EPrints 2.3 Documentation

Christopher Gutteridge

March 18, 2005

Contents

Chapter 1

Introduction

1.1   What is GNU EPrints?

GNU EPrints is generic archive software under development by the University of Southampton. It is intended to create a highly configurable web-based archive.

GNU EPrints primary goal is to be set up as an open archive for research papers, and the default configuration reflects this, but it could be easily used for other things such as images, research data, audio archives - anything that can be stored digitally, but you'll have make more changes to the configuration.

The system has been designed to encourage better quality data - that is to say accurate and unambiguous. Well defined metatdata can really help if you want to start exporting your archive data or making it interoperable with other systems.

1.2   Should I be installing EPrints 2, how much effort will it take?

This is version 2.3, it is considered ready for use. There are still some issues and bugs, but hopefully not too many or too major.

Setting up an archive may not (and should not) be a trivial task. The biggest single issue is actually deciding what you need and what policies the archive should have. As a first step we would suggest setting up a demonstration version using the default configuration and having your manager, or the committe who decide on what is needed, have a quick "play" with it and comment what features, data, colours should be changed, added or removed.

Previous experience has been that the initial set-up and configuration is quite a bit of work if you have complex requirements. It should be very easy if the only part of the default configuration you want to change is the colour!

We currently expect to release a few more versions of EPrints 2. We plan to make it as easy as possible to upgrade from earlier versions of EPrints 2.

1.3   What will it run on?

The EPrints software has been developed under GNU/Linux. It is intended to work on any GNU system. It may well work on other UNIX systems too. Other systems people have got EPrints up and running on include Solaris and MacOSX. There are no plans for a version to run under Microsoft Windows.

1.4   This Documentation

This documentation is written in perl POD format and converted into Postscript (which is 2 pages to a sheet for printing), ASCII, PDF, HTML and TexInfo.

The the latest version of this documentation can be obtained from http://software.eprints.org/

Chapter 2

Required Software

2.1 What Additional Software does EPrints Require?

In brief, EPrints requires Apache (with mod_perl), MySQL and Perl with some extra modules. Ideally you also want wget, tar and unzip.

EPrints bundles some perl modules which it uses, to save you installing them.

Where to get the Required Software

It's up to you. We have had best results with installing MySQL from RPM and apache from source.

The best place to get a software tool is the official site, but we've put a mirror of versions known to work at: http://www.eprints.org/files/tools/ - you don't need to install *everything* in the tools directory - just those described below.

2.2 MySQL

Tested on: 3.23.29a-gamma

Install a recent version of MySQL 3. You will need the .h and library files later to install the MySQL perl module. MySQL 4 is due soon, but we are not making plans to support it yet (if you try EPrints with MySQL 4 and it works, please let us know)

If installing from RPM you require: **mysql-server**, **mysql-devel** and **mysql** RPMs.

2.3 Apache/mod_perl

Tested on: apache 1.3.14 with mod_perl 1.25

Apache is the most commonly used webserver in the world, and it's free! EPrints requires Apache to be configured with mod_perl, as this allows Apache modules that are entirely written in perl, hence providing much improved efficiency.

Get Apache from http://httpd.apache.org/dist/httpd/

EPrints requires that the apache module **mod_perl** is enabled.

Apache with mod_perl Installation - Step by Step

Download mod_perl and apache sources

Make mod_perl, I use this command (in the modperl src dir):

```
% perl Makefile.PL APACHE_PREFIX=/usr/local/apache \
APACHE_SRC=../apache-1.3.14/src DO_HTTPD=1 USE_APACI=1 \
EVERYTHING=1
```

Remeber to change `../apache-1.3.14/src` to wherever your apache source is relative to this directory. The back slashes at the end of the line allow a single command to be split over multiple lines.

Make and install apache. From the mod_perl src dir, I use:

```
% make
% make install
```

(mod perl should have already run the apache ./configure script for us.)

2.4   Perl 5.6 and Perl Modules

EPrints is currently begin developed with perl 5.6.1, there are currently no plans for to make EPrints run under perl 6 on the theory of if-it-ain't-broke-don't-fix-it.

Some perl modules are bundled with the EPrints2 package, others must be installed by you.

Installing a Perl Module

This describes the way to simple perl module, some require a bit more effort. We will use the non-existant FOO module as an example.

Some archives can be installed direct from CPAN. That's great when it works. It doesn't always work, but it's the quickest and easiest, so give it a go first. To install a perl module from CPAN run:

```
 % perl -MCPAN -e 'install Foo::Bar'
```

Where `Foo::Bar` is the module you're installing.

I would like to make a list of which modules do/don't install OK from CPAN. If you're reading this before the end of Jan 2003, send me (Christopher Gutteridge) any comments on which ones worked, and on what operating system.

Download the archive.

Either from cpan.org, or from the tools directory on eprints.org described at the top of this chapter. Our example archive is `FOO-5.23.tar.gz`.

Unpack the archive:

```
% gunzip FOO-5.23.tar.gz
% tar xf FOO-5.23.tar
```

Enter the directory this creates:

```
% cd FOO-5.23
```

Run the following commands:

```
% perl Makefile.PL
% make
% make test
% make install
```

Perl Modules Bundled with EPrints

You don't have to install these. They are included as part of the EPrints distribution.

`XML::DOM`, `XML::RegExp`, `Filesys::DiskSpace`, `URI`, `Apache::AuthDBI`, `Unicode::Normalize`, `Proc::Reliable`.

Please note that these modules are not part of the EPrints system and are only included to make things easier. Please note that XML::DOM has has a few lines commented out to prevent it requiring additional modules.

Required Perl Modules (Which you will probably have to install)

This modules are not built into EPrints - you must install them yourself. We recommend installing them in the order they are listed.

Data::ShowTable

MySQL Interface Module requires this.

DBI

Tested with: v1.14

MySQL Interface Module requires this.

Msql-Mysql Module

Tested with: v1.2215

This one can be tricky. It requires access to .h and library files from MySQL. I install MySQL from source first, but some installs of MySQL don't put the lib and include dirs where this module expects. The answer to the first question is that you only need MySQL support.

Under Red Hat's GNU/Linux distribution, the **zlib-devel** RPM should be installed before you install this module.

MIME::Base64

Tested with: v2.11

Unicode::String requires this.

Unicode::String

Used for Unicode support. No known problems. Tested with v2.06.

XML::Parser

Tested with v2.30

Used to parse XML files. Requres the **expat library**. A .tar.gz and an RPM are available in the tools dir on eprints.org.

Apache

The perl Apache.pm module is acutally part of mod_perl - installing mod_perl as part of Apache should also have installed the perl Apache module.

Since version 2.3.7 The modules "Apache::Request" and "Apache::Test" (aka. "libapreq") are no longer required. They were a pain to install and the software has been redesigned to not use them at all.

Required Perl Modules (Which you will probably already have)

Most PERL 5.6 or later systems should already include the following modules, but you may have to install some by hand on certain platforms.

`CGI, Carp, Cwd, Data::Dumper, Digest::MD5, File::Basename, File::Copy,`
`File::Find, File::Path, Getopt::Long, Pod::Usage, Sys::Hostname.`

2.5 Optional GDOME support

Since EPrints 2.2 you may use either XML::DOM or XML::GDOME. XML::GDOME is recommended as it's faster and uses much less RAM, but it does require you to install a whole lot of extra libraries and perl modules. If you are running a pilot or demonstration service then XML::DOM is fine, and you can always

switch over later by installing the required tools and setting the GDOME flag in perl_lib/EPrints/SystemSettings.pm

Addional Libraries Required for GDOME support

```
libxml2
libxml2-devel
```

either get the tarball from: ftp://ftp.gnome.org/pub/GNOME/sources/libxml2/ or the RPMs (but we have had problems with complex RPM dependencies):

```
http://rpmfind.net/linux/rpm2html/search.php?query=libxml2
http://rpmfind.net/linux/rpm2html/search.php?query=libxml2-devel
```

The GDOME Library

Obtain this from

```
http://gdome2.cs.unibo.it/#downloads
```

You may either use the RPMs (gdome2 and gdome2-devel) or the tarball.

Additional Perl Modules Required for GDOME support

```
XML-LibXML-Common
XML-NamespaceSupport
XML-GDOME
```

All of which are in http://www.cpan.org/modules/by-module/XML/

2.6 Other Tools

File uploads

wget, **tar**, **gunzip** and **unzip** are required to allow users to upload files as .tar.gz or .zip or to captures them from a URL.

These all come installed with most modern versions of linux. If you can get them working, you can remove the option by edditing "archive_formats" in SystemSettings.pm

Tested with wget 1.6.

If there are problems you may need to tweak how these are invoked in SystemSettings.pm

Full Text Indexing

The full text indexer requires various tools to index each kind of document. These tools may or may not be already installed in your system. EPrints uses these tools to build a "words" file for each document (which contains the text of the document in UTF-8). If it can't run the tool, the "words" file will be empty and EPrints will not retry creating it unless you manually remove it.

PDF

To index pdfs you need "pdftotext" which is part of the "xpdf" package. RPM's are available.

ASCII

To index ASCII files you don't need anything. That's easy.

Microsoft Word

To index MS Word files you need a package called "wvware". It can be a bit of a bit of a pain to install.

HTML

To index HTML files you need a tool called "lynx". It's a text-based web-browser.

Latex Tools

There is an optional feature which allows you to set eprints to look in certain fields (eg. title and abstract) for stuff which looks like latex equations and display it as an image of that equation instead. These tools are only required if you want to use this feature.

latex, **dvips** and **convert** (convert is part of "imagemagick"). (These all ship with Red Hat's GNU/Linux distribution but you may have to install them yourself on other systems.)

This is a "cosmetic" feature, it only affects the rendering of information, so you can always add it later if you want to save time initially.

Chapter 3

How to Install EPrints (and get started)

3.1 Installation

(If you are upgrading an existing installation of eprints please see the section on upgrading elsewhere in this manual.)

EPrints needs to be installed as the same user as the apache webserver runs as. We suggest you install it as user "eprints" and group "eprints". Under some UNIX platforms, creating a user and group can be done using the "adduser" command. Otherwise refer to your operating system documentation.

Unpack the eprints tar.gz file:

```
% gunzip eprints-2.something.tar.gz
% tar xf eprints-2.something.tar
```

Now run the "configure" script. This is a /bin/sh script which will attempt to locate various parts of your system such as the perl binary. It will also check your system for required components.

```
% cd eprints-2.something
% ./configure
```

By default the system installs as user and group "eprints". You will need to change this if you are not installing as either "root" or "eprints".

The configure script accepts a number of options. All are optional. The most important are:

–help

> List all the options (many are intended for compiled software and are ignored).

–prefix=PREFIX

> Where to install EPrints (or look for a version to upgrade). By default /opt/eprints2/

–with-perl=[PATH]

> Path of perl interpreter (in case configure can't find it, or you have more than one and want to use a specific one).

–with-user=[USER]

> Install eprints to run as USER. By default "eprints".

–with-group=[GROUP]

> Install eprints to run as GROUP. By default "eprints".

–with-virtualhost=[VIRTUALHOST]

> Use VIRTUALHOST rather than * for apache VirtualHost directives.

–disable-diskfree

> Disable disk free space calls. This will be automatically set if configure fails its tests for the df call.

–with-toolpath=[PATH]

> An alternate path to search for the required binaries.

Once you are happy with your configuration you may install eprints by running install.pl:

```
% ./install.pl
```

Now you should edit the configuration file for your copy of apache. This is often /usr/local/apache/conf/http.conf or /etc/httpd/conf/httpd.conf

Add this line: (If you didn't install eprints in /opt/eprints2/ replace that with the location on your system).

```
Include /opt/eprints2/cfg/apache.conf
```

You may also wish to change the user and group apache runs as. The user *must* be the same as the user you installed eprints as. We recommend:

```
User eprints
Group eprints
```

3.2   Creating an Archive

EPrints 2 can run run multiple archives under one install. Multiple archives will require giving additional DNS aliases to the machine running EPrints, EPrints can then create all the parts of the apache configuration file needed to run the virtual hosts.

Creating the Archive

Make sure MySQL is actually running.

Change to your eprints user (probably "eprints").

Change directory to the eprints directory (`/opt/eprints2` by default) and run bin/configure_archive and answer the questions it asks. This will create a MySQL database, create a copy of the default archive configuration into `/opt/eprints2/archives/ARCHIVEID/`. Where archive name is the short text string identifier of this archive. It also creates a file called `/opt/eprints2/archives/ARCHIVEID.xml` which contains the configuration you just entered. If you want you can edit this file directly or re-run configure_archive.

Creating the Database Tables and Website.

You may want to repeat this several times as you will almost cetainly try several configurations before sticking with one.

If you want to totally erase the database, documents and website then run:

```
% bin/erase_archive ARCHIVEID
```

The following commands will generate the initial database tables, the initial website and the apache configuration files to run this archive:

```
% bin/generate_apacheconf
% bin/create_tables ARCHIVEID
% bin/import_subjects ARCHIVEID
% bin/generate_static ARCHIVEID
% bin/create_user ARCHIVEID USERID EMAIL admin PASSWORD
% bin/generate_views ARCHIVEID
```

Where USERID, EMAIL and PASSWORD are your choice for the initial administration account. Once you have made this account you can create new accounts via the web interface.

For more information on what these commands do, see the last section of this documentation or use the –man option.

After running generate_apacheconf or modifying the configuration you must restart your webserver for the changes to take effect. The example below to stop and start Apache might not work on your system - if you have a problem consult the apache documentation.

```
% /etc/rc.d/init.d/httpd stop
% /etc/rc.d/init.d/httpd start
```

Do not just use 'reload' or 'restart' as these do not force mod_perl to reload the perl modules, and EPrints currently only reads the configuration when the PERL modules are loaded.

3.3   Running a Live Archive

Creating a crontab

When you create an archive it will start out as a development system while you learn how to set it up (and your manager keeps changing his mind) but at some point (hopefully) you will declare your archive open for business.

At this point you should schedule certain scripts to run periodically. The best way to do this is to use "cron" which is an integral part of most UNIX systems.

To set up cron, run (as the eprints user):

```
% crontab -e
```

Exactly what to add to the cron table is described in the following sections - "Browse Views" and "Subscriptions".

There should be one set of crontab entries per archive.

Backups

You should also have made sure that the system is being properly backed up. This is gone into in more detail elsewhere in the documentation.

OAI

We would also encourage you to configure the OAI support for your archive and register it. It's quite easy - pretty much fill in the blanks in the ArchiveOAIConfig.pm file in the archive configuration directory.

EPrints 2.1 support OAI versions 1 and 2 at URL paths /perl/oai and /perl/oai2.

Once you register your archive (at http://www.openarchives.org) various search systems will be able to collect the metadata (titles, authors, abstract etc.) and allow more people to find records in your archive.

See http://www.openarchives.org/ for more information on the OAI protocol. For more information setting up the OAI interface archive see the section in this documentation about Configuring an Archive.

3.4   Browse Views

Once every so often you should run the "generate_views" script on each archive in your system to regenerate the browse views section of the site.

This is a set of static pages. By default one per subject, and one per year (only years with papers in that year not EVERY year ever!). Some users prefer to browse the system than search it. This also gives search engines a way to reach, and index, the abstract pages.

See the ArchiveConfig.pm config notes on how to edit the views it generates.

See the How-To section for some suggestions on how to set up views.

But I don't want this feature...

If you don't want to use this feature: don't, it's your archive. Remove the link from the template and front page. Don't run the generate_views script.

Setting it up

This is best done by using the UNIX "cron" command (as user "eprints"). Cron will email "eprints" on that machine with the output, so best use the –quiet option so it only bothers you with errors.

How often you want to run this depends on the size of your archive, and how fast the contents changes. This feature is roughly order "n". Which means if you double the number of items in your archive then you double the time it takes to run (ish).

Once an hour would seem a good starting point. If your archive gets real big, say more than 10000 records, then maybe once a day is more realistic - the one thing that you don't want to happen is for a new generate_views to start before the old one finishes as they will mess up each others output.

Run generate_views on the command line to find out how long it takes.

and add the line

```
23 * * * * /opt/eprints2/bin/generate_views I<archiveid>
```

This runs at 23 minutes past each hour. If you have more than one archive, don't make them all start rebuilding stuff at the same time, stagger it. Otherwise once an hour everything will slow down as it fights to run several intensive scripts at once.

See the crontab man page `man 5 crontab` for more information on using cron.

3.5 Subscriptions

Subscriptions provide a way in which users of your system can receive regular updates, via email, when new items are added which match a search they specified.

To automate sending out these subscriptions you must add some entries in the crontab (as for views). You need one set of these per archive.

For example:

```
# 00:15 every morning
15 0 * * * /opt/eprints2/bin/send_subscriptions dookuprints daily
# 00:30 every sunday morning
30 0 * * 0 /opt/eprints2/bin/send_subscriptions dookuprints weekly
# 00:45 every first of the month
45 0 1 * * /opt/eprints2/bin/send_subscriptions dookuprints monthly
```

Note the spacing out so that all 3 don't start at once and hammer the database. You may wish to change the times, but we recommend early morning as the best time to send them (midnight-6am).

But I don't want users to be able to do this!

Then remove the "subscription" power from each type of user in the archives ArchiveConfig.pm file.

3.6   Default Configuration

EPrints configures a new archive with a set of metadata fields aimed at an archive of research papers.

The initial "types" of eprint (book, poster, conference paper) are configured in metadata-types.xml

The initial subjects are a subset of the library of congress subjects. Feel free to totally replace them with your own subjects, but the more standard your subject tree the more useful your metadata will be to other people.

The authors and editors have the "hasid" option set which allows people to optionally use a unique id for a person in addition to their name (names are NOT unique!) - this can be useful for generating "CV" pages (see the views how-to) and possibly for generating statistics. Without it you will never be sure which "John Smith" wrote that paper. If you don't like this feature remove the "hasid" from the authors and editors - this will require you to recreate the tables, erasing the archive, so decide before you start. If you want to be more clear about what information goes in that field, edit the phrases `eprint_fieldname_authors_id` and `eprint_fieldname_editors_id` in the archive phrase file(s).

In general: Change it! It's not a recommended system setup, just a good starting point.

3.7   New Configurations

If you are setting up more than one archive which are related to each other, a "community", you may wish to establish common subjects and metadata.

