Searching Memory - So far nnn messages found". Non-content words are removed from the keyword list (user doesn't see this).

If the search would reduce the number of messages found to zero, it will be aborted with the (flashing) status line: "None of the messages shown have all of the index words you typed in". Hitting EXPLAIN would show "Since none of the messages in the list you see now have all of the words you typed into 'select if', the search you requested was not done. Try using fewer

or other index words in "select if".

8.3. Advanced Find Messages

Hitting the Show Advanced Options key would give you:

```
[OLDER] |Previous Screen Function: Previous Screen Title or Keywords
[NEWER] |FIND: Title or Keywords
[DONE]  |-------------------------------------------------------------
[FK1]-->|>Select if Keywords:
[FK2]-->|  Exclude if keywords:
[FK3]-->|  Include if keywords:
[FK4]-->|  Select messages Read between 1 and 9999999 times
[FK5]-->|  Select messages Added between Jan 1, 1980 and <today's date>
[FK6]-->|  Browse keywords used by currently selected messages
[FK7]-->|  Undo last Select, Exclude or Include
[FK8]-->|  Remove messages from list
[FK9]-->|  ---See titles off the screen ([1] through [7])---
[FK10]->|  View: [8] Title of next most recently used/added message
[FK11]->|  View: [9] Title of next most recently used/added message
[FK12]->|  View  [10] Title of next most recently used/added message
[FK13]->|  View  [11] Title of next most recently used/added message
[FK14]->|  View: [12] Title of next most recently used/added message
[FK15]->|  View  [13] Title of next most recently used/added message
[FK16]->|  View  [14] Title of next most recently used/added message
[FK17]->|  View: [15] Title of next most recently used/added message
[FK18]->|---See titles off screen ([16] through [10238])-------------
        | Push YES to narrow down these messages by subject, title or author
        ---------------------------------------------------------------
        [YES-Do what SL says]   [NO-Don't Do it]   [EXPLAIN-what SL says]
```

8.3.1. Exclude if keyword

Status line when selected reads: "Exclude any message that contains all of these keywords from the FIND". Each time this option is used the type-in area is cleared of previously entered keywords; this option does not accumulate these keywords.

8.3.2. Include if keyword

Status line when selected reads: "Include any message that contains all these keywords in the FIND".

8.3.3. Browse related keywords

Enter a special BROWSE screen that allows the user to see all the keywords used by the currently selected messages. The user can use this as a basis for further Selections, Exclusions and Inclusions. A future version of CM might allow the Select, Include and Exclude

function keys to appear on this screen allowing interactive world tree pruning.

8.3.4. Remove messages from list

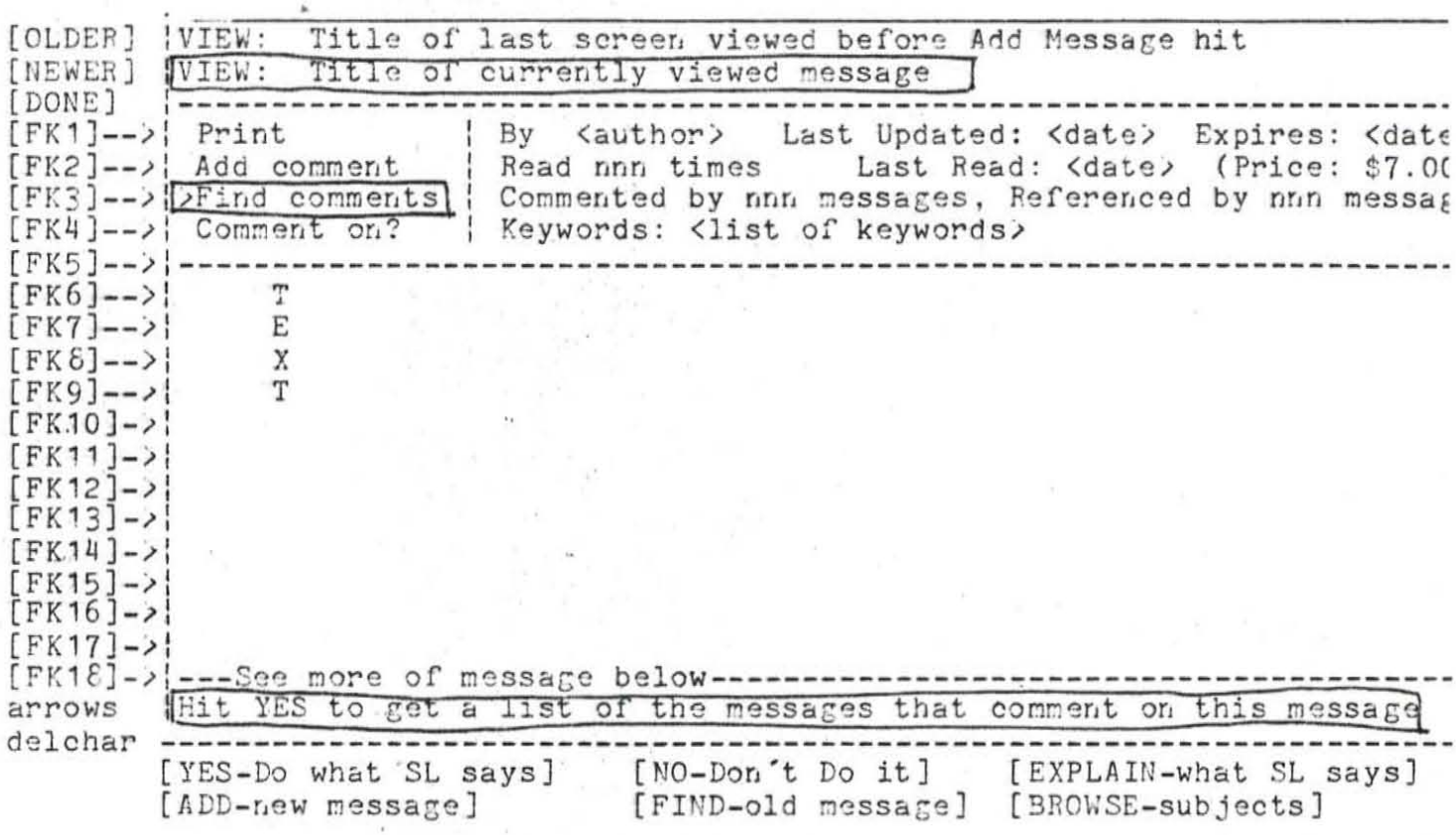
This turns on a mode (gasp!) in which touching a title will make it go away. The "Remove messages from list" key transforms into a "View messages on list" key (it changes the mode back). All the keys associated directly with titles transform from "View ..." to "Remove: ...". When these keys are selected, the word "REMOVE" flashes on both the key's label and on the status line description of the key.

