phpaga Documentation

Release 0.5

Florian Lanthaler

April 06, 2010

CONTENTS

Contents:

CHAPTER

ONE

INSTALLATION

Installing phpaga is not a fully automated process, there are a few simple steps that you must go through. If you are upgrading from a previous version you might be interested in the chapter *Upgrade*.

For a fresh install you need to do the following steps:

- Copy phpaga files to the destination folder(s).
- Install the requirements.
- Create a database and grant the necessary rights to the phpaga user.
- Set some base configuration parameters.
- Point your browser to phpaga - an installer will create the necessary tables and populate them with the core data.

This document will guide you through the installation and give you some help. If you don't know how to install PHP or how to compile it with support for the packages listed under "requirements", then refer to the PHP website and the official PHP Documentation.

1.1 Requirements

The following software packages have to be installed on your server in order to use phpaga. The version numbers mentioned below are known to work. Refer to each package's website for installation instructions.

1.1.1 RDBMS

Currently PostgreSQL (>= 7.4) and MySQL (>= 4.1.16) are supported. If you are unsure about which to pick go for PostgreSQL - phpaga is developed and tested under PostgreSQL, and database-related parts are better tested with that platform.

1.1.2 Web server

phpaga should work with any PHP-enabled web server. The development environment is running nginx with PHP as FastCGI. phpaga has also been tested with Apache.

1.1.3 PHP with PDO

>= 5.2.0

PHP with support for PDO and the PDO driver(s) for PostgreSQL (or MySQL), gettext and gd. Visit http://www.php.net/ for more information on how to compile and install PHP, and http://www.php.net/gettext for information about PHP's gettext support. For the various supported languages to actually work, make sure you

have the corresponding locales installed. Keep in mind that after installing a new locale you will need to restart the web server.

The `php.ini` settings `register_globals` and `magic_quotes_gpc` can both be `Off`. In case `magic_quotes_gpc` is `On` all escaping slashes in `$_REQUEST`, `$_GET` and `$_POST` will automatically be removed by phpaga.

The amount of memory available to PHP is defined by the memory_limit setting in `php.ini`. This value defaults to `8M`, but you might need to raise it to `16M` or `32M` if you encounter error messages like the following:

```
"Fatal error: Allowed memory size of 8388608 bytes exhausted (tried to
allocate 65211 bytes) in <some filename> on line <some line number>".
```

`etc/config.local.php` also provides the possibility to change this setting.

1.1.4 Graphviz

This package is optional. If you would like to see project relationship graphs you will need to install Graphviz and set the related option "Enable Graphviz" under the sitewide configuration to "Yes". Note that phpaga will work nicely without this package (provided you set the option "Enable Graphviz" to "No") - you will just be missing the relationship graphs.

1.1.5 ImageMagick

ImageMagick's `convert` is used to create preview thumbnails for uploaded files.

1.2 Installation

1.2.1 Installation layout

Only phpaga's `htdocs/` directory needs to be read by the web server. All other directories can and should be outside the web server's document root. This can be achieved either by creating a virtual host and have its document root point to the full path to `htdocs/`, or by installing phpaga completely outside the document root of an existing website and then adding an alias to the web server's configuration. This can be done under Apache with the following example:

```
Alias /phpaga /usr/local/phpaga/htdocs/
```

If you can't modify the web server's settings (for example because you are using the services of a webhosting company) you should use the following strategy:

Say your home directory is `/home/exampleuser`, and all your website content is stored under `/home/exampleuser/www/`, and you have unpacked the phpaga files to `/home/exampleuser/phpaga/`. Do this:

```
mv /home/exampleuser/phpaga/htdocs/* /home/exampleuser/www/
mv /home/exampleuser/phpaga/htdocs/.htaccess /home/exampleuser/www/
```

edit `/home/exampleuser/www/config.php` to point to `/home/exampleuser/phpaga/etc/config.php`, i.e.

```
require_once '../phpaga/etc/config.php';
```

If you want the phpaga setup to be accessible using a subfolder of your domain, i.e. www.example.com/phpaga, then do this instead:

```
mv /home/exampleuser/phpaga/htdocs /home/exampleuser/www/phpaga
```

edit `/home/exampleuser/www/phpaga/config.php` to point to `/home/exampleuser/phpaga/etc/config.php`
i.e.

```
require_once '../../phpaga/etc/config.php';
```

1.2.2 The database

Create a database and a database user with the necessary permissions and set the appropriate configuration settings in the file `etc/config.local.php` accordingly. Make sure the database is able to hold unicode/utf-8. We will name both the database and the user "phpaga" in the examples provided below.

If you are using phpaga with a webhosting provider, chances are that the provider will create the database for you and give you the credentials.

If you are upgrading from a previous version of phpaga refer to the section *Upgrade*. For information about converting from MySQL to PostgreSQL read the section *Migrating phpaga from MySql to PostgreSQL*.

For PostgreSQL

Note: If you are using PostgreSQL 8.x, use 'UTF8' as the encoding of the database. For PostgreSQL 7.x, use the encoding 'UNICODE'.

```
psql template1
template1> create database phpaga with encoding 'UTF8';
template1> create user phpaga password 'yourpasswd';
template1> grant all on database phpaga to phpaga;
```

