Andrew Consortium
School of Computer Science
Carnegie Mellon

The Andrew View

Volume 2, Number 1
April, 1993

Over the past few months at the Andrew Consortium we have been converting the sources to C++ code. News of our progress is below as is further details on these items:

> The Andrew Technical Conference will be held June 25 as previously announced. On the previous day, June 24, a tutorial will cover converting user code to C++ and writing ATK insets in C++.
>
> A patch to the CDrom release 5.1 is available to extend it to AIX 3.2 on the IBM RS/6000.
>
> Version 5.2.1 has been made available to members. Version 5.2.2 with the generator for stand-alone applications can be made available if you really want to try it.
>
> We have completed a User's Guide to the Andrew System.
>
> The survey we sent with the last newsletter reveals that more people are pure users of AUIS than developers.
>
> Metamail 2.4 is available.
>
> Xmosaic provides access to a treasure-house of information through the World-wide Web.

My own View

Wilfred J. Hansen, Director

Despite manufacturers claims to the contrary, they make only one size of disk: too small. Here at the Consortium we occupy a considerable volume of space:

100M for RCS
50M for the sources extracted from RCS
about 85M each for installed (unstripped) versions
of the system on four platforms
about 200M each for object modules created in
building the installed systems for four platforms

Our resources have been stretched as we attempt to maintain versions 5.1, and 5.2.1 while building 5.1.1, 5.2.2, and the C++ version.

Imagine then our pleasure in acquiring two new disks providing altogether more than one and a half gigabytes. We will at last be able to build systems without perpetual worry of space starvation. Do not concern yourself that we will not know what to do with this space; here are some of the things we hope to accomplish now:

Split the RCS tree in two to separate maintenance of 5.2.1 and the C++ version.

Build the system for three more platforms that we use but are not installing.

Create a distribution version with licenses for individual use so we can offer the latest version of AUIS to those working in organizations too poor to join the Consortium.

One of the things I do is to talk to people about AUIS. May 5 and 6 I will talk in Minneapolis and Rochester. If you are interested in inviting me to speak, please let me know. My current topic is "Comprehensive User Interface Environments," with ATK as an example of some things to do and some things not to do. A comprehensive environment covers much more than a typical user interface toolkit; it provides architectures for sharing the screen, user input devices, file streams, cut/copy/paste, memory and processor, application construction, extension language, and printing. Without these architectures, applications become separate entities, each offering excellent solutions in each area, but non-cooperating with any other and all confusing to the user who wants to combine information from one with another.

Andrew Technical Conference

The 1993 Andrew Technical Conference and Consortium Annual Meeting will be held on the last Thursday and Friday in June. The tutorial on Thursday the 24th will cover conversion to C++ of existing code and writing new ATK objects in C++. That evening will be the dinner and annual meeting, with the conference proper on the 25th.

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.

Possible Projects for 1993-1994

The annual meeting portion of the Andrew Technical Conference will discuss various possible future projects for AUIS. In preparation, we have prepared a list of possible projects for consideration. Please send us any comments, additions, or deletions you may have.

"Widgets": Provide a complete set of widgets including textfield, various flavors of button, slider, optionmenu, bargraphs, dials, and so on. Each widget should be able to: be inserted anywhere, print, interact with user, initiate the extension language, report values to the extension language, interface to a style selection service so authors can modify image. The ’figure’ inset needs to be enhanced, if necessary, to serve as a convenient substrate for widgets.

Decorations and other wrappers: Design a mechanism for insets to put on / take off arbitrary decorations (borders, scroll bars, etc) or wrappers (name space, script, size handles, ...). This should have a user-level interface in ADEW or some enhanced version thereof.

Template amenities: Shared templates so every text in a table can always have a template. Multiple and recursive templates so specifications can be distributed.

World-wide web: (See article below.) Build converters to/from the HTML document representation understood by WWW servers. Build a server so web requests can be met directly from document in ATK format (they would be translated to HTML on the fly). Build an AUIS application to access the web.

