GhostView - VMS

Version 2.0

January 1995

Johannes Plass

Johannes Gutenberg-Universität
Institut für Physik (THEP)
D–55099 Mainz, Germany
plass@dipmza.physik.uni-mainz.de

Contents

"A good previewer is worth one thousand trees."
Comment found on the hep–ph preprint server

1 Abstract

GhostView–VMS allows to view PostScript documents on an X display.

GhostView–VMS does not actively render any PostScript code. This task is transfered to the PostScript interpreter *ghostscript*[1]. *GhostView–VMS* provides the drawing areas for *ghostscript* and both programs communicate and cooperate to display the document in an organized manner.

Prior to displaying a document, *GhostView–VMS* analyzes its structuring comments to collect its characteristic properties. The obtained information includes, but is not restricted to, the number of pages contained in the document, the orientation of the pages and the page medias used. Whenever a page is selected for viewing, *GhostView–VMS* invokes *ghostscript*, restarting it if necessary, and passes, together with various display characteristics, the PostScript code of the particular page to the interpreter. *ghostscript* renders the page and notifies *GhostView–VMS* on completion. A new page may then be selected.

Full functionality of *GhostView–VMS* is restricted to documents that conform to the document structuring conventions (*DSC*). If a document does not conform to these conventions its characteristic properties can not be extracted and the larger part of the operations described in the following can not be used.

This application is derived from Timothy O. Theisen's *ghostview 1.5*. Much of the code used in *GhostView–VMS* is borrowed from the *original* and most of the ideas behind *ghostview 1.5* entered *GhostView–VMS* without or only with slight changes.
Without *ghostview 1.5 GhostView–VMS* would not exist[2].

The X Toolkit used to realize the user interface of *GhostView–VMS* consists of the *X Toolkit Intrinsics* and the *Xaw3D*[3] widget set.

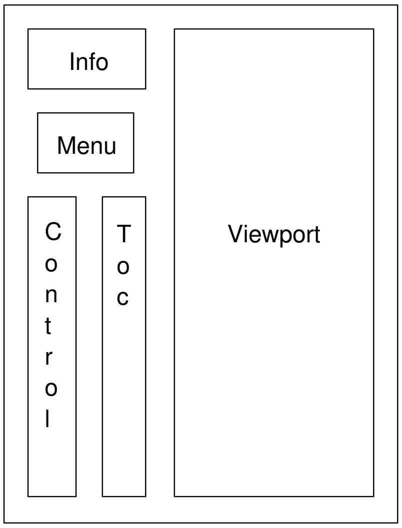
[1] See section 7 on page 25 for further information on the *ghostscript* interpreter.
[2] See section 7 on page 25 for further information on *ghostview 1.5*.
[3] See section 7 on page 25 for further information on this widget set.

2 The User Interface

2.1 The Main Window

The main window of the application is schematically shown in the figure below. Throughout this documentation the main window will be divided into disjoint regions *Info*, *Menu*, *Control*, *Toc* and *Viewport*. Every region contains a number of elements used to to control the application, to inform about its current state, or to display the chosen document. Starting *GhostView–VMS* with the "-ad gv_regions" command line option allows for a visual identification of these regions.

The document will be displayed in the *Viewport* region.
The *Info* region holds a number of informational labels.
The menus of this application are found in the *Menu* region.
The *Toc* region serves as a table of contents and shows a list of page labels as found in the *DSC* comments of the postscript file.
Additional buttons and a scrolling device appear in the *Control* region.



A more detailed description of the different, input accepting regions is given in the following sections.

2.1.1 The *Viewport* region

is the area in which the current page is displayed. The page is always clipped by the *Viewport* region. Obscured portions of the current page may be accessed by

- using keyboard accelerators to scroll the *Viewport* region. (see section 5, p. 19)
- using the scrollbars attached to the sides of the *Viewport* region.
- using the *Panner* found in the *Control* region (see section 2.1.5, p. 5).
- moving the mouse in the *Viewport* region while keeping mouse button 1 pressed.

Clicking mouse button 1 in this region will raise the applications window but does nothing otherwise.
A click with mouse button 2 pops up the *Zoom* popup, showing a magnified portion of the displayed page (see section 2.4.7, p. 8).
Clicking mouse button 3 in this region will *Update* the currently displayed file (see section 2.3, p. 6).

2.1.2 The *Info* region

holds a number of labels containing information about the current environment of the application.

The list below shows all informational labels that may appear in the main window of the user interface.

Title: Shows the title of the document as extracted from the document structuring convention (*DSC*) comments. If no title is provided the name of the displayed file is shown. The *Title* shown may appear clipped due to a limited size of of this label. Pressing mouse button 1 on this label will show the full information.

Date: Shows the creation date of the document as found in the *DSC* comments. If the creation date can not be extracted the modification date of the file is displayed. Pressing mouse button 1 on this label will display the full information.

Locator: Shows the position of the mouse cursor in the *Viewport* region relative to the default user coordinate system.

All informational labels are optional. They may be included or omitted by modifying the *Layout* of the application (see section 3.5, p. 14).

2.1.3 The *Menu* region

contains the menus of the application.

The menus realized in the user interface are labeled *File*, *Page*, *Magstep*, *Orientation* and *Media*. Their entries are listed below.

File:	Open ...	Display another document. The *File Selection* popup allows to choose the new file.
	Reopen	Reopen the currently displayed document.
	Print document ...	Print the current document. Depending on the value of the *confirmPrint* resource the *Dialog* popup will request confirmation to print the file. The *Dialog* also allows to alter the print command used for this job.
	Print marked ...	Print all *Marked* pages of the current document. If no pages are *Marked* the currently displayed page will be printed. (see section 2.1.4,p. 4 for the concept of *Marked* and *Unmarked* pages). As in the case of *Print document* the *Dialog* popup requests confirmation if the resource *confirmPrint* is set to `True`.
	Save document ...	Save the current document. The *File Selection* popup will request the specification of a destination file. For convenience the following convention has been chosen when data is to be saved: if the specification of the destination file includes a version number, *GhostView–VMS* first tries to save the data in the specified destination file. If this operation fails the version number will be removed and another attempt to save the data will be made.

	Save marked ...	Save all *Marked* pages of the current document. The *File Selection* popup will request the specification of a destination file. If no pages are *Marked* the currently shown page will be saved. Note that the destination file is subject to the same convention described in the context of the *Save all* entry.
Page:	Next	Show the next page.
	Redisplay	Redisplay the current page.
	Previous	Show the previous page.
	Center	Center the currently displayed page in the *Viewport* region.