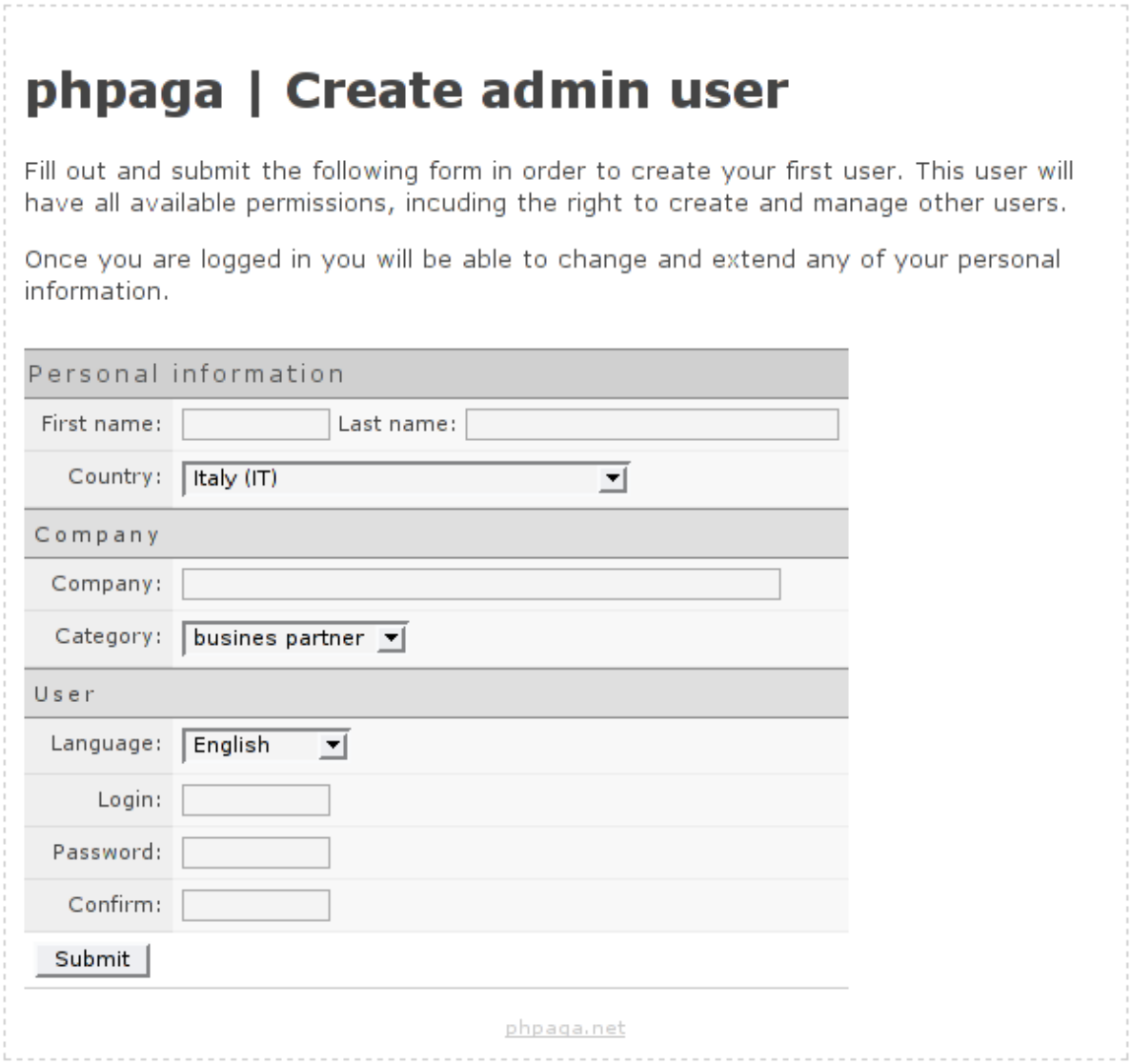
Depending on your setup, you may also need to modify `pg_hba.conf`. This file is part of PostgreSQL and controls which hosts are allowed to connect, how clients are authenticated, which PostgreSQL user names they can use, which databases they can access. Refer to the PostgreSQL documentation if you are unsure about the proper settings.

For MySQL

```
mysql -uroot -p mysql
mysql> create database phpaga default character set utf8 collate utf8_general_ci;
mysql> grant all on phpaga.* to phpaga@localhost identified by 'yourpasswd';
mysql> flush privileges;
```

1.3 Configuration

Fire up your browser of choice and point it to the right URL. A simple wizard will guide you and help you to create the database structure, populate it with core data, and create the first user.



Insert the data as required and submit the form. The user will be created with all available permissions, and you are invited to log in.

In case you have any problems accessing phpaga, try to open test.php in your browser and check its output for helpful information.

PLEASE NOTE: If you can log in and everything is running smoothly, **remove** `htdocs/test.php` (or chmod it so that the webserver does not have access to it) - it might expose critical information.

Once you are logged in you can configure the default settings by selecting "Sitewide settings" from the admin menu.

From there you have the option to manage your categorie sections (task/project/job and other categories) from the admin menu, before you (or your users) start to add tasks.

Only the JPG format will work for the graphic logo that can be included in the PDF output, anything else will cause the output to be invalid.

1.4 Development version

The following information is not needed if you install phpaga from an official (release) package.

If you fetched the source directly from phpaga's subversion repository you will need to install a few extra libraries (they come already bundled with official phpaga packages) and perform some additional installation steps.

1.4.1 Additional requirements

Smarty

phpaga uses Smarty as its templating engine.

R&OS pdf class

009e

The R&OS pdf class is a PHP class that allows the easy creation of PDF files. Get the file named pdfClassesAndFonts_009e.zip which contains the class, sample, and the required font metric files.

smarty-gettext

1.0b1

smarty-gettext is a Smarty plugin written by Sagi Bashari that allows to use gettext with Smarty templates.

1.4.2 Additional installation steps

If you fetched the source directly from phpaga's subversion repository you will need to perform the following steps. If you are installing an official package you can skip this section.

- Use the sample file `etc/config.local.php_sample` to create `etc/config.local.php` and fill in the necessary parameters.
- Give the web server full permissions to the `files/` directory. This directory will contain the files users can upload and associate to objects (project, person, company, ...).

  ```
  chown www-data files/
  chmod 0700 files/
  ```

- If you do not have administrative rights on the server and therefore cannot change the directory's ownership, simply chmod 0777 the directory.

  ```
  chmod 0777 files/
  ```

- Make sure your phpaga directory contains a templates_c directory which the web server can write into. If you have root access:

  ```
  cd /data/webs/phpaga/
  mkdir -p templates_c
  chown www-data templates_c
  chmod 700 templates_c
  ```

  or, if you are using shared hosting:

  ```
  cd /home/exampleuser/phpaga/
  mkdir -p templates_c
  chmod 777 templates_c
  ```

- Install required 3rd party packages

- If you have Smarty already installed on your server you can use it by editing the definition of SMARTY_DIR in `etc/config.local.php` accordingly.

  ```
  define("SMARTY_DIR", "/some/other/path/to/smarty/libs/");
  ```

  Otherwise install Smarty into `ext/` (no need to change `etc/config.local.php`).

  ```
  [ cd phpaga/ext ]
  tar zxf Smarty-x.x.x.tar.gz
  mv Smarty-x.x.x smarty
  ```

- Install the R&OS pdf class

  ```
  [ cd ext/ ]
  mkdir pdfclass
  cd pdfclass
  unzip pdfClassesAndFonts_009e.zip
  ```

- Install smarty-gettext

  ```
  [ cd ext/ ]
  tar xvzf smarty-gettext-1.0b1.tgz
  mv smarty-gettext-1.0b1 smarty-gettext
  ```

CHAPTER

TWO

UPGRADE

2.1 In general

CREATE A BACKUP OF YOUR DATABASE BEFORE PERFORMING ANY UPGRADE.