Architect a new printing model for ATK. Use PostScript(?). Use ghostscript or DisplayPostScript.

Printing amenities like floating keeps, two-column output, and setting the page size.

Toolset for the ‘image’ inset.

Simplify toolsets: If a toolset is visible and there is a click in an inset to which that toolset is relevant, then that toolset should be connected to that inset.

Replace keyboard and mouse hit models with something like events.

Provide a ‘graphics context’ model. This would be used instead of explicitly setting all graphics parameters for each drawing operation.

More and better programmer’s documentation.

Handle gracefully insets which do not exist.

Allow the data of insets to reside outside the file in which the inset is embedded.

Provide a way to filter newsgroups in ‘messages’ so you would only see headers for those messages the filter selects. This may have to be based on headers only to reduce server traffic.

Convert ‘messages’ to some more widely used message server mechanism.

Paragraph or line numbers visible for text (as for example numbering the stanzas of a poem or the lines of legal document).

Implement ‘Undo’.

Notes to members

Following release of 5.2 as reported in the last newsletter, we fixed a number of problems and released the result as 5.2.1. As usual this was made available to members via ftp; some members requested and received the patch on tape.

If any members are really anxious to try it, we can make available to them a copy of the most recent sources. These contain genstatl, as described in a later article, and some other features such as a color model we want to replace and an update to Ness to allow functions as values.

The rights and privleges of membership in the Consortium are spelled out in a descriptive handout. From time-to-time this is revised to clarify descriptions. Recently we changed it to clarify that membership fees are annual. If you need a current copy of the description, please contact wjh@andrew.cmu.edu.

Recent Accomplishments

In addition to the work on genstatl and C++ conversion reported below, we have completed a User’s Guide to Andrew and made a patch for the CDrom sources:

The User's Guide offers, for the first time, a comprehensive description of how to use the Andrew applications and insets. Chapters cover ez, help, typescript, messages, console, raster, figure, table, and many other facets of the system. The manual will be bundled with the CDrom and will be available separately for $15.

The CDrom includes binaries that execute on both AIX3.1 and AIX3.2 on the IBM RS/6000. However, the source code needs a few changes to compile under 3.2. These are available as a patch on a floppy which will be offered for sale at $10.

Projects in Progress

At present the Consortium staff is working full time on conversion of the source code to C++. A few sample programs have been converted, including the null inset (the one cloned by 'createinset'). We are on track with our hope to demonstrate an ez converted to C++ at the Annual Technical Conference.

The conversion process has one automatic step and one manual. The automatic step is to process the .ch and .c files through the converter. The manual step is to iteratively compile and add type information until compilation succeeds; the C++ language is much stricter about types than C has been. The next section is some of the doumentation of the converters.

As part of the conversion process, we are abandoning the 'index' scheme that has been used to map object names to the file in which the dynamically loadable module resides. We are still trying to limit file names to fourteen characters.

C++ Converters

There are two converter programs:

c.n converts .c and .ch files to C++

cH.n examines the .c files and uses the information to prototype
extern declarations in the .h files.

The processing of c.n is detailed below.

Limitations

#if directives may confuse the parser, though in most cases it can cope. One case where it cannot cope is when the first function definition in a file is under a #if, in this case it will add the forward declaration prototypes within the #if, just before the function.

Macros used in the construction of function definitions may confuse the converter. (Particularly when used to allow code to compile with and without prototypes.) A number of special names are recognized as synonyms for storage classes. (CONST,

STATIC, GLOBAL, GVAR, GLOCAL, METHODDEF, etc...)

If a method has a local variable with the same name as a member of the class, references to the local variable are replaced with ::foo. If this change were omitted, a reference to fork() in a class with a member function called fork() would do the wrong thing, even though it would compile cleanly.

c.n: Class file converter

Usage

    nessrun c.n <.c files> [<.ch files>]

Purpose
Converts the indicated class source files to C++, converting the corresponding header files as well.

Here are some of the conversions done to .c source files.