8.3.5. Undo last Select, Include or Exclude

This key only appears after one of Select, Include or Exclude has been done. It also disappear after it has been used (allowing only one level of undo). Its main purpose is for those situations where the last search left things worse off.

8.4. VIEW old message

You get here by hitting a title displayed by FIND. The statistics that are not usually immediately relevant to VIEWing (Author, Dates, Usage, Referenced-by) could be put up and taken down by an option key (not shown here).



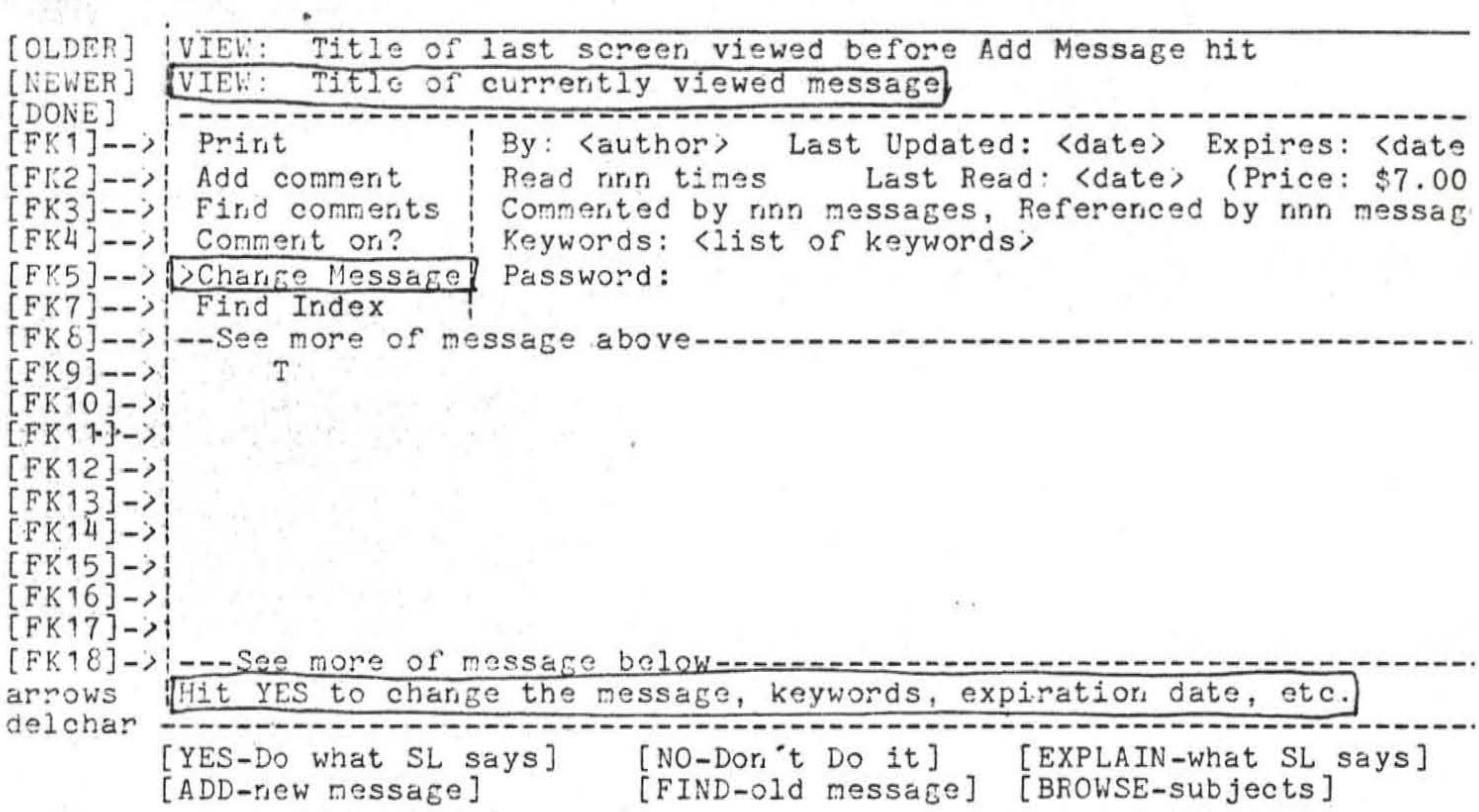
8.4.1. Comment On?

Status line would read: "Hit YES to view the message this is a comment on". Note that this option would not be displayed if a comment was not being viewed. Likewise the Find Comments option would not be seen if there were no comments

8.4.2. Add Comment

Go to ADD message screen with smart default title and keywords.