Magstep: Sets the scale at which the document is displayed. The value `0` corresponds to its natural size. Choosing *Magstep* n will scale the document by a factor 1.2^n relative to its natural size.

Orientation: Sets the orientation of the displayed page. When a new file is opened the *DSC* comments are examined to derive the document's preferred *Orientation*. The resulting setting is marked by the document icon in this case.
If the *DSC* comments do not provide a value for the document's *Orientation* the setting specified by the *orientation* resource is used and will be marked by the dot icon ●.
Using mouse button 2 when setting the *Orientation* will *force* the chosen value and override the value found in the *DSC* comments. In this case the selected *Orientation* will be marked by the hammer icon .

Media: Sets the *Page Media* used to display the current page. When a new file is opened the *Page Media* setting is extracted from the *DSC* comments. The *Page Media* is marked by the document icon .
If the extraction fails the *Page Media* setting given by the value of the *pageMedia* resource is used and the current *Page Media* is marked by the dot icon ●.
Using mouse button 2 when selecting a *Page Media* will override the *Page Media* information found in the *DSC* comments. The selected *Page Media* is marked by the hammer icon in this case.

2.1.4 The *Toc* region

serves as a table of contents and holds a list of all page labels as found in the *DSC* comments. The page label of the currently displayed page is marked with a '<'. Using mouse button 1 a new page may be displayed by clicking on its corresponding page label.
If the document conforms to the *DSC* comments a *Marked* property is defined for each page. Pages can be either *Marked* or *Unmarked*. *Marked* pages additionally show a '*' to the left of their page labels. The *Marked* property may be changed in the *Toc* region by

- click on a specific page label with mouse button 2. This will invert the *Marked* property of the corresponding page.
- selecting a range of page labels with mouse button 2 being pressed. This will invert the *Marked* property of all pages in the selected range.
- a triple click on a page label with mouse button 2. This will invert the *Marked* property of

all pages.

- a click on a page label with mouse button 3. This will reset the *Marked* property of all pages.

Some of the buttons found in the *Control* region also allow manipulation of the range of *Marked* pages.

2.1.5 The *Control* region

holds a number of additional control elements to handle the application.

The list below shows all buttons that are realized in the *Control* region. They are identified by the names used when customizing the *Layout* of the user interface (see section 3.5, p. 14).

toggleCurrentPage:	Inverts the *Marked* property of the currently displayed page.
toggleAllPages:	Inverts the *Marked* property of all pages contained in the document.
toggleEvenPages:	Inverts the *Marked* property of all even pages.
toggleOddPages:	Inverts the *Marked* property of all odd pages. By convention the page corresponding to the first page label shown in the *Toc* region is odd.
unmarkAllPages:	Resets the *Marked* property of all pages.
saveMarkedPages:	Pressing this button is equivalent to the use of *Save marked pages* in the *File* menu.
saveAllPages:	Equivalent to the *Save all* entry in the *File* menu.
printMarkedPages:	Equivalent to the *Print marked pages* entry in the *File* menu.
printAllPages:	Equivalent to the *Print document* entry in the *File* menu.
openFile:	Equivalent to the *Open* entry in the *File* menu.
showThisPage:	Equivalent to the *Redisplay* entry in the *Page* menu.
updateFile:	*Updates* the displayed file (see section 2.3, p. 6).
checkFile:	*Checks* the displayed file (see section 2.3, p. 6).
autoResize:	This button will be described in the context of section 2.2, p. 5.

The *Control* region also holds a *Panner* scrolling device. The rectangular *Panner* represents the current page, the smaller enclosed rectangle the *Viewport* region. Moving the mouse curser inside the *Panner* while keeping mouse button 1 pressed will scroll the page in the *Viewport* region.

All buttons listed above and the *Panner* are optional. Their appearance is determined by the *Layout* of the particular instance of the application (see section 3.5, p. 14).

2.2 Size and Resize Policy

At any time there is the notion of a *natural size* the user interface likes to have. It is the size obtained by taking all size related resources of the displayed document into account, possibly updated by preferences extracted from the *DSC* comments.
The *natural size* is subject to the size constraints given by the values of the *maximumWidth* and *maximumHeight* resources (see section 3.3.1, p. 10). *GhostView–VMS* will never attempt to resize

itself beyond the size given by these resources. The window manager may still enforce a larger size so that the user may enlarge the main window manually. In this case the *maximumWidth* and *maximumHeight* resources are adjusted automatically to respect the increased size.

If a new page is to be displayed *GhostView–VMS* recalculates its *natural size* and attempts to resize if the value of the resource *autoResize* (see section 3.3.1, p. 10) is `True`. No resize attempt will be made if this resource is set to `False`.

The value of the *autoResize* resource may be changed at run time by means of the *autoResize* button found in the *Control* region. If the button is labeled `Resize` *GhostView–VMS* tries to resize when the next change of its *natural size* is detected. If the button is labeled `Fixed Size` no attempts will be made to adjust the geometry of the application. Switching from `Fixed Size` to `Resize` causes *GhostView–VMS* to immediately adjust its geometry.

One exception for the size considerations outlined so far is made at startup. If the *geometry* resource (a resource supplied by the *X Toolkit*), is detected in the application's database *GhostView–VMS* enters a visual state with a size as specified by the value of the *geometry* resource. The *natural size* will be ignored in this case. This will also override any settings of the *autoResize* resource and forces *GhostView–VMS* to start in the `Fixed Size` mode.

2.3 Redisplaying udated documents

If a document changes while being displayed the updated file may be displayed without explicitly reloading it. The operations *Check File* and *Update File* handle this situation and are described in this section.

Check File examines the modification date of the file being displayed and reopens the file if a change is detected.

Update File first removes any version number from the file specification of the displayed file and then examines the modification date. If a change is detected the new file will be opened.

The *Check File* operation is performed whenever

- the current page is redisplayed or a new page is selected.
- the application is deiconified.
- the *checkFile* button in the *Control* region is pressed.

The *Update File* operation is performed whenever

- mouse button 3 is clicked in the *Viewport* region.
- the *updateFile* button in the *Control* region is pressed.

2.4 Popup Windows

This section describes additional windows that may appear if requested by the user or required by the application's current state.

2.4.1 The *Copyright* popup

displays a copyright message about *GhostView–VMS*.