1. If the string AUXMODULE does not occur in the file and there is a string of the form <foo.eh> or "foo.eh" then the corresponding .ch file will be processed as described below for .ch files. First the .c file is scanned to see if InitializeClass, FinalizeObject, and InitializeObject are present; for each, a flag is set to indicate that a suitable declaration must appear in the C++ header file.

2. Comments and constants are collapsed to unique characters. (To simplify further processing.) The contents are preserved elsewhere and restored later.

3. Words reserved in C++ but not in original K&R C are converted by prepending "word_"

4. ClassC declarations "struct basicobject" and "struct classinfo" are converted to the new names "ATK" and "ATKregistryEntry". Structs which are ATK classes are converted from "struct classname" to "class classname".

5. Functions are prototyped. As functions are prototyped, classheader and self arguments are removed from argument lists. Function bodies are corrected accordingly and explicit qualification is added where necessary. The latter means using ::foo, instead of foo, to reference a global variable shadowed by a class or instance variable.

6. The return types for InitializeObject and FinalizeObject are removed.

7. Method and classprocedure definitions are updated for the C++ syntax. (foo__bar -> foo::bar) For InitializeObject, return statements are replaced with the macro THROWONFAILURE(returnvalue), and the name InitializeObject is replaced with the classname, For FinalizeObject the name is replaces with ˜classname.

8. Calls on classprocedures and methods are updated to the C++ syntax.

9. References to foo->header.classname.bar are updated to read foo->bar. (Since data members inherit in C++.)

10. If not an AUXMODULE the ATKdefineRegistry macro is inserted before the first function definition.

11. Prototypes for all non-classprocedure and non-method functions are inserted just before the first function definition.

12. Comments and constants are re-expanded. A warning is given if the conversion process has deleted any comments or constants, since this may totally invalidate the generated files. The processed file is saved as classname.C.

CH files

All applicable operations described for .c files are also carried out for .ch files. In addition:

1. The class header is converted. If the keyword is "package" it is changed to "class". Name keys are eliminated. The keyword "public" is inserted before the superclass. If there is no superclass the superclass "ATK" is substituted. A virtual function declaration for ATKregistry is inserted as the first declaration in the class.

2. A check for duplication is made, and a warning reported if two members of the class have the same name. (This should only happen with data vs macro, method, or classprocedures.)

3. Class Procedures: InitializeObject and FinalizeObject are converted to classname() and ˜classname().

Other class procedures are made static, the return type shifted to before the function name, and the arguments checked for missing types. (Missing types are marked with the string MISSING_ARGUMENT_TYPE.)

4. Methods/Overrides: The return types are shifted to precede the function name and the arguments are checked for missing types.

5. MacroMethods: An attempt is made to intuit the return type from the body of the macro. Macros containing ’=’ are assumed to return void. Macros of the form self->member are assumed to return the type of the member. If the return type cannot be intuited the string MISSING_RETURN_TYPE or MISSING_RETURN_TYPE_SELF_member will be substituted. The arguments are checked for missing types. The storage class is made inline, and the body is converted to be a function body, with a single return statement if the return type is not known to be void.

6. Macros: The string MISSING_RETURN_TYPE is substituted for the return type. The arguments are checked for missing types. The storage class is made inline, and the body is converted to be a function body, with a single return statement.

7. Data: No processing is done.

Genstatl - Statically Linked ATK Applications

Some sites have wanted to create versions of ATK applications that did not involve dynamic loading (which is rather platform dependent). So Rob Ryan wrote a tool called ´**genstatl**' which constructs a Makefile which will build such an application.

Here's a version of the help file which explains:

> Genstatl is a program which assists in the creation of new standalone applications like runapp where some or all objects are dynamically linked. Many classes are left out of the default runapp since there are so many of them, and it isn't clear which will be used. Nor is it clear which classes are necessary to run any particular application.
>
> Genstatl reads a list of desired application classes and creates the instructions to build a statically linked executable that includes the desired classes *and all classes required to run the desired classes.* The actual output of genstatl is two files (Imakefile, and statl.c) in specified directory. For example, suppose you want to build an application consisting of ez and the X interface. Then you create the file 'desiredclasses' containing:
>
> eza
> xws
>
> You create the application in subdirectory newapp by saying

```
% genstatl newappdir < desiredclasses
% cd newappdir
% genmake
% make
% mv runapp newapp
```

> Then the new application can be run as 'newapp' from directory newappdir.

Survey results

In our last newsletter, we posed a survey. Thanks very much to those interested people at nineteen sites who sent in responses. Of the respondents, four are from research companies, nine are from universities, six are from corporations, and one is a personal system used at home by a husband and wife. Please note that these results are merely representative of Andrew usage. We know of numerous sites--including consortium members--from whom no response arrived.

The number of users at each site ranged from one to more than a thousand, with small sites being the more general rule. Of course, the majority of users are in large sites.

# of # of Sites	Users
4	1
7	2-10
4	10-100
2	100-1000
2	>1000

Most sites (eleven) reported running Andrew on more than one different hardware platform, with one running on eight.

# of sites	System Type
14	Sun
8	IBM
7	DEC
4	HP

Others platforms listed include Solbourne, SGI, i486PC.

AUIS is being used for mail management at all but one of the sites and document production at all but four. Just below half (eight) use it for project development.

One of the strong points of Andrew has been its advantages as a system for developing applications, so we anticipated that there would be many "AUIS programmers" reported. However, small sites generally use AUIS for its applications and have at most one AUIS programmer: five sites had none at all and eight sites had only one. Four of the larger sitesclaimed two to ten programmers and only two sites had more than ten.

Sites reported varied sorts of projects under development, many of which have appeared or will appear in the contrib section of the distribution. About a third of respondents--six of the eight reporting project development--expressed an interest in sharing their efforts with other sites. {This is a key part of the role of the Andrew Consortium --wjh}

Only five sites have AFS and of these, all but one have at least a hundred AUIS users.

When asked why they use AUIS, eight respondents indicated the price (free) was a factor. Six users mentioned that the AMS/AMDS interface played a part in their decision, while AFS integration was an important factor for two sites.

Comments included the following:

- I hope that the Andrew effort can continue to be periodically made available in binary form to the general public, not just consortium members. While I might be able to find the money to join the consortium and get future releases it is doubtful.

{The next public distribution will be on the X.V11R6 tape this December.}

- Our focus for ''AUIS'' is almost strictly for mail-related stuff. ATK is viewed as resource-expensive given that a typical configuration here is for several people to share a given workstation using X terminals. Thus, either VUI or BatMail is used as an AMS interface by most folks, who mostly then don't understand why they can't just use MH or UCB mail or whatever, since they don't see the glories of Messages.

{Too bad.}

- Please make any Motif behavior such that it can be disabled.

{Is and will be.}

- AUIS has pioneered many fine issues in User Interface Toolkit and multi-media application development. It is now competing with commercial products which are eating away at AUIS's universe one niche at a time. These commercial products do one thing well, and are supported by staffs of 10's or 100's of people. AUIS's ability to be visible at all in this context sould be recognized for the impressive achievement that it is. In order to continue to compete, the various subsystems must be more robust, and lose the flavor of some student's term project.

{Sigh. All too true. We are working as fast as we can.}

- Considering that Andrew is free, I think you've done a very good job. Though I think it would be nice to have some clearer programmer's documentation (my first few months with Andrew were very difficult!), and also better support for printing such as floating keeps, two-column output, and setting the page size (it's very difficult to produce professional documents without them).

{I've done two column output already by hacking lib/tmac/tmac.atk. Now we just need a better interface. -wjh}

- More cons than pros, but the pros weight more. We keep using AIUS.

{Thanks.}

> *- I really wish that there was an individual membership in the Andrew Consortium. The cost should be ˜$100 and should allow the individual access to the latest sources and non-voting rights.*

{How would it be if we sold copies of the latest to individuals?}

> *- Would be good if distribution included info on AFS, including vendors.*

{Good idea. Will do.}

> *- A lot of bugs exist, one has to read the code to understand it, "no short term plans" for completing programmer's documentation, limited reusability due to class-compiler constraints, very "poor man" object oriented enviroment (concerning inheritance of features), not portable to PC/MS-DOS/MS-Windows world.*

{Hmm. No surprises here. Sigh. Our current switch to C++ will satisfy those people who aren't too picky about their object oriented environments.}

Here are the questions asked on the survey:

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

xmosaic - World-Wide Web Access

An enormous volume of information is now available to X workstations via the 'xmosaic' package which accesses the World-Wide Web established by work at CERN (see below). The xmosaic window displays a document which may contain links to other documents. Many useful pieces of information have been linked into this web, which gets much of its power from its ability to access files via anonymous ftp, nationwide AFS, gopher, archie, and other sources.

Xmosaic sources are available via ftp from host ftp.ncsa.uiuc.edu in directory ./Mosaic. When started you may click on the entry for the World-Wide Web, which brings up the information below.

Although many document formats can be accepted, xmosaic is especially good at displaying documents in HTML format, which is a form of SGML tailored for hypertexts. There have been several requests that AUIS provide conversion to and from SGML and this might be a good place to start. With such converters several options become possible:

find information in HTML on the net and convert to ATK form
convert ATK documents to put them into the web
build an ATK interface to the web similar to xmosaic
    but offering ATK amenities like cut/paste and printing
build a server accessible via the Web's http protocol
    and converting ATK documents on demand to HTML

Possibly some enterprising reader will see these as interesting projects; each would be useful and a contribution to the community.

The following excerpt comes from a document maintained by the *National Center for Supercomputing Applications*. In xmosaic you could click on the words underlined here and display a subsequent or subsidiary document.

World Wide Web

The World Wide Web is a large-scale networked hypertext information system started by CERN, the European Laboratory for Particle Physics in Geneva, Switzerland.

Hypertext is text that is not necessarily linear. The text contains links to other texts or to graphics, videos, or sound. Links are words/phrases designated in a color and/or by underlining depending on your browser. You select a link by clicking on the highlighted word or pressing the return key, again depending on your browser. The word that indicates the link then either changes color or the underlining becomes broken. The same link may be included in multiple documents.

WWW uses a scheme called the ‘‘Universal Resource Locator’’ that helps you locate documents or graphics on the web. Assuming you are using NCSA Mosaic to view this document, notice that at the top of your current screen the Document URL box displays the document source (e.g., http, gopher) along with the directory path of the document or graphic you are viewing.

WWW is primarily hypertext-based and uses its own hypertext document description format called HTML (HyperText Markup Language). For more on HTML, network protocols, searching information, browsers, and the CERN HTML project, click here. Information about document production can be found here.

For more information on WWW, click here; click here for information about CERN staff.

Browsers

There are several types of browsers for hypertext information. Existing WWW browsers include ‘‘ScreenMode’’ browser as well as NCSA Mosaic .

Help Wanted

With the arrival of new Consortium members, we need to enlarge our staff to accomplish more than has been possible so far. We are willing to consider both temporary and permanent, full or part-time, depending on the qualifications of the individual. Here is the offical CMU posting for the full-time position:

JOB NO: 5533

DATE: 04-29-93 JL
TITLE: Systems Programmer
DEPARTMENT: Computer Science
SALARY RANGE: Negotiable
STARTING RANGE: Negotiable
FLSA STATUS: Exempt
HOURS: Full-time

This position is responsible for programming in C and C++ on Unix and X windows to maintain and enhance the Andrew User Interface System source code. Error correction includes problem examination, determination of repair strategy, execution of the repair, testing, and installation within the source code, including documentation. Enhancements includes studying alternative approaches, and making recommendations. Additional programming tasks may be assigned as necessary.

QUALIFICATIONS: BS in math or computer science or equivalent combination of training and experience required. C and Unix programming experience; graphics user interface programming experience; knowledge of the assembly language for at least one computer; and experience utilizing at least one object code debugger required. Experience in one or more of the following areas is preferred: reading and revising the source code of the Andrew Toolkit or the Andrew Message System, sendmail configuration, MIME and other internet protocols, PostScript, DCE, OLE, or CORBA.

This is a summary statement of the responsibilities and qualifications for this position.

AA/EEO EMPLOYER
DO NOT CONTACT DEPARTMENT DIRECTLY
CONTACT EMPLOYMENT OFFICE, SMITH HALL, 268-2046

If you are interested, or know someone who might be, please don't hesitate to let us know.

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 may themselves contain other objects, including text. The release includes many objects, including those mentioned above, along with a help system, a system monitoring tool (console), an editor based shell interface (typescript).

The underlying Andrew Toolkit architecture (ATK) supports not only screen display, but also file storage, cut/paste across windows, an application construction environment, an extension language, and printing. There is full support for programmers to create new objects and new applications.

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.

The following are components are available in version 5.2 and later versions. Some respond to frequently requested applications in X: word processor: ez, drawing editor: figure, mail and news reader: messages, font editor: bdffont, documentation presentation: help, directory browser: bush.

AUIS applications (graphic and interactive)

bush - directory browser
chump - schedule maintainer
console - shell interface / terminal replacement
ez - word processor and program editor
fdbbdf - font editor (for fonts in bdf format)
help - documentation browser (includes all AUIS help files)
launch - provides a menu of AUIS applications
messages - mail and news reader, manager, and composer
pipescript - viewer useful as stdout
prefed - preferences editor
sendmessage - application for sending a message or news posting
typescript - shell interface (terminal substitute)

Graphical, interactive editors that are both applications and insets

eq - equation inset
fad - animation editor
figure - drawing editor
layout - inset for arbitrary layout of enclosed insets
lset - display two adjacent insets
ness - extension and string processing language
org - display and edit hierarchies
page - allow flipping between pages
raster - editor for monochrome bitmapped images
table - table / spreadsheet inset
text - text, document, and program editor (the heart of ez and other applications)

Commonly used non-interactive AUIS applications

datacat - concatenate ATK files
ezprint - print an ATK document
nessrun - runs a Ness script (the extension and string processing language)
preview - preview an ATK document on screen

File format converters

from ATK to: RTF, ASCII, PostScript, troff
to ATK from: ppm, Scribe, RTF, troff, X window dump
convertraster - various raster formats

Embeddable insets not usually used as applications

clock - analog clock
eq - equations
fad - rudimentary animations
header - specify document headers and footers
image - display images
link - hypertext-like link
month - display one month calendar
note - annotation
text - the text object itself
timeoday - digital clock
writestamp - time file was written

Editing tools

compchar - type characters for European languages
compile - support compilation and error review
complete - filename completion for typescript
dired - directory browser
dsearch - dynamic search
ezdiff - compare two ASCII files
filter - process a region of a document through a shell command
isearch - incremental search
lookz - style table editor
spell - spelling correction (uses ispell)

Source text editing tools for

assembler, C++, C, Lisp, man pages, Modula, Modula-3, Pascal

Tools for constructing insets and applications

arb - ADEW application builder interface
bison - GNU parser generator (modified)
class - preprocessor for ATK object system header files
createcon - ADEW tool to generate C code for interface
createinset - create a prototypical inset under a new name
doindex - classC postprocessor
genmake - Makefile generator for classC
gentlex - lexical analyzer generator
makedo - classC post processor to create objects
runadew - ADEW application builder
whichdo - find an inset on the CLASSPATH

Additional insets available for building applications

button, display text string, entry of labeled text string, four position switch, multiple buttons, multiple sliders, multiple string entry, on/off switch, slider, slider controlling array of strings, text list, thumb knob

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, Contributing, 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

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 requests to
info-andrew-requests@andrew.cmu.edu
Send bug reports to
info-andrew-bugs@andrew.cmu.edu