Removing and adding types is easy. Removing and adding fields is a bit more work. All "screen" names of values are stored in the archives own "phrase file" which comes with phrases for the default config.

If you create a good default configuration for a different purpose or language(s) (and would like to share it), please contact the eprints admin who may want to put it online as an example or even include it as an alternate default in a later version.

Chapter 4

EPrints Structure and Terms

4.1 Terms

This is a definition of some terms used in the eprints documentation and comments. Many of these are "objects" within the code and the perl module which handle them is listed.

archive

```
EPrints::Archive
```

An archive is a eprints archive with it's own website configuration and data. One install of the eprints software can run serveral seperate archives. Sharing code but with totally different configurations.

session

```
EPrints::Session
```

A session is created every time a cgi script or a bin script is executed, and terminated afterwards.

eprint

```
EPrints::EPrint
```

An eprint is a record in the system which has one or more *documents* and some *metadata*. Usually, more than one *document* is to provide the same information in multiple formats, although this is not compulsary.

document

```
EPrints::Document
```

A document is a single format of an *eprint*, eg. HTML, PDF, PS etc. It can contain more than one file, for example HTML may contain more than one html page + image files. The actual files are stored in the filesystem.

user

```
EPrints::User
```

A user registered with the system. (NOT necesarily the author of the *eprints* they deposit).

subject

```
EPrints::Subject
```

A *subject* has an id and a list of who it's parents are. There is a build in *subject* with the id "ROOT" to act as the top level. A subject can have more than one parent to allow you to create a rich lattice, rather than just a tree, but loops are not allowed.

type or usertype or eprinttype

users, *eprints* and *documents* all have a "type". This controls how they are "cited" and also for *users* and *eprints* it controls what *fields* may be edited, and which are required.

dataobj or item

```
EPrints::DataObj
```

The "super class" of *subjects*, *users*, *eprints* and *documents*. In the very core of the system these are all treated identically and much of the configuration and methods of these classes of "thing" are identical. We use the term *item* to speak about the general case.

dataset

```
EPrints::DataSet
```

A dataset is a collection of *items* of the same type. It can be searched.

Some datasets all have the same "config id". The "config id" is used to get information about the dataset from the archive config - inbox, buffer, archive and deletion all have the same metadata fields and types.

Core datasets are:

```
    DATASET ID | CONFIG ID   | COMMENT
               |             |
    inbox      | eprint      | Eprints which users are still working on.
    buffer     | eprint      | Eprints submitted for editorial approval.
    archive    | eprint      | Eprints live in the archive.
    deletion   | eprint      | Eprints which have been deleted from the archive.
    user       | user        | Users registered.
    subject    | subject     | The subject tree or lattice.
    document   | document    | The documents belonging to ALL
               |             | eprints in the system.
 subscription | subscription | The subscriptions which users have requested.
```

database

```
EPrints::Database
```

The connection to the MySQL back end. *datasets* are stored in the MySQL system, but you do not have to address it directly.

fields or metadata fields

```
EPrints::MetaField
```

A single field in a dataset. Each dataset has a few "system" fields which eprints uses to manage the system and then any number of *archive* specific fields which you may configure.

subscriptions (sometimes called alerts in other archives)

```
EPrints::Subscription
```

A stored search which is performed every day/week/month and any new results are the mailed to the user who owns the subscription.
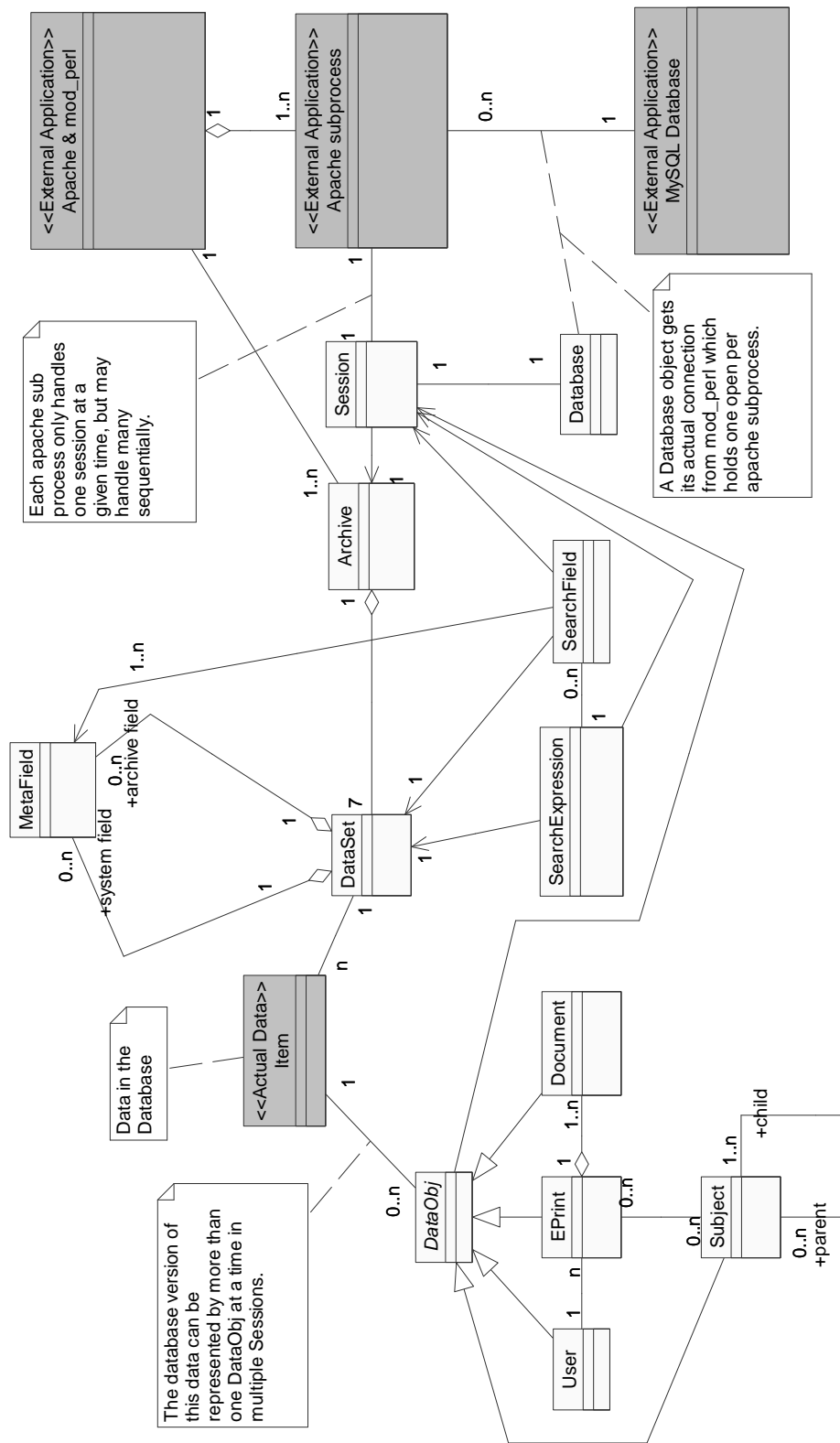
Figure 4.1: EPrints 2 Relational Diagram

This diagram does not show "Subscription". Subscription is a subclass of DataObj (like EPrint, User etc.). A Subscription is associated with one User. A User is associated with 0..n Subscription's.

Chapter 5

Configuring the System

5.1 EPrints General Configuration

This section describes all the configuration files in the EPrints system which do not relate to any specific archive.

EPrints Configuration Directory

The general EPrints configuration directory is usually /opt/eprints2/cfg/ and contains the following files:

apache.conf

> This file is generated by **generate_apacheconf**. See the documentation of **generate_apacheconf** for more information.

auto-apache.conf

> This file is generated and overwritten by **generate_apacheconf**. Do not edit it directly. See the documentation of **generate_apacheconf** for more information.

auto-apache-includes.conf

> This file is generated and overwritten by **generate_apacheconf**. Do not edit it directly. See the documentation of **generate_apacheconf** for more information.

languages.xml

> This XML file contains an (exhaustive) list of all ISO language ID's and their names.

system-phrases-*languageid*.xml

> One of these files per language needed for any archive in this system. These files contain the phrases needed to render the website and email in each

language, not counting names of things like metadata fields which vary between archives. It should not be edited by hand, but may be overridden. See the instructions on phrase files in the archive config documentation.

SystemSettings.pm

Described below.

SystemSettings.pm

This is a perl module which is created and edited by the eprints installer script when installing or upgrading EPrints. It's found in `perl-lib/EPrints/` SystemSettings contains system specific things:

base_path

The root directory of your eprints install. Normally /opt/eprints2/

executables

A hash of the path of various external commands such as sendmail and wget.

invocation

A hash of how eprints is to invoke various external commands. The variables with uppercase names - $(FOO) - are replaced with parameters from eprints, the lowercase names - $(sendmail) - are replaced with the strings in executables.

archive_formats

An array of id's of archive formats offered in the upload document page. For each their must be an entry in the archive_extension and invocation, $(DIR) is the where eprints wants the contents of the archive and $(ARC) is the archive file.

version_id

The id of the current eprints version.

version

The human readable version number.

user

The UNIX user eprints will run as. Usually "eprints".

group

The UNIX user eprints will run as. Usually "eprints".

virtualhost (Since v2.1)

If this is set, it is used for the VirtualHostName in the Apache configuration files. (By default EPrints uses "*").

disable_df (Since v2.1)

> If this is set to 1 then this disables the parts of EPrints which use the df call (disk free). If the "configure" script tested the "df" command and found that it failed the this function will initially be set to 1, otherwise 0.

enable_gdome (Since v2.2)

> If this is set to 1 then it enables the use of the XML::GDOME module, rather than XML::DOM. XML::GDOME is faster and less memory intensive but depends on a number of other libraries and modules which are not worth installing for a trial system.

Chapter 6

The Archive Configuration Files

6.1 EPrints Archive Configuration

This section describes all the configuration files in an single archive in the EPrints system.

Primary archive configuration file

Once you have created an EPrints archive the information you entered is placed in an XML file in /usr/local/eprint2/archives/ with the name *archiveid*.xml - this file is documented later in this section.

Archive configuration directory

The bulk of the archive configuration is copied from /opt/eprints2/defaultcfg/ into the archives own configuration directory (usually /opt/eprints2/archives/*archiveid*/cfg/ This directory will usually contain the following files and directories:

apache.conf

>This file is generated by **generate_apacheconf**. See the documentation of **generate_apacheconf** for more information.

apachevhost.conf (added v2.2)

>This file is generated by **generate_apacheconf**. See the documentation of **generate_apacheconf** for more information.

ArchiveConfig.pm

>The general configuration items which don't fit anywhere else are in this perl module. It is described fully later in this section of documentation.

This module "requires" the other 5 perl modules. They are in seperate files to make them easier to get to grips with.

ArchiveMetadataFieldsConfig.pm

This module configures the metadata fields and the default values.

ArchiveOAIConfig.pm

This module configures how the archive exports itself via the Open Archives protocol.

ArchiveRenderConfig.pm

This module contains subroutines which handle rendering the data into XHTML (mostly) for display as webpages.

ArchiveTextIndexingConfig.pm

This module handles turning UTF8 text strings into lists of index words for free text searches.

ArchiveValidateConfig.pm

This module contains subroutines which check the metadata for problems.

auto-apache.conf

This file is generated and overwritten by **generate_apacheconf**. Do not edit it directly. See the documentation of **generate_apacheconf** for more information.

citations-*languageid*.xml

One of these files for each *languageid* supported by this archive. These XML files describe how to turn metadata for an item into a citation (with markup). They are described fully later in this section of documentation.

entities-*languageid*.dtd

One of these files for each *languageid* supported by this archive. These DTD files are generated automaticly just before eprints loads the archives configuration and should not be edited directly.

metadata-types.xml

This XML file describes the various types of eprints, users etc. and which metadata fields are required or relevant to each. It is described fully later in this section of documentation.

phrases-*languageid*.xml

One of these files for each *languageid* supported by this archive. These XML files contain all the phrases which are specific to this archives such as the titles of metadata fields. They are described fully later in this section of documentation.

ruler.xml

> This XML file just contains the horizontal divider used in webpages created by the system. It is described fully later in this section of documentation.

static/

> This directory contains the data needed to create the static webpages such as the homepage, and about page. It is described fully later in this section of documentation.

subjects

> This file contains the initial subjects for the system. It is described fully in the documentation for **import_subjects**.

template-*languageid*.xml

> One of these files for each *languageid* supported by this archive. These XML/XHTML files describe the outline for webpages for this system. They are described fully later in this section of documentation.

6.2 XML Config Files in EPrints

This section contains some general information about the XML archive config files: template, phrases, ruler and citations. metadata-types.xml uses XML but these comments do not apply.

XHTML

This files use HTML elements (and other elements too). XHTML is a fairly new version of HTML which is back compatable with HTML 4 but written using XML not SGML. This means that it is much stricter but less ambiguous and easier to parse and modify. Assuming you know HTML, the main differences are as follows:

Tags must be closed

> All elements must be closed, even ones such as <li>. Tags which do not have a close tag in HTML, like `<br>` or `<img src="foo">` still must be closed eg. `<img src="foo"></img>` - this can be abbreviated to the neater looking: `<img src="foo" />`

All tags and attributes must be lower case

> Self explanitary.

Strict definition of what tags may appear within which others.

> Not actually checked by EPrints. It will let any rubbish past as long as it's valid XML. But that's no reason to be naughty.

All attributes must be wrapped in quotes

In HTML the values of attributes do not have to be wrapped in quotes, but in XML (and therefore XHTML) they do.

All attributes must have a value

In HTML some attribues did not require a value, for example in `<hr noshade>` elements. In XHTML it is represented as `<hr noshade="noshade" />`

So in summary, the HTML:

```
<img SRC=someurl>
<hr NOSHADE WIDTH=2>
<P>Foo bar</P>
```

should become in XHTML:

```
<img src="someurl" />
<hr noshade="noshade" width="2" />
<p>Foo bar</p>
```

And that's more or less it. See http://www.w3c.org/ for a complete description.

Language specific files.

phrases, template and citations have one instance per supported language. This allows the system to generate pages and emails in more than one language. Supporting a new language will require translating the all the english in the english config files currently shipped. If you do intend it do this (lots of work!) please get in touch with the eprints admin so that we can avoid duplicated effort.

Extra Entities

The XML files all use a DTD which defines a few extra entities. Entities are items in XML (or HTML) which start with "&" and end with ";" like &. These additional entities come from the entities DTD file created by generate_entities. One DTD is created per language, although currently the only variation is the archive name.

&archivename;

The name of the archive in the current language.

&adminemail;

The administrators email address.

&base_url;

> The base URL of the system (without a trailing slash)

&perl_url;

> The base URL of the CGI directory (without a trailing slash)

&frontpage;

> The URL of the system homepage.

&userhome;

> The URL of the user homepage.

&version;

> The current EPrints version.

&ruler;

> The XHTML of the standard divider.

Any XHTML character entity (since EPrints v2.1)

> You may now use any XHTML character entity, eg. `  é €`.

User configured entities

> You can generate your own entities by modifying the function which generates them in ArchiveConfig.pm

> None of these entities are not available in the citations file or the ruler file.

Name Spaces and XHTML

These files contain a mixture of custom tags and XHTML. To keep these distinct the XML files contain a name space definition in the first element. The pratical upshot is that all EPrints own tags have the prefix "ep:". The namespace information is actually ignored by the current version of the eprints system.

example of mixed tags (and entities):

```
<ep:phrase ref="lib/session:contact"><p>Feel free to contact
<a href="mailto:&adminemail;">&archivename; administration</a>
with details.</p></ep:phrase>

eprints elements: phrase
xhtml elements: p, a
eprints entities: archiveemail, archivename
```

6.3   The Primary Archive Configuration File

This XML file appears in the `archives/` directory, usually `/opt/eprints2/archives/`, it describes the most very basic details about the archive. It is generated (and modified) by configure_archive and will not normally need to be edited.

EPrints looks in this directory for XML files and attempts to load them all when starting the webserver.

This file should be chmod'd so that it can not be read by random users as it contains the database password.

The top level element is "archive" which has the attribute "id" which is the id of the archive. It should be the same as the filename. If this file is foo.xml then the id should be foo.

    <archive>   contains a list of XML tags enclosing some text. eg.

    `<host>stoatprints.org</host>`

The following tags are expected in no special order:

<host>

    The hostname of this archive.

<alias redirect="*yes-or-no*">

    This is optional and may be repeated. It has the attribute "redirect" which may be set to yes or no. This controls what virtual hosts are supported and if they should redirect to the main <host>.

<language>

    The ISO id of a language supported by this archive. Repeatable. One of these should also be the defaultlanguage. See below.

<port>

    The port number that the server is running on. Usually 80.

<urlpath>

    The directory from the root of the server name. Usually `/`

<archiveroot>

    The filesystem path of the rest of the archive configuration.

<configmodule>

    The path to the perl module which does the main configuration (Archive-Config.pm)

<dbname>

    The name of the MySQL database. Usually the same as the archive ID.

<dbhost>

> The host on which MySQL is running. Usually localhost.

<dbport>

> An optional MySQL port, if it's not the standard one. Should be empty if we are to use the default.

<dbsock>

> An optional MySQL socket. Should be empty if we are to use the default.

<dbuser>

> The username to use when connecting to MySQL, usually "eprints".

<dbpass>

> The password to use to connect to MySQL.

<defaultlanguage>

> One of the supported language. This is the default for this archive.

<adminemail>

> The email address of the archive administrator. I strongly suggest that this is an alias rather than a personal email address. If all your webpages contain "bob@footle.edu" and bill takes over from bob you would have to regenerate every page with "bill@footle.edu". Much better to set up an email alias or forward from "archive-support@footle.edu" and point it at bob (for now). Heed these words spoken from grim experience!

<archivename language="*langcode*">

> The name of the archive. This has an attribute "language" the value of which is an iso language id. There should be one of these archivename elements per supported language. eg.

```
<archivename language="en">White Lemur</archivename>
<archivename language="fr">La Archive d'Lemur Blanc</archivename>
```

> (apologies to the french, human languages aren't my strong suit)

<securehost> (since v2.2)

> Used for experiemental https support.

<securepath> (since v2.2)

> Used for experiemental https support.

