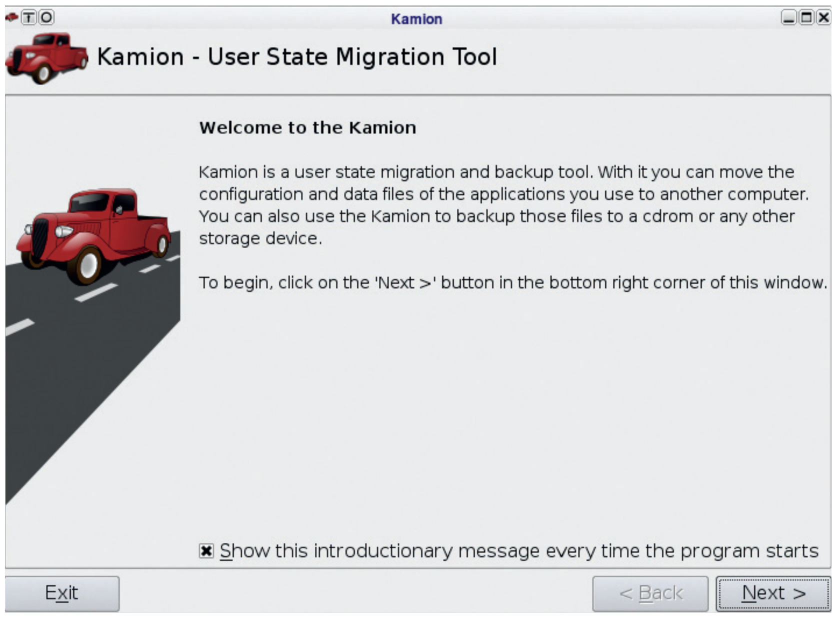
KDE 4 gets wheels

Moving house is a right pain. Moving between desktops can be a bit awkward too. **Nathan Sanders** reveals a new KDE app to ease the pain of migrations.

KDE pops up in more places than you might expect. Its focus on options, not simplicity, may alienate some, but control freaks tend to have a hard time using any other desktop environment and, now that the user state migration tool *Kamion* is here to help, being a control freak is about to get a lot easier. Here's how...

If you're running KDE across multiple boxes, you're probably not innocent of rummaging through dozens of settings dialogs, reprogramming each of your systems to act in nearly the same way. You're probably also the type who obsessively distributes backups of every last little bit of data to all your PCs. *Kamion* (which is part of the raft of new tools in KDE 4) is meant to aid both these compulsions. It uses a database and user input to intelligently pick out the significant bits of your personal and configuration files, then package them up in an archive so you can redeploy them on every errant box you can get your hands on. It's a user state migration tool, and can transfer the entirety of your setup from one KDE desktop to another.

› The old pickup truck might remind you of *The Beverly Hillbillies*, but *Kamion*'s methods for hauling junk are impressively up to date.


Kamion

Kamion - User State Migration Tool

Welcome to the Kamion

Kamion is a user state migration and backup tool. With it you can move the configuration and data files of the applications you use to another computer. You can also use the Kamion to backup those files to a cdrom or any other storage device.

To begin, click on the 'Next >' button in the bottom right corner of this window.

☒ Show this introductionary message every time the program starts

If you were to just copy and paste your **/home** directory to another machine, you'd waste time transferring bulky and unwanted browser caches. You'd also test your patience once you tried to launch programs on the second box with configuration files from the first box, since differing application versions can often cause conflict. Unlike this brute backup method, *Kamion* (Serbian for 'truck') knows what and what not to tote along to a new home. Ideally, every desktop application on your computer will submit an entry to *Kamion*'s application database describing

"Kamion can transfer the entirety of your setup from one desktop to another."

which of its myriad files is valuable, which versions of the application they are valuable to, and which are just waste by-products of daily operation. *Kamion* uses this information to compile archives of your files that can be restored on other computers to produce an environment that duplicates the original.

