Andrew Consortium
School of Computer Science
Carnegie Mellon

The Andrew View

Volume 1, Numbers 3 & 4
January, 1993

This double issue of The Andrew View announces and accompanies two major efforts

1) Release 5.2.0 of the Andrew User Interface System, and
2) a report of the feasibility of converting AUIS source to C++.

We also report on current projects and proposed new projects, and ask you to tell us about your site in response to a survey.

Release Plans and Version Numbers

Wilfred J. Hansen, Director

The current members-only release of the Andrew User Interface System, 5.2.0, has been sent to members, as detailed in the next article. If your organization is a member and you have not received this release, please let us know (info-andrew-request@andrew.cmu.edu). If your organization is not a member and you would like this release, this may be a good time to consider joining the Andrew Consortium (same address).

The release of 5.2.0 is a good time to review our future release plans, since they have some innovations. For the first time we will have more than one release in-the-works simultaneously.

In December, 1993, the X Consortium will release the R6 version of X. As part of the X distribution, we release version Andrew version 6.0.x (for a small value of x). Release 6.0.x will be the same as 5.2.x; that is, it will be version 5.2.0 with a few patches to correct core dumps and other egregious errors. At that time, the permissions statement on the sources will be changed to the normal X permissions, allowing general exploitation of the code.

Throughout the year, we will continue to enhance Andrew and release improvements to members. Patch releases correcting errors in 5.2.0 will be called 5.2.1, 5.2.2, and so on. However, releases to members with major enhancements will be labelled 6.1.0 and higher. These enhancements will not be included in the 6.0.x released to the X distribution.

(A future major release might resonably be called 7.0.0. However, some years ago there was a release 7 of Andrew. If anyone is still using it, please let us know and we may skip 7 this time around.)

(Here at the Consortium, we augment our version number with information as to the platform and date of compilation. This is done manually, by editing the file $ANDREWDIR/lib/atkvers.num; the value can be any string up to forty characters or so.)

In addition, we are also creating a small patch to 5.1.0, the CDrom version. This will fix known problems with that release and will augment the sources so they will build on the AIX 3.2 operating system for IBM/RS-6000’s.

Andrew User Interface System: Release 5.2.0

About the turn of the year, release 5.2.0 of AUIS was shipped to members of the Andrew Consortium. This release includes many new features introduced during the year just past. The full set of changes is described in a document entitled "Andrew User Interface System: New and Revised Features in Version 5.2.0." Here are the highlights.

--- New applications and insets ---

Launchapp - Displays a menu of AUIS applications and serves as an alternative to the shell for initiating applications. Give the command ‘launchapp‘; there is no help file; just click on the buttons. (Here at the School of Computer Science, the commands ‘atk‘ and ‘auis‘ also run launchapp. Sites can choose their own aliases.)

Prefed - An editor for the preferences file. It displays existing preferences and offers graphical options to change them; help is available on each preference. For help on the entire application, give the command ‘help prefed‘.

Figure - Drawing editor; faster and more reliable than zip, though it lacks zip’s capability for multiple levels of detail. ‘help figure‘

Image - Supplements the existing raster inset by providing for multibit pixels and thus color and greyscale images. Unlike raster, however, editing operations are not supported. A variety of image storage formats are supported such as jpeg, and gif. ‘help image‘

--- New general facilities ---

Menus now display the equivalent keystroke combinations.

File name canonicalization - In most instances, prompts for a file name will accept the following:

$XXX where XXX is an environment variable,
˜user/ or ˜user@afs-cell/ at the beginning of the file name,
/../ at the beginning of a filename to refer to superroot.

Keyboard-macro-to-ness-macro conversion - With this facility users can define new key stroke and menu operations dynamically. Macros are created with the usual keyboard macro { ˆX-(... ˆX-) } and then converted to Ness, after which they can be edited as Ness code. ‘help macros‘

The ATK copyright notice and disclaimer are now displayed in various key files such as the toplevel Imakefile and the README. It is displayed on starting up programs that use frame.

--- New documentation ---

An introduction to the **Andrew User Environment** for new users is installed into $ANDREWDIR/doc/intro.doc.

The answers to **frequently asked questions** have been amassed into an FAQ.ez file which resides at the top level of the source tree.

Class hierarchy browsing - A graphical representation of the class hierarchy can be created and viewed. The system installation process installs a tree of the standard Andrew classes into $ANDREWDIR/doc/atk/classes.org. You can create trees for other class hierarchies: ‘help mkbrowse‘

The papers from the **1992 Andrew Consortium Technical Conference** are in the source in directory ./doc/papers/atk/conference/

--- New preferences ---

For definitions and setting these preferences, see the ‘prefed‘ application described above.