8.5. Advanced VIEW message



8.5.1. Change Message

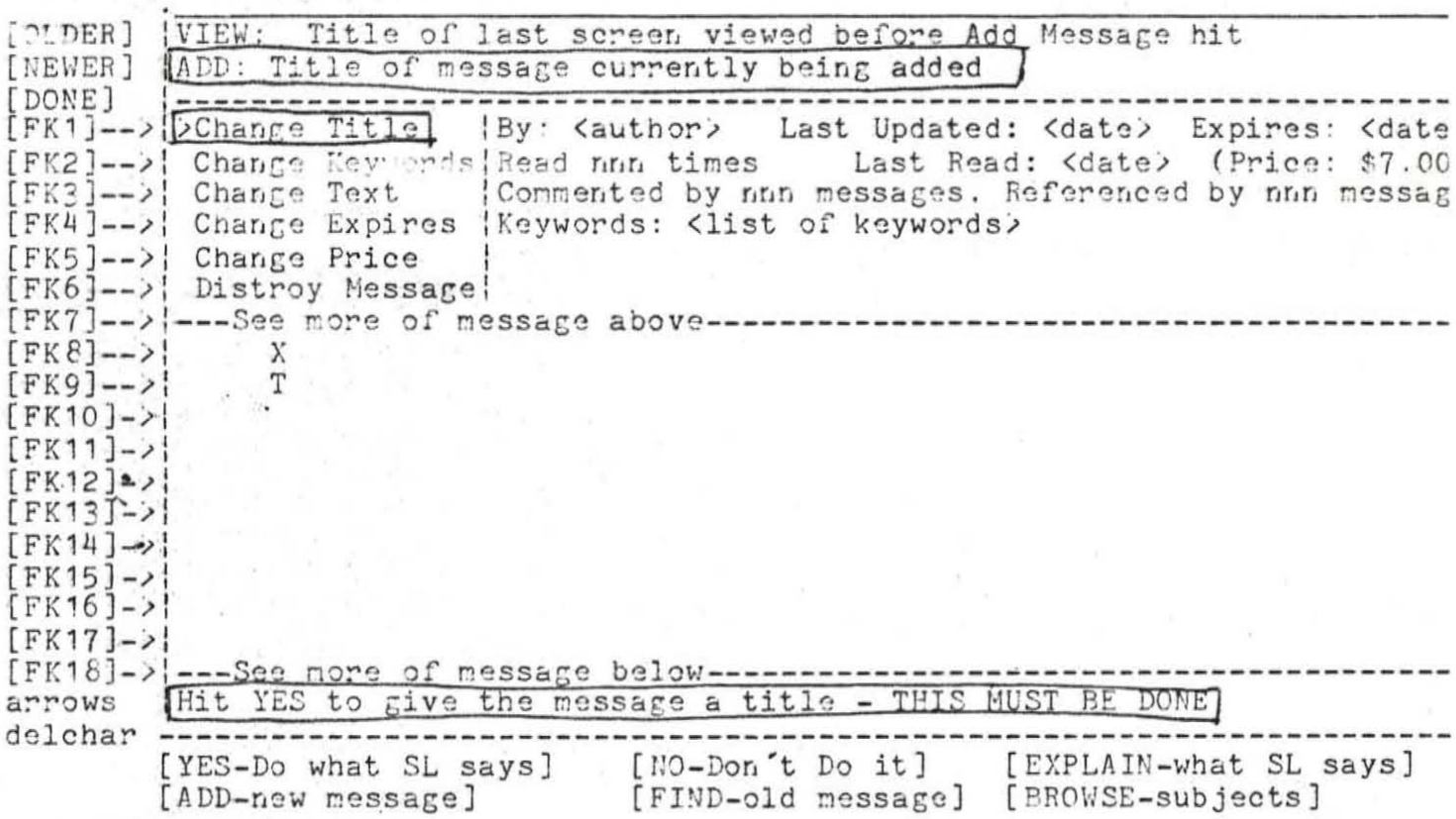
If the message is not public, the password will be requested and the cursor placed in the password window above. If everything checks out, the VIEW screen will transform into an ADD screen.

8.5.2. Find References

This will only be an option if the message has an index. In that case, a new FIND screen will appear (pushing the old one) that shows the index of references (analogy: looking up a footnote). It might be nice to allow direct VIEWing of a reference in the text (by pointing to it in the text or by number) instead of having to go through a FIND screen first.

8.6. ADD a message

Hitting the ADD button or doing Change Message in a VIEW screen gets you:



Messages are not really added to the system until the end of the session. When the GOODBYE key is hit, a GOODBYE screen appears that will warn the user about messages that have been ADDed that don't have a title or enough keywords The GOODBYE screen also gives the user the option to rate how well the system has served them and make comments and suggestions. GOODBYE also announces the titles of all messages that have been added in that session.

8.6.1. Change Text

This puts the user into the text editor. Pushing any of the hard or soft buttons to select a new function will cause the editor to exit.

8.6.2. Change Keywords

In addition to having keywords from the title and author, keywords can be added manually.

Another method of adding keywords to a message would be to allow the creation of a special BROWSE screen. In this BROWSE screen, hitting YES to a selected leaf of the world tree would add the name of

the leaf and the names of all the levels of categories leading to the leaf as keywords This way the user can walk through the world tree and touch several leaves if the message interrelates several things A more advanced user can also add new leaves and categories to the world tree in the same operation that adds the new keywords to a message.

8.7. Advanced ADD message

The advanced version would have in addition the functions: Edit references, Change/add password and Public edit.

8.7.1. Edit references

This creates a special FIND screen (pushing the current screen) showing all (may not be any) of the references (the index). Changing which messages are shown on this screen will change the index of the previous ADD screen. This screen should have a special title.