If you don't plan to use KDE 4, *Kamion* might still have something to offer you. Lead developer Ivan Cukic began the project as part of the Google Summer of Code spin-off The Season of KDE. He's building *Kamion* in two sections: a simple graphical interface and a back-end library. The back-end library, *libKamion*, will be usable by front-ends for other desktops or by individual applications to offer specialised backup capabilities.

A guided tour

For you brave souls using a development build of KDE 4 from *Subversion*, here's a brief tutorial for using *Kamion*. If you don't have KDE 4 and would like to learn how to install it, look for help at **http://developernew.kde.org/Getting_Started**. Even if you just want to see how *Kamion* looks and works, read on. Note that you'll need to find some application user state XML files to be able to make user state backups with *Kamion*.

The walkthrough on page 59 shows what the current version of the *Kamion Qt 4* GUI has to offer, but this represents only the beginning of *libKamion's* potential. On Cukic's to-do list is a mechanism for including arbitrary files and directories in a **.kamion** archive so that you can back up files from a **Documents** folder or an application that isn't automatically recognised. At the moment, there aren't any plans for adding a feature for keeping two machines' user states synchronised. Cukic feels that such functionality goes beyond the scope of his *Qt 4* interface, but he reckons that it would be a fairly easy component to build on top of *libKamion*.

Something for the developers

Kamion's future as a useful app is entirely dependent on how much support it gets from other applications. Apps that ship with user state XML files for *Kamion* to compile into its *SQLite 3* database (which is essentially just a fast cache for those XML files) will work seamlessly. *Kamion* will be able to back up and restore data and configurations for those applications effortlessly. Those that don't provide XML files will be ignored by *Kamion*, forcing you to back up and restore the user state manually, or simply abandon the idea altogether.

Fortunately, *Kamion* makes it fairly easy for applications to provide support. The XML files they are expected to provide, which have been submitted for Freedesktop.org standardisation, need only a few bits of information:

» The application's name.

» Some sort of general category for the application – 'web. browser', for instance.

» A brief description of the application. This could be as simple as 'IM program'.

» An icon to visually represent the program. Preferably, this will be the same one used throughout the desktop.

» An existence test file. This is the location of a file that's created when the application runs. The existence of the XML file itself demonstrates that the user has installed the application, but this test file demonstrates to *Kamion* that the user has used the application and would probably like to back up its user state.

» A list of resources that belong to the application's user state. These should include all of the configuration and data files that an application uses. Each resource must have a unique ID (such as 'kopete.history') and name (such as 'chat history'), file location (eg **$HOME/.kde/share/apps/kopete_history**), and description (eg 'Record of IM conversations').

» Some sort of version number. This could be the version of the application or the configuration file. If the version of the application is specified, *Kamion* will warn the user after every incremental update that their resource is incompatible. If a configuration file version is specified, *Kamion* could safely exchange files between many different versions of the application.

If the application developers are willing, this functionality could be extended. As it stands, *Kamion* will merely warn a user if they attempt to restore an incompatible configuration file. Applications could in future provide *Kamion* with conversion scripts to update the configuration files. Then again, it might be easier if the applications handled the conversion of outdated configuration files themselves, and simply instructed *Kamion* to allow configuration files from several different application versions.

Standardisation

Freedesktop.org standards already demand that applications submit other files used to integrate them with the desktop. Each application has its own desktop entry (**.desktop** files), which controls what appears in a user's programs menu and the way in which the programs are launched (ie which commands should be executed to make the app start). Asking applications to submit another standardised file – a *Kamion* user state specification – is therefore not overly demanding or even novel.