XStyleSelections and **StrictXStyleSelections** - In some X applications, the act of selecting text automatically makes it the source for any subsequent paste operation. (You can’t do "replace" operations!) These two preferences cause ATK applications to behave just like X applications.

DynamicMessageLineSize, **ResizableMessageLine**, **MinimumMessageLines** The message line can grow and shrink either dynamically or in response to user action. (I recommend setting DynamicMessageLineSize to Yes; it is disabled because others here disliked it in one annoying case that has since been fixed. -wjh)

HighlightToBorders - The highlight of the text selection has been changed so the newline is part of the preceding line rather than the following. Now, when the first word on a line is selected, the highlight includes only that word. The old behavior can be recovered by setting this preference to TRUE.

--- Improvements to existing applications and insets ---

Click boxes in help - All .help files have been converted to use the new ’Help Topic’ font for references. They will appear surrounded with dotted boxes and can be simply clicked on to traverse to the referenced text.

Raster editing - When editing a raster you can now select the "Toolset" menu item; a separate window will appear with a menu of raster operations. In addition, there is an option in the main raster menu to overlay an arbitrary inset, such as text, on top of a raster. After editing the overlay to a desired image, its bits can be "frozen" to become part of the raster image. Thereafter, they can be edited just like any other part of the raster.

Srctext - Andrew has always had a diverse collection of tools for editing files particular formats: ctext for C code, mtext for Modula code, and so on. These have now been superseded with a more generalized mechanism provided by IBM Rochester. The superclass "srctext" provides general source text editing mechanisms which are supplemented by subclasses to provide editors for even more source languages than before. General help is available via ‘help srctext‘. Languages supported include: assembly language, C++, C, Modula-3, Modula-2, and raw ATK data stream.

Bdffont - Numerous bug fixes and cleaning up. The font editor now works on all bdf format fonts we have been able to find.

MIME format - MIME support has been augmented. There is now better support for message bodies of type multipart/digest. And message bodies may contain items of type image/gif, image/pbm, image/ppm, image/pgm, and image/jpeg. There was a bug in the delivery system that kept MIME messages from netnews; this has been fixed. MIME documentation has been added to ‘help messages‘

Ness - Version 1.7 has the functions ReadRawFile, WriteRawFile, QueueAnswer, QueueCancellation, WriteObject, and a collection of functions which emulate REXX functions.

Parsing - A new set of tools has been introduced for token analysis and parsing. They are in the source tree as the atk/syntax subtree. Facilities include a version of the bison parser (fromthe Free Software Foundation), a parse object which uses tables generated by bison, a symbol table, a lexeme scanner and an associated lexeme description processor. These new tools are objects in the Andrew sense so it is easy to incorporate multiple instances in a single application. Moreover, many lexemes are understood from the bison grammar and need not be described redundantly to the lexeme processor.

Posixification - Many files were "Posixified", mostly by Todd Inglett. This meant converting to use posix defined libraries; in particular, strchr() and strrchr() are now used instead of index() and rindex(). The other things done relate to signal handling, particularly the return type, and to return values from wait. Also some prototypes were added under ifdef POSIX (related to signal handler functions), some, if not all of these have been expanded to test POSIX and **__**STDC**__** separately, since under AIX we compile with POSIX compliance, but old style C. In this process various additional corrections from IBM Rochester were incorporated into the sources.

Conversion to C++

One decision of the 1992 Annual Meeting was to study the question of converting the AUIS source code from C to C++. This study has been completed and is reported in an accompanying paper entitled "Prospects for Conversion to C++ of the Andrew User Interface System." The summary of the paper says:

> The Andrew User interface System is written in the C language with a few conventions which create an object-oriented programming environment. The industry has begun to adopt C++ as the object-oriented programming model, so the question addressed in this paper is whether to convert AUIS to C++.
>
> The non-technical aspects of the conversion include the cost of conversion--almost four programmer-years for the Consortium staff and more for member staffs--and the opportunity cost of not pursuing other enhancements to AUIS. After conversion to C++, there is some potential for greater acceptance of AUIS as a toolkit for X since existing toolkits are closely bound to C, however, the Fresco effort may be maturing at about the time the conversion would be done.
>
> The most important technical issue is dynamic loading. When the 40 mega-bytes of Andrew source code are statically linked in one object, the link time and memory occupancy are exorbitant. The only solutions are machine dependent, whether we remain in C or convert to C++. Even if dynamic loading is solved, it may not be feasible to convert AUIS, but experiments have indicated it can be done. One outcome from conversion

is the availability of C++ features such as multiple inheritance and in-line functions.