You are advised to back up your data before performing any upgrade.

Check whether the software requirements have changed at the section *Requirements*.

Since after 0.3, phpaga features a web interface for database upgrades which can be opened at "System / Database upgrades". Once you have installed the new release, open this page and follow the instructions shown there.

If you need additional help consider joining the user's mailing list.

2.2 Upgrading from 0.4 to 0.5

2.2.1 Invoices and payments

Support for payments on unpaid invoices/bills has been added. During the database upgrade from 0.4 to 0.5, all existing invoices will be automatically migrated.

If you have written your own billing plugin make sure to update it to make use of the payments system. Use one of the billing plugins that ship with phpaga to see what changes are necessary.

2.2.2 eZ Components / jqPlot

ezComponents are no longer used; graphs are now created with jqPlot, a plotting and charting plugin for the jQuery Javascript framework. phpaga comes with jqPlot (and jQuery) included, so no extra installation is required.

2.3 Upgrading from 0.3 to 0.4

2.3.1 JpGraph / eZ Components

JpGraph is no longer used; graphs are now created with ezComponents. See above about eZ Components installation.

2.3.2 PEAR replaced

PEAR is no longer required. All dependencies on PEAR packages have been replaced by phpaga's own code.

2.3.3 Billing plugins

Filed (tracked) expenses can now be added to invoices. If you have written your own billing plugin you need to apply a few little changes. First, and most important, a new parameter needs to be added to the function parameter. Then, the invoice calculation details need to be extended. Also, billing plugins are no longer supposed to return an error code. Refer to one of the billing plugins that come shipped with phpaga to see what needs to be changed.

2.3.4 File upload and pictures

Files can now be associated to persons. The script `tools/migrate_personpictures.php` can be used to migrate the existing pictures (from persons.pe_fotoname to files).

2.3.5 UTF-8: Locales and database

UTF-8 is now used as the encoding for all supported locales. Unfortunately, in previous releases phpaga used to store data in a different format (depending on your locale, most likely LATIN1). Therefore you need to convert the database to UFT-8 before using phpaga 0.4. You will also need to make sure that the desired locales are available in UTF-8.

If you have been storing data with different encodings in the same database, the conversion will most likely produce garbled data, no matter what database system you are using.

iconv is a tool to convert the encoding of given files from one encoding to another; it should be available on most *nix-like systems. If it is not already installed, consult your package management system. A Win32 port of **iconv** is available at gettext (& libiconv) for Win32 with downloads at http://sourceforge.net/project/showfiles.php?group_id=25167 (fetch the libiconv-*-bin.woe32.zip package).

PostgreSQL

If for some technical reason you are not able to perform the steps below (for example because you do not have administrative rights to drop and create a database) you can enable the setting PHPAGA_PGSQL_ENCODING in `etc/config.local.php`. Read more about this setting in its description directly in the file. It is highly recommended, though, that you perform the steps below and migrate the existing data and structure to UTF-8.

The following steps need to be taken to convert the existing database (and data) to use UTF-8:

- Create a database dump

  ```
  pg_dump -Fp -U phpaga phpaga >| /tmp/phpagadump.sql
  ```

- Convert the database dump to UTF-8

  ```
  iconv -f LATIN1 -t UTF-8 phpagadump.sql -o phpagadump.sql.utf8
  ```

  Replace LATIN1 with the appropriate encoding for your data. Most likely this is will be LATIN2 if you were using the Hungarian locale, KOI8-R if you were using the Russian locale, and LATIN1 for most other languages supported by phpaga in versions <= 0.4.

- Edit `phpagadump.sql.utf8` and make sure that the line

  ```
  SET client_encoding = 'LATIN1';
  ```

  is changed to

  ```
  SET client_encoding = 'UTF8';
  ```

- Connect to the database server from the terminal (psql)
- Drop the existing database

```
template1=> drop database phpaga;
```

- Create the database with the proper encoding ('UTF8' for PostgreSQL 8.x, 'UNICODE' for PostgreSQL 7.x)

```
template1=> create database phpaga with encoding 'UTF8';
```

- Connect to the database

```
template1=> \c phpaga
```

- Restore the database from the converted dump

```
phpaga=> \i phpagadump.sql.utf8
```

MySQL

The following steps need to be taken to convert the existing database (and data) to use UTF-8. Replace "latin1" in the following example with the encoding currently used by your database. The procedure described below is not guaranteed to work - while it can work for certain data and encodings, it can also produce garbled data under certain circumstances.

- Create a database dump

```
mysqldump --default-character-set=latin1 -p -u username phpaga > phpagadump.sql
```

- Edit `phpagadump.sql` and change the statement

```
SET NAMES latin1
```

to

```
SET NAMES utf8
```

Then replace all occurences of

```
DEFAULT CHARSET=latin1
```

with

```
DEFAULT CHARSET=utf8
```

- Connect to the database server from the terminal (mysql)
- Drop the existing database

```
mysql=> drop database phpaga;
```

- Create the database with the proper encoding ('UTF8')

```
mysql=> create database phpaga default character set utf8 collate utf8_general_ci;
```

- Exit from the database terminal
- Restore the database from the dump

```
mysql --default-character-set=utf8 -p -u username phpaga < phpagadump.sql
```

2.4 Upgrading from 0.2 to 0.3

After 0.2 the internationalization mechanism was switched from definitions to gettext. Make sure your version of PHP supports gettext and has the locales for the languages you want to use installed. (See *Requirements*)

All application settings that were formerly defined in `etc/config.php` are now managed within phpaga itself. Once you have configured `etc/config.local.php`, you can fire up your browser and change these settings from the Admin menu ("Sitewide settings").

Make sure you have PEAR DB version >= 1.6. Install the PEAR packages Log and Mail and all other required PEAR packages.