6.4   ArchiveConfig.pm

This module imports the other 5 perl modules. It allows lots of little tweaks to the system, which are all commented in the file.

It includes options to hide various features you may not want and to customise the browse, search and subscription functions.

Also you can customise what each type of user can and can't do, and how they authenticate their passwords.

This configuaration file contains perl methods which are called when a session starts and ends, to log things, to generate the entities for the entities file and security on non public files.

Browse Views

The browse views are generated by the script "generate_views" and what that script does is configured by the "browse_views" item in the config.

It is a reference to a perl array [], each item of which is a hash {}.

The hash has 3 required properties and a number of optional ones.

id (required)

The ID of this view - the view will be placed in a subdirectory of /views/ of this name. The ID is also used to identify the full name of this view in the phrase file.   `id=>"foo"`   would find it's title in the phrase "viewname_eprint_foo"

fields (required)

The list of the names of the fields to browse, seperated by a slash "/". This should normally be a single field unless you want to merge the values of two fields. The id part of a field may be specified by appending ".id" to the fieldname.

order (required)

A list of fields to sort by in order of priority, sepearted by slashes "/". A minus sign prefixing the fieldname "-" indicates reverse sorting on that field.

allow_null

Should we make a page for the "unset" condition? A page for items which do not have a year set may be useful. But for other fields this may be meaningless. Set it to 1 for true.

include

Generate a file for every value, ending in ".include" which contains the XHTML of the citations of records and the number of records, but without wrapping the site standard template around it.

nohtml

> Normally the system generates a page like that described for "include" with a .html suffix and the site template. If nohtml is set to 1 then it won't.

citation

> Normally the citation used is that for the "type" of eprint. If this is set then that citation (from the citations file) will be used for all items. This allows for some clever stuff if you want to make page which can get sucked into another website.

> Normally the system puts a paragraph tag around each citation, but if you use a custom citation this will not happen.

nocount

> Do not include the count of how many items at the top of the page.

nolink

> The system generates an `index.html` in `/view/` with a list of all the browse views available. Setting nolink to 1 will hide this item.

noindex

> Do not generate an `index.html` file in `/view/`*foo*`/` listing all the values of the view and linking to their respective pages.

notimestamp (since v2.2)

> Do not add the timestamp at the bottom of the view page.

hideempty (since v2.2)

> Only applicable to subjects. This option will supress subjects which do not have any records in. This is useful on "young" archives which look very empty if you have a large subject tree and only a few records, and those clustered in 3 or 4 subjects.

The most common view is to browse by subject:

```
{ id=>"subject", allow_null=>0, fields=>"subjects",
  order=>"title/authors", hideempty=>1 }
```

A more complex view generates a view on author & editor ID's which are not advertised but may be captured by some other software to build staff CV pages.

```
{ id=>"person", allow_null=>0, fields=>"authors.id/editors.id",
  nohtml=>1, nolink=>1, noindex=>1, include=>1,
  order=>"-year/title" }
```

For my example person id "wh" this will generate a webpage called /view/person/wh.include (and one for each other value of authors or editors ID's) which can be captured by an external automated system.

User Privs

The user permission configuration allows you to set what types of user can and can't do. The user home page will only show a user options which they can do.

New types of user, and which data about themselves they can edit is set in metadata-fields.xml.

Permissions are set by "type" of user. By default there are 3 kinds of user: "user", "editor" and "admin".

Admin can, by default, do everything.

subscription (since EPrints v2.1)

> If included then this kind of user can create subscriptions.

set-password

> Reset their password via the web registration system.

deposit

> Submit items into the archive.

view-status

> View the archive status page.

editor

> User can edit then approve submitted items into the main archive, or delete them, or return them to sender. Also can remove items from the archive back into the edit buffer for corrections, and move records into the deleted table (delete them).

staff-view

> User can perform a "staff search" of user or eprint records and view ALL the metadata.

edit-subject

> User can edit the subject tree via the online interface.

edit-user

> User can edit other users records.

change-email

> User can change their email address via the web interface. This is safer than allowing them to edit it directly as it ensures they cannot set it to an address which they recieve (it mails them a confirmation pin number)

change-user

> This allows the sinister feature which lets you log in as someone else. It still requires a password. This is useful if you want to perform admin tasks as a super user, then log-in as a normal user to deposit items.

no_edit_own_record (since v2.2)

> This supresses the "edit my user record" option. This may be useful if you disable web-registration and import the user records from some other database.

6.5 ArchiveMetadataFieldsConfig.pm

Fields Configuration

Metadata is data about data. The information which we store to describe each record (eprint) in the system. Users also have metadata.

This module is the configuration for the metadata. This is probably the most important part of the system.

See the chapter on metadata for all the configuration options.

Defaults

This section of the file contains subroutines which are called to set default values for Users, Documents and EPrints.

Automatics

These functions let you set automatic fields. This allows you to make fields which are updated automatically each time the item (User/EPrints/Document) is commited to the database.

This allows you to create "compound" fields. Such fields are created by processing the values of other fields rather than being edited directly.

For example, if you wanted to make an automatic int field which contains the *number* of authors, you could add the following to set_eprint_automatic_fields:

```
# no authors at all will be undef, not [] so check first
if( $eprint->is_set( "authors" ) )
{
        my $auths = $eprint->get_value( "authors" );
        $eprint->set_value( "authcount" , scalar @{$auths} );
}
else
{
        $eprint->set_value( "authcount" , 0 );
}
```

6.6 ArchiveOAIConfig.pm

This module configures how the archive exports its data via the OAI protocol.

For more inforamtion on the how and why of OAI see http://www.openarchives.org/

OAI allows a harvestor to request the metadata from your archive and other archives to provide a federated search. The next time the harvestor harvests your archive it only has to ask for items which have changed or been added since last time it asked.

The current version of EPrints supports OAI v2.0. OAI version one is no longer supported.

The base URL for your OAI v2.0 interface will be http://archivepath/perl/oai2

If you want to use the OAI system then you need to fill in the blanks, such as policy and the OAI-id of the archive.

You may create OAI sets in a similar manner to "browse views" in Archive-Config.pm.

If you want to change the way that an EPrint is mapped into Dublin Core then edit the make_metadata_oai_dc - which returns a DOM XML object.

To add a new metadata type you need to add a new mapping function and add entries to the namespaces, schemas and functions items near the top of the file.

6.7   ArchiveRenderConfig.pm

This module contains fuctions which turn data into XHTML for displaying on the web.

If you want to change the way a user info page, or an eprint "abstract" page is rendered then here's the place to do it.

There are also "full" versions of these functions which display all the internal variables and things. These are the views which the editors and admin see.

The XHTML is generated using DOM (Document Object Model), but eprints provides some functions for easily generating XHTML DOM. The only method of DOM you should need to use is appendChild - which adds an element to this element.

EPrints API functions which return XHTML objects.

Note, all text strings should be in UTF-8.

Example:

```
my $page = $session->make_doc_fragment();
my $h1 = $session->make_element( "h1" );
$h1->appendChild( $session->make_text( "Title" ) );
$page->appendChild( $h1 );
$page->appendChild(
   $session->make_element(
      "img",
      src=>"/images/cheese.gif",
      width=>128,
      height=>53 ) );
```

$page now contains:

```
<h1>Title</h1><img src="/images/cheese.gif" width="128" height="53" />
```

Many of the EPrints modules are now properly(!) documented. For an example try running:

```
% perldoc /opt/eprints2/perl_lib/EPrints/Archive.pm
```

The functions most useful to extacting and rendering information are documented here:

`$session->make_text( $text )`

> Returns a DOM object representing that text.

`$session->make_doc_fragment()`

> Returns a document fragment. This renders to nothing but is a container to which you can add stuff.

`$session->make_element( $name, %opts )`

> Makes a simple XHTML element. %opts is an optional series of attributes.
>
> To make `<h1 class="foo">...</h1>` you would call:
>
> ```
> $session->make_element( "h1", class=>"foo" );
> ```

`$session->render_ruler();`

> Returns the default ruler for the archive (from ruler.xml).

`$session->render_link( $uri, $target )`

> Returns the XHTML element (with URI properly escaped):
>
> ```
> <a href="uri"></a>
> ```
>
> Which you can appendChild stuff into. If $target is specified then a target attribute is included - to make it pop up a new window.

`$item->render_value( $fieldname, $showall )`

> $item is either an EPrint, a User or a Document.
>
> $fieldname is the name of the field you want to render. If $showall is 1 then ALL values are rendered in a multilang field.

`$item->render_citation( $style )`

> Renders the citation of the item using the citation for the item's type from the citation file.
>
> If $style is set then it uses the citation with that id instead.

`$item->render_citation_link( $style )`

> This renders a citation as above, but links it to the url of the item.

`$item->render_description()`

> This renders a simple description of the item using the default citation for this dataset eg. for eprint it uses citation type "eprint".

`$session->html_phrase( $phraseid, %opts )`

> Returns the item from the phrase file. If you don't care about supporting multiple languages then just use make_text instead, it's easier.
>
> It looks first in the archive field from the current language.
>
> Then in the archive phrase file for english.
>
> Then is the system phrase file for the current language.
>
> Then is the system phrase file for the english.
>
> The %opts are a series of DOM elements to place in the "pin" items in the phrase file.

Some other useful functions you may need

`$item->get_value( $fieldname, $no_id )`

> Returns the value of field $fieldname from the item. An optional second parameter may be set to 1 to return the value without the "id" part, to keep things simple.

`$item->is_set( $fieldname )`

> Returns true if the field is set on this object, false otherwise.

`$eprint->get_all_documents()`

> Return an array of the document objects belonging to this eprint.

6.8   ArchiveTextIndexingConfig.pm

This module you probably won't need to change unless you want to modify how eprints does searches for words in strings.

When a record is added to the system eprints uses this module to turn a string into a list of values which are indexed. By default these are words with 3 letters or more except some predefined stop words. It also turns latin characters with acutes into the their plain ascii (no acute/grave) versions.

It then does the same with the search string and looks for these keys.

Example:

`The rain in spain falls mainly on the plains.`

Is turned (by default) into the keys:

```
rain spain fall mainly plain
```

Thus searching for "rain" or "plain" or "plains" or "MaiNlY" will all match this string.

You may wish to add your own "stop words". eg. If you are running an archive about badgers, a search for the word "badger" will return almost all the records.

At a more complex level you may wish to add handling for non-european character sets (I have no idea how well the default setting will work on these), or do "stemming" - removing "ed", "ing", "ies", "s" etc. from the end of words so that "land" will match "land", "landed", "landing" and "lands". (It current removes 's').

Another suggestion is using *soundex* or similar techniques to match words which sound similar.

Changing the indexing on a live system will require you to regenerate the indexes using the reindex script. (If you don't then some of the search results will be wrong).

6.9   ArchiveValidateConfig.pm

This module handles validating data entered by users. Each subroutine is described in more detail in the module itself.

Each subroutine returns a list of DOM elements, each of which describing a single problem. Any problems will prevent the user from continuing with editing until they correct the problems.

As with the rendering functions, if you don't care about making this work in more than one language then you can just make the DOM items by calling $session->make_text("problem explanation")

The eprint & document validation routines have a flag **$for_archive** which, if true, indicates that the item is being checked before going into the actual archive. You can use this to force an editor to enter fields which the user may leave blank.

Validation Functions

validate_field

> Called for all fields. Use it to check individual field values. By default checks that url's look OK.

validate_eprint_meta

> Check the metadata of an eprint. Use this to test dependencies between fields. eg. if you have a requirement that field "A" OR field "B" must be set.

validate_eprint

> Validate the whole eprint. The last part of the validation of an eprint.

validate_document_meta

> Validate the metadata of the document (as with eprint_meta)

validate_document

> Validate the whole document, files and metadata.

validate_user

> Validate a user record.

6.10   citations-*languageid*.xml

The ciations file describes how to render an item (eprint/user/whatever) into a short piece of XHTML. Each citation has a "type". There are 3 kinds of citation:

default citation

> This is a very short description of the item. Usually "the title or failing that, the id". The type id is just the name of the dataset. eg. "eprint"

type citation

> These are richer descriptions which vary between type of eprint, user or document. The type id is dataset_type eg. eprint_preprint.

other citation

> Used by custom browse views. Any name you like.

The citation file contains a list of citation elements:

```
<ep:citation type="...">
```

Each one may contain text and tags. The text may also include the names of fields in the record being rendered. These names should be between @ symbols. eg. @authors@ or @title@. These will be replaced with a rendered version of the value in that field. (if you need an actual @ symbol for some reason two @@ with nothing inside will be rendered as a single @).

Note. The @title@ style was introduced in EPrints 2.2. Before that this file used XML entities such as &title; but this caused problems and didn't solve any. Use of entities is still supported, but deprecated.

In addition you may use XHTML elements and the following elements in the eprints namespace. These elements are always removed but they control if their contents is kept or not. Conditional elements may be placed inside each other since v2.2.

```
<ep:linkhere>
```

> This element is replaced with an XHTML anchor linking to the item. If this citation is being rendered without a link then it is just removed (but not the contents).

<ep:iflink>

The contents of this element are only preserved if we are rendering this citation as a link. Maybe an icon which you don't want if it's not a link.

<ep:ifnotlink>

The opposite of iflink.

<ep:ifset ref="fieldname">

The contents of this element are only preserved if the field "fieldname" has a value.

<ep:ifnotset ref="fieldname">

The contents of this element are only preserved if the field "fieldname" does not have a value.

<ep:ifmatch name="fieldname(s)" value="searchparam">

This is the swiss army knife of the world of conditional rendering. It is also a bit complicated, and few people will need to use it. This actually works like a single search element. The attributes are:

name

This is the name of one or more fields, specified as in the search fields configuration. eg. "title/abstract"

value

This is a value to search for. Treated like the value entered in a search field.

merge (optional)

Can be ANY or ALL. Works like the match all? in a search form.

match (optional)

Can be IN, EQ, or EX. In, Equal or Exact. Exact on subjects means that subject, but not any below it in the heirarchy.

For example:

```
@year@<ep:ifmatch name="year" value="-1949">
(approx)</ep:ifmatch>
```

This will render (approx) after years before 1950. Neat eh?

<ep:ifnotmatch name="fieldname(s)" value="searchparam">

Like ifmatch but only includes the values inside if the search does not match.

6.11   metadata-types.xml

This file allows you to configure the types of eprint, user, document and document security level.

When you add a new type you should add it's name to the archive phrases file(s). The phraseid is "dataset_typename_typename" eg. "document_typename_pdf", and you should add a new citation to the citations file. Any fields which are not required but appear in the citation should probably be inside a <ep:ifset> so that you don't get see "UNSPECIFIED" if they are not, er, specified.

The main element is "metadatatypes". This contains a list of "dataset" elements each of which has a name attribute.

The "type" elements in user and eprint "dataset"s should contain a list of "field" elements. This describes the fields which may be edited for this type and the order that they appear on the form.

You may include system fields in this list, but be careful if you do.

Multi-page metadata (2.3.0+)

You may optionally add <page name="pagename" /> elements to the field list. These break the submission process into smaller stages. The pagename is used to identify the sub-page, for purposes of validation etc. Pages only have an effect on eprint types, not user, document etc.

See the section on paged metadata.XX

Attributes for "field" element

name (May not be ommited)

The name of the metadata field.

required

If set to "yes" then this field may not be left blank. Some system fields are always required no matter how this is set.

staffonly

This field only appears on the "editor" edit eprint form, not the user one. Or, in the case of the user dataset, the staff edit-user page.

The "security" dataset

This is a handy place to define the security levels. The type with no name is special. It is the "public" security type. All other types will require a valid username and password. If that username is acceptable for a given document is decided by the can_user_view_document subroutine in ArchiveConfig.pm

The "document" dataset

By default eprints requires at least one of ps, pdf, ascii or html to be uploaded before an eprint is valid. You may change this list in ArchiveConfig.pm - any more complicated conditions will have to be checked in the eprint validation subroutine.

6.12 phrases-*languageid*.xml

This file contains a list of XML "phrasees". Everything eprints "says" to users is stored in this file and its system-level counterpart. If you want the site to run in more than one language, you need one phrase file per language.

The phrase file is XML and contains a toplevel "phrases" element. This contains the list of phrases.

Each phrase has a "ref" attribute to identify it and contains text and optionally some XHTML tags. It may also contain eprints entities such as &archive-name; and also some phrases should contain "pin" elements, described below.

The phrases in the archive phrase file are specific to that archive, the system phrase file contains non-archive specific phrases. The id's of most of the phrases in the archive phrases are generated from the id's of the fields, datasets, types etc.

The archive phrase file contains: names of dataset types, names of metadata fields, help on entering each Ametadata field, the names of options in "set" fields, the description of different search ordering options, names of browse views, phrases used in the render and validation routines, mail which eprints sends out and phrases which override those in the system file.

pins

Some phrases need some "pin" elements to show eprints where to insert values. Usually pins don't contain any elements but occasionally they do when they represent what to place a link around.

Overriding System Phrases

If you don't like some of the phrases in the main system phrases file you can override them by creating a phrase with the same "ref" in the archive file.

Don't edit the system file, if you upgrade eprints to a newer version it will get over-written.

Emails

EPrints sends out emails when a user registers/changes their password, when a user changes their email, when a deposited item is rejected/deleted by an editor and when the system is low on resources. These mails can be customised in the phrase file.

Make sure you wrap your text in paragraph <p> tags. EPrints will automatically word wrap these in the email. <hr /> elements in a mail are turned into a line of dashes.

When eprints sends a mail it will send it as plain ASCII text, unless it contains latin-1 elements, in which case it will be latin-1 encoded. If it contains unicode characters not in the latin-1 charset then it will be utf-8 encoded.

6.13 ruler.xml

This file configures the horizontal divider which eprints uses, which is inserted in place of &ruler;

If you have no great dislike of <lt>hr<gt> horizontal rulers then you can leave it alone.

You can't use entities like &frontpage; in ruler.

6.14 The static/ directory

This directory contains the static pages for the site - the frontpage, the help pages, images, the stylesheet etc.

`static/` contains one directory per language, eg. `en`. Plus a `general` directory which contains files which don't need translating like images and the stylesheet.

When you run the `generate_static` command it copies the files for each language, and the gerneral dir, into the static site for that language.

See the `generate_static` documentation for more details.