Over the next months, Consortium members will have to determine whether the potential benefits justify the conversion effort.

Andrew Technical Conference

Proceedings are available for the 1992 conference. Send email to info-andrew-request@andrew.cmu.edu for ordering information.

The 1993 Andrew Technical Conference and Consortium Annual Meeting will be held on the last Thursday and Friday in June. A tutorial will be on Thrusday the 24th and the conference proper on the 25th with the annual meeting at the dinner on the evening between the two days.

Your participation in the conference is welcome. Papers are appropriate on any aspect of Andrew:

applications
experience with users
new objects
proposals for revision of internal protocols

We expect to have an RS/6000 with video projector available if you would like to do a demonstration.

Survey

To learn more about our users we are distributing the following survey. Please take a few moments to complete it and return it to susan+@andrew.cmu.edu. We will share the results of the survey in a later edition of The Andrew View.

(It is not our intention to publish individual answers other than to quote comments; however, we are not taking cryptographic precautions and cannot guarantee the confidentiality of what you send us. If we’ve asked questions you’d rather not answer, please feel free to omit them from your reply.)

~~~~~~
~~~~~~

1993 Survey of Users of the Andrew User Interface System
~~~~~~

Name:
Net address:
Company/division:
Address:
Phone:
Number of machines at site running AUIS:
    1?   2-10?   10-100?   100-1000?   over 1000?
Number of AUIS users at site:
    1?   2-10?   10-100?   100-1000?   over 1000?
Number of AUIS programmers at site:
    1?   2-10?   10-100?   100-1000?   over 1000?
Does your site have AFS:
What system type(s) do you run AUIS on:
What are you using AUIS for:
    Mail
    Document production
    Project development
    Other uses
Which AUIS applications/insets do you most commonly use?
Why are you using AUIS? Did your site consider alternatives?
Other comments:

Please answer the following if you are doing development work using AUIS.
What subject areas are you working on?
What applications/insets have you developed?
Are you interested in sharing code and/or information with other AUIS sites?

MetaMail 2.4

MetaMail is used in AMS, the Andrew Message System, for handling some parts of reading MIME format messages. A version has been incorporated in the Andrew CDrom, but a newer version--MetaMail 2.4--is now available for anonymous ftp from thumper.bellcore.com in directory pub/nsb. For more information contact Nathaniel Borenstein

    Bellcore Room 2D-296
    Morristown, NJ 07962-1910
    (201) 829-4270
    <nsb@thumper.bellcore.com>
~~~~~~

What follows is an extract from the README announcing MetaMail 2.4:

> On behalf of Bellcore, I am happy to announce the availability of version 2.4 of the "metamail" software to the email community. This package, which is available free of charge for unlimited use by anyone for any purpose, is offered in the hope of making multimedia mail (using the MIME standard) more widespread. ...
>
> Metamail is a package that can be used to convert virtually ANY mail-reading program [in our case, the Andrew Message System] into a multimedia mail-reading program. It is an extremely generic implementation of MIME (Multipurpose Internet Mail Extensions), the proposed standard for multimedia mail formats on the Internet. ...
>
> [At heart, the package] allows the easy configuration of mail readers to call external "viewers" for different types of mail. The distribution includes viewers for a number of mail types defined by the MIME standard, so that it is useful immediately and without any special site-specific customization or extension. Types with built-in support in the metamail distribution include:
>
> 1. Plain US ASCII (i.e., English) text, of course.
> 2. Plain text in the ISO-8859-8 (Hebrew/English) character set.
> 3. Richtext (multifont formatted text, termcap-oriented viewer)
> 4. Image formats (using the xloadimage program under X11)
> 5. Audio (initial "viewer" for SPARCstations)
> 6. Multipart mail, combining several other types
> 7. Multipart/alternative mail, offering data in multiple formats.
> 8. Encapsulated messages
> 9. Partial & external messages (for large data objects)
> 10. Arbitrary (untyped) binary data

Projects in Progress

The Consortium staff are currently working on these projects:

- Documentation Request name completion has been added to help. An introduction suitable for publication is being prepared so users can be given a document in addition to being told to use the help application. (A number of other documentation enhancements are incorporated in 5.2.0.)

- Static loading Mechanisms are just about completed for buiilding ATK applications as standalone programs. To build a standalone application, you specify a set of desired objects and then run a utility which determines the complete set of objects needed and creates the commands to build the system.

- Pattern specification in Ness Ness is a string processing language but has so far had only rudimentary patterns. An experimental pattern specification scheme has been designed and will be implemented over the course of the year.