2.4.2 The *Dialog* popup

expects input of additional parameters required to proceed with the application.

It may appear if data is to be printed. If the *confirmPrint* resource is set to `True` the *Dialog* popup requests confirmation to print the selected data. It also allows changes to the print command used for this particular print job.

2.4.3 The *File Selection* popup

requests a file specification whenever necessary to proceed with the application.

It appears if a new file is to be opened or if data is to be saved to a file.
The *File Selection* popup shows two text fields, the upper one is used for path specifications, the lower one for file names. The path shown in the upper text field points to the currently selected directory.
Also shown is the contents of the current directory, a hierarchically ordered list of its parent directories and a list of its subdirectories.
A click with mouse button 1 on a parent directory or subdirectory opens the selected directory. All displayed information will be updated accordingly.
A click with mouse button 1 on a file contained in the current directory selects it and inserts its file specification into the two textfields.
The *Rescan* button reads the contents of the specified directory and updates the displayed information.
The *Filter* button causes the *File Selection* popup to display only those files in the current directory that match the file name provided in the lower text field. The wildcard "`*`", representing an arbitrary string, is allowed to appear in the lower text field in this context.
The button labeled `Save Marked` (or *`Open File`*) notifies the application to proceed with the file specified in the two text fields and pops down the *File Selection* popup.
A double click with mouse button 1 on a file contained in the current directory selects this file, advices the application to proceed with this file and pops down the *File Selection* popup.
The *Cancel* button interrupts the operation the file was requested for and pops down the *File Selection* popup.

2.4.4 The *Info* popup

is used to display the output of *ghostscript* subprocesses.

If procedure verification is enabled in the application's process, for instance due to a

```
$ set verify
```

prior to the start of the application, the *Info* popup also displays the commands used to start the *ghostscript* interpreter. This might be used to check how additional, possibly case dependant parameters (specified by the *arguments* resource of the *ghostview widget* or by the *-arguments* command line option) are passed to *ghostscript*.

2.4.5 The ***Note*** popup

informs the user about encountered errors.

It appears if, for instance, data can not be printed or if a selected file can not be accessed.

2.4.6 The ***Options*** popup

allows to edit some of the basic operational defaults.

This version of *GhostView–VMS* allows to modify the following resources at run time:

- *printCommand* (see section 3.3.1, p. 11)
- *confirmPrint* (see section 3.3.1, p. 11)
- *scratchDir* (see section 3.3.1, p. 11)
- *defaultSaveDir* (see section 3.3.1, p. 11)

Pushing the *Apply* button notifies the application of the changes and pops down the *Options* popup. The *Cancel* button pops down the *Options* popup. Any changes made will be ignored if this button is pressed.
Pushing the *Save* button will save the changes to a resource file. If, by means of the *-ad* command line option, an additional resource file was specified at startup, the modified resources will be saved to this file. Otherwise the changes enter the application–specific user resource file DECW$USER_DEFAULTS:GV.DAT.

2.4.7 The ***Zoom*** popup

shows a magnified portion of the current page.

It appears whenever mouse button 2 is clicked in the *Viewport* region. The magnified region of the current page is a rectangle centered with respect to the postion at which the mouse click occured.

3 Customizing *GhostView–VMS*

3.1 Introduction

Like many other X Window applications *GhostView–VMS* makes use of the X Resource Manager to collect data that either is required by the application to function properly or reflects user specific preferences. This data forms the set of resources known to the application and is provided in associated resource files that conform to the resource file syntax defined by the X Resource Manager (see *Xlib - C Language Interface*, *X Toolkit Intrinsics - C Language Interface*). Command line options as described in section 3.4 are additionally taken into account and may be used to override a subset of the resources provided in resource files.

All resources are collected during initialization and enter a common database associated with the particular instance of the application. The construction of this database is usually completed before the application enters a visible state.

3.2 Resource fetching

This section outlines the construction of the resource database associated with a particular instance of *GhostView–VMS*. Steps 1-5 shown below summarize the sources of resource specifications as they may enter the database and may be used to customize *GhostView–VMS*. Possible sources not directly related to *GhostView–VMS* are not listed.

1: The application–specific user resource file

DECW$USER_DEFAULTS:GV.DAT

is loaded into the database.

2: The application–specific class resource file

DECW$SYSTEM_DEFAULTS:GV.DAT

is loaded into the database. If the database and the file contain entries that are equally qualified for a specific resource the entry found in the file will be ignored.

3: Resources resulting from command line options are merged into the database and will override equivalently specified settings.

4: The current database is queried for the resource *userDefaults*. If present the value of this resource points to a context specific user resource file that will be merged into the database. Resources found in this file have the highest priority and will override equivalently specified settings.

The value of the resource *userDefaults* must be either a valid file specification or the name of a file located in DECW$USER_DEFAULTS. In the latter case the default extension `.DAT` may be omitted.

The resource *userDefaults* may be set by using the command line option *-ad <file>* (see section 3.4, p. 13).

5: The current database is queried for the resource *version*. If not present the application–specific class resource files

DECW$USER_DEFAULTS:GV_CLASS.DAT

DECW$SYSTEM_DEFAULTS:GV_CLASS.DAT

are loaded in this order without overriding equivalent prior specifications. This step is usually skipped since the *version* resource might enter the database already in step 2 above.

If one of the above listed sources can not be accessed it will be skipped and the next source is checked. The resulting database will be queried for the *version* resource to detect possible incompatibilities between different versions of the application and the resources loaded so far. If the value of *version* is accepted *GhostView–VMS* proceeds by building the user interface. If not the application will exit.

All entries in the class and user specific resource files are usually prefixed with the application class identifier GV. Using the `-name <name>` X Toolkit option when starting *GhostView–VMS* causes `<name>` prefixed resources to have precedence over those prefixed `GV`. `<name>` prefixed resources should enter the user specific resource file. `<name>` must be a string containing only the letters "a"-"z","A"-"Z","0"-"9","-","_".

3.3 Customizing the resources

3.3.1 Application Defaults

The following table lists all application defaults. The default values shown are provided by the application and serve as fallback values if the sources described in section 3.2 do not contain specifications of these resources.

Name	Class	Type	Default Value
autoCenter	AutoCenter	Boolean	`True`
autoResize	AutoResize	Boolean	`True`
minimumMagstep	Magstep	int	`-5`
maximumMagstep	Magstep	int	`5`
maximumWidth	MaximumWidth	String	`screen-20`
maximumHeight	MaximumHeight	String	`screen-44`
magstep	Magstep	int	`0`
orientation	Orientation	XtPageOrientation	`XtPageOrientationPortrait`
page	Page	String	`NULL`
pageMedia	PageMedia	String	`A4`
forceOrientation	Force	Boolean	`False`
forcePageMedia	Force	Boolean	`False`
confirmPrint	ConfirmPrint	Boolean	`True`
printCommand	PrintCommand	String	`?`
scratchDir	ScratchDir	String	`?`
defaultSaveDir	PreferredDir	String	`?`
selectedBitmap	SelectedBitmap	Pixmap	`None`
documentBitmap	DocumentBitmap	Pixmap	`None`
forcedBitmap	ForcedBitmap	Pixmap	`None`
locatorFormat	LocatorFormat	String	`%d x %d`
version	Version	String	`?`

autoCenter **:** Whether the page should be centered in the *Viewport* region whenever the page size changes.

autoResize **:** Determines the initial resize behaviour of the application (see section 2.2, p. 5).

minimumMagstep*, *maximumMagstep **:** Determines the range of *Magsteps* that may be set.

maximumWidth*, *maximumHeight **:** These resources define the allowed maximum size of the application. *GhostView–VMS* will not try to resize beyond these boundaries. *GhostView–VMS* may be resized to a larger size by the window manager, however. The values of these resources may change at run time and reflect the largest size of the application since *GhostView–VMS* was started (see section 2.2, p. 5).

When deriving suitable values for these resources the window decorations provided by the window manager should be taken into account.

These resources may be specified as a positive integer or as the string `screen`, optionally combined with an offset in the given direction. The string `screen` will be replaced by the size of the screen in the particular direction. The default values shown above provide an example for this syntax.

magstep **:** The initial *Magstep*.

orientation **:** The initial *Orientation*. A resource converter is registered for the values `portrait`, `landscape`, `upside-down` and `seascape`.

page **:** The initial page to be displayed. The value of this resource must be the label of the particular page as shown in the *Toc* region.

pageMedia **:** The initial *Page Media*. Valid standard values for this resource are listed below, the sizes shown are given in units of postscript points.

Value	Width	Height
`Letter`	612	792
`LetterSmall`	612	792
`Tabloid`	792	1224
`Ledger`	1224	792
`Legal`	612	1008
`Statement`	396	612
`Executive`	540	720
`A3`	842	1190
`A4`	595	842
`A4Small`	595	842
`A5`	420	595
`B4`	729	1032
`B5`	516	729
`Folio`	612	936
`Quarto`	610	780
`10x14`	720	1008

forceOrientation **:** If `True` the value of the resource *orientation* overrides the *Orientation* as found in the *DSC* comments.

forcePageMedia **:** If `True` the value of the resource *pageMedia* overrides the *Page Media* as found in the *DSC* comments.

printCommand **:** The command used to print postscript data. The data to be printed is stored in a temporary file, then *printCommand* is applied to the temporary file.

confirmPrint **:** Determines whether to pop up a window that allows to change the print command before individual print operations.

scratchDir **:** The path of the application's scratch directory. Temporary data, like files designated for printing will be created there.

defaultSaveDir **:** The path initially offered in the *File Selection* popup when data is to be saved.

selectedBitmap **:** The Bitmap shown in menus to indicate current settings.

documentBitmap **:** The Bitmap displayed in the *Orientation* and *Page Media* menu if the current setting is prescribed by the *DSC* comments of the displayed document.

forcedBitmap **:** The Bitmap indicating *forced* settings in the *Orientation* and *Page Media* menu.

locatorFormat **:** The format used to display the pointer coordinates in the *Locator* informational Label. The value of this resource should contain two `%d` conversion specifiers and must not consist of more than 40 characters.

***version*:** Defines a version identification for the application defaults file. *GhostView–VMS* will check the value of this resource to detect incompatibilities between the resources and the *GhostView–VMS* executable. *version* is not a public resource and should not be modified.

3.3.2 Resources of the *ghostview widget*

Some of the resources used by *GhostView–VMS* are registered by the *ghostview widget* which is control of all *ghostscript* interpreter subprocesses.
Resources of this widget that may be set by the user are listed below. The shown default values represent the settings found in the source code of this widget.

Name	**Class**	**Type**	**Default value**
arguments	Arguments	String	NULL
busyCursor	Cursor	Cursor	watch
cursor	Cursor	Cursor	crosshair
interpreter	Interpreter	String	gs
palette	Palette	XtPalette	XtPaletteColor
quiet	Quiet	Boolean	True
safer	Safer	Boolean	True
useBackingPixmap	UseBackingPixmap	Boolean	False

***arguments*:** The string value of this resource enters the command line that starts the *ghostscript* interpreter and may be used to pass additional options or files to *ghostscript*.

***busyCursor*:** This is the cursor shown in the widget's window whenever the *ghostscript* subprocess is busy.

***cursor*:** The cursor shown in the widget's window if (the *ghostscript* subprocess) associated with this instance of the widget is idle.

***interpreter*:** The foreign command used to start the *ghostscript* interpreter.

***palette*:** Causes *ghostscript* to restrict to the palette given by the value of this resource when rendering. A resource converter is registered for the values `monochrome`, `grayscale` and `color`.

***quiet*:** If the value of this resource is `True` *ghostscript* will be started with the `-dQUIET` option to suppress informational messages.

***safer*:** If the value of this resource is `True` *ghostscript* will be started with the `-dSAFER` option.

***useBackingPixmap*:** The widget always tries to maintain the contents of obscured parts of its window.

If the value of this resource is `False` the *X Server* is advised that saving off–screen pixels of the widget's window (maintaining backing store) would be beneficial. The widget itself does not actively maintain the contents of its window but relies on the *X Server*in this case. The server is, however, always free to stop maintaining backing store.

If the value of this resource is `True` the *X Server* is advised that maintaining backing store is not useful. The widget tries to allocate a sufficiently large pixmap to store the contents of its window in this case. Note that the server is still free to ignore the request from this widget and to maintain backing store on its own.

3.3.3 Pseudo–resources

Some resources used by *GhostView–VMS* are not known to the X Toolkit. They enter the resource database and are queried explicitly if necessary. The default values shown are as provided in the application–class specific resource files.

Name	Class	Type	Default value
messageFile	MessageFile	String	`DECW$USER_DEFAULTS:GV_MESSAGE.DAT`

***messageFile*:** The full spcification of the file containing text messages to be displayed if, for instance, the application is invoked with the `-?` command line option.

3.4 Command line options

The tables below show some of the command line options accepted by the application. Options not registered by the application, for instance those provided by the *X Toolkit Intrinsics*, are not shown.

Most options override resources provided by the application. They are listed first.

Option	Resource	Value
–ad *file*	*GV.userDefaults*	*file*
–page *label*	*GV.page*	*label*
–center	*GV.autoCenter*	`True`
–nocenter	*GV.autoCenter*	`False`
–magstep *magstep*	*GV.magstep*	*magstep*
–portrait	*GV.orientation*	`Portrait`
–landscape	*GV.orientation*	`Landscape`
–seascape	*GV.orientation*	`Seascape`
–upsidedown	*GV.orientation*	`Upside-down`
–legal	*GV.pageMedia*	`Legal`
–letter	*GV.pageMedia*	`Letter`
–tabloid	*GV.pageMedia*	`Tabloid`
–ledger	*GV.pageMedia*	`Ledger`
–legal	*GV.pageMedia*	`Legal`
–statement	*GV.pageMedia*	`Statement`
–executive	*GV.pageMedia*	`Executive`
–a3	*GV.pageMedia*	`A3`
–a4	*GV.pageMedia*	`A4`
–a5	*GV.pageMedia*	`A5`
–b4	*GV.pageMedia*	`B4`
–b5	*GV.pageMedia*	`B5`
–folio	*GV.pageMedia*	`Folio`
–quarto	*GV.pageMedia*	`Quarto`
–10x14	*GV.pageMedia*	`10x14`
–force	*GV.Force*	`True`
–forcemedia	*GV.forcePageMedia*	`True`
–forceorientation	*GV.forceOrientation*	`True`

Some options registered by the application override resources specific to the *ghostview widget*.

Option	**Resource**	**Value**
–monochrome	*GV*Ghostview.palette*	`Monochrome`
–grayscale	*GV*Ghostview.palette*	`Grayscale`
–color	*GV*Ghostview.palette*	`Color`
–quiet	*GV*Ghostview.quiet*	`True`
–noquiet	*GV*Ghostview.quiet*	`False`
–safer	*GV*Ghostview.safer*	`True`
–nosafer	*GV*Ghostview.safer*	`False`
–arguments *args*	*GV*Ghostview.arguments*	*args*
–xdpi *dpi*	*GV*Ghostview.xdpi*	*dpi*
–ydpi *dpi*	*GV*Ghostview.ydpi*	*dpi*
–dpi *dpi*	*GV*Ghostview.Resolution*	*dpi*
–resolution *dpi*	*GV*Ghostview.Resolution*	*dpi*

The remaining options are not related to the X Window system and do not override any resources. If one of the options listed below is detected during command line parsing the application displays a message in the terminal and exits.

Option	**Action**
–?	Display a short help message and exit.
–h	Equivalent to the –h option.
–help	Display a more explicit help message and exit

The message activated by the –? and –h switches merely list the set of known options. This message is also displayed if an error occurs while the command line is parsed. The message activated by the –help switch is intended to be more explicit and may contain remarks on local customizations. The messages displayed are customizable and are retrieved from a message file that is identified by the value of the *messageFile* resource as found in the application–specific class resource file.

3.5 Customizing the user interface

Geometry management of the user interface of *GhostView–VMS* is provided by the *layout widget* written by Keith Packard. The *Layout* of the application is defined as the combination of all widgets contained in the main window together with the geometrical constraints imposed on these children.
To a large extent the *Layout* of *GhostView–VMS* is determined by the *layout* resource of the *layout widget* responsible for the geometry management within the main window.
This resource is usually specified in an application–class specific resource file (by default in GV_CLASS.DAT) but may equally well enter a customization file which is loaded at startup by means of the *-ad* command line option.

Customization of the user interface may result in

- a modification of the visual appearance of *GhostView–VMS*.

  The *Layout* described in the resource file *gv_landscape.dat* provides an example for this kind of customization. It arranges the *Menus* and the informational labels in a horizontal direction. This layout achieves a better ratio between the visible part of a displayed page and the total size of the application when viewing documents with a horizontal orientation.

- inclusion or exclusion of a subset of all optional widgets supported by *GhostView–VMS*. Widgets affected by this kind of customization are listed in section 6.1, p. 21. *GhostView–VMS* searches the *layout* resource associated with its main window for the names of optional widgets. If a name is detected the corresponding widget will appear in the main window.

  The *Layout* described in the resource file *gv_regions.dat* provides an example for this kind of customization by inclusion of additional labels and lines.

The syntax and interpretation of a *Layout* description is subject of the article *Layout.ps* by Keith Packard. This article is included in the documentation of the *Xaw3D* widget set.

The user should be aware that improper *Layout* descriptions cause *GhostView–VMS* to not function properly.

4 Actions

4.1 Actions registered by the application

This section describes most of the actions registered by *GhostView–VMS*. The default event to action binding is defined in the application class specific resource file. Some of the actions are used internally and should not be used for customization purposes.

GV_Center() Centers the displayed page in the *Viewport* region if the displayed page is partially obscured.

GV_CheckFile(*kind*) Checks if the displayed file has changed and redisplays the new version if necessary. The kind of checking performed is prescribed by the required string parameter *kind*. If *kind* is "date" the modification date of the file is examined. If *kind* is "update" the version number in the file specification of the displayed file is removed, then the modification date is checked. If the specification of the displayed file does not include a version number the use of "date" or "update" result in equivalent actions.

GV_ClearBackground() Clears the background of the *ghostview widget* at startup if *GhostView–VMS* is invoked without a file specification. This is not a public action.

GV_Copyright(*action*) Pops up (down) the *Copyright* popup if the parameter *action* equals the string "popup" ("popdown").

GV_DeleteWindow() Implements Pops down the *Popup Window* in which the received event originated. The main window is not affected by this action.

GV_Dialog(*action*) Pops up (down) the *Dialog* popup if the parameter *action* is equal to the string "popup" ("popdown").

GV_DismissPopup() Pops down the *Popup Window* in which the received event originated. The main window is not affected by this action.

GV_EraseLocator() Erases the displayed coordinates in the *Locator* informational Label.

GV_Force(*flag*) Prepares a forthcoming call of the GV_SetOrientation and GV_SetPageMedia action. GV_Force sets the *forced* flag to the value of the parameter *flag*. If *flag* is "true" ("false") the next call of GV_SetOrientation or GV_SetPageMedia may (may not) override the settings found in the *DSC* comments of the displayed documents. If *flag* is "default" a decision on the it forced behaviour is based on the value of the application resources *forceOrientation* and *forcePageMedia*.

GV_Info(*action*) Pops up (down) the *Info* popup if the parameter *action* has the value "popup" ("popdown").

GV_MovePage(*action*, *smoothing*, *amount*, *relfactor*, *absfactor*) Used to scroll the displayed page in the *Viewport* region. The *action* parameter "start" initializes this action. The triggering event of this invocation must be a *ButtonPress* of any mouse button. If called with the "start" parameter the remaining parameters are ignored and may be omitted.

The *action* parameter "move" represents the state in which scrolling is possible. Scrolling is performed by comparing the events associated with subsequent invocations of this action with the "move" parameter. If invoked with this parameter all remaining listed parameters are required and are interpreted as described below. The triggering events must originate from mouse movement in the *Viewport* region (must be of kind *MotionNotify*). If GV_MovePage detects a preferred direction (either horizontal or vertical) while scrolling is in progress it tries to keep the mouse on track. It does so only if subsequent triggering events of the "move" action are not too far apart. The required parameter *smoothing* is a measure for this upper bound and must be a positive integer.

The parameter *amount* regulates the amount of smoothing. Scrolling horizontally or vertically will become smoother for increasing integer values of the parameter *amount*.

The required parameter *relfactor* relates the scrolled distance for two subsequent events to the ratio between the *Viewport* region and the size of the displayed page. *relfactor* must be a positive integer and scrolling speed will increase for larger values.

The required parameter *absfactor* relates the scrolled distance to the absolute distance between subsequent events. *absfactor* must be a positive integer. Larger values for *absfactor* result in increased scrolling speed.

The *action* parameter "stop" stops the scrolling in progress. GV_MovePage action must be called with the "stop" parameter to properly terminate the overall scrolling action. If invoked with the "stop" *action* the remaining listed parameters may be omitted.

All events triggering the GV_MovePage action must originate in the *Viewport* region, otherwise the action has no effect.

GV_Next() Displays the next page of the document currently in use.

GV_Note(*action*) Pops up (down) the *Note* popup if the value of the parameter *action* is the string "popup" ("popdown").

GV_Open() Opens a new file and displays it in the *Viewport* region. The *File Selection* popup allows to choose the new file.

GV_Options(*action*) Pops up (down) the *Options* popup if the parameter *action* is equal to the string "popup" ("popdown").

GV_Panner(*switch*) Initializes (resets) the *Panner* scrolling device if *switch* is "on" ("off"). This is not a public action.

GV_PopupMenu() Used to pop up the menus associated with the *Date* and *Title* informational labels.

GV_Previous() Displays the previous page of the document currently in use.

GV_Print(*range*) Prints all pages of the current document contained in a range of pages described by the required *range* parameter. If *range* equals the string "all" the whole document will be printed. If *range* equals "marked" only marked pages are printed. If no pages are marked the currently displayed page will be printed if the *action* parameter "marked" is passed to this action. The resource *printCommand* defines the print command that will be used. If the resource *confirmPrint* is `True` the *Dialog* popup asks for confirmation and allows to modify the print command for this particular job.

GV_Quit() Causes the application to exit.

GV_Reopen() Reopens the currently displayed document.

GV_Save(*range*) Saves all pages of the current document contained in the range of pages described by the required *range* parameter to disk. If *range* equals the string "all" the whole document will be saved. If *range* equals "marked" only marked pages are saved. If no pages are marked the currently displayed page will be saved. GV_Save pops up the *File Selection* popup to choose the destination file. The initial path shown in the *File Selection* popup is defined by the value of the application resource *defaultSaveDir*.

GV_Scroll(*direction*) Scrolls the page in the direction described by the string value of the required parameter *direction*. *direction* can be one of the set "up", "down", "left", "right". This action is intended to be bound to keyboard events only.

GV_SetMagstep(*magstep*) Sets the magnification used to display the document. *magstep* must be either an integer in the range specified by the *minimumMagstep*, *maximumMagstep* resources or "+" or "-". The value "+" ("-") will increase (decrease) the currently chosen magstep by 1.

GV_SetOrientation(*orientation*) Sets the orientation used to display the document to *orientation*. *orientation* must be one of the strings "portrait","landscape", "upside-down", "seascape". If not *forced* due to a previous call of the GV_Force action the chosen setting will be accepted only if the *DSC* comments of the displayed document do not prescribe a different orientation.

GV_SetPageMark(*range,action*) Modifies the *Marked* property of the given *range* of pages according to the parameter *action*. Both parameters are required.

If *range* is "all" ("even", "odd") all (only even, only odd) pages as listed in the *Toc* region will be affected. The first page listed is defined to be odd by convention.

If *range* is "selection" the pages that appear highlighted (selected) in the *Toc* region will be subject to the *action*. If *range* is "current" the *Marked* property of the currently displayed page will be modified.

If *action* is "mark" ("unmark") the pages contained in *range* will be marked (unmarked). The *action* parameter "toggle" causes the *Marked* property to be inverted for all pages contained in *range*.

GV_SetPageMedia(*media*) Sets the media used to display the document to *media*. *media* must be the name of a standard media as listed on page 3.3.1 or the name of a media defined in the displayed document. As in the case of the GV_SetOrientation action the chosen *media* will not be set unless it is *forced* or unless the *DSC* comments of the document do not prescribe another media.

GV_Show() Redisplays the current page.

4.2 Actions registered by the *ghostview widget*

The actions listed in this section are defined by the *ghostview widget* and may also be used to customize the event to action bindings in the resource files.

notify(*width*, *height*, *xdpi*, *ydpi*) Calling this action causes *GhostView–VMS* to pop up a *Zoom popup* using a width (height, horizontal resolution, vertical resolution) as given by the *width* (*height*, *xdpi*, *ydpi*) parameters. All parameters are optional.

If *ydpi* is omitted the resolution in both directions will be extracted from the *xdpi* parameter. If *xdpi* is also omitted a default resolution of 300 will be used for both directions. If *height* is not passed to this action the size of the *Zoom* popup is given by the *width* parameter. If *width* is omitted a default size of 300 will be used.

If *GhostView–VMS* is notified with *width* and *height* being 0, the *Locator* informational label will be updated.

5 Keyboard Accelerators

The following table shows the default keyboard event to action bindings as defined in the application class specific resource file GV_CLASS.DAT.

Keys	Action Description
C	Center page
Q	Quit *GhostView–VMS*
O	Open new file
R	Reopen File
Shift-S	Save document
S	Save marked pages (current page if no page is marked)
Shift P	Print document
P	Print marked pages (current page if no page is marked)
BackSpace, Delete, B, Prior	Show previous page
space, Return, F, Next	Show next page
period, Ctrl L	Redisplay current page
M	Mark current page
N	Unmark current page
Shift-5	Set magstep -5
Shift-4	Set magstep -4
Shift-3	Set magstep -3
Shift-2	Set magstep -2
Shift-1	Set magstep -1
0	Set magstep -0
1	Set magstep 1
2	Set magstep 2
3	Set magstep 3
4	Set magstep 4
5	Set magstep 5
+	Increase magstep by 1
-	Decrease magstep by 1
U	Scroll up
D	Scroll down
H	Scroll left
K	Scroll up
J	Scroll down
L	Scroll right
Shift-Ctrl-Up	Force orientation portrait
Shift-Ctrl-Right	Force orientation landscape
Shift-Ctrl-Down	Force orientation upside-down
Shift-Ctrl-Left	Force orientation seascape
Shift-Up	Set orientation portrait in force mode "default"
Shift-Right	Set orientation landscape in force mode "default"
Shift-Down	Set orientation upside-down in force mode "default"
Shift-Left	Set orientation seascape in force mode "default"
Up	Scroll up
Right	Scroll right
Down	Scroll down
Left	Scroll left

6 The Widget Tree

The application class of *GhostView–VMS* is GV.

The name of a particular instance of the application defaults to its class name, but may change if the *X Toolkit Intrinsics* option `-name <name>` is used when the application is started.

The tables in this section list all widgets created by the application. The indentation of the rows indicates the parent-child relationship of the widgets shown.

The **Widget** field shows the names of the widgets as they appear in the source code of this application and are not of public interest. Widgets labeled *internal* in this field are created internally by other widgets.

The **Name** field shows the creation names of the widgets. These entries may be used to refer to a particular widget when resources are set.

Correspondingly the **Class Name** field holds the name of the widget class a particular widget belongs to. These entries may be used to set resources for all widgets in this class.

6.1 The Main Window

The name of the application shell of *GhostView–VMS* defaults to the appplication class GV.
This name may change if the *X Toolkit Intrinsics* option `-name <name>` is used.

Widget	**Name**	**Class Name**
toplevel	*see above*	ApplicationShell
control	control	Layout
fileButton	fileButton	MenuButton
fileMenu	menu	SimpleMenu
openEntry	open	SmeBSB
reopenEntry	reopen	SmeBSB
printAllEntry	printAllPages	SmeBSB
printMarkedEntry	printMarkedPages	SmeBSB
saveAllEntry	saveAllPages	SmeBSB
saveMarkedEntry	saveMarkedPages	SmeBSB
lineEntry	line	SmeLine
optionEntry	options	SmeBSB
copyrightEntry	copyright	SmeBSB
quitEntry	quit	SmeBSB
pageButton	pageButton	MenuButton
pageMenu	menu	SimpleMenu
nextEntry	next	SmeBSB
showEntry	show	SmeBSB
prevEntry	prev	SmeBSB
lineEntry	line	SmeLine
centerEntry	center	SmeBSB
magstepButton	magstepButton	MenuButton
magstepMenu	menu	SimpleMenu
magstepEntry[0]	-5	SmeBSB
⋮	⋮	⋮
magstepEntry[10]	5	SmeBSB
orientationButton	orientationButton	MenuButton
orientationMenu	documentMenu	SimpleMenu
portraitEntry	portrait	SmeBSB

landscapeEntry	landscape	SmeBSB
upsidedownEntry	upsidedown	SmeBSB
seascapeEntry	seascape	SmeBSB
pagemediaButton	pagemediaButton	MenuButton
pagemediaMenu	documentMenu	SimpleMenu
Letter	Letter	SmeBSB
Tabloid	Tabloid	SmeBSB
Ledger	Ledger	SmeBSB
Legal	Legal	SmeBSB
Statement	Statement	SmeBSB
Executive	Executive	SmeBSB
A3	A3	SmeBSB
A4	A4	SmeBSB
A5	A5	SmeBSB
B4	B4	SmeBSB
B5	B5	SmeBSB
Folio	Folio	SmeBSB
Quarto	Quarto	SmeBSB
10x14	10x14	SmeBSB
toc3d	toc3d	Form3d
toc	toc	Text
pageview	pageview	Viewport
internal	clip	Core
page	page	Ghostview
internal	horizontal	Scrollbar
internal	vertical	Scrollbar

A number of widgets appear in the user interface on an optional basis. All optional widgets are children of the control widget above (the Layout widget). If the name of one of the widgets listed below is included in the *Layout* description of the application the associated widget will appear. The indented lines indicate children that are automatically created together with their parent. The names of these children should not be used in the *Layout* description.
As mentioned in section 3.5 (p. 14), new *Layouts* may be loaded at startup by means of the *-ad* command line option.

Widget	**Name**	**Class Name**
titlebutton	titleButton	Label
titlemenu	labelMenu	SimpleMenu
entry	title	SmeBSB
datebutton	dateButton	Label
datemenu	labelMenu	SimpleMenu
entry	date	SmeBSB
locator	locator	Label
w_toggleCurrentPage	toggleCurrentPage	Command
w_toggleAllPages	toggleAllPages	Command
w_toggleEvenPages	toggleEvenPages	Command
w_toggleOddPages	toggleOddPages	Command
w_unmarkAllPages	unmarkAllPages	Command
w_printAllPages	printAllPages	Command
w_printMarkedPages	printMarkedPages	Command
w_saveAllPages	saveAllPages	Command
w_saveMarkedPages	saveMarkedPages	Command
w_openFile	openFile	Command
w_autoResize	autoResize	Command
w_showThisPage	showThisPage	Command
w_updateFile	updateFile	Command
w_checkFile	checkFile	Command
panner3d	panner3d	Form3d
panner	panner	Panner

A small number of labels and lines may be used to support a particular *Layout*.
The number of optional labels is restricted to 9.