6.15 subjects

This file is not used by the core eprints system. It is used by import_subjects to set up the initial subjects. For more information see the instructions for import_subjects.

6.16 template-*languageid*.xml

This file is the shell of every page in the system. It is more or less a normal XHTML page but you can use the eprints &foo; entities in it and it should contain "pin" elements like a phrase. The pins it should contain are:

<ep:pin ref="title" />

This is where to put the title of the page. It can be used more than once - in the title in the page header and somewhere in the body. If placing it in the title in the head of the page you must use the additional attribute textonly="yes" which only works here. It removes images from the title (which can happen if using the "Latex" mode).

<ep:pin ref="head" />

> This goes somewhere in the head of the page. It shows eprints where to insert the "meta" and "link" elements.

<ep:pin ref="pagetop" />

> This goes at the top of the body. It is sometimes used as a "target".

<ep:pin ref="page" />

> Where to place the bulk of the content of the page.

Chapter 7

Configuring the Archive Metadata

7.1 EPrints Metadata

Introduction

Metadata is data about data. In this case information about the documents we are storing.

This section describes how to configure the metadata of an archive, and gives information on the various properties of metadata in GNU EPrints.

From the point of view of the database, all eprint records have the same metadata fields. Although each eprint type will only expose a subset of those fields to the user interface.

Modifying the Metadata fields in an Archive

To add metadata fields you must edit `ArchiveMetadataFieldsConfig.pm` and then erase and recreate the database tables. This will destroy all your data. If you want to add or modify fields to an archive without destroying your data, you will need to go into the SQL.

All types of record in GNU EPrints have metadata. All have a core set of fields which may not be modified as these are required by the software. The "eprints" and "users" table also have a list of metadata fields which are used for storing the information about users and eprints.

In addition to the metadata types there are functions in `ArchiveMetadataFieldsConfig.pm` which can be modified to set default values of fields and to set certain fields based on values from other fields (automatics).

Automatics are useful as they allow you store the simple answer to a more complex question. Eg. "Does this record have a document which has a security

level public?" which a user may want to make a search criteria. If this is stored as a simple field in the record then it is easy to search on it.

Default Configuration

The default configuration of metadata fields, as of v2.3.0, has been designed as part of a collaboration between EPrints and library staff as a good starting setup for an institutional archive. It will still probably want some change or other depending on your requirements.

There are also so default values for some fields and some automatics.

The default value for the "hideemail" of user records is "TRUE" because we don't want to show peoples emails unless we were given explicit permission.

If the "frequency" field of a user record is not set, it automaticall gets set to "never". This is because undefined means pretty much the same as "never" but it makes it clearer not to have it listed as "unspecified". "frequency" only applies to editors. It's how often they get sent updates on what needs approving.

The default security level for a document is "" (public). The default language for a document is the language of the current session (whatever requested by the user doing the deposit). This is not important unless you have an archive which cares what languages individual documents are in.

The default type for an eprint is "article". This means that eprints can never have an undefined type.

There are several automatics for eprint records. If the eprint is a "monograph" or "thesis" and it does not have the "institution" field set, then nothing happens... but there is some code you can uncomment to make it set "institution" to the name of your University (or whatever).

If the eprint is of type "patent" then it is automatically set to "published" as we should never be storing unpublished patents.

If the eprint is of type "thesis" then it is automatically set to "unpublished" as thesis' do not get published. If this is incorrect for your archive it is easily disabled.

The "effective date" field is set to the same value as the "date of issue". Unless it's undefined in which case it's set to the value of the "date of submission". If they are *both* undefined then it is set to the datestamp of the metadata record. This means that "effective date" contains the "best" date for this record, to use when searching and ordering. It's not actually rendered anywhere.

"full_text_status" is set to a value indicating the status of the full text. One of "none" - no full text, "public" - full text is available, "restricted" - full text is available but is restricted in some way (security setting is not public).

There is also a rather ugly hack to set the value of "fileinfo" to info about the documents so that it can be rendered as icons in citations.
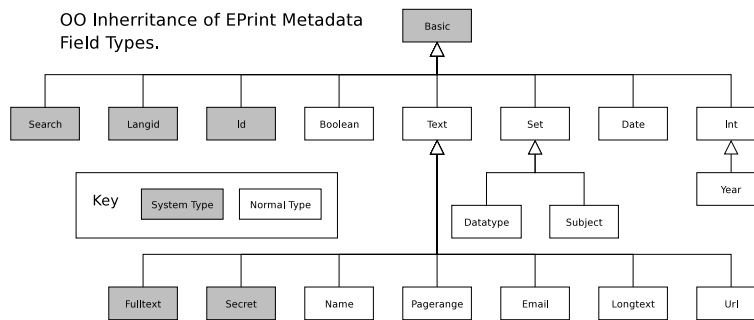
Figure 7.1: EPrints 2 Metadata Types Inherritance

7.2   Fields Configuration

Fields have a number of properties. The only required properties are "name" and "type". Name is the name of the field. This is used to identify this throughout the system. The other properties depend on what type the field is.

When you add a field you need to add the "human readable" version in the phrase file, this seperation allows you to change the description without changing the field itself. When you add a field named "foo" to the "eprint" metadata you should add "eprint_typename_foo" to the phrases. You may also wish to add "eprint_typehelp_foo" which is the explanation given to the user on the metadata input page.

The following types of field are supported, along with their special property options.

(there are some internal types not mentioned here. There use is not recommended.)

7.3   Metadata Types

There are a number of different types which are stored, input, rendered and searched differently.

Some types extend more simple types. Eg. "Year" extends "int", but forces a limitation of 4 digits and all the descriptive text is different.

It is theoretically possible to add your own types which inherrit from the inbuilt ones. This should be approached with caution. Pagerange is a good example to look at, when considering making your own types. You must make sure your ArchiveConfig.pm has a "use" for the module for your new type as it won't be loaded otherwise. The module name is the same as for the type, except that the first letter is capitalised. Field type "latlong" would be described by module `EPrints::MetaField::Latlong`.

int

Optional properties: digits

This type describes a positive integer. Stored as an `INT` in the database.

year

Where possible use a "date" field with a minimum resolution of "year" instead of the "year" type. That way the field can be treated as a date in the searches rather than an int.

This type describes a year. It works pretty much like "int" but is always 4 digits long. Stored as an `INT` in the database.

longtext

Optional properties: input_rows, input_cols, search_cols

This type describes an unlimited length text field. Used for things like titles and abstracts. It can't be effiently searched as a single value, the system indexes the words. See "free text indexing" section. Stored in MySQL as a `TEXT` field.

date

Optional properties: min_resolution

This type describes a date, always expressed as YYYY-MM-DD, eg. 1969-05-23. It is stored as a `DATE` in the database.

boolean

Optional properties: input_style

This is a simple yes/no field which is stored in the database as `SET(` `'TRUE','FALSE' )`. It can be rendered as a menu, a check box or radio buttons. (See input_style)

name

Optional properties: input_name_cols, search_cols, hide_honourific, hide_lineage, family_first

This type is used to store names of people (eg. authors). It is split into 4 parts: honourific, given names, family name and lineage. This may seem over fussy but it avoids people putting "Reverend" in the given names or "Junior" in the family name. If you dislike this you can hide honourific and lineage (See ArchiveConfig.pm).

We use "family name" rather than "last name" in the hope of avoiding international confusion (some countries list family name first, so their last name is what I would call their "christian", or "first", name.

Names are stored using 4 SQL fields. The name field "supervisor" would be stored as supervisor_honourific, supervisor_given, supervisor_family, supervisor_lineage. Each is a `VARCHAR(255)`.

set

Required properties: options

Optional properties: input_rows, search_rows

This type is a limited set of options. The list of options must be specified. Each option must also be added to the phrase file. Option "foo" of field "bar" in the "user" dataset will have the phrase id "user_fieldopt_bar_foo".

Stored in the database as a `VARCHAR(255)`, containing the id of the option.

text

Optional properties: input_cols, maxlength, search_cols

This is a simple text field. It normally has a maximum length of 255 ASCII characters, less if non-ASCII characters are used as these are UTF-8 encoded.

Stored in the database as a `VARCHAR(255)`.

secret

Identical to "text" except that the input field is a starred-out password input field, and it is only ever written to the database, it can't be read back. Writing an empty value will NOT change the previous value.

url

Identical to "text" except it is rendered and validated differently.

email

Identical to "text" except it is rendered and validated differently.

subject

Optional properties: top, showtop, showall, input_rows, search_rows

This is a hierarchical subject tree. At first glance it works like sets, but it can be searched for all items in or below a given subject. Subjects may be added to the live system.

The subject tree starts at a subject with the id "ROOT" but a subject *field* only offers all the items below the subject with the id "subjects". This can be changed using the "top" property, so that you can have two fields which options are different parts of the same tree.

Subjects may have more than one parent. eg. *biophysics* can appear under both *physics* and *biology*, while still being the same subject.

See the bin/import_subjects manpage for more information on seting up the initial subjects.

You may have more than one "subject" field, eg. Subject and Department, with unrelated parts of the subject tree as their "top".

A later version of eprints2 will have a feature which allows an admin user to limit an editor user to a certain subject (and things below it). So that in the above example you can declare an editor of either a Subject (capital-S) or a Department.

pagerange

A range of pages, eg 1-44. Currently not searchable.

Stored in the database as a `VARCHAR(255)`.

datatype

Required properties: datasetid

Optional properties: input_rows, search_rows

This field works like a set, but gets its options from the types of the dataset specified.

For example, if you specified the datasetid "user" then, unless you've changed the defaults, would give the options "user","editor" and "admin" - which are the types of user specified in **metadata-types.xml**.

Options are:

user

The types of user.

document

The types of document.

eprint

The types of eprint.

security

Security levels of a document (probably not very useful).

language

All the languages specified in **languages.xml**

arclanguage

The languages supported by this archive. Configured in ArchiveConfig.pm. Stored in the database as a `VARCHAR(255)`.

langid

This is used internally, it contains an ISO language ID. You probably don't want to use it. Stored as a CHAR(16).

id

This is also used internally, it contains the ID part of a field with the hasid property. Don't use it! Stored in the database as a `VARCHAR(255)`.

search (Since EPrints v2.1)

>   Required properties: datasetid, fieldnames
>
>   Optional properties: allow_set_order
>
>   This type describes a stored search acting on the named dataset. The fields that can be searched are described by fieldnames.
>
>   This field type is quite unusual and you are not really expected to use it. It was created for use in the systems field of the Subscription dataset.
>
>   This field is stored in MySQL as a `TEXT` field.

>   Field Properties:

"status" indicates either "system" or "cosmetic" or "other". "system" properties cannot be changed without erasing and recreating your archive. "cosmetic" fields only effect the display of data and can be safely changed. "other" is explained in the description.

name

>   Status: system
>
>   Required by: all
>
>   Default: NO DEFAULT
>
>   The name of the field. Strongly recommended to only be lowercase a-z only.

type

>   Status: system
>
>   Required by: all
>
>   Default: NO DEFAULT
>
>   The type of field. One of the list described above.

browse_link

>   Status: cosmetic
>
>   Optional on: all
>
>   Default: undef
>
>   This is the id of a "browse" view. This will hyperlink this value to the browse for that value when rendering it.

confid

>   Status: cosmetic
>
>   Internal use only. Sets the confid if a field is being created without a dataset. The confid is used as a fake dataset for generating phrase ids.

datasetid

> Status: other
>
> Required by: datatype
>
> Default: NO DEFAULT
>
> Used to set which dataset's types are this fields options.
>
> Changing this on a live system could cause some confusion, as values in the old dataset may exist.

digits

> Status: cosmetic
>
> Optional on: int
>
> Default: 20
>
> Maximum number of digits for this number.

input_rows

> Status: cosmetic
>
> Optional on: longtext, set, subject, datatype
>
> Default: set in ArchiveConfig.pm
>
> The number of input rows in a text area, or options to display at once in a menu. Setting to 1 will make a pull down menu (unless this is a "multiple" field).

search_cols

> Status: cosmetic
>
> Optional on: text, longtext, url, email, name, id
>
> Default: set in ArchiveConfig.pm
>
> The width of the search field. If searching multiple fields at once then the value is taken from the first field in the list.

search_rows

> Status: cosmetic
>
> Optional on: datatype, set, subject
>
> Default: set in ArchiveConfig.pm
>
> The number of items to display in a search field list. If searching multiple fields at once then the value is taken from the first field in the list.

input_cols

> Status: cosmetic
>
> Optional on: text, longtext, url, email
>
> Default: set in ArchiveConfig.pm
>
> The width of the input field.

input_name_cols

> Status: cosmetic
>
> Optional on: name
>
> Default: set in ArchiveConfig.pm
>
> The width of the input fields of a "name" field.

input_id_cols

> Status: cosmetic
>
> Optional on: fields with "hasid" set.
>
> Sets the width of the ID input field on a field with an ID.
>
> Default: set in ArchiveConfig.pm

input_add_boxes

> Status: cosmetic
>
> Optional on: fields with "multiple" or "multilang" set.
>
> Default: set in ArchiveConfig.pm
>
> How many boxes to add when the user hits the "more spaces" button.

input_boxes

> Status: cosmetic
>
> Optional on: fields with "multiple" set.
>
> Default: set in ArchiveConfig.pm
>
> How many boxes to initially show on a multiple field.

input_style

> Status: cosmetic
>
> Optional on: boolean
>
> Default: undef
>
> By default booleans render as a check box. These other formats look a bit clearer on the input field:
>
> **menu**
>
> > Display as a pull-down menu. You will need to set the phrases *dataset*_fieldopt_*fieldname*_TRUE and *dataset*_fieldopt_*fieldname*_FALSE (where dataset & fieldname are the ids of the dataset and field). These are the menu options.
>
> **radio**
>
> > Display as radio buttons (ones which deselect when you select another one). You will need to set the phrase *dataset*_radio_*fieldname*. This phrase should have two "pin" elements: true and false, which are the positions to place the radio buttons.

input_assist

> Status: cosmetic
>
> Optional to: all
>
> Default: undef
>
> Add an internal button which reloads the page, with a "#" jump to make the page load at the current input field. The assist button does not do anything except cause the page to be reloaded. This is intended to work with the input_advice fields.

input_advice_right

> Status: cosmetic
>
> Optional to: all
>
> Default: undef
>
> If defined this should be a function pointer which takes params ($session, $field, $value)
>
> value is the current value of the field.
>
> The return value of this should be an XHTML chunk based on the value. This XHTML will appear to the right of the input fields for the value.
>
> This is intended to give useful advice, which as if the field is an int which is the eprintid of another eprint, this feature could render then name and a link to that eprint to appear next to the integer input box.

input_advice_below

> Status: cosmetic
>
> Optional to: all
>
> Default: undef
>
> As with input_advice_right, only the results appear below, not to the right. Both _right and _below may be used.

fromform

> Status: cosmetic
>
> Optional to: all
>
> Default: undef
>
> A reference to a perl function which will process the value from the form before storing it. The function will be passed ($value, $session) where value is the value from the form and session is the current EPrints::Session. It should return the processed value.
>
> This could be used, for example, to turn a username "moj199" into a userid "312" for internal user.

toform

Status: cosmetic

Optional to: all

Default: undef

A reference to a perl function which will process the value just before it is displayed in the form. The function will be passed ($value, $session) where value is the value from the database and session is the current EPrints::Session. It should return the processed value.

This could be used, for example, to turn a userid "312" being used internally by your systems into more human-friendly username "moj199".

If you use toform then you should probably set fromform to change your values back again.

maxlength

Status: cosmetic

Optional to: text, email, url, secret

Default: 255

The maximum length of the value.

hasid

Status: system

Optional to: all

Default: 0

This adds an additional "ID" property to the field. This is most useful on a "name" field which is "multiple". It associates an additional value with the name, for example a username, or email address, which can be used to *uniquely* identify that person. If you want to get an accurate list of all of someones papers then their name is NOT good enough.

You might also wish to make a "publication" text field have an ID which is an optional ISSN, but it makes more sense in "multiple" fields.

multilang

Status: system

Optional to: all (but silly for date, year, int, boolean)

Default: 0

If set this makes the field "multilingual". That is to say it can have more than one value, one value per language.

For example, the "canadian stuff" archive may wish to make your title and abstract multilang so that authors can enter them in both french and english.

This is more useful than having title_en and title_fr as eprints *understands* it and can render the version of the field appropriate to the viewer (if they set a language preference).

multiple

Status: system

Optional to: all (but silly for date, year, int, boolean)

Default: 0

If set this property makes the field a LIST rather than one value and handles rendering it as a list and inputing it. The input field will appear with a default of 3 inputs and a "more spaces" button which will reload the page with more if you need more than 3.

This causes the field to be stored in a seperate SQL table.

options

Status: other

Required by: set

Default: NO DEFAULT

This should be a array of options. eg.

```
[ "blue", "green", "red" ]
```

Removing options on a live system could leave invalid values floating around. Adding options is fine. Don't forget to add them to the phrase file too.

required

Status: system

Optional to: all

Default: 0

This indicates that this field is *always* required. It is not recommended to set this, but rather indicate requirednes of fields by type in the metadata-types.xml file.

Either way you set it, required fields will cause the item they are in to fail to validate unless the field has a value.

requiredlangs

Status: other

Optional to: fields with "multilang" property

Default: []

A list of languages which are required for this multilang field. eg. you can force an "en" (english) entry, while allowing them to optionally add others.

eg. ["en", "fr"]

A list of codes can be found in languages.xml

Adding more requiredlangs does not magically give you values for these languages in existing data.

showall

Status: cosmetic

optional to: subjects

Default: 0

By default subjects are only shown if they are "depositable". This option makes all subjects, depositable or not, options.

showtop

Status: cosmetic

optional to: subjects

Default: 0

If set then the topmost item in the subject is shown. Usually this is a container, eg. "subjects", and should remain hidden.

top

Status: cosmetic

optional to: subjects

Default: "subjects"

Sets the top node in the tree. The options are all the children (and their children).

idpart

Used internally.

mainpart

Used internally.

render_single_value

Status: cosmetic

Optional to: all

Default: undef

This overrides the rendering of a single item. In a multiple, multilang field it will be called on each value of the language to display.

This is a reference to a function which takes ($session, $field, $value) and returns a XHTML DOM fragment.