Execute the scripts required to upgrade the database structure. Depending on the RDBMS you are using, this will be `sql/upgrade_0.2-0.3.pgsql` for PostgreSQL or `sql/upgrade_0.2-0.3.mysql` for MySQL. If you use MySQL, execute also `sql/mysql-innodb.mysql`; this will convert all database tables to the InnoDB format.

phpaga 0.3 introduces a permission system. In order to migrate to the permission system, the former administrator gets all available permissions. It is then in her responsibility to assign the proper permissions to the remaining users. The necessary SQL statement to perform this migration is contained in the upgrading scripts (see above paragraph).

phpaga 0.3 introduces news to the billing system. Invoices can now have line items. In order to convert the existing invoices to the new format, you will need to run a conversion script. This script must be run *after* `upgrade_0.2-0.3.(pgsql|mysql)` has been applied and *before* any new invoice is created. The script is named `convert_invoices.php` and located in the directory `tools/`. Open the file with an editor and set the parameters in the first section according to your configuration. Then run the script from the shell.

2.5 Upgrading from 0.2rc1 to 0.2

Configuration settings are now defined via web. You will need to replace your old `etc/config.php` with the new one from the package, and later change the settings from the Admin menu ("Sitewide settings").

Smarty and the R&OS pdf class are no longer bundled with phpaga. Refer to the *Requirements* section to find instructions on where to find and how to install these packages.

2.6 Upgrading from earlier versions

2.6.1 PostgreSQL

Execute the proper `upgrade*.sql` script from `sql/`.

2.6.2 MySQL

From 0.1.1 to 0.2

The following instructions give you the necessary information to upgrade your phpaga database under MySQL from phpaga 0.1.1 to phpaga 0.2.

Since phpaga 0.2 we use Pear::DB to abstract database access. Since MySQL does not support sequences directly (it has the 'autoincrement' feature), Pear needs a set of help tables that contain the sequence values.

- Apply `sql/upgrade_0.1.1-0.2.mysql` and `sql/phpaga_sequences.mysql`:

```
$ mysql -uroot -p phpaga < sql/upgrade_0.1.1-0.2.mysql
$ mysql -uroot -p phpaga < sql/phpaga_sequences.mysql
```

- The user passwords need to be migrated from MySQL password() to md5 hashes, in order to have a common way to manage encrypted passwords on different rdmbs. You can use md5sum to generate md5 hashes. Example (under Debian GNU/Linux):

  ```
  $ md5sum.textutils --string oldpassword
  d5b5fffc89f961903fb3c9a173f1b667  "oldpassword"
  mysql> UPDATE users SET usr_passwd = 'd5b5fffc89f961903fb3c9a173f1b667'
  WHERE usr_login = '@YOUR_LOGIN@';
  ```

  If you do not have md5sum.textutils you can create a small php script that contains the following lines:

  ```
  <?php
  echo md5('@PASSWORD@');
  ?>
  ```

  and either parse it via the command line php or via a php-enabled webserver and the output string will be the correct md5 value for your password.

From 0.2-rc1 to 0.2

In order to upgrade from phpaga 0.2-rc1 to 0.2 release simply apply `sql/upgrade_0.2_rc1-0.2.mysql`:

```
$ mysql -uroot -p phpaga < sql/upgrade_0.2_rc1-0.2.mysql
```

2.6.3 Migrating phpaga from MySql to PostgreSQL

Starting with version 0.2-rc1 phpaga features database abstraction through Pear::DB. As of version 0.2 the following rdbms are supported:

- PostgreSQL
- MySQL

If you want to migrate your phpaga data from MySQL to PostgreSQL you might find the following instructions of help.

- Create a mysqldump of phpaga containing just the data:

  ```
  mysqldump -uphpaga -pyourdbpasswd -c -t phpaga > phpaga_data.sql
  ```

- Connect to the PostgreSQL phpaga database with pgsql using the phpaga db user:

  ```
  psql phpaga phpaga
  phpaga=> \\i phpaga_data.sql
  ```

  Check the output for errors.

- Set the sequences to the correct values:

  ```
  phpaga=> \\i /path/to/phpaga/sql/phpaga_setseqvals.pgsql
  ```

This should be it. Please report any problems via the mailing list.

CHAPTER

THREE

BILLING PLUGINS

phpaga gives you the ability to write your own custom billing plugins which decide how totals and subtotals are calculated when you generate PDF bills to send to your clients. phpaga comes pre-installed with a few user-contributed plugins to calculate taxes for various countries, and a very simple plugin, which returns the total, not including any taxes. You can use these files as a reference implementation of what a billing plugin should look like. If you happen to write your own generalized plugin feel free to contribute it back.

3.1 Mini HOWTO

Billing methods can be created as plugins. The plugin file can be freely named and must be installed under the directory `lib/bill/`

The billing method function in the plugin **always** has to be called `phpaga_bill_calculate` and accept the following parameters:

```
(-> in |  <- out)

-> float startsum    # The startsum from which to calculate the bill
-> int   currency-ID # ID of the currency to be used
<- real  endsum      # The full sum that is to be paid
<- array details     # Array containing the full calculation with
                     # description and amounts
-> float expensesum  # Amount of expenses (optional)
-> float paymentsum  # Sum of payments (optional)
-> float feesum      # Sum of late fees (optional); developed for notaxes
                     # plugin, so may require new code to account for taxes
                     # that accrue in total amount due i.e. endsum
```

The "details" array has to be of the following structure:

```
details[]["text"]
details[]["amount"]
```

Please refer to the sample files shipped with phpaga (to be found under `lib/bill/`); you can use one of them as a base for your personalized billing method.

When a user selects a billing method, the corresponding file will be included, and the function will be called (with the necessary parameters).

This should provide a standard interface with which users can create their own plugins that create their bills exactly the way they (or their legislation) needs them.

As of phpaga 0.3, billing method plugins can be installed/uninstalled from within phpaga. The files are not physically installed or removed, but the database entries can be managed from the administrator's interface.

CHAPTER

FOUR

GETTING STARTED GUIDE

Sean Hull $Date: 2005-07-10 15:57:50 +0200 (Sun, 10 Jul 2005) $

(**Note**: *This guide has been written by Sean Hull in 2005. Some things have changed since then, but the core information still applies. Florian Lanthaler*)

4.1 Introduction

phpaga is one of those Open Source projects that really fits a niche. Are you an independent contractor, sole proprietor, freelancer, or perhaps you run a small business? Then surely you need a system to handle your billing. phpaga not only fits the need, it will do much more and surprise you with information about your business that you may not have paid attention to.