8.7.2 Change password

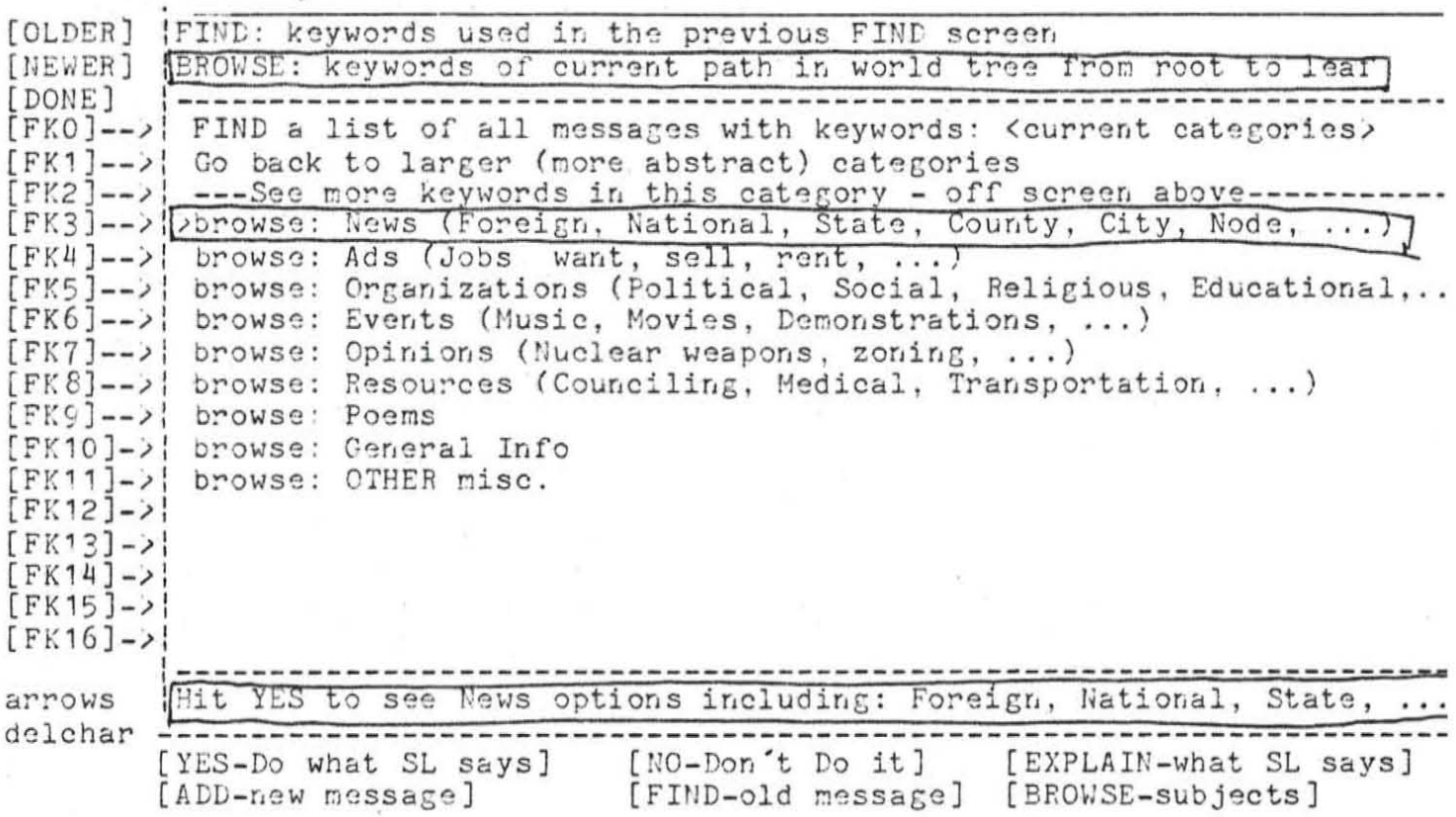
Allows the user the option to change or add (as the case may be) a password to the message for future editing. The default is to make the password be the same as the one specified on the WELCOME screen.

8.7.3. Public edit

This allows the message to be edited by anyone.

8.8. BROWSE screen

Hitting the BROWSE key puts up the screen:



8.8.1. FIND messages with these keywords

Creates a new FIND screen (pushes current browse) with the current tree path as keywords to Select If.

8.8.2 Back to larger categories

The allows the user to move up the world tree. (could use a more descriptive label here)

8.8.3. See keywords off screen above

Scrolls the screen. A similar message would appear at the bottom if there are keywords not shown below.

8 8.4 Browse: keyword

Categories are shown in order of number of messages contained from most to least. The first several sub-categories are shown parenthetically for each category. Hitting a category key adds the category to the current screen title line and shows keywords in that category.

Leaves would show (parenthetically) how many messages they have in them. If a leaf is hit it could cause a FIND on keywords selected already. In the case of a special BROWSE for adding keywords to a message, hitting a leaf would cause keywords to be added (the special BROWSE should have distinguishing marks telling what message it is associated with; also each leaf should say "Hit YES to add these keywords to message titled ...".)

8.9. Advanced BROWSE options

Hitting show advanced options would get you:

```
[OLDER] |FIND  keywords used in the previous FIND screen
[NEWER] |BROWSE: keywords of current path in world tree from root to leaf
[DONE]  |-------------------------------------------------------------------
[FK0]-->|  FIND a list of all messages with keywords: <current categories>
[FK1]-->|  Go back to larger (more abstract) categories
[FK2]-->|>Add a new keyword to this category called:
[FK3]-->|  Add a synonym to <the selected keyword> called:
[FK4]-->|  Show synonyms for each category
[FK5]-->|  ---See more keywords in this category - off screen above----------
[FK6]-->|  browse: Events (Music, Movies, Demonstrations, ...)
[FK7]-->|  browse: Opinions (Nuclear weapons, zoning, ...)
[FK8]-->|  browse: Resources (Counciling, Medical, Transportation, ...)
[FK9]-->|  browse: Poems
[FK10]->|  browse: General Info
[FK11]->|  browse: OTHER misc.
[FK12]->|
[FK13]->|
[FK14]->|
[FK15]->|
[FK16]->|
        |-------------------------------------------------------------------
arrows  |Hit YES to add another index-word to the category <category name>
delchar ---------------------------------------------------------------------
        [YES-Do what SL says]    [NO-Don't Do it]    [EXPLAIN-what SL says]
        [ADD-new message]        [FIND-old message]  [BROWSE-subjects]
```

8.10. Add new keyword

Adds the world tree to be expanded. Note that categories and leaves can not be deleted once added.

8.11. Show synonyms

Changes the subcategory display to a list of synonyms. This keys label changes to "Show subcategories" for toggling back to the normal display.

8.12. Add a synonym

Allows the user to add a synonym to the next keyword selected.

9. GOODBYE

This screen gives the user a list of the changes being made to the data base (added/edited messages, previously unknown keywords and synonyms). The user is given one last chance to keep any of these changes from actually occuring Warnings are issued if a message does not have a title or enough keywords and the user is allowed to go back to an appropriate ADD screen to change these. The user also has the (advanced) option of creating a special BROWSE (same as that created from ADD) that allows each novel keyword to be added to the world tree.

The user also has the option of rating the CM system. Another function key would allow suggestion messages to be added by setting up a special (simplified) ADD screen with appropriate default title and keywords

The user is given the option of terminating the session from this screen. At any point before hitting the "all done" key (and the YES key) in the GOODBYE menu, the user can start new screens (pushing the GOODBYE screen) or go back to previous screens.

10. Future features

10.1. Polling and rating

There should be support for the creation and usage of opinion polls and ratings on specific issues as well as a general evaluation of a message's merit A message that states an issue can have an index of opinions, each with voting information attached. New opinions can be added to an issue.