Set this to \&EPrints::Latex::render_string to make eprints try and spot latex in this fields values and render it as images instead!

(Since EPrints v2.1) Set this to \&EPrints::Utils::render_xhtml_field to make eprints read this field as XML and place that XML right in the XHTML web page. (Normally the system would escape all the greater-than and less-than characters.

render_value

Status: cosmetic

Optional to: all

Default: undef

This is a reference to a function which will render the entire value of the field, overriding eprints own renderer. It should take as parameters: ($session, $field, $value, $alllangs, $nolink)

The function should return an XHTML DOM fragment.

If $alllangs is set then the function should render all values on a multilang field, rather than just the "best" one.

If $nolink is set then no HTML anchor links should be used, eg. to link a URL.

render_opts (v2.3.0)

Status: cosmetic

Optional to: all

Default: undef

This allows you to specify certain minor tweaks in how this fields values are displayed without going to all the trouble of creating a custom render_value subroutine.

See the section on "Metadata Field Render Options" for details.

export_as_xml

Status: cosmetic

Optional to: all

Default: 1

If this attribute is set to zero then this field will be ommitted from the output of the XML export script.

make_value_orderkey

Status: other

Optional to: all

Default: undef

This may be a reference to a subroutine which returns a single string which can be used to alphabetically sort this string. It is used to order the results within the database. The function is passed the following parameters ($field, $value, $session, $langid). You may wish to sort certainly fields differently for different languages.

For example - for some reason you may want a field formated with a single character then an integer (a934 or b3) - If you sort this alphabetically then a2 would come after a11. So you make the orderkey function do something like:

```
$value =~ m/^(.)([0-9]+)$/;
return sprintf( "%s%08d", $1, $2 );
```

This would turn a2 into a00000002 and a11 into a00000011 which will sort correctly alphabetically. Don't worry - these values are only ever used for sorting, they should never get output.

You should probably use the bin/reindex command on the dataset in question (probably "archive" or "user" after adding or changing this property to a field. This may take a significant amount of time.

make_single_value_orderkey

Status: other

Optional to: all

Default: undef

This is a slightly more simple version of make_value_orderkey. It only takes ($field, $value) as parameters. It is only ever passed single values of $value and lets eprints takes care of multiple values (or multilang values) by calling the function once per value.

As with make_value_orderkey you should reindex after meddling with orderkeys.

fieldnames

Status: cosmetic(ish)

Required by: search

Default: NO DEFAULT

This should be a reference to an array of field names - exactly like the ones used in ArchiveConfig.pm to configure search, advanced search and subscriptions.

Adding fields to this will cause no problem. Removing fields will mean that those fields are ignored when turning values of this field back into searches.

can_clone (since v2.2)

> Status: changeable (but changes functionality)

> Default: 1

> If can_clone is set to zero then this field will not be cloned when the record is cloned. This may be useful for automaticly generated fields or fields with meaning such as "content has been spellchecked" or somesuch.

sql_index (2.2)

> Status: system

> Default: 1

> If this field is set to zero then an SQL index will NOT be created for it. This means the field should never be used in a "value exactly matches" search as it may be very slow. MySQL has a limit of 32 indexes per table, which is why you should use this field if you go over that limit.

id_editors_only (2.2)

> Status: cosmetic

> Default: 0

> Optional on: fields with "has_id" set.

> It means that the "id" part of the field only appears in the editor view, not the normal user submission form. Some archives may wish to do this to save confusing the person making the deposit.

allow_set_order (2.2)

> Status: changeable (but changes functionality)

> Default: 1

> Optional on: search

> Prompt user for a search order in addition to the search fields.

min_resolution (2.3.0)

> Status: changeable

> Default: day

> Optional on: date

> If this is set to "month" then the "day" part of date field will be made optional in the input form and validation.

> If this is set to "year" then both the "day" and "month" parts will be optional.

> This allows you to allow users to only enter "2003" if that's all they know, without preventing them give the exact date if relevant and known.

hide_honourific (2.3.0)

> Status: changeable
>
> Default: 0
>
> Optional on: name
>
> If set to true (1) then the honourific field does not appear in the input form for this field.

hide_lineage (2.3.0)

> Status: changeable
>
> Default: 0
>
> Optional on: name
>
> As for honourific.

family_first (2.3.0)

> Status: changeable
>
> Default: 0
>
> Optional on: name
>
> If set to true (1) then the input form presents the "family" field before the "given" field. This seems to make librarians happy.

Chapter 8

Configuring the functions of an Archive

8.1 Multi Page Metadata Input (v2.3.0)

If you want to split the metadata input into more than one page, you can, by adding <page name="foo" /> elements in between <field> elements in metadata-types.xml.

The "name" attribute is used so that EPrints knows which page it's currently on. It can also be used to define a custom title for a page of fields, and to specify validation requirements for that page.

Metafield input page name

Eg. The title of a metadata input page is taken from the phrase "meta-page_title_*pagename*". It may have any of the following pins:

type

> The type of the current submission. Article, Book, or whatever.

eprintid

> The ID number of the current submission.

desc

> The short description of the item. Usually the title.

Per-page Validation

The simple validation will be checked for each field on the sub page. This means that an invalid URL will raise a problem and not let the submitter continue. However if you have a more complex validation issue, such as an exclusion or a

co-dependancy, you will need to edit the **ArchiveValidateConfig.pm** config file, and edit this subroutine:

```
sub validate_eprint_meta_page
{
        my( $eprint, $session, $page, $for_archive ) = @_;


        my @problems = ();


        return @problems
}
```

The options are as for validate_eprint_meta except that $page is the sub-page to validate. @problems should be an array of XHTML objects describing any problems with the data submitted for that page.

8.2   Submission Customisation XX

8.3   Filters XX

8.4   Searches XX

8.5   OAI XX

8.6   Latest Tool XX

8.7   Metadata Field Render Options (v2.3.0)

Render options are settings for a metadata field which control how it is rendered (but nothing else). Some render options are only meaningful for certain types.

Setting in Metadata Fields Configuration

Render options can be specified as properties of a metadata field in ArchiveMeta-dataFieldsConfig.pm in which case they apply to that field (unless over ridden). In this case they are a hash reference, for example:

```
{ name => "creators", type => "name", render_opts=>{ order=>"gf" } },
```

This sets the "order" render option of the creators field to be "gf".

Setting in views and citations

Render options can also be specified in views and citations. If you don't want them to apply except in the given view or citation. For example, in citations:

```
@title;magicstop@
```

Magicstop is a boolean option so this is the same as saying:

```
@title;magicstop=1@
```

In views you can use

```
"some_date_field;res=year"
```

To make a view that browses by the values of a date field as if it were a "year" field.

Available options

Boolean options with no value default to true (1).

magicstop

Boolean. Applies to text and longtext fields. If true then render the value with a full stop on the end unless the value already ends with "." "!" or "?". Handy for getting citations right.

noreturn

Boolean. Applies to text and longtext fields. Turns all Carriage Return and Line Feed characters into whitespace. Handy when you have authors entering titles with linebreaks in which should only be displayed under some circumstances.

order

"gf" or "fg". Applies to name fields. Override how this name field will be rendered. Either "given-name family-name" or "family-name, given-name".

quiet

Boolean. If true then and the value is not set, don't print the ugly "UN-SPECIFIED" just print an empty string.

res

"day", "month" or "year". Default is "day". Applies to date fields only. Resolution at which to deal with the dates. @foo;res=year@ will always render just the year part of the "foo" field.

Chapter 9

Troubleshooting

9.1 Trouble Shooting

This section covers some things which can go wrong and why. If you have a suggestion for this section, let us know!

It will grow as people suggest new problems and solutions. Check the http://www.eprints.org/ website for the latest version.

9.2 Installation of EPrints and Required Software

Apache Crashes with a segmentation fault

Possible cause: apache linked against "expat" library. If you did not install apache from source then it is possible it was linked against the "expat" library. The problem arrises that it is also linked against mod_perl and when we use the XML::Parser module, that is also linked against expat. 2 expat's in one apache make it seg-fault.

(Under SuSE Linux) Apache has problems compiling the mod rewrite module

With an error something like:

```
 In file included from mod_rewrite.c:93:
 mod_rewrite.h:133: ndbm.h: No such file or directory
```

Possible cause: Missing the ndbm library which is required (for some reason).

Solution: It comes as part of gdbm which is free. If working from a package you need gdbm-devel to get the header files (.h files).

9.3   Setting Up and Configuring a New Archive

System gives a "500 Internal Error" when viewing advanced search or submitting a document

Possible cause: No Subjects, Bug in code.

Solution: Run **generate_subjects**

If this fails: Look at your apache error log for clues. If reporting a bug, include the errors from the apache error log (often, but now always, found at `/usr/local/apache/logs/error_log`)

9.4   General

Solution: Build apache following the detailed instructions in the "requried software" section of the documentation.

Changes to the configuration didn't appear on the website

Possible cause: Several.

Solution: Rebuild everything by re-running (for the archive in question) **generate_static**, **generate_views**, **generate_apacheconf** then stop and start apache. **generate_abstracts** can take a long time, so don't run it unless you want to update the abstracts themselves.

Browse View page gives a "404 not found" error or fails to update.

Possible Cause: You didn't run the script which makes them!

Solution: Run generate_views, and ideally set it up to run automatically: see "Browse Views" in the installation section.

Apache takes a really long time to start (over a minute) and so do the command line scripts.

Possible Cause: EPrints loads several XML files at start up, and for some reason this requires a DNS lookup if DNS lookup is unavailable then it has to timeout.

Solution: Make sure that the machine can perform DNS look-ups.

Chapter 10

How-To Guides

10.1   HOW TO: Set up a Complex Custom View

Simple instructions are in the ArchiveConfig.pm section.

Example situation: On my main website www.foobars.ac.uk I have a page per research project, of which we have hundreds. Each project has a short unique id code, eg. "manticore". I have a field in my eprints archive (eprints.foobars.ac.uk) which is configured:

```
{ name=>"projectcodes", type=>"text", multiple=>1 }
```

I add the following browse view:

```
{ id=>"by_project", allow_null=>0, fields=>"projectcodes",
order=>"-year/title" }
```

This will generate a page http://eprints.foobars.ac.uk/views/by_project/manticore.html with a list of all papers in that project, we can link to that URL!

Making the Field link to the Browse Page

If you want a subject to link the subject browse page of that value, add the **browse_link** property to the field (and regenerate the abstracts if you want).

If you remove the browse view you should remove the browse_link or it will be a broken link.

Values rendered inside citations which are used to link to the main record will not link into the browse view for obvious reasons.

Including a view in another page

If you change that to:

```
{ id=>"by_project", allow_null=>0, fields=>"projectcodes",
order=>"-year/title", noindex=>1, nolink=>1, nohtml=>1, include=>1 }
```

This will generate a view which is NOT listed on the /view/ page and it will not skip making the .html file and make a .include file per value. This will only contain the "count" of items and the XHTML of their citations. This can be used as part of a page on the http://www.foobaars.ac.uk/ site; either by using php and capturing it on-the-fly using `readfile` or scripting it with perl or NFS exporting the filesystem onto the main server (or just doing it all on one computer) and using server side includes to place it in a page.

Customising the way each item is cited

I want to list the project papers in a strict format in a table with 4 columns: title, author(s), year and an icon which links to the document abstract page...

I add some *more* options to the browse view:

```
{ id=>"by_project", allow_null=>0, fields=>"projectcodes",
order=>"-year/title", noindex=>1, nolink=>1, nohtml=>1, include=>1,
nocount=>1, citation=>"project_table" }
```

Now I add a new citation to the citations config file:

```
<ep:citation type="project_table"><tr><td>&title;</td><td>&authors;</td>
<td>&year;</td><td><ep:linkhere><img
src="http://www.foobars.ac.uk/images/paperlink.png" alt="view" width="32"
height="64" border="0" /></ep:linkhere></td></tr></ep:citation>
```

That should generate for the manticore project, in the ".include" file (I've cut the contents of the "img" tag for readability:

```
<tr><td>Making Stuff</td><td>Guy, A.</td><td>2001</td><td><a
href="http://eprints.foobar.ac.uk/archive/00000923/"><img ... /></a>
</td></tr>
<tr><td>Eating Food</td><td>Herring, Walter and Chips, Bob</td><td>
2000</td><td><a href="http://eprints.foobar.ac.uk/archive/00000445/">
<img ... /></a></td></tr>
```

A "CV" page - a list of all of Alices records

This is where the authors and editors field having an ID comes in handy.

Say we use local username to identify people in the "Person ID" fields, we can now set up a view:

```
{ id=>"by_person", allow_null=>0, fields=>"authors.id/editors.id",
order=>"-year/title", noindex=>1, include=>1 }
```

This will generate both .html pages and .include pages. An member of your organisation can get a list of their records either by linking to /views/by_person/alice.html (where alice is their username) or by snarfing the URL /views/by_person/alice.include into his own homepage.

10.2 HOW TO: Add a New Field

This convers adding a new field to a new system, not a live system. It is possible to add a new field to a live system but invloves SQL hacking.

In this example we add a new "set" field called "local" which will have 3 options "yes","no" and "partial" - this will indicate if the item in question was produced in our organisation or not.

Add the Field to ArchiveMetafieldConfig.pm

Add the field to the appropriate part of ArchiveMetafieldConfig.pm (the "eprint" section in our example)

```
{ name => "local", type => "set", input_rows => 1,
    options => [ "yes", "no", "partial" ] }
```

input_rows being set to one will make it appear as a pull-down menu.

Add the Field to metadata-types.xml

If you want the user to be able to edit this field for any or all types of eprint/user then you need to add it to each appropriate type in metadata-types.xml (this can be changed on a live system without any serious consequencies).

```
<field name="local" required="yes" />
```

Add the Field Information to the Archive Phrase File(s)

Normally we just need to add fieldname and fieldhelp, but this is an option field so we need to add names for each option. If we run the archive in more than one language then we add this to each phrase file (but in the appropriate language).

```
<ep:phrase ref="eprint_fieldname_local">Produced Locally</ep:phrase>
<ep:phrase ref="eprint_fieldhelp_local">Please indicate if this item was
produces in the foo organisation, or not.
<ep:phrase ref="eprint_fieldopt_local_yes">produced locally</ep:phrase>
<ep:phrase ref="eprint_fieldopt_local_no">not produced locally</ep:phrase>
<ep:phrase ref="eprint_fieldopt_local_partial">only partially produced
locally</ep:phrase>
```

Other things you may wish to change after adding a new field

Add it to the citations file

This is optional, only do this if you want it to appear in the cited forms.

In our example case we only want this to appear when citing technical reports, so we change that entry to:

```
<ep:citation type="eprint_techreport"><ep:linkhere><span
class="citation">&authors; <ep:ifset name="year">(&year;)
</ep:ifset>&title;. Technical Report<ep:ifset name="reportno">
&reportno;</ep:ifset><ep:ifset name="department">, &department;
</ep:ifset><ep:ifset name="institution">, &institution;</ep:ifset>.
&local;.</span></ep:linkhere></ep:citation>
```

All we've done is add &local;. to the end. It's not inside <ep:ifset
name="local">   as it is a required field and will (should) always be set.

Add it to the the Abstract (or View-User) page.

This is also optional. If you want it to appear on the web page for this item
then edit ArchiveRenderConfig.pm and select the appropriate function,
either eprint_render or user_render.

In our example we only want to mention items if an item was not produced
locally. We'll add it below the documents and above the abstract...

Single language example:

```
if( $eprint->get_value( "local" ) ne "yes" )
{
     # don't need to "my $p" as it's done earlier.
     $p = $session->make_element( "p" );
     $p->appendChild( $session->make_text( "This item was " ) );
     $p->appendChild( $eprint->render_value( "local" ) );
     $p->appendChild( $session->make_text( "." ) );

     # Append our new paragraph to the page.
     $page->appendChild( $p );
}
```

Multiple-language example:

If you want to make it handle more than language then we'll need to use
the archive phrase file - we would add something like this to each languages
file:

```
<ep:phrase ref="page:itemnotlocal"><p>This item was <pin ref="status" />.</p>
```

And to the ArchiveRenderConfig.pm file:

```
if( $eprint->get_value( "local" ) ne "yes" )
{
     my $localmsg = $session->html_phrase(
             "page:itemnotlocal",
             status=>$eprint->render_value( "local" ) );
     $page->appendChild( $localmsg );
}
```

You may prefer to use this method even if you are only using a single language.

Add extra Validation Routines

If you need to validate this field in a special way, add code into Archive-ValidateConfig.pm

Add it to the OAI metadata (eprints only)

If this field can be rendered into Dublin Core (or other metadata formats you are using) then add it to the appropriate place in the ArchiveOAIConfig.pm file.

Add a browse view (eprints only)

If you want to be able to browse this values items. See elsewhere in the docs for how to do this.

If you add a field you will need to run erase_archive and create_tables before you will see a change. EPrints will fail to run if you change the fields and do rebuild the tables.

10.3 HOW TO: Remove a Field

The quick answer is to say "the opposite of adding one".

Remove the Field to metadata-types.xml

Removing it from here will mean that nobody can enter values for that field. Which is possibly enough, and means you can put it back later.

Remove the Field to ArchiveMetafieldConfig.pm

This will remove it from the database (and require a rebuild as with adding a field).

Remove it from the phrase file(s)

This is optional, unused phrases are just ignored.

Remove it from the citations file

If it's used there.

Remove it from the the Abstract (or View-User) page.

If it's used there.

Remove extra Validation Routines

In the unlikely event you added some validation which looks at this field.

Remove it from the OAI metadata (eprints only)

> If it's being used. Fields used to generate the default dublincore are: title, authors, subjects, abstract, year and month. It also uses "type" which is a system field so you can't remove it!

Remove it from browse views (eprints only)

> If it's used there.

10.4   HOW TO: Add a new eprint type

Add the eprint type to **metadata-types.xml**. Some fields should probably be "required".

Add the name of the type to the phrase file.

Add a citation for this type to the citations file.

10.5   HOW TO: Remove an eprint type

Remove it from metadata-types.xml

You can remove it from the phrase file & citations but it won't hurt to leave it there.

10.6   HOW TO: Add a new document type

Add it to **metadata-types.xml**. This does not need any fields.

Add the name of the document type to the phrase file.

Add a citation for this document to the citations file.