Presumably if you're already here reading these docs you're already *SOLD* on the LAMP or LAPP framework. If not I'll say a word or two about that. If you're using a desktop application to do your finances you have to worry about licensing and upgrades, bugs & fixes. What's more you're confined to the desktop, can't share the information, accross the city when you're at various client sites, or across the world if you're sharing the information, exchanging ideas with colleagues or have business partners in other locales. A web-based application moves the data out of your office and into the datacenter, where it can easily be part of your regular backup routine or hosted by a hosting provider who upgrades the OS, monitors security, and provides backups and so on.

4.2 Features

A picture or in this case a website is worth a thousand words. So before I dig into the features of phpaga, I'd like you to take a minute and check out the live demo (**Note**: *the demo is no longer online*) with user:demo pass:demo. It already has lots of companies, projects, invoices, quotations, and resources so you can see all the graphs and reports in action.

phpaga is a billing application that can keep track of your personal or small business finances, and is built on the LAMP or LAPP framework. What else can we say. Well it is international, handling billing for Australia, Canada, Germany, Italy, and the USA. And that's just with the builtin plugins. If you don't find what you need, you can write your own plugin quite easily. It can handle invoices based on hourly work, or named items with a cost. It can handle projects with multiple resources each with different hourly rates. And if you use those folks on another project, they can have different rates on those projects too. It can produce spectacular PDF quotations and invoices for your clients at week-end, month-end, or however you like. Furthermore it has extensive reports such as paid and unpaid invoices (turnover), summary reports for people, projects, and categories. The main page or dashboard also includes a nice 52 week graph of weekly manhours work. All this information and presentation helps you begin to get a bigger picture of your business, and finances to help you run your business better.

- Major Features
  - custom user logins
  - persons (project resources) without logins

  - default and per-project customizable hourly billing rates
  - company (client) can have more than one project
  - projects + subprojects
- Reports
  - weekly manhours reports
  - paid/unpaid invoices
  - project timeline
  - summary by persons
  - summary by project
  - summary by task
  - summary by customer
- Internationalization
  - international language support
  - international currency support
  - billing plugins for Australia, Canada, Germany, Italy & USA
- Invoices
  - hour based line items at billable rate
  - task based line items at fixed cost
  - printable/emailable PDF for client
  - enable/disable letterhead, tasklist, timesheet
- Quotations
  - customizable line items
- Miscellaneous
  - customizable company, person, project, and project status categories
  - customizable color scheme
  - default hourly billing rates
  - customizable hourly billing rates by project

4.3 Quickstart

A quick start is what everyone wants don't they. I know when I install a new piece of open source software, I quickly jump to this section, skim through the steps, and wing it. If I can't get it to work I go back to that section, and only if I absolutely have to do I touch the real docs. So, here goes.

The first thing you probably want to do when you test out this system is to generate a PDF invoice, right? Ok, here's what you do. After installation, you'll have an "admin" account, so login with that.

- Create a client - the recipient of your invoice
  - click on the "Companies" link (MAIN section)
  - click the "add company" link
  - fill in all the information and click "submit"
  - follow these steps a-c to create your parent company as that is disassociated from the user you login as

- Create a project
  - click on the "Projects" link (MAIN section)
  - click "Add Project" link in the "Selected projects" section
  - fill in the information, be sure to select the "Customer" who is the recipient, and "Billing Company" who is the owner of the project
- Create a person
  - click on the "Persons" link (MAIN section)
  - click on the "Add Person" link in the "Persons" section
  - fill in the info and select a company
  - default hourly rate can be changed for each project this person works on
  - Don't jump to the project resource section yet, this person won't show up. First you have to create a user.
- Create a user
  - click on the "Users" link (ADMINISTRATION section)
  - click "Add user" under the Users section
  - enter the login, and select the "Person" you created in step 3
- Add project members and tasks
  - click "Projects" link (MAIN section)
  - under "Selected projects" click your project name
  - click the "Add Project Member" link
  - select a Person from the popup and their default job category and default hourly wage will be filled in automatically! Of course if they are different for this project, you may change them.
  - click "submit" when you're done
  - when the popup window for "Add Project Member" closes click "Add Task" on the "Project: your-project-name" page.
  - select the "Project member"
  - enter the start/end date or use the popup calendar to select them. The first + last day of the month make a lot of sense here.
  - Fill in the duration as total hours worked.
  - enter a description of what that resource did this month
- Review project page
  - click the project name under "Project information" section
  - you'll see some changes there, and the graphs and details will include that new task you've added
- Create your invoice
  - click on the "Invoices" link (FINANCE section)
  - click "Add invoice with tasks" under "Invoices" section
  - if these are your first invoices and tasks it won't be obvious, but at this point it's worth noting that only *unassigned* tasks which don't appear on another invoice will show up.
  - click the "Create Invoice" link
  - select or deselect tasks as appropriate, or add custom line items
  - click recalculate if you've removed items

- when you're done, click "Create line items for associated tasks" button
- finally if all is ok click "Create Invoice" button
- note a *Notice* box may be displayed at the top if you don't select a recipient company
- enter a custom invoice number if necessary
- the resulting window will be your invoice in HTML format
- click "Export PDF" + select options
- select save-as or print from your Acrobat client

4.4 Understanding the workflow

4.4.1 Users vs Persons

The separation of users and persons revolves around the idea of those who can login to phpaga (users) and those who are just in the contact database (persons). At this point in time a user can exist, that is a login to phpaga, with associated permissions to use and administrate the system, *without* being part of any project or company. What this means is that you can use phpaga as your contact database, but all of those Persons won't show up in a resource list when you are creating or modifying a project. Only if they have a User defined will they show up in such a popup list.

Permissions for a user, and how they can use the phpaga system must be modified *after* that user is created. Click the "Users" link in ADMINISTRATION section and you'll see a "Users" table. (You can also search of course) If you click the "Login" column, you'll get user details and all of the permissions checkboxes. If you click the "Name" column, you'll get the user's profile (same as you do when you are logged in as yourself, and click the "Profile" link under your named section. All of the various permissions are available here, allowing the administrator to control what other phpaga users can do.

4.4.2 Companies

As explained in the quick start section above, a company is any financial entity that needs to be represented in phpaga, whether it is the *billing* company, that is your company that is using phpaga, or your clients - recipient companies.

4.4.3 Projects

Projects are associated with companies so you must have the *OWNER* company created before you create a project. That company can have 0 or more projects associated with it. However, and here's where it gets tricky, but very FLEXIBLE in ways you may not need now, but could need later. The billing company must also exist when you create a project.