Unfortunately, in a *Linux Format* straw poll of application developers, most respondents hadn't even heard of *Kamion*. Even developers working on other KDE 4 applications were completely unaware of its existence. Given that *Kamion* has yet to receive any »

User state migration in Windows

With the release of Windows XP, Microsoft introduced the *Files And Settings Transfer Wizard* (*FAST*) to help users upgrading from Windows 95, 98, ME and 2000. Microsoft also released a more customisable version called *User State Migration Tool* (*USMT*) for corporate customers. With Vista came a new version of *USMT*, 3.0, and *Windows Easy Transfer*, a replacement for *FAST*.

Unlike *Kamion*, these tools seem more focused on actually migrating users to a new PC or operating system than simply making backups or duplicating functionality across two simultaneously running setups. To this end, they have a lot of nifty features for the immediate transfer and restoration of data on other PCs over USB cables, crossover Ethernet cords and other quick hacks. They can also transfer user accounts and other low-level system details that might create complications in Unix environments.

Like *Kamion*, these Windows user state migration tools are designed to back up and restore personal and configuration files. Just like *Kamion*, Microsoft's tools can preserve files only from the specific set of applications that it recognises (mostly Microsoft's own products) without demanding user intervention. *Kamion* is not long into development so it doesn't have much application support yet, but with a Freedesktop.org-standardised user state XML file format, it could eventually have tremendous compatibility.

Backup vs user state migration

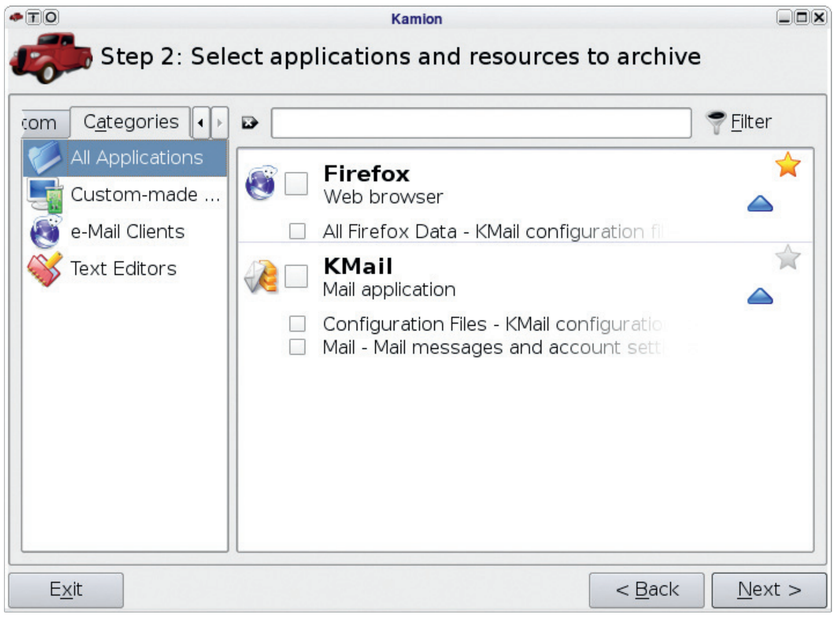
We've all used backup tools, but how are they different from user state migration tools?

Task	Backup tool	User state migration tool
Archives personal files	✔	✔
Compresses archives	✔	✔
Archives application configuration files	✔	✔
Warns against restoring incompatible configuration files	✗	✔
Automatically ignores caches and other junk files	✗	✔
Prevents restoring configuration files for unused programs	✗	✔

» sort of promotion, this is not surprising. Nonetheless, it indicates that substantial publicity work will be necessary to ensure widespread compatibility with applications. Attaining Freedesktop.org standardisation is the first step in this process.

Ivan Cukic is developing *Kamion* with KDE 4 and a Linux environment in mind, but his code has the potential to go much further than one platform. *Kamion* should run with no problems on other Unix-like operating systems, such as Solaris and the various BSDs. One platform that might be beyond its reach is Windows, which Cukic seems to have no interest in supporting. A Windows version would require significant modification to support applications that do not, as Cukic says, "follow the Unix style of storing data (dot files and directories in the home directory)".

Following Trolltech's announcement that *Qt 4*, the toolkit on which KDE 4 is based, will be available for Windows under the GPL, KDE developers have succeeded in porting much of the desktop to run natively on Windows. The Microsoft operating system's large user base offers obvious potential for the spread of KDE applications and technologies as well as for luring in new developers. Nonetheless, some parts of KDE 4 – such as the window manager *KWin*, and perhaps *Kamion* – will be of little use on Windows and will not be ported. Indeed, tools similar to *Kamion* already exist for Windows *(see User State Migration In Windows box, previous page)*.


Kamion
Step 2: Select applications and resources to archive
Categories
All Applications
Custom-made ...
e-Mail Clients
Text Editors
Filter
Firefox
Web browser
All Firefox Data - KMail configuration fi
KMail
Mail application
Configuration Files - KMail configuratio
Mail - Mail messages and account sett
Exit
< Back
Next >

› ***Kamion*'s *Qt 4* interface is pretty flashy. You'll see smooth animations when you expand the resource lists and text that fades out if it's too long to fit the window.**

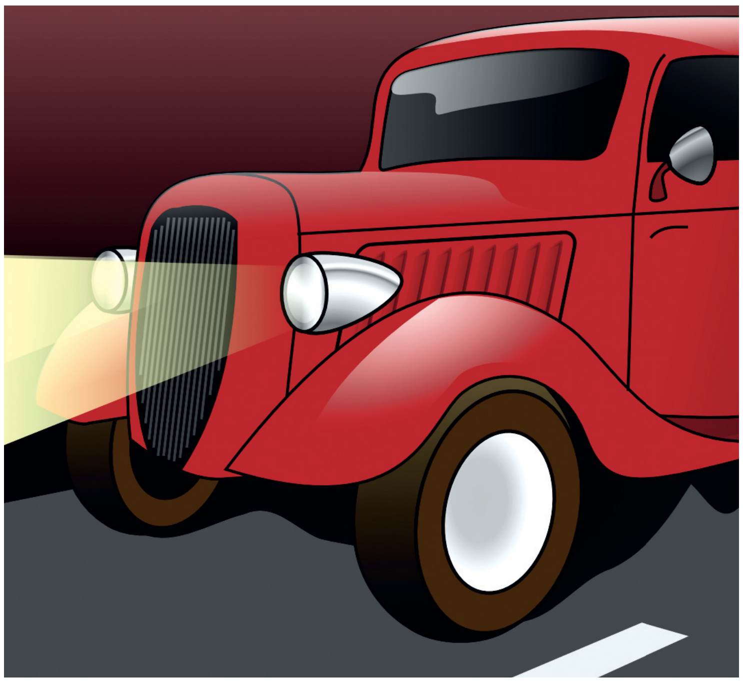
Beyond KDE 4

Although Cukic is not willing to support *Kamion* on Windows, it wouldn't be hard for others to do it for him. His *Qt 4* interface for *Kamion*, the only GUI for his user state migration tool that exists, is merely a front-end to an underlying library called *libKamion*. From the initiation of the *Kamion* project, it was Cukic's intention to maintain the separation between the back-end library and the user interface so that other developers can write their own UIs. Individual applications could even implement personalised *libKamion* front-ends. An email app, for instance, could use *libKamion* to let users package their own mail backups. Cukic has already written simple console-based tools that use *libKamion*.

Developers need not wait to start taking advantage of *libKamion* – Cukic declared it "100% completed" last August. The back-end library provides all the necessary functionality for building a user state migration tool. It can query a database for application information, collect and archive necessary files, and restore files from archives. *libKamion*'s only dependencies are *SQLite 3* and *Qt 4*, although Cukic hopes to eventually release a version that uses the *Xerces-C* XML parser and the *PStreams* process controller in order to remove the *Qt 4* dependency.

It may seem likely that Gnome developers will construct their own user-state migration solution rather than adopt *Kamion*, but it's not unprecedented for technologies to be integrated into multiple desktops (see, for instance, the PDF viewer *Poppler* and the messaging system D-BUS). As all of Cukic's code is released under the GPL 2, port developers have almost free rein.

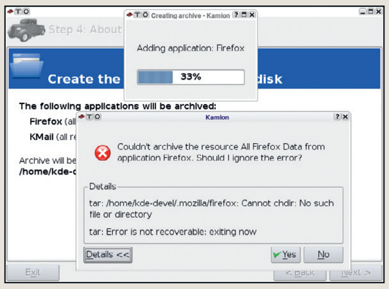
Kamion will not succeed unless every application you use submits its user state specifications to *Kamion* in an XML file. And depending upon how many developers the project recruits, *Kamion* might not even be shipped as a core part of KDE 4; if it doesn't get enough support it could be relegated to the KDE extragear module. If application developers are willing to cooperate, though, *Kamion* could become KDE's Ellis Island. Visit **http://kamion2.sourceforge.net** to get involved. LXF



The .kamion file

The product of a user state backup operation in *Kamion* is a single **.kamion** file, which is nothing more than a tar archive with a simple internal structure for storing resources (your personal and configuration files). If you untar a **.kamion** file you will find **Manifest.xml**, which lists all of the resources in the archive and what applications they came from. The resources themselves are compressed and stored in tarballs with the designation **app name.resource name.tar.bz2**. Also contained in the **.kamion** archive will be a file called **Wishlist.xml**. **Wishlist.xml** lists all the resources that the user tried to back up, and differs from **Manifest.xml** only in the event that an error prevents something from being included in the archive.

Kamion's GUI does not currently make use of the **Wishlist.xml** file, but it could prove invaluable when you're trying to figure out what happened to all of the emails you thought you backed up.



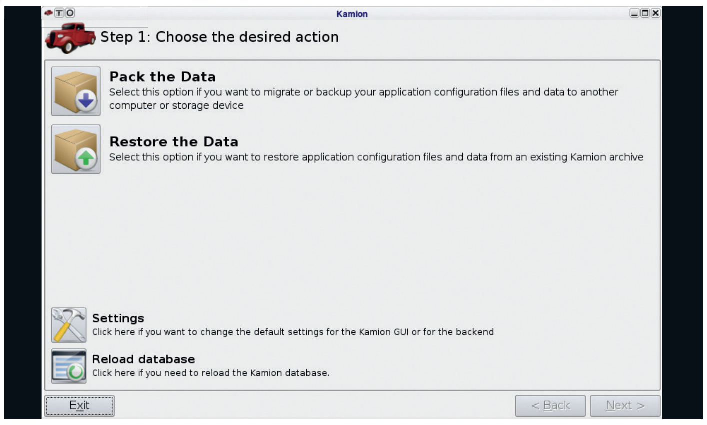
Couldn't archive the resource All Firefox Data from application Firefox. Should I ignore the error?

Details

```
tar: /home/kde-devel/.mozilla/firefox: Cannot chdir: No such file or directory
tar: Error is not recoverable: exiting now
```

› **Your wishlist won't always be the same as your manifest. Here, we've deleted our *Firefox* data directory before we compiled our archive. *Kamion* lets us know what happened and gives us the option to continue without the *Firefox* data.**

Step by step: Migrating a KDE setup with Kamion



Step 1: Choose the desired action

Pack the Data
Select this option if you want to migrate or backup your application configuration files and data to another computer or storage device