If you want this to be one of the must-have-one-of document types then add it's id to the list in ArchiveConfig.pm

If you want to do something "clever" on the abstract page then edit the ArchiveRenderConfig.pm file. If you don't then it will use the citiation you created to render it in the list, as with PDF, HTML etc.

10.7   HOW TO: Add a Discussion Forum for Each EPrint

The UK Open University (open.ac.uk) have set up a service which allows you to create a discussion for every EPrint in your archive.

The really easy way to do this is to use their discussion server. If you want to run your own d3e server the software is available from http://d3e.sourceforge.net/

Using d3eprints.open.ac.uk

Just add the following code to the ArchiveRenderConfig.pm file just before the
if($has_multiple_versions) bit.

Please note that this code is not internationalised.

```
######################################
# Begin D3Eprints links

my $ol = $session->make_element( "ol" );
my $li = $session->make_element( "li" );
$a = $session->render_link(
        EPrints::Utils::url_escape(
                "http://d3eprints.open.ac.uk/disc.php?url=".$eprint->get_url ),
        "_top" );
$a->appendChild( $session->make_text(
        "View public discussion of this document" ) );
$li->appendChild( $a );
$ol->appendChild( $li );

$li = $session->make_element( "li" );
$a = $session->render_link(
        EPrints::Utils::url_escape(
                "http://d3eprints.open.ac.uk/private/disc.php?url=".
                        $eprint->get_url ),
        "_top" );
$a->appendChild( $session->make_text(
        "Create private discussion of this document" ) );
$li->appendChild( $a );
$ol->appendChild( $li );

$table->appendChild( _render_row(
        $session,
        $session->make_text( "D3Eprints discussion" ),
        $ol ) );

# End of D3Eprints links
######################################
```

10.8   HOW TO: Make the latest additions to your archive appear on your main website

The contents of the "latest" page - /perl/latest - can be included via a cron tab
using wget and a server side include or using something like PHP's command
to do:

```
 readfile( "http://eprints.foo.org/perl/latest?mainonly=yes" );
```

The "mainonly=yes" flag is a hack which supresses the template of any eprints page in the /perl/ area so that it can be included, but it is most useful for "latest".

WARNING: If you have a script which imports 1000 records in one night then latest isn't currently bright enough to truncate the list so your homepage could get kinda messy.

10.9   HOW TO: Add full text searching

EPrints does not support this natively but there are several options.

htdig, or similar software

There is plenty of software which will provide a full text search of a website. To add non-eprints cgi scripts to your site create a directory in the cgi dir:

```
mkdir /opt/eprints2/cgi/local
```

and place your scripts in there, they will have URLs under `http://yoursite.com/perl/`

Using an external search engine

This is very easy, but will only index public documents. Any search engine will work, but Google is a good choice. Google provide a site-search service which allows you to register with them and then have a form which searches your site using google and adds your logo and colourscheme to the results.

A really easy solution is to just make a form which links to our "google_site" script which just adds "site:yoursite.com" to the google request to limit the search results. The HTML for this would be something like:

```
<form action="/perl/google_site">
<p>Use Google to search this site:
<input name="q" value="" />
<input type="submit" name="go" value="Search" />
</form>
```

For other search engines; see their documentation for how to make a form to search only your site.

10.10   HOW TO: Make the referencetext field link to the items referenced

This should have been in the 2.2 docs but I didn't have time. Check the latest version of the documentation at software.eprints.org, if it's not there then bug me at support@eprints.org

10.11 HOW TO: Make the password controled parts of the site use HTTPS

This should have been in the 2.2 docs but I didn't have time. Check the latest version of the documentation at software.eprints.org, if it's not there then bug me at support@eprints.org

10.12 HOW TO: Customise the way the the search results are formatted

This should have been in the 2.2 docs but I didn't have time. Check the latest version of the documentation at software.eprints.org, if it's not there then bug me at support@eprints.org

Chapter 11

VLit Transclusion Support

11.1 What is VLit?

** This feature is still experimental **

VLit is a method for quoting a part of a document directly. It is available on any file in the public documents area of a text mime-type (`text/plain`, `text/html` etc). It allows a range of characters to be extracted rather than the entire document. This allows external systems, rather than quote by cut-and-paste, to quote in context of the original.

If you use it, please edit the vlit page in `/opt/eprints2/archives/`*foo*`/cfg/static/en/vlit.xpage` to indicate under what terms the feature may be used. When using "human" mode, described below, this page is what the (c) copyright symbol links to, by default.

Don't forget to re-run **generate_static**.

11.2 But I don't want this Feature...

If for some reason you want to disable VLit, you may do so in ArchiveConfig.pm - you will then need to re-run `generate_apacheconf` and restart the web server.

Edit the vlit page (as described above) to indicate that the feature is not available on your archive.

11.3 Invoking VLit

Simple Parameters

A simple *transclusion* is achieved by adding the following to the end of the URL of a text/* document:

```
?xuversion=1.0&locspec=charrange:offset/length
```

Replacing "offset" and "length" with an integer. An offset of 0 and a length of 1 will return the first character. eg.

```
http://www.weebleprints.co.uk/archive/00000543/01/notfalldown.txt
?xuversion=1.0&locspec=charrange:1403/130
```

(All one URL, only split to fit on the page) This will return characters from offset 1403 to 1533.

Human Mode

An optional "mode" parameter may by used. The "human" mode returns the character range as HTML with characters like & properly escaped and new line characters turned into HTML "br" break tags. It will place two links before the text: a (c) link which will link to an explanation of transcopyright - If you want to change this URL you'll have to hack VLit.pm - and a TRANS link which will take you to the context of the quote - 1024 extra characters before and after but with the quote highlighted in red. Clicking TRANS on the context view will take you to the full raw document.

XML-Entity Mode

You may also set mode to be xml-entity, eg:

```
http://lemur1.ecs.soton.ac.uk/archive/00000134/01/xuDation-d18.txt?
locspec=charrange:10429/488&xuversion=1.0&mode=xml-entity
```

This returns the transclusion as a piece of XML which can be included as an external entity by other XML documents.
eg:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<transclusion xlmns="http://xanadu.net/transclusion/xu/1.0"
href="http://lemur1.ecs.soton.ac.uk/archive/00000134/01/xuDation-d18.txt"
offset="10429" length="488">A more recent interactive screen
implementation of transpointing windows (screen shot fig. 5) was created
by Ka-Ping Yee in 1999.  This was served, like Walker's, from our xu88
server (rechristened as Udanax Green).  Yee's PYXI is a skeleton front
end that shows and edits text, links and transclusions, and may be
extended to reach more of the server's functionality.  Both Udanax Green
and PYXI were released under an open source license in 1999 (120), and are
available at udanax.com. </transclusion>
```

11.4   Area locspec

There is an 'area' locspec which requires the 'convert' unix command. It returns an area of an image specified as `locspec=area:page=n/hrange=startx,endx/vrange=starty,endy`

where startx,endx,starty,endy and n are all positive integers. Any parts may be omitted: To specify the first 50 rows of page 3 `locspec=area:page=3/vrange=,50`.

Chapter 12

Backing-Up your System

Why Backup?

It is almost certain that you will be storing valuable information in your Eprints server. Even assuming that the eprints code is 100% bug free and that you will never delete 8000 records when you run the wrong script at 3am, you *still* need to back up! Drives and fans break. Computers get stolen. Server rooms get flooded (that happened to us!). Without proper backups this could be a disaster.

What to Backup

You need to backup two things.

The `/opt/eprints2/` directory (or whatever you called it). Not all the subdirectories *have* to be backed up, but it is much easier to backup the whole thing. Make sure that you back up any symbolicly linked directories too.

Each MySQL database which your archives use. See the MySQL manual for more information on backing up MySQL databases. The `mysqldump` command will dump the whole of a database as a big list of SQL commands to re-create it.

Best Practice

We strongly recommend that you:

* Regularly backup your EPrints archive and database.

* Keep mulitple sets of backups.

* Keep a recent backup physically seperate from the archive - either in another room or ideally another site.

* Regularly check that you can actually restore from your backup. It's not uncommon for people to produce a daily backup for years without checking it. When they come to need it, they discover that something has gone wrong and the backup is useless.

* Assume that you will be restoring to different hardware - the tape drive
may be stolen or melted too, and you'll be unabled to get one just the same 'cus
they stopped making them! Check that your backups work on hardware other
than that used to create them.

* Decide who is responsible for backups. Their responsibilities should include
making sure that the above policies are implemented even if they are ill or
unavailable and making sure that someone else knows how to take over making
and restoring the backups if they leave or are hit by a bus.

If you can't do all of these, which is admittedly a lot of extra work, then do
as many as you can.

Fortune favours the backed-up. It always seems to be the un-backed-up
systems that have disk crashes. Life's like that...

Chapter 13

Problems, Questions and Feedback

13.1 Bug Report Policy

There is currently no online bug tracking system. Known bugs are listed in the
BUGLIST file in the distribution and a list will be kept on the http://eprints.org/
site.

   If you identify a bug or "issue" (issues are not bugs, but are things which
could be clearer or better), and it's not already listed on the site, please let
us know at support@eprints.org - include all the information you can: what
version of eprints (see SystemSettings.pm if you're not sure), what operating
system etc.

13.2 Where to go with Questions and Suggestions

There are some mailing lists for EPrints which may be the right place to ask
general questions and start discussions on broad design issues.

eprints-tech

eprints-tech is the technical mailing list for people who want to argue over XML
processing and which libraries to use. UNIXy people.

   To subscribe send an email to majordomo@ecs.soton.ac.uk containing the
text

```
subscribe eprints-tech
```

eprints-underground

eprints-underground is the non-technical mailing list for people who actually care what the system does, not how. This is for discussion of metadata, politics (how to get people to fill the damn data in) and what people actually plan to use the eprints for.

To subscribe send an email to majordomo@ecs.soton.ac.uk containing the text

```
 subscribe eprints-underground
```

OAI Lists

The Open Archives Protocol has some mailing lists of its own, see the http://www.openarchives.org/ site for information on these.

Chapter 14

Updating from Previous Versions

Generally speaking when upgrading EPrints v2 you should unpack and install eprints to the same path as your current version. The installer will detect the existing version and upgrade it. Existing files which have been altered by you should be automatically backed up so they don't get lost. But if your hacks are important then you should probably back them up by hand before upgrading.

Always stop apache before upgrading.

Always make sure your system is fully backed up before upgrading.

Then you should follow the specific instructions for each stage from your old version to your new version...

14.1 Updating from EPrints 1

A tool for assisting with this is available from http://software.eprints.org/files/

14.2 Updating from EPrints2-alpha-2 to EPrints 2.0

Too much of the configuration of the archive has changed to document individually. See the `CHANGELOG` file if you're really interested. If you have more than one archive, then do each step for each one. Our example is the archive with the ID *foobar*.

To upgrade we suggest you use the UNIX `diff` command to make a records of changes you made to the archive config directory.

```
% diff /opt/eprints2/defaultcfg/ /opt/eprints2/archives/foobar/cfg/
```

Store the output from diff somewhere safe.

Remove `/opt/eprints2/archives/foobar/cfg/`
Now install eprints 2.0 and agree to upgrade when it asks.
Copy the new default configuration into your archive dir:

```
% cp -R /opt/eprints2/defaultcfg/ /opt/eprints2/archives/foobar/cfg/
```

Work through the "diff" you produced, and re-apply the changes to the
contents of `/opt/eprints2/archives/foobar/cfg/`.
You also need to execute the following SQL command.

```
UPDATE document SET security='' WHERE security='public';
```

To connect to mysql via the command line run:

```
% mysql -u eprints foobar -p
```

And enter the database password for that archive.
Now re-run `generate_apacheconf` and restart apache.
Upgrading from 2.0 should be less painful than this, the config files should
change less dramatically from now on.

14.3   Updating from EPrints 2.0 to EPrints 2.0.1

This release is mostly a bug fix release. There are a few bugs in the config files
which you should fix by hand.

After upgrading EPrints, repeat these steps for the cfg directory of each
archive you have created:

- In `citations-en.xml` in the citation for poster, the year field is incorrecly
  named &years; rename it to &year;

- (This is not essential) Also on the citations file for the "eprint" and "user"
  citation types: add a <ep:linkhere> .... </ep:linkhere> just inside the
  <ep:citation> tag.

- There was a bug in `ArchiveFreeTextIndexingConfig.pm` which meant
  that "return" was treated as part of a word, rather than a word break.
  Unless you have modifed that file, just replace it with the new one from
  `/opt/eprints2/defaultcfg/`

- (This is not essential) The title for the abstract block never gets added to
  the page. Edit `ArchiveRenderConfig.pm` - search for eprint_fieldname_abstract
  and add the following line as the next line:

  ```
  $page->appendChild( $h2 );
  ```

14.4 Updating from EPrints 2.0.1 to EPrints 2.1

These release fixes a few bugs but the main point is adding the subscriptions feature and support for OAI 2.0. See the NEWS file in the distribution for more information.

After upgrading EPrints, repeat these steps for each archive you have created:

- Upgrade the SQL tables - EPrints 2.1 needs to make some small changes to the database. Run:

  ```
  % bin/upgrade ARCHIVEID
  ```

  This script will explain the changes it is making. It will require the MySQL root password.

  If you want an extra level of protection, you may want to run mysqldump to backup the database.

- If you are not using the OAI system then just copy the new ArchiveOAIConfig.pm configuration file over your old one. The new one can be found in /opt/eprints2/defaultcfg/ArchiveOAIConfig.pm.

  If you have already configured the OAI system then you need to add the required configuration for OAI 2.0. These can be copied from /opt/eprints2/defaultcfg/ArchiveOAIConfig.pm. The two relevant sections are the block titled "OAI-PMH 2.0" and the subroutine make_metadata_oai_dc_oai2. Copy these into your archives ArchiveOAIConfig.pm and modify them if needed.

- Look at the section on setting up subscriptions in the "Installation" chapter.

  You need to add some more items to the cron tab to mail out the subscriptions. One set of new cron entries per archive.

- The call for EPrints::EPrint->new() has changed the order of the parameters (to standardise it with Subject, User, etc.).

  In ArchiveRenderConfig.pm edit:

  ```
  my $target = EPrints::EPrint->new( $session,
          $session->get_archive()->get_dataset( "archive" ),
          $eprint->get_value( "commentary" ) );
  ```

  to be:

  ```
  my $target = EPrints::EPrint->new( $session,
          $eprint->get_value( "commentary" ),
          $session->get_archive()->get_dataset( "archive" ) ) ;
  ```

  (swapping the order of the last 2 parameters)

14.5   Updating from EPrints 2.1 (or 2.1.1) to EPrints 2.2

Upgrade eprints software to the new version then (for each archive)

- Upgrade the SQL tables - EPrints 2.2 needs to make some small changes to the database. Run:

  ```
  % bin/upgrade ARCHIVEID
  ```

- cfg/ArchiveOAIConfig.pm

  In the sub eprint_to_unqualified_dc change this line:

  ```
  push @dcdata, [ "creator", EPrints::Utils::tree_to_utf8(
      EPrints::Utils::render_name( $session, $author, 0 ) ) ];
  ```

  to this:

  ```
  push @dcdata, [ "creator", EPrints::Utils::make_name_string( $author ) ];
  ```

- cfg/ArchiveRenderConfig.pm (optional)

  You may wish to add the following to "eprint_render", after the commentary section. It will add 'type' to the abstracts page (although not unless you run generate_abstracts

  ```
          $table->appendChild( _render_row(
                  $session,
                  $session->html_phrase( "eprint_fieldname_type" ),
                  $eprint->render_value( "type"  ) ) );
  ```

- cfg/ArchiveConfig.pm (optional)

  In the 2.1 default configuration a user could not view their own secure documents if they were not an editor. This was silly. Fix it in sub can_user_view_document:

  Change the block:

  ```
          if( $security eq "staffonly" )
          {
                  # If you want to finer tune this, you could create
                  # a new priv. and use that.
                  return $user->has_priv( "editor" );
          }
  ```

  to:

```
        if( $security eq "staffonly" )
        {
                # If you want to finer tune this, you could create
                # new privs and use them.