For example, suppose your 5-man consulting enterprise wants to use phpaga to manage it's business invoicing. You have three projects, but you decide to break the company into Windows-arm, and Linux-arm. Two of your new projects are on Linux, and one is on Windows, for GE Capital. You create project A and select the customer as "GE Capital" and the Billing company as "Linux-arm". For project B it's the same thing. However for project C you select the Billing company as "Windows-arm".

For each project you can select a parent project. This allows phpaga to keep track of and visualize relationships between projects. If you go to live.phpaga.net (**Note**: *the demo is no longer online*) and log in (with "demo" and "demo") and then go to the details page for the project "phpaga parent", you can see a relationship graph under the section "relations". This graph is created on the fly with the Graphviz package. Try clicking on one of the boxes representing a related project and you this project's page will load. The currently selected project will always show in green, whereas the other projects are shown in grey.

4.4.4 Creating Quotations

4.4.5 Tasks + Hours

As you can see in the section above, you can enter tasks at any time while using phpaga. At the time you record a task, you can set a duration, or a start and stop time, plus a description. At that time you specify the project to which the task belongs, but you *DON'T* specify the invoice. Again, seemingly complex but you will find that it is really FLEXIBILITY in disguise. You may have worked some hours in May but you want, or the client asked you to bill them on your June invoice. That's no problem.

When you add tasks they go into a pool of unassigned hours. Under the FINANCE section you can click "Unbilled hours" at any time to see the list. Of course when you go to create an invoice, and assign those tasks, they will no longer be included in this list.

4.4.6 Creating Invoices

This should be your favorite part of using phpaga, and your favorite part of running your business! When you go to create an invoice you can do it with description/price line items or with tasks. Your choice. You may have to bill your client for equipment only one month, for that you would simple "Add invoice" not "Add invoice with Tasks".

4.4.7 Enter Invoice Paid Information

When you get a check from a client, you want to let phpaga know about it. Click on "Invoices" under the FINANCE section, and click the invoice from the list, or do a search. The HTML detail page for the invoice will display, with a "Set Payment Date". Click the calendar button and select the date it was paid. This will update various graphs, and reports in phpaga, letting you know where your finances are.

4.5 Understanding the reports

4.5.1 Main report - weekly manhours

By clicking "Main" under the MAIN section you can see two manhours breakdowns, one for the year, and one by day for the current month.

4.5.2 Finance report

This report is a summary of billed and collected money for your business. When a check comes in from one of the invoices you've sent to a client, you record the date it was paid (see 7. Enter Invoice Paid Information above).

4.5.3 Summary reports

Under the ADMINISTRATION section you see a link for "Summary report". There are actually *FOUR* great reports hidden under this little link. One provides a projects breakdown, by hours per project. The next one provides breakdowns by persons or resources who worked on all projects, and how much they contributed to the total. Next is a category breakdown pie chart, which as described in the user profile section below, if you have people who work as managers, programming and development, consulting, sales, marketing, human relations, and so on, this will break up each of their tasks so you can see where the bulk of billable time went. The last one is the turnover graph.

4.5.4 Project timeline

(**deprecated** - *this feature has been removed*)

There is another report hidden away in this application which is quite useful, and you'll be glad when you find it. Click on "Projects", and under "Selected projects" click the "Timeline" link next to "Add project". Under the project column you'll see each project you're working on, and based on the estimated start and end times, the timeline will display bars so you can see which projects overlap, and thus when thinks might be more rocky, or conversely when you have time for some new work.

4.5.5 User profile report

In this report you'll see three sections. Manhours is the same detail as seen in the main dashboard report, but just for you. It is a breakdown of the number of hours worked per week on a yearly scale. Very useful for viewing your business at a glance. The Task graph is the next section you will see and it is broken up nicely into type of work you do and how much time you spend. If you wear a lot of different hats, for instance business manager, DBA, Unix Administrator, marketer, sales-drone, and so on, you can see the breakdown easily.

4.6 Settings and customizing

Except for the user settings below, all of these settings are under the link "Sitewide settings" in the ADMINISTRATION section.

4.6.1 Graphs

JpGraph is the PHP module you installed to provide graphs for phpaga. You can set the width and height for the various graphs, change the work-week to 35 hours for France, and 70 hours for New York!! Also you can changes settings for the Gantt Chart showing financial summary of paid and unpaid invoices.

(**Note**: *Charts are now generated via a JavaScript library, JpGraph is no longer required.*)

4.6.2 Invoice Letterhead Formatting

The Letterhead section includes all the fields you will need to control how the PDF will format. It may make sense to put the street address in the "Location" field, and the "New York, NY 10003" entry in the "Street address" field just because of the order they show up on the PDF. Experiment until you like the output. In the US if you're a C-Corp you'll fill out Company tax number, but leave personal tax number blank, and vice-versa if you're a sole proprietorship. Other countries regulations are different of course, so take that into consideration. The footer line can be left blank, or filled in as appropriate for your needs, as can the logo file.

4.6.3 PDF Settings

In this section there are some general PDF settings for invoices, quotations, and the "Print" button you'll see in various places throughout phpaga. They are fairly self-explanatory, and direct how the PDF will be generated so that Acrobat can do it's magic.

4.6.4 Misc. Settings

4.6.5 Categories, Currencies + Countries

Just declared your own soverign state, you can add it into phpaga! Want to add another currency besides the included USD and Euro abbreviations, go for it. The categories edit is probably the one you'll use most often.

Types of companies, types of tasks in a project, types of jobs or positions held within a company, categories for resources, and even project statuses can all be configured.

4.6.6 Themes

In the layout section phpaga will list the CSS stylesheets that it finds in your install directory `htdocs/styles/*.css` so if you want a new one, copy one of those files, rename it, and edit the settings inside there. See Wikipedia for details on CSS.

4.6.7 User settings

Under your name section, the "Settings" link jumps to a page allowing you to change your user login information, and your permissions.

4.6.8 Other Layout settings

Do you like to use '.' to separate numbers and decimals, or do you prefer ','? You can configure that here. Like to have your month/day/year date layout, no problem. Set the sitewide date format as well. All sorts of other settings can be configured here, such as how many items to display for tasks, projects, invoices, and so on.

4.7 Miscellaneous items

4.7.1 Forgotten Password

If you forget your password, you have to get your SQL thinking cap on and change it. If you don't recall the name of your database, do:

```
$ md5sum.textutils --string oldpassword
d5b5fffc89f961903fb3c9a173f1b667  "oldpassword"
$ mysqlshow
```

Then login as follows:

```
$ mysql phpaga
mysql> select usr_id, usr_login, usr_passwd from users;
+--------+-----------+----------------------------------+
| usr_id | usr_login | usr_passwd                       |
+--------+-----------+----------------------------------+
|      2 | sean9     | 4f2a1493c661c0f2d2ee9a37040b8082 |
|      3 | neal9     | 3e7023ed317ed603851f22d510924ca1 |
|      4 | akahn     | b27fad92c6ddeddf0bfd6eb9871a8c79 |
+--------+-----------+----------------------------------+
3 rows in set (0.00 sec)
mysql> update users
set usr_passwd = 'd5b5fffc89f961903fb3c9a173f1b667'
where usr_id = 4;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Note that if you don't have **md5sum.textutils** installed you can also use this bit of php code to get the hash string:

```
<?php print md5("oldpassword")."\n"; ?>
```

4.7.2 Using billing plugins

There is a howto on creating billing plugins in the `docs/` directory. Read that for details. You'll basically put a new php file into `lib/bill/` and then select it in your "Sitewide Settings" so it will be the default. You can copy one of the existing ones in that directory, and edit it as appropriate.

4.7.3 Customizing templates

Templates are found in the `templates/` directory. You select one in "Sitewide Settings".

The customized template sets are an option, if someone has particular layout needs.

The theme (.css) should be enough to suit most people's needs. In case you need a rather exotic layout, say you don't need certain information or you want the layout to be readable on a handheld device over a slow network link, you can create your own template set. Simply copy the whole `templates/phpaga/` directory to a new name, say `templates/mytheme/`, and edit the files at will. You can, for example, leave out all the graphs and optimize the layout for 200x160 pixel screens.

In general, it is not recommended that users create their own template set unless they really want to keep up with development. Whenever we introduce a new variable (or remove an existing one) in one of the template files, this change would need to be applied to the customized template set too. If you do venture into this area, be sure to contact us here at phpaga, and contribute those changes for everyone to benefit from. That's what open source is all about after all.

4.7.4 Customizing themes

Themes are found in `htdocs/styles/` so copy one of those css files to rename the file, and then edit the contents for your needs. Wikipedia CSS Info

CHAPTER

FIVE

CHANGES

5.1 phpaga 0.5

5.1.1 New features

- Charts are now generated via JavaScript by using the jqPlot library. Therefore ezComponents are no longer required.
- Late Fees - Manual process; Adds late fee to invoice for unpaid balance. Late fee amount calculation based on percentage specified in Sitewide Settings. Updated bill page and pdf invoice to display late fees
- Payments - Support for payments on unpaid invoices/bills.
  - Updated finances reports to use payments in calucations
  - Updated bill page and pdf invoice to display payments and balance due; ticket #42
  - Payment date automatically set when payments equal or exceed invoice amount.
  - Added option to Sitewide Settings to show/hide 'Set Payment Date' link. May be times when date needs to be set manaully.
- Updated all existing billing plugins to include support for payments and late fees. NOTE to authors of billing plugins: Please verify changes are backward-compatible.
- Added Notes field to invoices and payments.
- Added ability to enable/disable display of currency abbreviation after monetary values.
- Added field to Sitewide Settings for monetary symbol to be displayed in front of monetary values e.g. $ or €
- Added Bar Chart highlighting to display exact monetary amount for each bar on Finances Invoiced/Received/Missing chart. Also added option to Sitewide Settings to enable/disable highlighting.
- New billing plugin by Tim Esselens (Belgium VAT)
- Ticket #55: Allow an invoice's payment date to be modified
- Ticket #72: PDF template sets
- Ticket #74: Show/hide the unbilled hours panel

5.1.2 Fixed

- Replaced _REQUEST with REQUEST_DATA that contains only _GET and _POST, addressing a problem reported and analyzed by Jools Wills (cookie collision).
- Fixed bug making it possible to bill again already invoiced expenses.
- Ticket #42: Balance Due now calulated and displayed on invoices when payment has been made.

- phpaga.css: references missing sortable.gif; fixed path and added sortable.gif to htdocs/img/phpaga/
- Fixed bug causing paid invoices with due date in the past to be displayed with red due date text.
- A company's financial information is no longer displayed if the user does not have the required permission
- Ticket #78: Handling of time in task entry inconsistent
- Ticket #59: Cannot delete expenses
- Ticket #57: Finance report chart: Wrong value for the "missing" bar
- Ticket #54: Do not allow project removal when related invoices exist

5.1.3 Other

- Documentation is now written in reStructuredText and built via Sphinx http://sphinx.pocoo.org/

5.2 phpaga 0.4

5.2.1 New or changed requirements

- UTF-8 locales
- PostgreSQL >= 7.4, MySQL >= 4.1.16
- PHP >= 5.2 is required with PDO and the PDO driver for PostgreSQL or MySQL
- eZ Components are required
- JpGraph is no longer used
- PEAR packages are no longer required - all dependencies from PEAR have been removed.
- The project timeline has been removed
- Picking a date is now handled by the "Unobtrusive Date-Picker Widgit" by frequency-decoder.com. (The coolest DHTML calendar widget is no longer required.)

5.2.2 New features

- Simple recurring invoices. (Ticket #7)
- Improved databased abstraction, new error handling, HTML cleanup.
- Files can be uploaded and associated to persons. (The "foto" feature has been removed from person's details. A migration script is available at tools/migrate_personpictures.php.)
- UTF-8 support (read the relevant sections in the INSTALL file before upgrading your existing installation)
- An optional contact person has been added to invoices and quotations. (Ticket #34)
- An invoice due date has been added. (Ticket #35)
- When a quotation is turned into an invoice, the quotation number can optionally be used as invoice number. (Ticket #30)
- It is no longer possible to store two invoices with the same number within the same year. (Ticket #31)
- Filed expenses can be added to an invoice (existing custom billing plugins need to be slightly changed).
- Jeremy from omnitechpro.com has submitted the following patches:
  - Show materials amount on unbilled hours page
  - Show invoice amount on invoice list page

  - Add “unpaid” bar to finances graph
  - Add invoice popup on Unbilled Hours screen

5.2.3 Updated translations

- Norwegian (Roger Bystrøm)

5.2.4 New billing method plugins

- Netherlands: nld_btw_hoog.php (Angelo Höngens)

5.2.5 Fixed

- Ticket #48: Letterhead layout problems with PDF documents in “letter” format.
- Ticket #25: With MySQL new projects would show up as blank under the unbilled hours page.
- Unbilled hours would show up on invoices.