Polls should allow voting by open and secret ballot. An open ballot adds a reference to the user's name-address-phone-who-am-I message (can't be defaulted to J. Doe) to a petition index (if it is not already on the list). A secret ballot simply counts the number of votes issued (and is less immune to misuse).

The user should only be allowed to see the current statistics about a poll or rating after they have voted. There are other protection issues, e g. should we check for a secret ballot vote on the same issue twice in the same session?

11. Criticisms

All feedback is welcome I know that I am writing in the dark. I don't know what issues have been discussed and what has been decided. The flip side is that I have no biases related to CM's history, allowing me to see the problem from a fresh perspective. Having said all this, I now open myself up to all ideas that anyone wishs to throw at me (not too hard please) about the CM user dialog

Ken suggests: evaluate in terms of specific applications
- Skills exchange
- Conferencing
- ???

Proposed CM Front End

Phil Kohn
Michael Moore

Community Memory Project

This is a working document on the CM front end design. It contains a description of buttons, screens and options available to users during their dialog with CM. A users' viewpoint is taken: for each thing the user can do, the resulting messages, actions and options are presented. Menu options and buttons are presented in order of importance (i.e. implemented first and most visible in layout)

June 7, 1983

Proposed CM Front End

Phil Kohn
Michael Moore

Community Memory Project

1. Issues

1.1. Dumb vs. Intelligent terminals

This design is based on the assumption that (at least for development purposes) CM should be able to run on a dumb terminal. Dumb terminals may require a slightly different menu system than the one described here; a long time lag between pressing an arrow key and highlighted of a new option may cause frustration.

The upgrade path to local intelligence should be smooth. Menu selection can be easily extended to handle mice, joysticks, etc.

1.2. Hard vs. Soft buttons

Most first time users tend to look at the menu for hints about what to do next, they forget about the buttons on the keyboard. It is important for the novice to be able to use CM with a minimal amount of knowledge. A user who can only deal with selecting from menus should still be able to use all CM functions.

Each menu has a "Pause from <finding, reading, adding>, go back to MAIN menu"" option. (The menu that was in progress is saved for continuation later.) The MAIN menu allows new find and add menus to be started (and old menus in progress to be continued). This option would go away in ADVANCED mode to make room for other menu options.

Hard buttons allow new find and add menus to be started immediately, without having to go through the MAIN menu.

1.3. Coin Box algorithms

CM should charge in units that are understood by everyone. People should have a clear idea of what they

are getting for their quarter. For example, charging by CPU time would require some idea of how long it takes computers to do things -- A difficult problem.

A general equation that includes all the ideas I have heard and all of their linear combinations is:

$$COST = ANTE + \sum_{\text{for each new find, add, or read}} RATE * LOADCOMP * t + FIXED$$

Where RATE and FIXED are different for finding, adding and reading. The RATE for adding should be zero or very small. Loadcomp is a function that decreases when the system is heavily loaded.

The problems with charging by time are:

A) A heavily loaded system will be frustrating because the user will wait around and think about their money slowly disappearing.

B) It is bad to have to interrupt the user when money runs out. They may be in the middle of doing something and may be out of change, etc.

The advantages of charging by time:

A) It is easy to understand.

B) It discourages "hogging" of the machine.

C) It is a partial solution to the dangling session problem.

Load compensation is supposed to attack problem A. Though it tends to reduce advantage A. It can also backfire since people may notice the lower rates and decide that it is a good time to use CM.

1.4. Message ID numbers

Advantages:

A) Shows chronology of messages at a glance

B) Concise way to refer to messages

C) Numbers are easier for non-typists than letters

D) Messages can refer directly to many other messages by number

Disadvantages:

A) Numbers on screen can be alienating

B) Contrary to spirit of system. "Unlinks" commenting concept. People won't "browse". Discourages creative use of labels.

Maybe it should be an advanced feature.

1.5. Union vs. Intersection => Scoring

In FIND, messages are shown in an order related to how desirable they are to the user. The user can manipulate two lists of labels (four lists are available in advanced mode, see below): labels that messages should have and labels that messages should NOT have. Which labels are on these lists totally determines what messages the user will get back and in what order. (Thus the user can backtrack by removing labels from a list.) For every "should have" label a message has, it receives one point. Similarly, for every "should NOT have" label a message has, it loses a point. Titles of messages with the highest score are shown first. If a message has all the "should have" labels and none of the "should NOT have" labels, it will be seen first. (This is the intersection.) Users are encouraged to enter more labels in a search since there is no penalty -- it can only help. (For intersection, the penalty for using too many labels is getting a null search; for union, the penalty is getting too many messages to look at.)

In the case of a tie in score, the newer message will be shown first.

In ADVANCED mode, the user has four lists of labels that determine what messages are seen. In addition to "should have" and "should NOT have" there is "Only see messages with all of these labels" and "Never see messages with any of these labels". These options provide coarser (less fuzzy, more logical) tools for hacking away messages.

A fast algorithm that does not need to do sorting or merging and shows messages on-the-fly has been

sketched out. One trick is to notice that you don't have to look at many messages to realize if you can reduce the highest possible score a message will be able to get.

1.6. "World tree" => label aids

CM provides a way to help users choose labels. Exactly the same label aids are provided to the user when entering labels into the above lists in FIND and when entering labels for a new message in ADD. (Also, when adding personal labels in READ.) Hopefully, some of the weird reasons people have for choosing certain labels should be the same in both situations.

The user can alternate between typing labels into a list and using the EXPLAIN button to get a menu of labels to choose from (in a window overlaying the bottom of the screen). Labels selected (by the usual menu selection method) are added to the list being used as if they had been typed in manually.

Every label has a set of related labels (references) coming from it. This forms an associational network of labels. For example, the label CAR may have related labels like: year, model, color, accessories, van, automobile, ... When editing or entering labels on a list, pressing EXPLAIN shows the menu of labels related to the label that the cursor is currently on. If the cursor is not under a label (e.g. there are no labels on the line), pressing EXPLAIN will show labels related to the special label called "TOPICS".

For example, the user types in "car". The status line will read: "Press EXPLAIN to see more labels related to "car"". The user presses EXPLAIN and sees a list of labels on the bottom of the screen and a cursor. The user selects the label "color" from the list by putting the cursor over it. The status line reads: "Press YES to add "color" to list, Press EXPLAIN to see labels related to "color"". The user presses YES, the word "color" appears at the end of the list of labels being entered. The status line changes to: "Press YES to remove "color" from the list, Press EXPLAIN to see a list of colors". The user can still go and select other words related to "car". The user may press EXPLAIN to see a list of colors, or may press NO/DONE and type in a list of colors manually (fill in the form style).

Every list of related labels has (at the end of it) a special label called "OTHER - add a new one". Users can create labels by simply typing them in; by using EXPLAIN and selecting OTHER from the menu, users

can relate any label to any other label.

Every so often, a census is taken in which the number of messages that use each label is determined. The menus of related labels are sorted so that the most often used labels will be on the top of the list, and least used ones will sink to the bottom. (It would be better to keep track of how many times each label is used from a particular menu of labels; some popular labels are more appropriate in one list of related labels than another.)

There are special labels that are ranges of numbers, prices and dates. Range labels can be used anywhere in the network. They are like regular labels except that when they are added to a list they must be followed by a value and optionally the word "to" and another value. When a range label is selected from a menu, the user is prompted for the lower and upper limits; the values are put in the above syntax when added to the list.

The TOPICS label is seeded with a list of transactions. What follows is a proposed seed for this default menu (it can grow though the OTHER option):

wanted	for-sale	trading	renting
news	events	opinions	resources
services	transportation	political	social
skills-exchange	consumer-reports	graffitti	rent-control
people			
date-message-added		number-of-times-read	

1.7. Indexes => personal labels

Personal labels provide a simple way for individuals to reorganize the messages on CM. Users can take advantage of each others efforts by explicitly asking for message under a particular person's label.

Just as the author of a message can add labels to it in ADD, anyone can add personal labels to any message while in READ. People can see each other's personal labels when READing a message. Personal labels make use of the pen-name that can be set in the MAIN menu; pen-names are unique and can be protected with passwords. Every personal label is automatically prefixed by the pen-name of the person who adds it. This keeps personal labels from directly interfereing with regular labels or other people's personal labels.

Also there is a special (because you can't just add new labels to it with OTHER) label called "PEOPLE"

that has related labels that are pen-names. Each pen-name has related labels that include all the personal labels for that person.

Personal labels can be used to do what indexes do. For example, Tom's restaurant guide may consist of messages labeled "Tom's favorite-restaurants". Tom can add any new restaurant messages he READs by using the "add personal labels" option and then typing (or using EXPLAIN and selecting) "favorite-restaurants". Other people can do FINDs listing "Tom's favorite-restaurants" as a "should have" label (or if they always disagree with Tom's tastes, it can be listed under "should NOT have").

Personal labels can also be used as named hit lists or note-pads. Messages can be added and removed from a note-pad by using READ's "add/remove personal labels" and typing or selecting the name of the pad. FIND can be used to look up messages in a particular pad.

The full power of FIND is available for mixing personal labels from different people and regular author's labels. This allows indexes and note-pads to be searched and refined. Also, different ways of organizing messages can be compared to see what they have in common or in difference. For example, to see restaurants that Tom recommends and Carl doesn't recommend, one would enter "Tom's favorite-restaurants" in the "should have" list and "Carl's favorite-restaurants" in the "should NOT have" list.

2. Buttons

There are two major groups of buttons: bottom line buttons and menu buttons.

2.1. Bottom line buttons

These buttons are press to respond to what the bottom line says.

2.1.1. Yes

Causes what the bottom line says to actually happen. Answers 'yes' to questions on the bottom line. Tells CM that you are done typing.

2.1.2. Explain

Show a longer explanation of what the bottom line is trying to say. This would appear in a larger window overlaying the status line. EXPLAIN is also used to get help while typing in (esp. labels).

2.1.3. No/STOP/Done

Causes what is happening to stop. Answers 'no' to questions on the bottom line.

Most often the response to pressing NO will be an explanation of how to do menu selection, e.g. "Use up and down arrow buttons to choose what you want to do, then press YES"

2.2. Menu buttons

These buttons put what you are doing on hold and show you a fresh menu of things you can do. To return to what you were doing before, press the MAIN button. The MAIN menu gives you a choice of continuing anything that is on hold. A menu button can be press at any point while using CM.

2.2.1. MAIN

Status: "Press YES to pause from <finding, reading, adding> and go back to the MAIN menu"

2.2.2. ADD

Status: "Press YES if you want to add a new message to community memory".

2.2.3. FIND

Status: "Press YES to look at messages in community memory".

2.2.4. GOODBYE

Status: "Press YES to clean up loose ends and finish using CM".

2.3. Arrow buttons

These move the highlighted line around (or letter while editing). This is the primary method for selecting from enus on dumb terminals.

2.4. ADVANCED OPTIONS button

Status: "Press YES to start showing more advanced options". Pressing YES: "Advanced options are displayed, Press ADVANCED again to make them go away".

3. Menus

Before money is put in, a TEASER menu is displayed. Sessions start in the MAIN menu. From there you can go to ADD and FIND.

3.1. TEASER

3.1.1. Short description of CM

3.1.2. Blurb about paying

3.1.3. List of latest added titles (?)

3.2. MAIN

*** MAIN menu of the Community Memory Message Exchange ***

1) I have never used Community Memory before and I want help

2) I want to look at messages in Community Memory

3) I want to add a message to Community Memory

4) I am done using Community Memory

5) I want to start a new pen-name for myself on Community Memory

6) I want to use an old pen-name (any message I add will be labeled with this)

7) I would like to give Community Memory more information about myself

8) Continue <reading, adding> message: <title>.
OR Continue finding messages about: <labels>.

3.2.1. I have never used Community Memory before

This gives an interactive introduction to using CM. It should provide the minimal needed information (terminology, buttons, etc.) to avoid initial turn-off by overstimulation. It should have little game-like demos that allow the user to proceed when the desired

concept is learned. There should be a reality check menu to allow individual confusions to be clarified (esp. a menu of CM words).

3.2.1.1. Description of CM

Community Memory is a way of communicating with other people. Messages can be added to Community Memory in the same way that pieces of paper can be tacked on a bulletin board. A message is put up in a part of the board that is labeled for that type of message. For example, a message about a bicycle for sale would be tacked up under the label "FOR SALE". Community Memory has more space and many more labels than an ordinary bulletin board. For example, there is also a label "BICYCLE" and a label "1Ø-SPEED". Unlike a normal bulletin board, Community Memory allows you to tack your message under as many labels as you want. Your bicycle for sale can be listed under "FOR SALE" and "BICYCLE" and "1Ø-SPEED".

To find the messages you want to look at, give Community Memory the label to look under. You can also give a bunch of labels and Community Memory will try to show you the messages that have as many of those labels as possible.

3.2.1.2. Demo the YES, NO and EXPLAIN buttons

3.2.1.3. Demonstrate the bottom line

3.2.1.4. Demo the up and down arrow buttons

3.2.1.5. Explain Menus

3.2.1.6. Demo the MAIN button

3.2.2. I want to look at messages

Status: "Press YES to show a menu of messages that you can choose to read".

3.2.3. I want to add a message

Status: "Press YES if you want to type a new message into Community Memory".

3.2.4. I am done using Community Memory

3.2.5. I want to start a new pen-name for myself on CM

If the name the user types is already in use, ask for another name. There is an option for passwording a name to protect it from unauthorized use.

3.2.6. I want to use an old pen-name

Ask for password if needed.

3.2.7. I would like to give CM more information about myself

Status: "Press YES to show a menu of things you can choose to tell CM about yourself". Causes a sign-in menu.

3.2.8. Continue ...

Status: "Press YES to return to where you left off <reading, adding, finding> ..." Pressing YES causes a return to that menu. When that menu is paused again to use another menu (by pressing a menu button or using the "Back to MAIN" option), it is added to the top of the continue list. If there is more than a screen full of options the last menu item will be: "*** SEE MORE MENUS YOU CAN CONTINUE BY PRESSING THE DOWN PAGE BUTTON ***".

3.3. SIGN-IN

*** Sign-in to Community Memory ***

1) I want to type in my NAME

2) I want to type in my ADDRESS

3) I want to type in my PHONE NUMBER

4) I want to type in a list of labels related to my INTERESTS

5) I want to type in a SELF DESCRIPTION

6) Done telling CM about myself, go back to MAIN menu

As they are used the menu prompt changes from "I want to type in ..." to "I want to change ... from: <current value>".

3.4. FIND

*** Look at messages in Community Memory ***

1) Type in labels that messages you want SHOULD be under

2) Type in labels that messages you want should NOT be under

3) <n> messages found, choose titles below to read (list of titles)

4) Type in labels that messages you want to see MUST have

5) Type in labels that messages you want to see MUST NOT have

6) Done finding these messages, go back to MAIN MENU of other things you can do

3.4.1. Go back to MAIN menu of other things you can do

This item goes away when ADVANCED OPTIONS are on.

3.4.2. Type in labels that messages you want should (or should NOT) have

Pressing YES causes the menu option to change to: "Desired labels: <cursor>". The status line says: "Press EXPLAIN to get a list of topic labels to choose from, or type in labels, Press DONE when done".

If the user presses EXPLAIN, an overlay window will appear at the bottom of the screen, with a list of TOPIC labels. This overlayed window should be clearly delimited and should have a title: choose labels you want, press DONE when done. The first label on the screen will be highlighted. The status line will read: "Press YES to add <label> to desired labels, Press EXPLAIN to see labels related to <label>, Press DONE when done". As the arrow keys are used to highlight labels in the list, the status line changes to show the currently selected label. As many labels as desired can be added to the list from the menu (by selecting and pressing YES); Putting the cursor under a label (while typing or while in a menu) and pressing EXPLAIN always allows you to trace the associational paths to labels related to the current one.

If the user types in a label, it is case converted and looked up. As the label the cursor is under changes and if the label is known to CM, the bottom line reads: "Press EXPLAIN to see labels related to <label>, Press DONE when done". If it is unknown, the status line: "Don't know label <label>, Press YES and give a word that is related to it".

When DONE is pressed after the first time labels are entered, the menu option turns into: "Change

desired labels from: <label-list>". Using this menu option puts you in a position to type in new labels (cursor at end of list) or change the old ones by using left-arrow and the editor. Putting the cursor under old labels and pressing EXPLAIN will show related labels.

Labels are typed in as a set of words separated by spaces. Multi-word labels have to use minus or underscore (they should map to the same character during case conversion). There are two special types of labels: Personalized labels and Range labels.

Anyone can add their own personal labels to any message in CM (see READ menu). A personal label is associated with a users (unique) pen-name, or with the pen-name "Doe" if the user did not give one. Personalized labels are distinguished from the usual author's labels by being proceeded with a pen-name followed by "'s". For example, Jude may label all of the messages she finds interesting with "Jude's interesting-messages". Carl may do the same, except that his will be labeled as "Carl's interesting-messages" since he signed-in with the pen-name Carl (and entered the optional password he may have associated with his pen-name). Anyone can see messages Jude finds interesting by doing a find on "Jude's interesting-messages". One can also see messages that both Jude and Carl find interesting by doing a find o "Jude's interesting-messages Carl's interesting-messages". Regular labels can also be used, so "politics Carl's interesting-messages" would find political messages that Carl found of interest. All the personal labels on a message can be seen during READ. Everyones personal labels can be browsed by going into the PEOPLE branch of the TOPICS label.

Ranges come in three flavors: price ranges, date ranges and number ranges. A range will have a special label associated with it that begins with one of the words "price", "date" or "number". For example, the label "date-added" would receive special treatment: the following word would be read as a date, if the word after that is the reserved word "to" then the following word would also be read as a date (if there is no "to" then assume the end of range is the same as the beginning). To see messages added to CM between 1-1-83 and 1-1-84, one would say "date-added 1-1-83 to 1-1-84". If an attempt to parse a word as a date, number or price fails, prompt the user for the field in question (e.g. "couldn't understand xxxxxx, type in latest date-added: "). When ranges are entered by being selected from a related label menu, CM will prompt for both lowest and highest values and put them in the

above format on the label list.

Message titles are shown in order from most to least points based on the number of "should have" labels minus the number of "should NOT have" labels. When messages have the same score, they are shown from most to least recently added. Messages with all the "should have" labels will be seen first. Labels that don't have any messages are in effect ignored (the user should probably be informed that the label was not recognized).

3.4.3. Choose from message titles

Use the up and down arrows to highlight the title of the message you want to read. A four line window on the bottom of the list shows some more information about the message (same format as message header used by READ). Status: "Press YES to read a message about: title". Pressing YES causes a new READ screen. If there are more titles than will fit, the last option shows: "*** Press DOWN PAGE to see <n> more titles below ***" If there are titles off the top of the screen, there is a similar message on top. Both can also be selected using the up and down arrow and pressing YES.

A fresh FIND screen might show all messages on CM in chronological order from most to least recently added. Advantages:

A) user doesn't have to type at all to read messages (only up/down arrows and YES button are required)

B) all recent messages get a crack at being seen even if their labels are fucked

C) some users may want to scan all messages added since the last time they looked

3.4.4. Type in labels that messages you want to see MUST/MUST NOT have

This is an advanced option. When labels are added to these lists, the menu items become: "Only find messages about all of: label label ..." and "Never find messages about any of: label label ...". Before scoring, a message will be thrown out if it has any of the "MUST NOT have" labels If the message does not have all of the "MUST HAVE" labels it will also be disposed of before scoring.

3.5. ADD

*** Add a new message to Community Memory ***

1) Type in labels to put this message under

2) Type in a title for this message (this MUST be done)

3) Type in message

4) Destroy message

5) Allow message to be changed at a later time {advanced}

6) Go back to MAIN menu of other things to do

3.5.1. Type in labels to put this message under

This works in exactly the same way as label entry during FIND. We might want to add to the list of labels all of the single words contained in multi-word labels. This allows FINDs on multi-word labels typed in without minus or underscore to have a chance of working. No personal labels can be entered this way (see READ).

3.5.2. Type in message

Goes into editor in the message window. Changes menu option to "Change or add to message".

3.5.3. Destroy message

Must confirm to do it.

3.5.4. Allow message to be changed at a later time

Ask if a password is desired, if not, make it public edit.

3.6. READ

*** Read a message on Community Memory ***

1) Add a comment to this message

2) Print this message {invisible if no printer}

3) Find comments on this message {invisible if no comments on message}

4) See what this message is commenting on {invisible if not a comment}

5) Go to next message on list: <title>

6) Make changes to this message {advanced}

7) Add or Remove personal labels to this message {advanced}

8) Read message by message id number {advanced and contraversial}

9) I agree/disagree with this message

1Ø) Done reading this message, go back to MAIN menu.

3.6.1 Go to next message on list

This option eliminates the need to switch back and forth between FIND and READ when reading a fair percentage of the found messages. The title of the next message in a find is shown. Pressing YES causes a READ of that message. Pressing NO causes the title of the next message to be moved down the hit list in that FIND.

3.6.2. Add or Remove personal labels on this message

Each label is automatically prefixed with the current pen-name. Can not add labels under another pen-name without doing the "use old pen-name" option of the MAIN menu. The EXPLAIN button can be used while entering labels to get a list of all of the labels the current pen-name has. If the label entered is already on the message, ask if the user wants to remove it, otherwise add it.

3.6.3. I agree/disagree with this message

Ask user to rate this message. In return for doing the rating, show how many people have rated it and what the current average rating is.

3.6.4. Message window

The top of the message has a boxed in heading. The header can scroll off the screen as it is just part of the message. It would look like:

```
| Title:     <title>
| By: <pen-name>       Date-added: <date>     Date-last-read: <date>
| Expires: <date>      Commented by <number> messages,   read <number> times
| Labels:  <list of labels>
```

If the message is too long to fit in the window, the last line will show "*** press DOWN PAGE to SEE MORE BELOW (<number> lines) ***". When there is message off screen on top the top line of the window shows: "*** press UP PAGE to see more above (<number> lines) ***".

4. Dialog without special buttons

It is important to provide a concise dialog option for more experienced users.

1) The YES button also appears on the keyboard for use while typing.

2) Pressing the question mark key is the same as pressing EXPLAIN.

3) Pressing the escape key get you back to the MAIN menu.

4) Pressing NEW LINE is the same as pressing YES except while in the editor.

USER INTERFACE FOLKLORE WANTED

More and more work is being done in the area of user interface design, also known as man-computer interaction. Much of this research focuses on the low level details of the interface: keyboard layouts, efficiency of short commands, and the structure of command selection.

But there is currently a bunch of folklore in the CS community that gives us higher level principles. These principles are much less quantitative than the low level principles, and they are also harder to support scientifically. But, I feel that this folklore is very important to people designing REAL computer interfaces.

Won't you please take some time and think of some principles that you use when you design user interfaces, or that your friends use? I am also interested in finding out how you discovered or learned the guidelines that you use. References to previous studies will also be useful.

I will be preparing a summary of what I have gathered, as well as comments by both Cognitive Psychologists and Computer Professionals. Send me mail if you have something to contribute or if you want my summary. The summary will be available about mid June.

To give you the flavor of what I'm asking for, here are some principles that I have heard or used:

1) Principle of least astonishment:
   The actions of a computer system should be predictable by the user of the system. They should fit the mental model that the user has of the system.

2) Make actions retractable:
   It is often desirable make a system easy to learn. In these situations the system should allow the user to retract his commands, as this encourages experimentation. For example, this can be done by providing inverse operations for commands (the ability to move the cursor up OR down) or by providing an undo facility.

3) Let the punishment fit the crime:
   For hard-to-retract changes in the state of a system, the amount of effort required to make the change should be proportional to the magnitude of the change. For instance, it should be easy to delete one file, and much harder to delete all of your files.

I have learned these maxims from colleagues, and don't know the origins beyond that.

--Bob Mayo

Uucp net: ucbvax!mayo
ArpaNet: mayo@berkeley
CSnet: mayo@berkeley
Phone: (415) 642-9716
US Mail: 573 Evans Hall, Computer Science, UC Berkeley, Berkeley, CA 94720
From [illegible]