                # people with priv editor can read this document...
                if( $user->has_priv( "editor" ) )
                {
                        return 1;
                }

                # ...as can the user who deposited it...
                if( $user->get_value( "userid" ) ==
                        $eprint->get_value( "userid" ) )
                {
                        return 1;
                }

                # ...but nobody else can
                return 0;

        }
```

- cfg/ArchiveRenderConfig.pm (optional)

  You may wish to add subscriptions to the user_render_full method so staff
  can see what subscriptions a user has. Just before

```
        $info->appendChild( $table );
```

Add:

```
my @subs = $user->get_subscriptions;
my $subs_ds = $session->get_archive->get_dataset( "subscription" );
foreach my $subscr ( @subs )
{
        my $rowright = $session->make_doc_fragment;
        foreach( "frequency","spec","mailempty" )
        {
                my $strong;
                $strong = $session->make_element( "strong" );
                $strong->appendChild( $session->make_text(
                        $subs_ds->get_field( $_ )->display_name( $session ) ) );
                $strong->appendChild( $session->make_text( ": " ) );
                $rowright->appendChild( $strong );
                $rowright->appendChild( $subscr->render_value( $_ ) );
```

```
                            $rowright->appendChild( $session->make_element( "br" )
                }
                $table->appendChild( _render_row(
                        $session,
                        $session->html_phrase(
                                "page:subscription" ),
                        $rowright ) );

        }
```

- Subject Editors (optional)

  If you want to add the feature which allows the editor to set a search
  which limits the scope of what an editor may edit (eg. only books, or
  only things under oceanography in the subject table) you must add in
  metadata-types.xml ...

  To the end of <type name="editor">

  ```
  <field name="editperms" staffonly="yes" />
  <field name="frequency" required="yes" />
  <field name="mailempty" required="yes" />
  ```

  And to the end of <type name="admin">

  ```
  <field name="editperms" />
  <field name="frequency" required="yes" />
  <field name="mailempty" required="yes" />
  ```

  And then to ArchiveConfig.pm add (near "subscription fields")

  ```
  # Fields used for limiting the scope of editors
  $c->{editor_limit_fields} =
  [
          "subjects",
          "type"
  ];
  ```

- Translated Phrase Files (non-english archives only)

  There is a tool in http://software.eprints.org/files/contrib/update_phrases/
  which will assist with identifying changes made in the English version
  which should be applied to your translated version(s).

- Rehash Documents (optional but recommended)

  If you want to use the "probity" feature (see elsewhere in documentation
  for more information) you should run the rehash_documents command to
  generate a hash of each document file in the system.

- Updating the website (optional)

  You may also wish to update the website to reflect that eprints 2.2 is OAI 2.0 compliant. Also re-running generate_static will update the version number for eprints where mentioned.

- matches per page (optional)

  10 matches per search page was a little low. You may wish to up it to 100, which seems to work better for people.

- GDOME support

  If you wish to enable GDOME support (faster, better XML library) install the additional required libraries then edit the flag in SystemSettings.

  The additional 2 changes are required for GDOME support.

  In ArchiveConfig.pm in sub get_entities change

  ```
          $entities{ruler} = $archive->get_ruler()->toString;
  ```
  to
  ```
          $entities{ruler} = EPrints::XML::to_string( $archive->get_ruler() );
  ```

  and in ruler.xml change the XML header to:

  ```
   <?xml version="1.0" ?>
   <!DOCTYPE ruler >
  ```

- Home grown plugins and scripts (API changes)

  If you've made your own scripts using the eprints modules then you should be aware of some changes to those modules:

  $metafield->get_dataset()
  > MetaFields no longer known what dataset they belong to

  $metafield->set_dataset()
  > MetaFields no longer known what dataset they belong to

  $metafield->get_values($session)
  > Now use: $metafield->get_values($session, $dataset)

  XML functions
  > All the XML handling now uses EPrints::XML as an abstraction to the differences between XML::GDOME and XML::DOM.
  >
  > Please use EPrints::XML::to_string($node) rather than $node->toString as in GDOME toString does not work as expected on DocumentFragments.

  EPrints::Config::parse_xml()
  > Removed. Use EPrints::XML methods instead.

$db->get_values($field)
>   Now use: $db->get_values($field, $dataset)

XHTML Page generation functions changed
>   Should not cause any problems, but might.

$searchfield->set_value()
>   Removed.

14.6   Updating from EPrints 2.2 (or 2.2.1) to EPrints 2.3

This version adds some more functionality and uses apache in a slightly different way. The full text indexing tool adds some more requirements.

Recompile Apache

EPrints 2.3 requires an additional mod_perl hook, TRANS. I suggest you compile apache with "EVERYTHING=1" instead of "AUTHEN=1 AUTHZ=1" (see the required software section of the 2.3 manual). mod_rewrite is not used any more.

libapreq

Version 2.3.7+ of EPrints do not require this module. It was a pain to install. 2.3.0 to 2.3.6 do require it, but we recommend you run 2.3.7 or later.

Install tools used by full-text indexer

See the chapter "Required Software", section "Full Text Indexing". Follow the instructions.

Upgrade eprints software to version 2.3

As usual download and extract the latest (stable) version and run ./configure
>   Stop the webserver
>   Run ./install.pl as usual
>   For each archive run

```
% bin/upgrade ARCHIVEID
```

>   Update the apache config:

```
% bin/generate_apacheconf
```

>   Start the webserver

Add the new icons & update the style sheets

The new input interface has up and down buttons to let you rearrange the values in lists, such as authors. If you don't copy them into your live archives you'll probably see ugly text of "UP" and "DOWN". You need to put them in your archives "images" dir then use generate_static to copy them to the live website.

```
% cp defaultcfg/static/general/images/multi_*.png archives/ARCHIVEID/cfg/static/general/image
```

Copy in the new VLit stylesheet.

```
% cp defaultcfg/static/general/vlit.css archives/ARCHIVEID/cfg/static/general
```

Copy any new styles you want from

```
defaultcfg/static/general/eprints.css
```

to

```
archives/ARCHIVEID/cfg/static/eprints.css
```

I suggest everything after the line which starts ".subject_browser_opts, .subject_browser_selections, .subject_browser_search". This isn't essential, but will make your site look nicer.

Now run:

```
% bin/generate_static ARCHIVEID
```

Start the indexer

EPrints 2.3 does not create the search index in the same way as 2.2 did. It has a background daemon which builds new indexes and then copys them over the old indexes when they are ready. This can potentially take quite a long time, but means that editing records will be much faster (no need to keep reindexing).

If you have some buggy or experimental archives which you do not wish the indexer to process add <index>no</index> to the main XML config file for that archive.

Only one copy of the indexer runs, no matter how many archives you have. To start it run:

```
% bin/indexer start
```

You should make the indexer automatically start and stop if the machine is rebooted. See the main documentation about the indexing system.

14.7   Updating from any of EPrints 2.3.0 through EPrints.2.3.6 to EPrints 2.3.7

Both the below changes are optional, although recommended.

Short URLs

You may wish to enable the short urls feature; just add $c->{use_short_urls} = 1; to ArchiveConfig.pm

XSLT for OAI

2.3.7 introduced an XSLT stylesheet for OAI requests. If you install it and view your OAI requests in Mozilla or IE it should be rendered in a human readable form.

```
% cp defaultcfg/static/general/oai2.xslt archives/ARCHIVEID/cfg/static/general
```

Chapter 15

EPrints History (and Future Plans)

15.1   A Brief History of EPrints

The EPrints project was created by Professor Stephen Harnad.

April 2000

Rob Tansley begins work on EPrints

June 2000

EPrints beta-1 released

Cogprints archive created. http://cogprints.soton.ac.uk/

September 2000

EPrints beta-2 released

November 2000

EPrints 1.0 released, contains OAI 0.2 support

Rob Tansley leaves the EPrints Project

Christopher Gutteridge joins the EPrints Project

January 2001

EPrints 1.1 released, contains OAI 1.0 support

Work begins on EPrints 2

June 2001

Mike Jewell joins EPrints, working primarily on installer software

August 2001

EPrints 2 Alpha-1 (Anchovy) released.

Jan 2002

   EPrints 2 Alpha-2 (Pepperoni) released.

Feb 14 2002

   EPrints 2.0 (Olive) released.

Apr 17 2002

   EPrints 2.0.1 (Tuna) released. Mostly bugfixes.

July 1 2002

   EPrints offically joins GNU Project.

July 4 2002

   GNU EPrints 2.1 (Pineapple) released. Added subscriptions and OAI 2.0
   support.

October 31 2002

   GNU EPrints 2.2 (Pumpkin) released. Added subject editors and GDOME
   support.

January 12 2004

   GNU EPrints 2.3.0 released.

February 5 2004

   GNU EPrints 2.3.1 released.

February 25 2004

   GNU EPrints 2.3.2 released.

March 4 2004

   GNU EPrints 2.3.3 released.

July 6 2004

   GNU EPrints 2.3.4 released.

August 6 2004

   GNU EPrints 2.3.5 released.

August 9 2004

   GNU EPrints 2.3.6 released.

November 25 2004

   GNU EPrints 2.3.7 released.

Chapter 16

The EPrints Logo

16.1 The EPrints logo

The Folder Jigsaw logo was created by Christopher Gutteridge. It was supposed to encompass "archives","documents", OAI, "skywriting", "modularity/extensibility" and the IAM research groups visual identity.

The base version is a 684x668 XCF file (For the **GIMP Editor**. You are welcome use it and to modify it for your own (EPrints-related) use, maybe have your own logo in the jigsaw.

Figure 16.1: EPrints Logo

Chapter 17

Command Line Tools

17.1 configure_archive command

NAME

configure_archive - create a new EPrint Archive, or edit the basic options of an existing archive

SYNOPSIS

configure_archive [*archiveid*] [**options**]

DESCRIPTION

Create a new EPrint Archive, or edit the basic options of an existing archive.

This script will prompt the user for various basic information about the archive and then will:

```
  - create a basic XML config file
  - make the archive directory
  - copy default files into the archive directory
  - create the MySQL database (but not the tables)
```

After running this script you should edit the files created in the archives **cfg** directory, especially **ArchiveConfig.pm** then you will need to run a number of setup scripts, starting with **create_tables**.

ARGUMENTS

archiveid

The ID of the EPrint archive to modify (or create).

OPTIONS

–help

> Print a brief help message and exit.

–man

> Print the full manual page and then exit.

–quiet

> This option does not do anything.

–verbose

> Explain in detail what is going on. May be repeated for greater effect.

–version

> Output version information and exit.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
 EPrints c/o Christopher Gutteridge
 Department of Electronics and Computer Science
 University of Southampton
 SO17 1BJ
 United Kingdom
```

COPYRIGHT

This file is part of GNU EPrints 2.

Copyright (c) 2000-2004 University of Southampton, UK. SO17 1BJ.

EPrints 2 is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

17.2 create_tables command

NAME

create_tables - create SQL tables for an EPrints archive

SYNOPSIS

create_tables *archiveid* [**options**]

DESCRIPTION

Create the SQL tables which the EPrints software will use.

You should run **import_subjects** after this, actually it will remind you unless you make it **–quiet**.

ARGUMENTS

archiveid

> The ID of the EPrint archive to effect.

OPTIONS

–force

> Go ahead and try and create the tables, even if the script encounters things it thinks are wrong like too many indexed fields or too the database not being empty.

–help

> Print a brief help message and exit.

–man

> Print the full manual page and then exit.

–quiet

> Be vewwy vewwy quiet. This option will supress all output unless an error occurs.

–verbose

> Explain in detail what is going on. May be repeated for greater effect.

–version

> Output version information and exit.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
EPrints c/o Christopher Gutteridge
Department of Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
```

COPYRIGHT

This file is part of GNU EPrints 2.

Copyright (c) 2000-2004 University of Southampton, UK. SO17 1BJ.

EPrints 2 is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

EPrints 2 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with EPrints 2; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

17.3 create_user command

NAME

create_user - add a new user to an EPrint archive

SYNOPSIS

create_user *archiveid* [**options**] *username email usertype* [*password*]

DESCRIPTION

Create a new user to an eprint archive.

This is handy for making the initial admin user at the very least.

ARGUMENTS

archiveid

> The ID of the EPrint archive to add a user to.

username

> The requested username for the new user. If a user with this name already exists then the script will abort with a error.

email

> The email address of the new user.

usertype

> The type of the new user. The type of a user sets how much they can do in the system. The default EPrints configuration provides 3 types of users:
>
> *user*
>
> > Normal everyday joe public users.
>
> *editor*
>
> > Editors may approve eprints for addition, browse the submission buffer, and check the archive status.
>
> *admin*
>
> > Administrators may do everything. That's probably what for yourself if you are the person setting up the archive.

password

> The initial password for this user. You don't have to specify it here if you don't want to. It will be encrypted, by EPrints, using UNIX crypt. This should not be a problem unless you are using a different method to authenticate users.

archiveid

>   The ID of the EPrint archive to add a user to.

OPTIONS

–help

>   Print a brief help message and exit.

–man

>   Print the full manual page and then exit.

–quiet

>   Be vewwy vewwy quiet. This option will supress all output unless an error
>   occurs.

–verbose

>   Explain in detail what is going on. May be repeated for greater effect.

–version

>   Output version information and exit.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher
Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information
on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be con-
tacted in the real world at

```
 EPrints c/o Christopher Gutteridge
 Department of Electronics and Computer Science
 University of Southampton
 SO17 1BJ
 United Kingdom
```

COPYRIGHT

This file is part of GNU EPrints 2.

Copyright (c) 2000-2004 University of Southampton, UK. SO17 1BJ.

EPrints 2 is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

EPrints 2 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABIL-ITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with EPrints 2; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

17.4 erase_archive command

NAME

erase_archive - Erase all database tables and files associated with an EPrint archive

SYNOPSIS

erase_archive *archiveid* [**options**]

DESCRIPTION

This script completely erases the archive contents, including all database tables and eprint document files and the web site. After running this, the metadata configuration can be safely updated and the creation scripts run again.

Without the −**force** option, this script asks for confirmation before actually erasing anything.

After this script you will need to run **create_tables** before you can use the archive.

ARGUMENTS

archiveid

The ID of the eprint archive to use.

OPTIONS

−**help**

Print a brief help message and exit.

–man

> Print the full manual page and then exit.

–quiet

> Be vewwy vewwy quiet. This option will supress all output unless an error occurs.

–verbose

> Explain in detail what is going on. May be repeated for greater effect.

–version

> Output version information and exit.

–force

> Don't ask before making the changes.

–noerasedb

> Don't delete the database tables.

–noerasefiles

> Don't erase the document and website files.

–rootpass *password*

> This option allows you to specify the MySQL root password, which will skip the bit where you get asked for it. It's not very secure to put a root password on the command line. Consider yourself warned...

BUGS

This script assumes that the root mysql user is called root.

You may run into problems if your mysql install is not on the same machine as EPrints. If this happens use **–noerasefiles** and delete and recreate the database yourself.

You can specify **–noerasedb** and **–noerasefiles** at the same time, but that's really stupid, as then nothing will happen.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
EPrints c/o Christopher Gutteridge
Department of Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
```

COPYRIGHT

17.5 erase_fulltext_cache command

NAME

erase_fulltext_cache - erase the cached full text search index files.

SYNOPSIS

erase_fulltext_cache *archiveid* [**options**]

DESCRIPTION

This erases all the .words and .indexcodes cache files from your archive, forcing the indexer to rerun the tools used to extract full text from your documents.

This is useful if you only setup the fulltext indexing after your archive is already live, or if you discover there has been a problem.

Without the –**force** option, this script asks for confirmation before actually erasing anything.

ARGUMENTS

archiveid

    The ID of the EPrint archive to use.

OPTIONS

–help

    Print a brief help message and exit.

–man

    Print the full manual page and then exit.

–quiet

    Be vewwy vewwy quiet. This option will supress all output unless an error occurs.

–verbose

    Explain in detail what is going on. May be repeated for greater effect.

–version

    Output version information and exit.

–force

    Don't ask before making the changes.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

    Chris Gutteridge may be contacted at **support@eprints.org**

    Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
EPrints c/o Christopher Gutteridge
Department of Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
```

COPYRIGHT

17.6 explain_sql_tables command

NAME

explain_sql_tables - explain what the tables in the SQL database mean

SYNOPSIS

explain_sql_tables *archiveid* [**options**]

DESCRIPTION

This script describes what the tables in the SQL archive are used for.

ARGUMENTS

archiveid

> The ID of the EPrint archive to use.

OPTIONS

–help

> Print a brief help message and exit.

–man

> Print the full manual page and then exit.

–quiet

> Be vewwy vewwy quiet. This option will supress all output unless an error occurs.

–verbose

> Explain in detail what is going on. May be repeated for greater effect.

–version

> Output version information and exit.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
EPrints c/o Christopher Gutteridge
Department of Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
```

COPYRIGHT

This file is part of GNU EPrints 2.

Copyright (c) 2000-2004 University of Southampton, UK. SO17 1BJ.

EPrints 2 is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

EPrints 2 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with EPrints 2; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

17.7 export_hashes command

NAME

export_hashes - export the hash values for all documents in an archive.

SYNOPSIS

export_hashes *archiveid* [**options**] [*filename*]

DESCRIPTION

Every time a document in eprints is modified a new .xsh file is generated containing a hash of each file. This script creates a hash of each of these .xsh files and create a super .xsh file containing each of those hashes.

If no filename is given this script outputs to standard out.

The XML file produced may then be archived safely. You can then create an MD5 of that file and do something to prove you had it on the date you created it.

For example, publish it in an small-ad in a national paper.

Now you can prove you had that MD5 on that day, which proves you had the file output by export_hashes that day. The MD5's in that file prove that you had the .xsh file of a given document. Those files should prove that you had a given file.

This all assumes that nobody works out a way to do MD5's in reverse. And there's no legal precident yet.

That's why this is an experimental feature.

ARGUMENTS

archiveid

The ID of the EPrint archive to use.

filename

A filename to write to. If omited this will write to stdout.

OPTIONS

–help

Print a brief help message and exit.

–man

Print the full manual page and then exit.

–quiet

Be vewwy vewwy quiet. This option will supress all output unless an error occurs.

–verbose

> Explain in detail what is going on. May be repeated for greater effect.

–version

> Output version information and exit.

–all

> Output hashes of ALL .xsh files for each document, not just the most recent. This takes longer but should be logged periodically.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
EPrints c/o Christopher Gutteridge
Department of Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
```

COPYRIGHT

This file is part of GNU EPrints 2.

Copyright (c) 2000-2004 University of Southampton, UK. SO17 1BJ.

EPrints 2 is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

EPrints 2 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with EPrints 2; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

17.8 export_xml command

NAME

export_xml - export a dataset as XML.

SYNOPSIS

export_xml *archiveid* [**options**] [*dataset*]

DESCRIPTION

This command outputs the named dataset as XML.

ARGUMENTS

archiveid

> The ID of the EPrint archive to use.

dataset

> The name of the dataset to export, such as "archive", "subject" or "user".

OPTIONS

–**xmlversion** *version*

> The version of the eprints data XML format to use. Either "1" or "1.1".
> Default is (currently) "1". 1.1 is still under development and may change.
> We strongly recommend not using 1.1 until we finalise it in a later version
> of EPrints.

–**help**

> Print a brief help message and exit.

–**man**

> Print the full manual page and then exit.

–**quiet**

> Be vewwy vewwy quiet. This option will supress all output unless an error
> occurs.

–**verbose**

> Explain in detail what is going on. May be repeated for greater effect.

–**version**

> Output version information and exit.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
EPrints c/o Christopher Gutteridge
Department of Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
```

COPYRIGHT

This file is part of GNU EPrints 2.

Copyright (c) 2000-2004 University of Southampton, UK. SO17 1BJ.

EPrints 2 is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

EPrints 2 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with EPrints 2; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

17.9 force_config_reload command

NAME

force_config_reload - Force the webserver to reload the configuration of this archive.

SYNOPSIS

force_config_reload *archiveid* [**options**]

DESCRIPTION

This command forces the webserver to reload the config files for the given archive.

This is very useful, but the webserver will only reload the config files in the "forked" versions of itself, but the origional. This will generate a major extra load each time the webserver forks of a process.

This command offers a **temporary** solution to the problem only. You should still restart apache at some point.

All this file really does is create an empty file named .changed in the archives own var/ directory. The main system, running in the webservers mod_perl, looks at the time this file was last modified. If this file has changed since the archive configuration was loaded then it forces a reload.

ARGUMENTS

archiveid

The ID of the eprint archive to force a reload on.