- Fixed problems related to Javascript behaviour under various browsers.

5.3 phpaga 0.3a

5.3.1 Fixed

- Fixed a minor issue with the installer in 0.3 that could be triggered when trying to create the database on a remote MySQL server.

5.4 phpaga 0.3

5.4.1 New requirements

- PHP with gettext enabled
- smarty-gettext
- PEAR DB >= 1.7.6
- PEAR HTTP_Upload >= 0.9.1
- HTTP_Download >= 1.1.1
- HTTP_Header >= 1.2.0
- locales for desired supported languages
- Graphviz and PEAR IMAGE_Graphviz (optional, if you want to see project relationship graphs)
- Removed Overlib dependency, tooltips now shown with Walter Zorn’s DHTML JavaScript Tooltips library (comes bundled with phpaga)
- If you are using MySQL then MySQL >= 4.1.15 is required
- 3rd party libraries are now moved to their own subdirectory ext/

5.4.2 New features

- A web interface for assisted database upgrade during phpaga upgrades has been added.
- First time "wizard" has been enhanced to create database structure and import core data.
- Added the possibility to assign default hourly rates per task type; per task type per project; per task type, project and project member.
- Multiple invoices can be exported to a single PDF file.
- Invoices can be created from within a project page, and tracked material and tasks can be added.
- Quotations can be deleted as long as they are not tied to a project.
- Material can now be tracked per project.
- Product management added. Also, if a product with the product code of a new line item exists in the database, the description and price are added automatically to the line item.
- Files can be uploaded and associated to projects.
- The day and month of the beginning of the fiscal year can be defined. All financial reports are now based on the fiscal year.
- A default term of payment can be saved per company. This value can be fetched when creating an invoice.
- Let only users with the right permissions see all projects; other users see only projects they created or they are a member of. This is both true for the projects search interface and for the projects timeline.
- Hide a project's financial information (hourly rates, human cost) from "common" project members. Show "common" project members their own expenses but nobody elses.
- Added the possibility to upload and assign a picture to a person.
- When a project is created from a quotation then add a reference to said quotation
- Added a simple system information and an application log viewer
- Added project priority (suggested by Mark Parssey)
- Search bills/quotations between two dates
- Enable/disable installed billing plugins. Disabled plugins don't show up in drop-downs when creating new invoices/quotations
- Line items for invoices and quotations (conversion script for existing inv/quot)
- Search interfaces for persons and for companies (Mark Parssey)
- Only persons with a user account are shown when adding a new project member (Mark Parssey)
- Show project relationship graph (optional - Graphviz and PEAR Image_Graphviz required)
- Added a "summary overview" page
- An individual color can be specified for each operation category
- Persons: show projects person is owner or member of
- List summary overview of unbilled hours per customer and project; create invoice from unbilled hours
- Projects search interface; print project list to PDF (Mark Parssey)
- Projects: Added deadline (Mark Parssey)
- Projects: Added estimated manhours, estimated cost manhours, estimated cost material, parent project, billable status
- Get persons's default job category when adding a project member (Mark Parssey)
- Default hourly rate per person (this rate is suggested when adding a new project member)
- First time setup "wizard" (to create first user and person via web)

- Permission system
- All user submitted data is escaped/quoted by PEAR DB's methods before being stored in the database
- When selecting a language it is first checked whether the required locale is installed
- Second address field added to companies and persons
- Delete users
- Added task category matrix/graph to company page
- A list of invoices and the billing summary is shown on a company's details page
- Added a detailed overview to the project page that shows the sum of time per task category per person in a matrix
- Delete companies
- Billing method plugin installer/uninstaller
- Delete invoices that have not been paid yet
- Translations handled by gettext
- New billing method plugin by Daniel Cabezas
- Main page: show only projects the current user created or is a member of
- Limit number of characters shown for project title and task category title in task lists

5.4.3 New billing method plugins

- Spain: esp_liberoprof_euro.php (Daniel Cabezas)
- Canada: cdn_gstpst.php (Ken)
- USA: usa_sales.php (Arif Hamirani)
- Germany: de_mwst.php (Jens Bierkandt)

5.4.4 New translations

- Norwegian (Sverre Farstad)
- French (Benoit Nicolas)

5.4.5 Fixed

- Include JpGraph libraries only when actually creating a graph
- When changing a task, the task is no longer "taken over" by the changing user
- Thousands and decimal separator settings were not respected on the financial overview page
- New invoice/quotation numers greater that 10 would not be calculated correctly

5.5 phpaga 0.2

5.5.1 New features

- Create an invoice from a quotation
- Clone operations and invoices

- Added “average weekly work hours” line in weekly manhours graph
- Separated pdf layout (templates) from library, making it easier to personalize the pdf output
- Additional task search parameters (sponsored request)
- Added a detailed overview to the person page that shows the sum of time per task category per project in a matrix
- Added a field to projects that can contain the company that issues the invoice (sponsored request)
- Changed various text links to icons (using gnome icons from http://jimmac.musichall.cz/ikony.php3)
- Added a billing plugin (sales of goods) written by Alessio Bogani
- Added basic expense tracking per project (i.e. travel, hotel, meals)
- Allow multiline input for company location in the letterhead (patch by Robert Paskowitz)
- New web-based configuration system
- Added robots.txt (to prevent spiders from indexing phpaga sites)
- Added combined financial graph
- Added contrib/ for contributed stuff. Moved contributed style sheet(s) to contrib/
- Give detailed information when unable to connect to the database when PHPAGA_ERR_VERBOSE is defined and true
- Added a turnover history graph
- Added human costs calculation for invoices and project information. The necessary information is taken from the project memberships table (hourly rates field).
- When adding a task, administrators can now file hours for other users, not just themselves

5.5.2 New translations

- Danish (Per Christiansen)
- Hungarian (Fehér János)
- Russian (Dmitry Stroganov)
- Spanish (Daniel Cabezas)

5.5.3 Fixed

- Gantt chart (project timeline) shows cleanly, projects bars are linked to projects
- Various code, templates and stylesheets cleanups; interface improvements
- Email address check did not allow upper case letters
- Moved more display logic into the Smarty templates
- Fixed a bug regarding the handling of timestamps in PostgreSQL

CHAPTER

SIX

APPENDIX

6.1 Bugs

Feel free to drop a message to the phpaga-users mailing list to report bugs.

6.2 Contact

Comments, questions and praise are welcome on the mailing list.