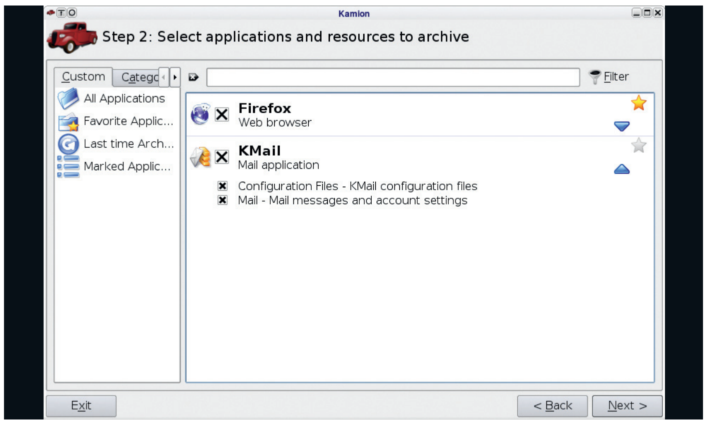
Restore the Data
Select this option if you want to restore application configuration files and data from an existing Kamion archive

Settings
Click here if you want to change the default settings for the Kamion GUI or for the backend

Reload database
Click here if you need to reload the Kamion database.

1 Choose backup

Here's where you choose whether to Pack The Data (make a user state backup) or Restore The Data. You can also access the Kamion Settings dialog from this screen, which lets you manually set the location of *Kamion*'s configuration files or select a disc-burning or mail program with which to export your data.



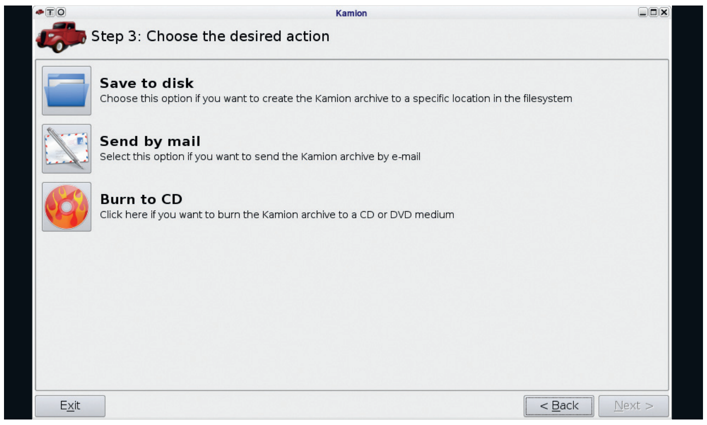
Step 2: Select applications and resources to archive

Firefox
Web browser

KMail
Mail application
Configuration Files - KMail configuration files
Mail - Mail messages and account settings

2 Select applications and resources

Kamion will show you a list of apps that have submitted user state XML files to its database. Use the arrow buttons to the right to see all of the resources you can select for inclusion in your backup archive. The sidebar and the filter bar enable you to enter criteria with which to search for applications.



Step 3: Choose the desired action

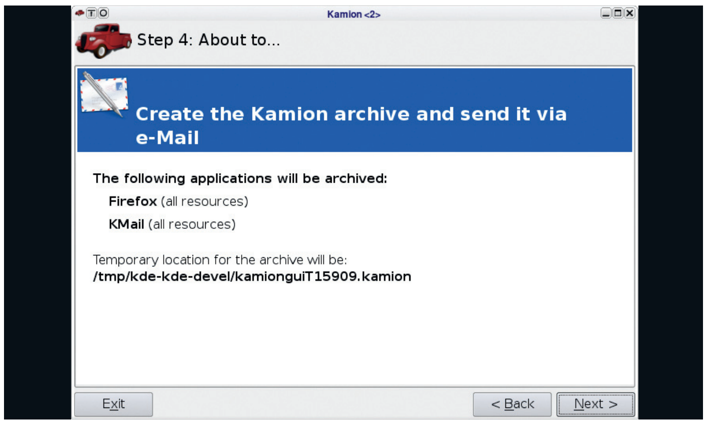
Save to disk
Choose this option if you want to create the Kamion archive to a specific location in the filesystem

Send by mail
Select this option if you want to send the Kamion archive by e-mail

Burn to CD
Click here if you want to burn the Kamion archive to a CD or DVD medium

3 Store your user state backup

Click on Save To Disk to bring up a standard KDE file selector dialog so that you can choose where to save your packaged *Kamion* archive. The Send By Mail option lets you email the archive via whatever application you specified in the Settings dialog. You can also burn to CD using *K3b* or another app.