- Patches to the CDrom A few patches have been found that will increase the utility of the CDrom release. These will be available as a floppy disk accompanying the CDrom and will include adaptation to AIX-3.2 on the IBM RS/6000.

In addition to the work above, one of our members has submitted a sketch of some projects they would like us to undertake. If others have comments, we will be happy to hear them, and will accommodate them as best we can. Comments can include suggestions for different priorities, different projects, and possible implementations. It is unlikely that we will complete all of these projects in 1993 (especially if we decide to convert to C++.)

0. Two important bugs to fix. (A) Backwards scrolling should work smoothly even if the text has large insets. (B) Color shading should be more predictable. Only the background color of a view should dictate the shadow colors. The current scheme of ‘‘blending’’ the foreground and background colors is not intuitive.

1. Text should allow multiple simultaneous templates and recursive templates. Multiple templates are needed for packages like compchar which add a template to the document; the current strategy of copying one of the templates into the datastream is not desirable. Recursive templates are needed by our users who frequently customize templates. It will allow them to override styles while still including the system installed template.

2. Motif style ‘‘widget’’ classes. A comprehensive set of widgets is needed to support application construction. Widgets must support interfaces for inclusion in text or applications, printing, setting attributes, and attaching user actions to widgets. Most widgets should support label fields for text or icon labels of the object. To prove the merits of the widget classes and to improve system consistency and appearance, the new widget set should be used for the control panels of raster, figure, lookz, and other applications.

Especially important widgets are:

> *textfield* - a single line of editable text. A feature would tie an optionmenu to the textfield to provide a list of possible ‘‘canned’’ values.
>
> *pushbutton* - A simple 3D pushbutton. The button should support a text label or an icon image (X bitmap or Xpm pixmap).
>
> *checkbutton* - A simple box that may be ‘‘checked.’’ It may be best if checkbutton (like the existing sbutton) supports rows and columns of buttons to allow them to line up neatly.

radiobutton - A radiobutton is like a checkbutton except it works in a group and only one radiobutton in the group may be pressed at a time.

optionmenu - An optionmenu is a small button that when pressed posts a simple popup menu. The user may choose an item from that menu which now becomes the new selection in the button. A nice additional feature would provide a way for the user to ‘‘rotate’’ through the list of options rather than raise a menu (both are useful--click vs. press and hold?).

grouping - Some way to group ‘‘widgets’’ into a user interface. The existing figure, text, and table insets should be made suitable as grouping widgets and mechanisms should be provided to facilitate the creation of other such widgets.

3. Rectangular cut & paste for text. (This feature may be less useful when not working with a constant width font family.) Here is one way to view the problem:

> Text could support multiple selected regions. Sweeping text in ‘‘rectangle mode,’’ for example using shift-button1, would be a short hand way to select many small regions of text (partial lines). A copy will put these regions concatenated with newlines together into the cut buffer. The lines can be pasted in one of two ways. A normal paste will work as it does today. A "rectangular paste" will insert each line from the cut buffer at the same character position as the starting point in consecutive lines of text.

Other approaches are possible; the entire idea requires considerable design work.

4. Paragraph numbers for text. In this mode, a separate fixed width column at the left of a textview will show a paragraph number at the start of each paragraph. This display can be toggled on and off with a command procedure. Clicking on the numbers could possibly have some affect, such as automatically selecting the paragraph’s text. Additionally, perhaps, an inset may be allowed to post icon characters in the paragraph number view. For example, a debugger might toggle the paragraph numbers on and post an arrow icon on a specific line.

5. Architect a new printing model for ATK. It must support both postscript, plaintext and MIME. Inset implementations should be independent of postscript (as they are today independent from X).

6. Develop a new previewer. Base it on ghostscript and/or Adobe’s Display Postscript. Discard the old preview program.

7. Image toolset. Similar to raster’s new toolset for painting/editing. Discard raster.

8. Event-based keyboard model. The keyboard model should change from the old ASCII terminal model into one where all keys/modifiers are available. The keyboard and mouse hit mechanisms might be somehow merged in the toolkit. Views should also be allowed to see other events such as ‘‘enter’’ and ‘‘leave.’’

9. Graphics should support a Graphics Context model. A view would be able to switch among multiple contexts instead of setting all context values for each operation. For compatability, we must provide a default graphics context and reimplement the current methods as macromethods using this default context. Provide new general methods taking a GC as an argument.

10. Text should gracefully handle insets which do not exist. Perhaps it can generate a note inset containing the datastream for an unknown inset?

11. Generalize textview_BalanceCmd. Have it quit checking C syntax (comments, apostrophes, etc). Make it a method so ctext can override it.

12. Eliminate "Update Document" from lookz. Our machines are fast enough for the updating to be dynamic. A preference may be used for slow machines if you think it is necessary.

13. Make the begin/end flags on environments be settable attributes on a per style basis. This will allow srctext to better control specific styles (comments, etc).

14. Implement an "include" mechanism for preferences. Perhaps we could support something like this:

```
*.preferences: file1, file2, file3
```

Where the preferences in the main preferences file would be used first, then this preference will be queried for additional files which will be read in order, followed by global.prf.

15. Undo. Begin by introducing an undo method in the data object class, then have each data object override this method. Simpletext and text are the obvious first candidates for an undo function. An undo command procedure would ask the dataobject to undo the last function. The dataobject may save a history of ‘‘undo information’’ up until the last time it was written. Alternatively, some dataobjects may only support one level of undo (some kind of feedback would be useful to warn the user of this inconsistency). A dataobject would not be expected to undo operations to an embedded inset. It may be expected to delete an inset that was inserted, however. Can you undo an undo operation? We should probably look at some example implementations to see what has been tried.

New Staff

Mary Anne Cowden has joined the staff as secretary and general assistant after previously working for the Center for Machine Translation here at Carnegie Mellon. When not working, she indulges in the creative arts, including drawing, fabric constructions, gastronomic concoctions, and astrology. She has two sons in middle school, one of whom is currently specializing in investment in potential-antique illustrated-story magazines.

The Masthead

The Andrew User Interface System

The Andrew User Interface System (AUIS) is a portable user-interface environment and toolkit that runs under X11. It provides a dynamically-loadable object-oriented environment wherein objects can be embedded in one-another. Thus, one could use our 'generic-object' editor (ez) to edit text that, in addition to containing multiple fonts, contains embedded raster images, spreadsheets, drawings, equations, simple animations, etc. These embedded objects could themselves contain other objects, including text. With the toolkit, programmers can create new objects that can be embedded as easily as those that come with the system. Many objects, including those mentioned above, along with a help system, a system monitoring tool (console), an editor based shell interface (typescript), and support for printing multi-media documents, are included in the release, making it useful to programmers and non-programmers alike.

The Andrew Message System(AMS) is a component of AUIS and provides a multi-media interface to mail and bulletin-boards. AMS contains many advanced features including authentication, return receipts, automatic sorting of mail, vote collection and tabulation, enclosures, audit trails of related messages, and subscription management. It also provides a variety of interfaces that support character-based terminals and low-function personal computers in addition to high-function workstations.

Consortium Services and Offerings

For information about services and offerings of the Andrew Toolkit Consortium please contact us at:

Information Requests
ATK Consortium
Carnegie Mellon University
Smith Hall 106
5000 Forbes Avenue
Pittsburgh, PA 15213-3890
USA
phone: +1-412-268-6710
info-andrew-request@andrew.cmu.edu

We offer:

Memberships at three levels: Full, Distributing, and Associate
Source tape
CDrom with both source and executables
Bibliography
Copies of papers
Videotapes
Conference proceedings

Sources and binaries are available online via anonymous ftp from

emsworth.andrew.cmu.edu (128.2.45.40)

where the full CDrom release is in the ./cdrom directory. See the README there. Another ftp site is

export.lcs.mit.edu (18.30.0.238)

in directory ./contrib/andrew; this is cloned on many servers, world-wide. On the nationwide AFS file system, AUIS is available in

/afs/andrew.cmu.edu/itc/sm/releases/X.V11R5/ftp/cdrom/

Remote Andrew Demo service:

finger help@atk.itc.cmu.edu

News groups:

info-andrew+@andrew.cmu.edu

This is a distribution list offering the news with full AUIS formatting; send subscription requests to info-andrew-request@andrew.cmu.edu. For newsgroup in plain ASCII, see

comp.soft-sys.andrew

Platforms

Andrew has been successfully used on (at least) these platforms:

IBM: RT AOS 3.4, RT AIX 2.2.1, RS/6000 AIX3.1, PS/2 AIX1.2
SUN: Sun3 3.5, Sun3 4.0, Sun4 4.0, Sun3 4.1, Sun4 4.1, Sun4 Mach
DEC: Vax Ultrix 3.1, Vax Ultrix 4.2, Vax BSD, DEC MIPS, Pmax Mach
other: HP, SCO I386, SGI IRIX 4.0, Apollo, Macintosh II MacMach, 486 Mach

Send bug reports to:

info-andrew-bugs@andrew.cmu.edu