Widget	**Name**	**Class Name**
internal	label1	Label
⋮	⋮	⋮
internal	label9	Label

If n labels should enter the application the names "label1", . . . , "label$n$" must appear in the associated *Layout*. Numbering must start at 1 and no number must be left out.

The number of optional lines is restricted to 9.

Widget	**Name**	**Class Name**
internal	line1	ThreeD
⋮	⋮	⋮
internal	line9	ThreeD

The same numbering rules as for the optional labels have to be obeyed.

6.2 Popup Windows

All *Popup* windows are direct children of the toplevel ApplicationShell.

6.2.1 *Copyright* Popup

Widget	**Name**	**Class Name**
versionpopup	versionPopup	TransientShell
versionLayout	versionLayout	Layout
versionLabel	versionTexta	Label
versionLabel	versionTextb	Label
versionLabel	authora	Label
versionLabel	authorb	Label
versionLabel	authorc	Label
versionLicense	license	Label
versionDone	versionDone	Command

6.2.2 *Dialog* Popup

Widget	**Name**	**Class Name**
dialogpopup	dialogPopup	TransientShell
dialogLayout	dialogLayout	Layout
theLabel	dialog_textLabel	Label
theText3d	dialog_textText3d	Form3d
theText	dialog_textText	Text
messageLabel	message	Label
doneButton	done	Command

cancelButton	cancel	Command

6.2.3 *File Selection* Popup

Widget	Name	Class Name
FileSel_popup	fileSelPopup	TransientShell
FileSel	fileSel	FileSelection
internal	rescan	Command
internal	filter	Command
internal	pathform	Form3d
internal	path	Text
internal	fileform	Form3d
internal	file	Text
internal	paned	Paned
internal	topform	Form3d
internal	topview	Viewport
internal	clip	Core
internal	toplist	List
internal	horizontal	Scrollbar
internal	vertical	Scrollbar
internal	grip	Grip
internal	curform	Form3d
internal	curview	Viewport
internal	clip	Core
internal	curlist	List
internal	horizontal	Scrollbar
internal	vertical	Scrollbar
internal	grip	Grip
internal	subform	Form3d
internal	subview	Viewport
internal	clip	Core
internal	sublist	List
internal	horizontal	Scrollbar
internal	vertical	Scrollbar
internal	button1	Command
internal	button2	Command
internal	button3	Command

6.2.4 *Info* Popup

Widget	Name	Class Name
infopopup	infoPopup	TopLevelShell
infoform	form	Form
infotext	text	Text
infodismiss	dismiss	Command

6.2.5 *Note* Popup

Widget	Name	Class Name
notepopup	notePopup	TransientShell
noteLayout	noteLayout	Layout
messageLabel	message	Label
doneButton	done	Command

6.2.6 *Options* Popup

Widget	Name	Class Name
optionpopup	optionPopup	TransientShell
optionLayout	optionLayout	Layout
theLabel	confirmPrintLabel	Label
confirmPrintToggle	confirmPrintToggle	Toggle
theLabel	printCommandLabel	Label
theText3d	printCommandText3d	Form3d
theText	printCommandText	Text
theLabel	scratchDirLabel	Label
theText3d	scratchDirText3d	Form3d
theText	scratchDirText	Text
theLabel	saveDirLabel	Label
theText3d	saveDirText3d	Form3d
theText	saveDirText	Text
doneButton	done	Command
saveButton	save	Command
cancelButton	cancel	Command

6.2.7 *Zoom* Popup

Widget	Name	Class Name
zoom	zoom	TopLevelShell
zoomform	form	Form
zoompage	page	Ghostview
zoomdismiss	dismiss	Command

7 References

This application is based on Timothy O. Theisen's *ghostview 1.5*. Many changes have been applied to the original code but certainly the relation to its origin is quite intimate. Note however, that the author of *ghostview 1.5* is not related to *GhostView–VMS* otherwise. Therefore questions, suggestions or bug reports should not be send to Timothy Theisen.
ghostview 1.5 may be obtained via anonymous ftp from

site:	ftp.cs.wisc.edu
directory:	pub/ghost

ghostview 1.5 contains the following copyright notice:

Author:	Tim Theisen	Systems Programmer
Internet:	tim@cs.wisc.edu	Department of Computer Sciences
UUCP:	uwvax!tim	University of Wisconsin-Madison
Phone:	(608)262-0438	1210 West Dayton Street
FAX:	(608)262-9777	Madison, WI 53706

Kaleb S. Keithley turned *Xaw* into the *Xaw3D* widget set used to realize the user interface of *GhostView–VMS*. The original sources of this widget set may be obtained via anonymous ftp from

site:	ftp.x.org
directory:	contrib/widgets/Xaw3d

All rendering of any PostScript code is done by the *ghostscript* interpreter. *ghostscript* may be obtained via anonymous ftp from

site:	ftp.cd.wisc.edu
directory:	pub/ghost

8 Acknowledgements

I'd like to thank L. Peter Deutsch and Timothy Theisen for the *free* software that makes up the excellent *ghost** series.

Kaleb S.Keithley has to be thanked for preparing the extremely nice and useful *Xaw3D* widget set. In this context I'd also like to thank Keith Packard for his *layout widget*.

Many people have *beta–tested GhostView–VMS* and the VMS port of the *Xaw3D* widget set. I'd like to thank them all.

Especially Dan Pirjol (pirjol@dipmza.physik.uni-mainz.de) disrespectfully continued complaining

about any convenient features he was missing in earlier versions of *GhostView–VMS*. I hope he is content now.

Special thanks is due to Jim Jennis (James_H_Jennis@mail.mmmg.com). If there is something like the notion of a *chief–beta–tester* he deserves this attribute. His careful and devoted testing revealed many severe errors. I hope they are fixed now.

9 Author

GhostView–VMS is written by:

Johannes Plass plass@dipmza.physik.uni-mainz.de
Department of Physics
Johannes Gutenberg University
Mainz, Germany

GhostView–VMS borrows much of its code from *ghostview 1.5*, written by

Author:	Tim Theisen	Systems Programmer
Internet:	tim@cs.wisc.edu	Department of Computer Sciences
UUCP:	uwvax!tim	University of Wisconsin-Madison
Phone:	(608)262-0438	1210 West Dayton Street
FAX:	(608)262-9777	Madison, WI 53706