Step 4: About to...

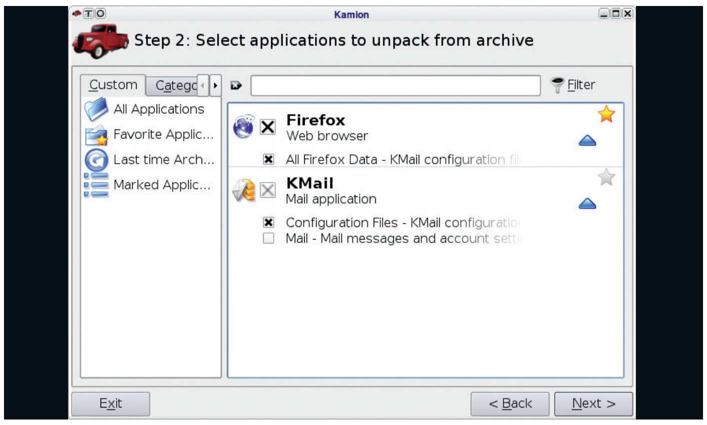
Create the Kamion archive and send it via e-Mail

The following applications will be archived:
Firefox (all resources)
KMail (all resources)

Temporary location for the archive will be:
/tmp/kde-kde-devel/kamionguiT15909.kamion

4 Mission complete

Kamion will show you a list of the resources to be saved, mailed or burned, and give you one last chance to back out or make changes by clicking on Back. Remember: as with all KDE applications, you can use KIO slaves to save or load a *Kamion* archive from a remote storage device.



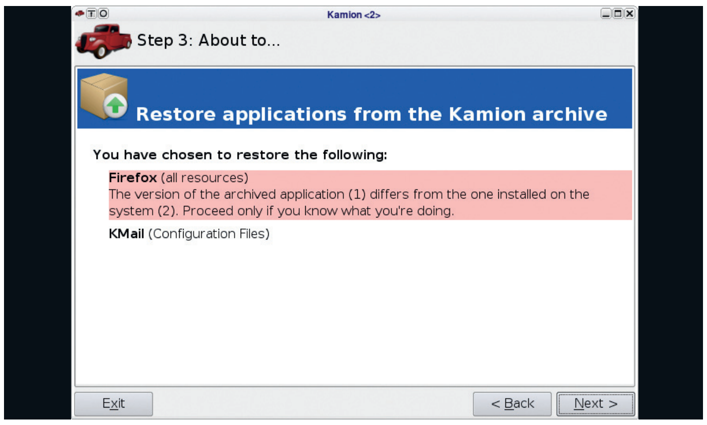
Step 2: Select applications to unpack from archive

Firefox
Web browser
All Firefox Data - KMail configuration fi

KMail
Mail application
Configuration Files - KMail configuratio
Mail - Mail messages and account sett

5 Bring it all back

The dialog for restoring a user state archive looks just like the one for making it. *Kamion* will show you only the resources and applications present in your archive, and will let you leave out any of your resources that you don't yet want to restore. You can still filter for applications by keywords, categories, favourite status, archival history and selection status.



Step 3: About to...

Restore applications from the Kamion archive

You have chosen to restore the following:
Firefox (all resources)
The version of the archived application (1) differs from the one installed on the system (2). Proceed only if you know what you're doing.
KMail (Configuration Files)

6 The inevitable

Of course, you can't expect everything to go smoothly. It appears that we have a newer version of *Firefox* on this computer than on the machine on which we made the backup. Unless we're sure that the two versions have compatible data formats, we should probably click Back and unselect the *Firefox* resources. *Kamion* would also warn us if *Firefox* wasn't installed on this computer.