OPTIONS

–help

Print a brief help message and exit.

–man

Print the full manual page and then exit.

–quiet

Be vewwy vewwy quiet. This option will supress all output unless an error occurs.

–verbose

Explain in detail what is going on. May be repeated for greater effect.

–version

Output version information and exit.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
EPrints c/o Christopher Gutteridge
Department of Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
```

COPYRIGHT

This file is part of GNU EPrints 2.

Copyright (c) 2000-2004 University of Southampton, UK. SO17 1BJ.

EPrints 2 is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

EPrints 2 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with EPrints 2; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

17.10   generate_abstracts command

NAME

generate_abstracts - Regenerate all the static abstract pages for an EPrint archive

SYNOPSIS

generate_abstracts *archiveid* [**options**] [*eprintid*]

DESCRIPTION

This script recreates every static abstract page for an eprints archive. To save load on the database, as archived data should not change, EPrints creates static webpages containing the summary of each eprint. If you change the way the abstracts are rendered or change the site template then you will want to run this script.

ARGUMENTS

archiveid

> The ID of the eprint archive to use.

eprintid

> An optional integer indicating that only the abstract page for record *eprintid* should be updated. Handy for testing new configurations.

OPTIONS

–help

> Print a brief help message and exit.

–man

> Print the full manual page and then exit.

–quiet

> Be vewwy vewwy quiet. This option will supress all output unless an error occurs.

–verbose

> Explain in detail what is going on. May be repeated for greater effect.

–version

> Output version information and exit.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
EPrints c/o Christopher Gutteridge
Department of Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
```

COPYRIGHT

This file is part of GNU EPrints 2.

Copyright (c) 2000-2004 University of Southampton, UK. SO17 1BJ.

EPrints 2 is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

EPrints 2 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABIL-ITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with EPrints 2; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

17.11   generate_apacheconf command

NAME

generate_apacheconf - Create the apache config files needed for EPrints

SYNOPSIS

generate_apacheconf [**options**]

DESCRIPTION

This script generates the apache config files which will be used by EPrints. In the simple case all you need to do is run this script then add a line to your main apache configuration file - often, but not always, **/usr/local/apache/conf/httpd.conf**

```
Include /opt/eprints2/cfg/apache.conf
```

Or elsewhere if you installed EPrints somewhere other than /opt/eprints2. This file then uses the "Include" directive to include all relevant apache config files from this EPrints installation.

By default the virtualhost directives are

```
VirtualHost *
```

But the * can be changed to something different by editing the `virtualhost` option in SystemSettings.pm

ARGUMENTS

OPTIONS

–help

> Print a brief help message and exit.

–man

> Print the full manual page and then exit.

–quiet

> Be vewwy vewwy quiet. This option will supress all output unless an error occurs.

–verbose

> Explain in detail what is going on. May be repeated for greater effect.

–version

> Output version information and exit.

FILES

EPRINTS/cfg/apache.conf

> This file is not updated if it already exists, so you can add system-wide apache configuration directives here. By default it just includes the two other system wide files: auto-apache-includes.conf and auto-apache.conf

EPRINTS/cfg/auto-apache-includes.conf

> This file is updated with Include lines to each of the archive specific apache config files. This file should not be edited by hand,

EPRINTS/cfg/auto-apache.conf

> This file contains the system wide apache directives required by EPrints. This file should not be edited by hand.

EPRINTS/archives/ARCHIVEDIR/cfg/apache.conf

> This file is not updated if it already exists, so you can add archive-specific apache configuration directives here. By default it just includes the automatically generated archive specific file: auto-apache.conf

EPRINTS/archives/ARCHIVEDIR/cfg/auto-apache.conf

> This file contains all the configuration directives needed for an archive. This is where the bulk of the configuration appears, the clever stuff, if you will. This file should not be edited by hand.

EPRINTS/archives/ARCHIVEDIR/cfg/auto-secure.conf

> Only created if the archive has a securehost configured. This file contains all the configuration options that need to be included into the secure server virtualhost.

EPRINTS/archives/ARCHIVEDIR/cfg/apachevhost.conf

> This file is not updated if it already exists, it is included into the virutalhost in auto-apache.conf so that you can a couple of additional directives if you need to. For example, redirects or additional log directives.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
EPrints c/o Christopher Gutteridge
Department of Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
```

COPYRIGHT

17.12   generate_static command

NAME

generate_static - Generate static pages of an EPrint archive using the template.

SYNOPSIS

generate_static *archiveid* [**options**]

DESCRIPTION

This script creates the static web site for EPrints (or, if you are running in multiple lanugages it generates the websites).

It processes every file in **EPRINTS/archives/ARCHIVE/cfg/static/LANGID/**. For each language processes all the files in /LANGID/ and /generic/ into **EPRINTS/archives/ARCHIVE/html/LANGID**. If that sounds confusing, don't worry, it's not that bad, just put your webpage outlines in static/en/ and your image files and the like in static/generic/ and run this script and see what happens.

Most files are copied into the target directory as is and directory structure is preserved.

Files with a .xpage or .xhtml suffix are processed as they are copied.

.xpage

This is an XML file with the following structure:

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE page SYSTEM "entities-en.dtd" >
<page>
  <title>This is the page title</title>
  <body>
    <p>Some XHTML body</p><p>Which is <b>neat</b></p>
  </body>
</page>
```

The resulting file will be a .html file (foo.xpage becomes foo.html). It will take the template for this archive and insert the title and body from the appropriate places. It will also cause the the special EPrints entities to be converted as it is copied. See the main documentation.

.xhtml

This is a normal XHTML file but with the following XML header:

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE html SYSTEM "entities-en.dtd" >
```

This will cause the the special EPrints entities to be converted as it is copied. See the main documentation for more about these entities. It will also be renamed to .html for example, foo.xhtml will become foo.html

NOTE FOR THE NON-ENGLISH MAJORITY

If you are running EPrints in a language other than English then place the static files in a directory of your ISO language ID instead of **en**, for example French is **fr**. The generic directory is for language neutral stuff. Which is extra handy if you want to run the site in more than one language. Also the entities file should be renamed from -en to -whatever eg. **entities-fr.xml**.

ARGUMENTS

archiveid

>   The ID of the eprint archive to use.

OPTIONS

–help

>   Print a brief help message and exit.

–man

>   Print the full manual page and then exit.

–quiet

>   Be vewwy vewwy quiet. This option will supress all output unless an error
>   occurs.

–verbose

>   Explain in detail what is going on. May be repeated for greater effect.

–version

>   Output version information and exit.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher
Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information
on mailing lists and the like.

>   Chris Gutteridge may be contacted at **support@eprints.org**

>   Should you need a real world address for some reason, EPrints can be con-
tacted in the real world at

```
 EPrints c/o Christopher Gutteridge
 Department of Electronics and Computer Science
 University of Southampton
 SO17 1BJ
 United Kingdom
```

COPYRIGHT

This file is part of GNU EPrints 2.

Copyright (c) 2000-2004 University of Southampton, UK. SO17 1BJ.

EPrints 2 is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

EPrints 2 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABIL-ITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with EPrints 2; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

17.13 generate_views command

NAME

generate_views - Generate static browse pages for an EPrint archive

SYNOPSIS

generate_views *archiveid* [**options**]

DESCRIPTION

This script renders static "browse views" for an EPrint Archive.

What this does is generate browse pages for each field configured as browsable in **ArchiveConfig.pm**. It creates a static web page for each value of that field, and index pages to navigate to them.

For example, if we make "year" browseable then this script will generate one page for each unique value of the year field. So a user can then view the 1995 page and see links to all the 1995 eprints.

Advantages of this are that this puts less load on the database than user searches. Assuming you pick two or three sensible fields to make browsable.

This script should be run every hour or so, but that should once a day or even once a week on large archives, as the more eprints the longer it will take to run. The rough length of time to run this is of the order of O(**languages * eprints * browsable fields**). You can automate running this with the **cron** system.

ARGUMENTS

archiveid

The ID of the eprint archive to use.

OPTIONS

–help

> Print a brief help message and exit.

–man

> Print the full manual page and then exit.

–quiet

> Be vewwy vewwy quiet. This option will supress all output unless an error occurs.

–verbose

> Explain in detail what is going on. May be repeated for greater effect.

–version

> Output version information and exit.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
EPrints c/o Christopher Gutteridge
Department of Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
```

COPYRIGHT

This file is part of GNU EPrints 2.

Copyright (c) 2000-2004 University of Southampton, UK. SO17 1BJ.

EPrints 2 is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

EPrints 2 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with EPrints 2; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

17.14   import_eprints command

17.15   import_subjects command

NAME

import_subjects - rebuild an EPrint archive subjects list from the contents of a file

SYNOPSIS

import_subjects *archiveid* [**options**] [*subjectfile*]

DESCRIPTION

Import a set of subjects into an EPrints archive. The subjects are the heirarchical tree of options for "subject" type metadata fields in an eprint archive.

Use the staff admin subject editor for little tweaks. Use this command for the initial setup or bulk editing subjects. Use the exporter to dump the current subjects if you (an administrator) have edited them online.

This script should also be run after **create_tables**.

Without the –**force** option, this script asks for confirmation before actually erasing anything.

ARGUMENTS

archiveid

> The ID of the EPrint archive to use.

subjectfile

> This is the file to import the subjects from. If you do not specify it then the system will use "subjects" from the given archives cfg directory (or "subjects.xml" if you used –**xml**).

OPTIONS

–help

> Print a brief help message and exit.

–man

> Print the full manual page and then exit.

–quiet

> Be vewwy vewwy quiet. This option will supress all output unless an error occurs.

–verbose

> Explain in detail what is going on. May be repeated for greater effect.

–version

> Output version information and exit.

–xml

> Instead of using colon separted subject file, the input file is in the XML format which the exporter uses.

–nopurge

> Do not purge the existing records from the subject table before importing this file. Rather than do this, it's probably easier to export the current subjects as XML, then combine in your new file and reimport it.

–force

> Don't ask before making the changes.

FILE FORMAT

There are two different file formats excepted, the default colon seperated file and XML in the eprints export format.

The colon seperated ASCII is easier to edit, but is more limited. It is not intended for UTF-8 encoded characters and can only specify subject names in the default language.

The XML format can contain any unicode characters and also allows multiple languages for the names of subjects. You may wish to dump the current subjects out of eprints as XML. Edit it. Then re-import it. The downside is that this format is far more verbose.

The ASCII Default Format   This is the default format.
Comments may be placed on lines begining with a hash (#)
Each (non-comment) line of the file should be in the following format:
subjectid:*name*:*parents*:*depositable*
eg.
blgc-bphy:Biophysics:blgc,phys:1
Please see the main documentation for the meaning of these fields.

subjectid

An ASCII string which is a unique ID for this subject.

name

The name of this subject, in the default language of the archive.

parents

A comma seperated list of the parents of this subject. Be careful not to
cause loops. The top level subject id is ROOT and should not be imported
as it always exists.

depositable

A boolean value (1 or 0) which indicates if this subject may have eprints
associated with it.

The XML File Format   This is the standard eprints export format. It looks
like this:

```
<eprintsdata>
  <record>
    <field name="subjectid">phys</field>
    <field name="name"><lang id="en">Physical Sciences &amp; Mathematics</lang></field>
    <field name="parents">subjects</field>
    <field name="depositable">FALSE</field>
  </record>
  .
  .
  .
</eprintsdata>
```

The fields have the same meaning as described for the ASCII format, with
the following variations. The name field can (and should) have a name for each
language supported by the archive. Multiple parents are indicated by multiple
<ltfield name="parents<gt>> elements. Depositable should be either TRUE
or FALSE.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher
Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
EPrints c/o Christopher Gutteridge
Department of Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
```

COPYRIGHT

17.16   indexer command

NAME

indexer - Indexing daemon for EPrints

SYNOPSIS

indexer start [**options**]
   indexer stop
   indexer status

DESCRIPTION

This daemon runs in the background and creates index files for all eprints archives. It builds indexes for each archive in turn then starts again.

Messages and errors are logged to /opt/eprints2/var/indexer.log unless you change the log options. If it appears to be having problems try raising the log level and examining the log.

Once all the archives have been indexed, the indexer rolls the logs (up to logfile.5) and then starts again. See –rollcount for ways to customise this.

OPTIONS

–help

> Print a brief help message and exit.

–man

> Print the full manual page and then exit.

–quiet

> Be vewwy vewwy quiet. This option will supress all output unless an error occurs.

–force

> Start up, even if the PID file exists (implying another copy is running). This is useful for starting after a crash, but be carefully not to run to copies at once as BAD THINGS will happen.

–verbose

> Explain in detail what is going on. May be repeated for greater effect.

–logfile *filename*

> Log to *filename* rather than default indexer log.

–loglevel *level*

> Set the level of detail to log. Level may be 0-5.
>
>   1. Do not log anything.
>   2. Report start, stop and errors.
>   default All above, plus report names of archives being indexed and warnings.
>   3. All above, plus report processes starting and stopping and each dataset being indexed.
>   4. All above, plus report details of index process.
>   5. All above, plus report id of every item indexed.

–rollcount *number*

> Set the number of once-through logs that should be kept. If set to zero then indexer will never roll the logs but rather just keep writing to the main log.

–notdaemon

> Do not become a daemon, remain attached to the current terminal (errors still go to the log file).

–once

> Only index each archive once rather than loop for ever.

–version

> Output version information and exit.

Making into a service

This has only been tested under redhat linux. It make work on other OS's, but not promise.

To make the indexer into a service which starts and stops on reboots etc. like httpd and mysqld do the following (as root):

```
ln -s /opt/eprints2/bin/epindexer /etc/init.d/epindexer
chkconfig --add epindexer
chkconfig epindexer on
```

The epindexer script runs as root, changes user to "eprints" (or whatever uid your eprints install runs as) and then calls indexer.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
EPrints c/o Christopher Gutteridge
Department of Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
```

COPYRIGHT

17.17 list_user_emails command

17.18 rehash_documents command

NAME

rehash_documents - Regenerate all the hashes of all documents in the archive.

SYNOPSIS

rehash_documents *archiveid* [**options**] [*documentid*]

DESCRIPTION

This command regenerates the hash values for all documents. This will, by default, create an MD5 hash of all the files in the document in alphabetic order of their path. This value is then stored as a string of hexidecimal characters in the metadata for the document.

This value is automatically re-calculated if more files are uploaded or any files are removed.

It is useful as a quick reliable way of seeing if the document contents has changed. If you generate other formats of a document on-the-fly the MD5 can be useful as part of caching, or you may wish to pass the hashes to a third party to allow you to later verify that an item was in the archive since a given date.

archiveid

>   The ID of the eprint archive to rehash the documents of.

OPTIONS

–help

>   Print a brief help message and exit.

–man

>   Print the full manual page and then exit.

–quiet

>   Be vewwy vewwy quiet. This option will supress all output unless an error
>   occurs.

–verbose

>   Explain in detail what is going on. May be repeated for greater effect.

–version

>   Output version information and exit.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher
Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information
on mailing lists and the like.

   Chris Gutteridge may be contacted at **support@eprints.org**

   Should you need a real world address for some reason, EPrints can be con-
tacted in the real world at

```
 EPrints c/o Christopher Gutteridge
 Department of Electronics and Computer Science
 University of Southampton
 SO17 1BJ
 United Kingdom
```

COPYRIGHT

This file is part of GNU EPrints 2.

Copyright (c) 2000-2004 University of Southampton, UK. SO17 1BJ.

EPrints 2 is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

EPrints 2 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with EPrints 2; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

17.19   send_subscriptions command

NAME

send_subscriptions - Send out all the email subscriptions for a given frequency.

SYNOPSIS

generate_views *archiveid frequency* [**options**]

DESCRIPTION

This script sends out all the subscription emails for the specified archive and frequency. frequency must be one of daily|weekly|monthly.

This script should probably be called from your "cron" system, soon after midnight. Something like:

```
# 00:15 every morning
15 0 * * * /opt/eprints2/bin/send_subscriptions dookuprints daily
# 00:30 every sunday morning
30 0 * * 0 /opt/eprints2/bin/send_subscriptions dookuprints weekly
# 00:45 every first of the month
45 0 1 * * /opt/eprints2/bin/send_subscriptions dookuprints monthly
```

Note the spacing out so that all 3 don't start at once and hammer the database.

ARGUMENTS

archiveid

The ID of the eprint archive to use.

frequency

> Which "frequency" of subscriptions to send - the daily, weekly or monthly ones.

OPTIONS

–help

> Print a brief help message and exit.

–man

> Print the full manual page and then exit.

–quiet

> Be vewwy vewwy quiet. This option will supress all output unless an error occurs.

–verbose

> Explain in detail what is going on. May be repeated for greater effect.

–version

> Output version information and exit.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
 EPrints c/o Christopher Gutteridge
 Department of Electronics and Computer Science
 University of Southampton
 SO17 1BJ
 United Kingdom
```

COPYRIGHT

17.20   upgrade command

NAME

upgrade - Upgrade the stucture of the database to be compatable with a newer version of eprints.

If upgrading requires more than one step then the system will perform the upgrade step by step. If a step fails then you can continue from where it left off.

SYNOPSIS

upgrade *archiveid* [**options**]

DESCRIPTION

Some versions of eprints require modifications to be made to the database tables used by earlier versions.

Run this script on each archive after upgrading the eprints software.

ARGUMENTS

archiveid

The ID of the EPrint archive to effect.

OPTIONS

–help

Print a brief help message and exit.

–man

Print the full manual page and then exit.

–quiet

> This option doesn't do anything. You REALLY don't want to run this script without knowing what's happening.

–verbose

> Explain in detail what is going on. May be repeated for greater effect.

–version

> Output version information and exit.

AUTHOR

This is part of this EPrints 2 system. EPrints 2 is developed by Christopher Gutteridge.

VERSION

EPrints Version: 2.3.11-99-1-beta

CONTACT

For more information goto **http://www.eprints.org/** which give information on mailing lists and the like.

Chris Gutteridge may be contacted at **support@eprints.org**

Should you need a real world address for some reason, EPrints can be contacted in the real world at

```
EPrints c/o Christopher Gutteridge
Department of Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
```

COPYRIGHT

This file is part of GNU EPrints 2.

Copyright (c) 2000-2004 University of Southampton, UK. SO17 1BJ.

EPrints 2 is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

EPrints 2 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with EPrints 2; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA