

GS Plan

Team Commitment Planning and Tracking

© Jan Schrage 2008

Table of Contents

1Introduction.....	4
2Getting and installing the software.....	5
2.1Prerequisites.....	5
2.2Getting the software.....	5
2.3Setting up the software.....	5
2.4Receiving updates.....	7
3Use.....	7
3.1Logging in.....	7
3.2Periodicity.....	8
3.3The planning dashboard.....	8
3.4My team.....	10
3.5Review.....	11
3.6Reports.....	11
3.7The cumulative reporting dashboard.....	14
4Audit trails.....	15
5Maintenance.....	16
5.1The Maintenance Screen.....	16
5.2Import of time data.....	16
5.3User Administration, Roles and Rights.....	20
5.4Users vs. employees, how teams and countries work.....	22
5.5Projects and work types.....	22
6Reporting a problem.....	23
7License.....	23
83rd Party Components.....	24

Symbols and Notation

This documentation uses the following notation:

Code and commands you should execute are in red sans monospace: `rake test:units`

Files and folders are in blue sans monospace: `config/database.yml`

Internet links are in blue sans serif: <http://www.ruby-lang.org>

The following symbols are used:



Caution. Ignore at your peril.



Example



Additional information

1 Introduction

GSPlan is a team commitment planning and tracking tool. Its main purpose is to enable teams that have their own projects and collaborative projects to plan this, make commitments, and track them.

It is not a project planning tool: It does not and will not support planning of tasks, resources etc. at a level that is suitable for project planning.

GSPlan can

- Aid your teams in planning and committing to what they will do next month.
- Track the execution of those commitments.
- Aid in quality control through a review tracking system that is coupled to the project status workflow.
- Serve as an aid in team and cross-team discussions on the project portfolio.
- Promote transparency and efficiency.
- Help you to keep an overview of the status of your project portfolio.

The goal on creation was simplicity in use and data quality: There are no hidden menus or submenus, there is no distracting eye candy just for the sake of it, actions can mostly be taken on the elements on the screen and only those parts you can or are allowed to use are visible at any time. Database updates have a series of checks that enforce consistency and will alert you if things do not go as planned, are missing, or simply wrong.

GSPlan is licensed under the [GPL v3](#). Please make sure to review and understand this license before you use GSPlan. A copy of the license is included in [doc/License.txt](#). By using GSPlan you agree to this license.

2 Getting and installing the software

2.1 Prerequisites

In order to use this software you will need:

- git - <http://git.or.cz/>
- Ruby - <http://www.ruby-lang.org>
- Ruby on Rails - <http://www.rubyonrails.org/>
- ImageMagick - <http://www.imagemagick.org/>
- Rmagick - <http://rmagick.rubyforge.org/>
- A database. Rails supports a number of databases including Oracle, DB2 and MySQL. For the full list see <http://wiki.rubyonrails.org/rails/pages/DatabaseDrivers>

Optional but recommended:

- SQLite DB (for playing around before production and for testing) – <http://www.sqlite.org>
- Apache as production web server – <http://www.apache.org>
- A Unix-based server OS, such as Linux or Solaris.



The use of a standards-compatible browser, such as Firefox, Opera or Safari is highly recommended. While you should be able to use Internet Explorer I do not guarantee that anything looks good or even works on IE and there are some known visual quirks. See also [Reporting a problem](#)

If you are using Linux you can install all or most of these packages via your package administration without any need for manual installation. (Ubuntu, on which GSPlan was developed, comes with all of them so you only have to make sure they are actually installed.)

2.2 Getting the software

GSPlan is hosted on GitHub at <https://github.com/janschrage/gspln/>

You can clone the repository with

```
git clone git://github.com/janschrage/gspln.git
```

Alternatively, you can download a tar or a zip file from the hosting site. If you want to receive upgrades cloning with git is the suggested method, as it allows for easy integration of changes made to the central repository.

2.3 Setting up the software



All examples are on Linux and can be executed from a shell there. All commands must be

executed from the installation directory unless noted otherwise.

In order to make sure the software is in a workable condition after download (including DB setup etc) you should first run the test suite:

```
rake test
```

You should not receive any errors.

Please note that by default the test suite uses sqlite. If you have not installed it you will get errors now. I would recommend installation. Alternatively you can change the test environment by editing

`config/database.yml`

The next step is to set up the database.

You will have to

1. Create a database
2. Configure GSPlan to access it
3. Load the DB schema
4. Populate the database

The following paragraph describes how to do that with MySQL.

Create a database:

```
mysqladmin -u root create GSPlan
```

Configure GSPlan for access:

Edit `config/database.yml`

Assuming this is your production database you need to edit the production section:

```
production:
  adapter: mysql
  encoding: utf8
  database: GSPlan
  username: root
  password:
  socket: /var/run/mysqld/mysqld.sock
```

If you did not set a user and a password for the GSPlan database this is sufficient.



You should set a user and a password for the database and update the configuration file accordingly. Do not run a productive system without a database password. A step-by-step description can be found in the MySQL manual.

Load the DB schema:

```
RAILS_ENV=production rake db:schema:load
```

Populate the database. This will set up the default roles and associated rights. It will not set up

users.

```
RAILS_ENV=production rake db:populate_roles
```

Start the application:

```
mongrel_rails start -e production -d
```

You should now be able to point your browser at <http://localhost:3000> and get started. As no users are in the database there will be no login screen.



You should, at this point, make sure you have a solid and reliable backup for your database in place.

2.4 Receiving updates

GSPlan is in ongoing development and you may want to receive updates or bugfixes.

In your installation directory execute

```
git pull
```

This will pull all updates from the project. You may have to migrate the database after an update with

```
rake db:migrate
```



Do not try this on a productive system. You should first review and test all upgrades to make sure you know what you are doing. You should also verify the commit logs on the project's site at github to see what updates you will receive. Please note: What happens on your systems is entirely your responsibility, including loss of data, nerves, limb and life, and the destruction of the neighbourhood. You have been warned.

3 Use

3.1 Logging in

The login is case sensitive. After installation, with no users, you will not need to login. As soon as you create the first user a login and all authority checks are enforced. Please make sure to review chapter 5.3 (User Administration, Roles and Rights) before you create users. The first user should have the admin role assigned; otherwise you will lock yourself out of the system.

3.2 Periodicity

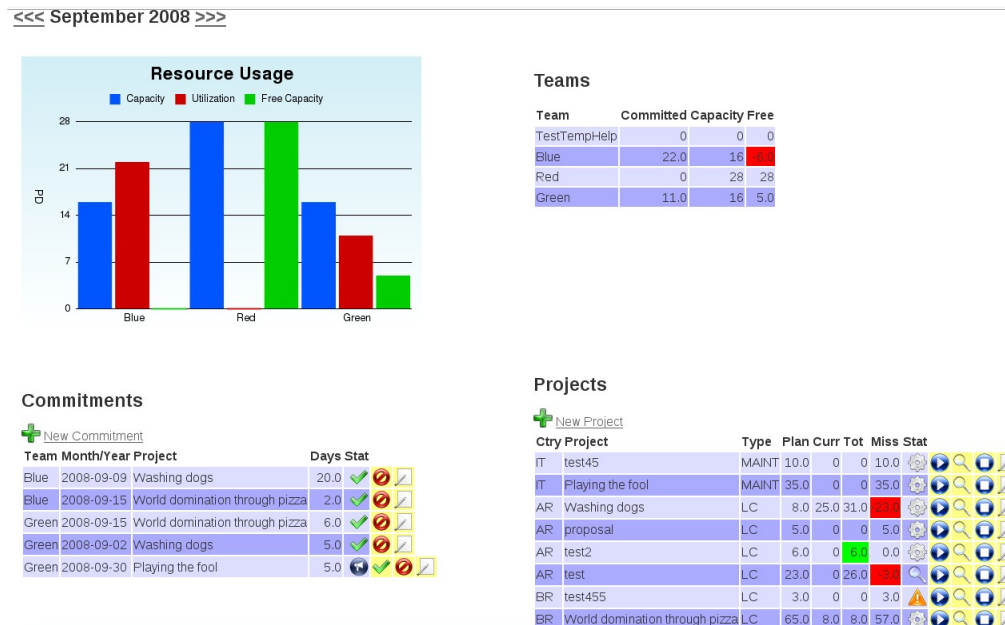
GSPlan operates on periods of one month.

3.3 The planning dashboard

The planning dashboard has four parts:

- The team capacity graph (top left)
- The team capacity list (top right)
- Current team commitments (bottom left)
- Current projects (bottom right)

In general status and information fields are in shades of blue. Action fields have a pale yellow background. Action fields will only be visible if you are authorized for this action.



Navigation between months is done with the arrows to the left and the right of the current month at the top of the screen.

The capacity graph and the capacity list

Both graph and list show the same data: For each team the capacity is calculated based on the persons assigned to the team. The usage is based on the commitments the team has for this month. Only accepted commitments are counted.

Capacity is calculated as 16 days per person and month at 100%. This accounts for vacation, illness, trainings etc. If you want to change that you will need to edit

[app/helpers/statistics.rb](#)





Change the line

`DaysPerPerson = 16;`

to something that is more suitable for you.

The commitment list

The commitment list shows all commitments of all teams for the selected month. Each team can commit any number of days to any project. Commitments must be unique, i.e. you cannot have two entries for the same team and the same project. The column „Stat“ shows the status of the commitment. By default any new commitment has the status „proposed“. It is only counted when „accepted“.

Symbol	Meaning
	Status: accepted (days count) Action: accept
	Status: proposed (days don't count)
	Action: delete
	Action: edit

With the default role setup team leads can propose commitments and administrators can accept them, e.g. in a common planning session.

The project list



The project list shows all projects that:










- fall into the selected month or
- end before the selected month but are not closed (will be marked with status „overdue“)

The columns show:

Pre?	Ctry	Project	Type	Plan	Curr	Tot	Miss	Stat
Work type is part of preload (red up arrow) / no preload (blue down arrow)	Country (or functionality), e.g. UK or UK-XYZ	Project name	Type of project (e.g. maintenance, documentation)	Total planned time (person days)	Committed this month (only accepted commitments)	Total committed time (only accepted commitments) (green if fully committed)	Missing time (Total planned – total committed) (red if overcommitted)	Project status (see below for symbols)

Symbols used:

Symbol	Meaning
	Status: proposed
	Status: not started

	Status: in process
	Status: testing/piloting Action: set to „testing“
	Status: closed Action: accept
	Status: rejected Action: reject
	Action: set to „in process“
	Action: set to „closed“
	Action: edit
	Status: overdue (planned end reached but project not closed)
	Status: Parked. This project is on hold and not being worked on or considered in planning. Parked projects will not appear in monthly planning and reporting. There is a dashboard report „Parked projects.“

Project workflow:

- When created, a project has the status „proposed“. It can then be accepted or rejected. By default team leads can create projects, but not accept or reject. This can be done by admins, e.g. in a common planning session.
- Rejected projects cannot be changed from this screen.
- If accepted, the status is set to „not started“.
- The „play“ button will set the status to „in process“. This is not possible if a project is in testing or closed.
- The „test“ and „stop“ button will set the status to testing and closed, respectively. If the project type needs reviews this is only possible if all reviews have been passed. Please see section Review for further explanation.

Successful execution of an action will highlight the row with a pale yellow flash. An error will highlight the row with a red flash. An error is normally due to missing reviews. Please note the comments on the workflow above.

3.4 My team

The „My team“ page combines elements of

- the planning dashboard

- project reporting
- review reporting

for the user's assigned team. Refer to those sections for details.

3.5 Review

The review workflow

The review section is intended to help you keep an overview of your review process. Through a link to the project workflow reviews can be enforced.

The review screen displays all current projects (i.e. overlapping with the selected month) and their review status. Currently specification, design and code reviews are handled.

GS Plan

Plan Report My Team Review Maintain

Listing current projects for review

Country	Project	Type	Begin Date	End Date (Planned)	Spec?	Design?	Code?	
AR	Washing dogs	LC	2008-08-01	2008-09-10	✓	✗	✗	Review
BR	World domination through pizza	LC	2008-09-01	2008-10-15	✗	✗	✗	Review
AR	proposal	LC	2008-08-28	2008-09-28	✗	✗	✗	Review
AR	test	LC	2008-08-09	2008-09-09	✗	✗	✗	Review
AR	test2	LC	2008-08-09	2008-09-09	✗	✗	✓	Review
BR	test455	LC	2008-08-09	2008-08-09	✗	✗	✗	Review

The „review“ link allows a reviewer to add a review with comments. The reviewer is stored automatically. A project can have multiple reviews of any type.

The status of each review can be

- „OK“
- „OK with comments“
- „Fail“

Only failed reviews will prevent testing and closing projects.



The current default setup for the database does not include a specific reviewer role. Refer to section User Administration, Roles and Rights for an explanation on how to do this.

How to add new review types

The database model is flexible enough to handle this, but you will need small code modifications in a few places:

- model `Review` (`app/model/review.rb`): Add the new review type to the constants
- helper `ReviewsHelper::review_type_list` (`app/helpers/reviews_helper.rb`): Add the review type text to the list
- view `app/views/reviews/current_projects.html.erb`: Add a field for the new review type to the table
- model `Project::reviews_ok_before_pilot_or_close` (`app/model/project.rb`): If you want a check on the new review type for the project workflow it has to be added in this

method, similar to the existing ones.

3.6 Reports

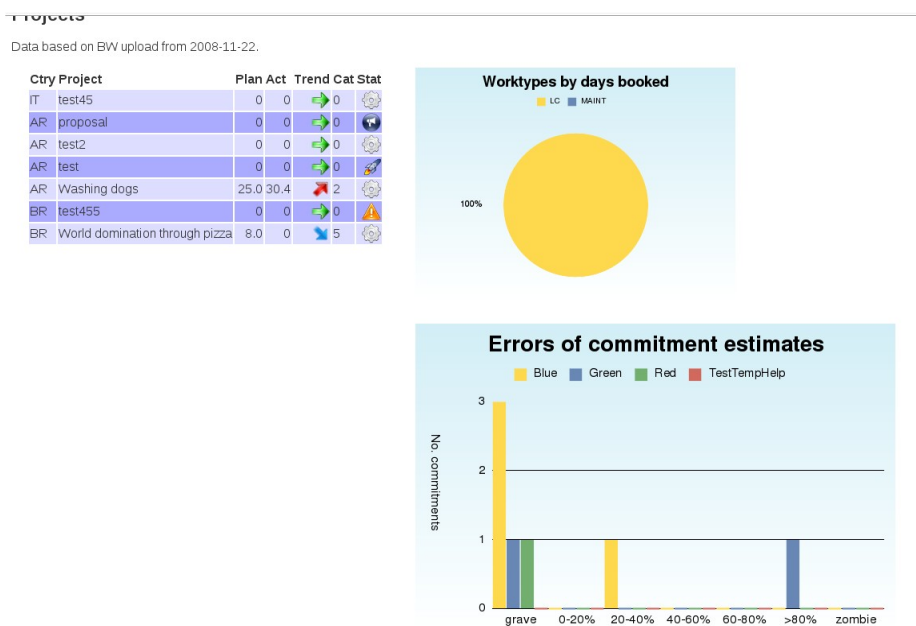
The report page displays a list of all projects that

- overlap with the current month or
- are overdue

It also shows a graph on the distribution of project types and another on the quality of project planning/execution.



Currently all reports are only for the current month. Cumulative reports are planned for the near future.



The first information on the screen is the date when the time data was uploaded (refer to section Import of time data).




The project list

The project list shows the following information:

Ctry	Project	Plan	Act	Trend	Cat	Stat
Country	Project name	Committed days (selected month)	Booked days (selected month)	Trend committed vs. Booked (see below)	Quality of execution/planning (see below)	Project status (as described in the planning dashboard)

Project trend

This indicates the trend of actually spent time compared to the commitment:

	Project on commitment +- 10%
	Project under commitment, delta > 10%. This is expected for most projects before a month is closed.
	Project over commitment, delta > 10%.

Quality of execution/planning

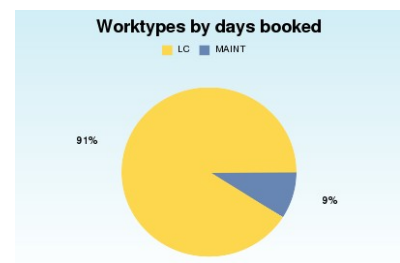
This category describes, depending on how you look at it, how good your planning was or how well it was executed. The values mean:

0	1	2	3	4	5	6
No time committed, none spent	Estimation error 0-20%	Estimation error 20-40%	Estimation error 40-60%	Estimation error 60-80%	Estimation error > 80%	No time committed, time spent

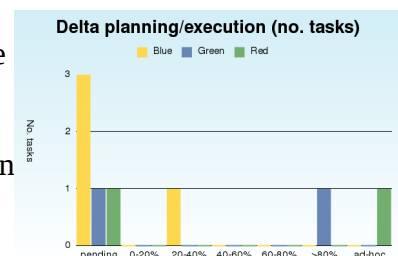
Apart from the obvious categories 1-5 category 0 are projects that are just lying around (and should possibly be closed) and category 6 is the intransparent part of ad-hoc work, where one thing was planned and something completely different done. While this is always likely to be present it should be kept low.

The graphs

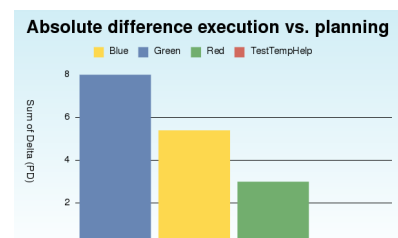
The worktype distribution graphs show you how time was distributed between different types of work.



The graph „Delta planning/execution“ shows you the planning quality categories discussed above per team (**not** per country, as the project list does). The y-axis shows the number of commitments in a certain category. „pending“ is category 0 from above, and „ad-hoc“ is category 6. Note that before a month is closed the estimation errors for that month will be quite high as not all times have been booked.



The graph „Absolute difference of execution vs. Planning“ gives a different view of the same data: While the previous graphs counts



the number of projects this graph shows the sum of absolute differences of planning vs. execution per team in days. (Absolute means it does not matter if the estimate was above or below the time spent, it adds to the sum.)

3.7 The cumulative reporting dashboard

In addition to the monthly overview report there is a dashboard for cumulative and tracking reports over a period of time. It can be reached via the button „Dashboard“ in the menu.

All reports display their data in a normalized table that makes further analysis in a spreadsheet or other tool easy. If you are using Firefox the add-on [Table2clipboard](https://addons.mozilla.org/de/firefox/addon/1852) (<https://addons.mozilla.org/de/firefox/addon/1852>) will allow you to copy&paste the data into a spreadsheet or a csv.



The default time range is 3 months from now into the past, including the current month. In effect this will give you a rolling 3 months report.

Work type distribution – tracking

This report displays the work types per team, work type and month between begin and end date, to the extent to which data is available. Like in the monthly report the last upload will be used for each month. In the example on the right you can see that while the end date is January 09 data is only available till September 08.

Team	Month	Work type	Days booked
Blue	Aug	LC	22.0
Green	Aug	LC	8.4
Blue	Sep	LC	22.0
Green	Sep	LC	8.4
Red	Sep	LC	6.0
Red	Sep	MAINT	3.0

Work type distribution – cumulative

This report works exactly like the tracking report, but cumulates over the months between begin and end date. Please note that missing months are not easy to see here. You should make sure that data is actually available over the period chosen.

Team	Worktype	Days booked
Blue	LC	44.0
Green	LC	16.8
Red	LC	6.0
Red	MAINT	3.0

Project age for current projects

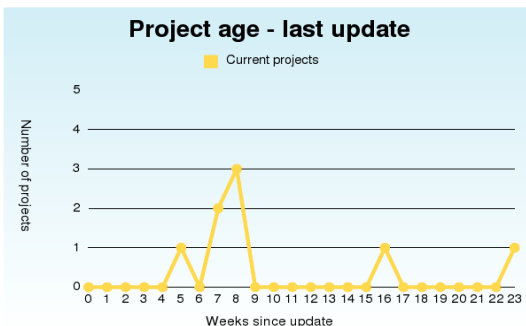
The purpose of this report is to identify projects that are just lying around without being touched (and possibly should be closed). It displays data on currently open projects, including the number of weeks since they were created, their status and the number of weeks since they were last updated. Please note that „updated“ refers to an update in the system, e.g. a change in status from in process to testing. This report uses the time stamps on database changes. The graph counts the number of projects per bin. In the example graph and report you can easily see two outliers with very low efforts.

Report type: **Project age - current projects**
Go

Report Data

Project age - current projects

Project	Ctry	WType	PD(Plan)	Wks s/creat.	Wks s/upd.	Stat
Playing the fool	IT	MAINT	35.0	7	7	
Test helper move	AR	LC	2.0	7	5	
Washing dogs	AR	LC	8.0	23	16	
World domination through pizza	BR	LC	65.0	23	8	
proposal	AR	LC	5.0	16	7	
test	AR	LC	23.0	23	8	
test2	AR	LC	6.0	23	8	
test455	BR	LC	3.0	23	23	



The project age report does not use begin and end date from the selection screen. It evaluates all projects that are currently not closed or rejected. It does not evaluate projects with a begin date in the future.

Process cycle time (PCT)

This report evaluates the process cycle time for all projects finished between the selected begin and end date. It also shows work in progress, i.e. the number of projects that are currently open, in process, or in testing. The date used for selecting closed projects for the evaluation is the planned end date.

The report also shows process lead times for projects (PLT). The date used for calculating the PLT is the date the project was last updated. A closed project should not be updated any more from a process point of view. (A project that gets updated is *not* closed.) The PLT report also shows the PLT as a percentage of the planned effort in months (as the planning cycle is monthly with supposed monthly updates every project will have between 28 and 31 days PLT, no matter how long it really takes, increments will also be monthly):

$$PLT(\%) = PLT / \text{time needed} * 100/30$$

For ease of use a month is calculated with 30 days here.

The projected days to deliver for the project pipeline is calculated as $PCT * WIP$.

For these figures only projects whose work type is not marked as „continuous“ are considered (i.e. ongoing work such as maintenance etc. may be disconsidered).

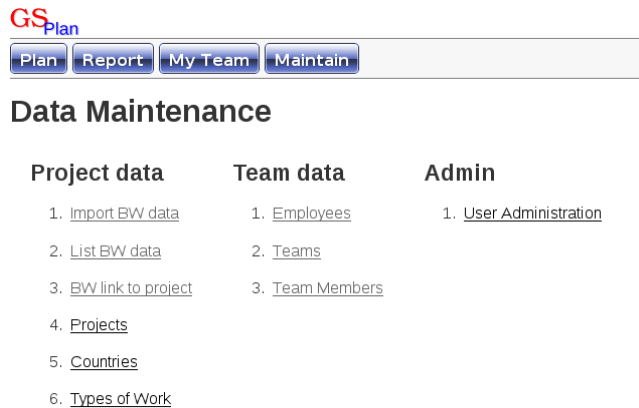
4 Audit trails

All relevant database changes are automatically logged in a table called `audit_trails`. There is currently no interface to view them. If needed the audit trail can be accessed from a suitable db administration tool.

5 Maintenance

5.1 The Maintenance Screen

The maintenance screen allows for maintenance of project data, team data, and user administration. Currently not implemented are roles and rights administration, as they are hardly ever needed.



5.2 Import of time data

Time data is imported from a CSV file (comma separated values) that has bookings per employee and task. During import that data is mapped to the teams in-memory, personal data is at no point retained or stored on disk.

Several tasks can be mapped to one project (e.g. specification, design, development, test) allowing for a granularity in your time data that differs from your commitment planning. (Project planning, and planning time per tasks, is not the purpose of GSPlan in any way whatsoever.)

GSPlan operates on a monthly basis, but you can import several sets of time data per month. For all calculations the most current set will be used.

We will now

1. Look at the format of the import file
2. Discuss the steps of the import
3. See how to link a task from the import file to a project
4. Briefly discuss how to adapt the software to a different input format

The format of the input file

- The import file must be a .csv, fields are separated by a semicolon.
- The decimal separator can be a comma or a decimal point. There must not be a thousands separator.
- Numbers are assumed to be days, not hours.

- During import non-project time is read from a different column than project time. The mapping for both, however, is identical. That means if you wish to book this time you should have a project that takes the task type „Not assigned“.

Column	A	B	C	D	E	F
Content	Personnel number	Employee name	Task name	Time (unproductive)	Time (productive)	unused
Use	Identifies a person. If empty, time data is assumed to belong to the last person seen. Used to identify the team to book time for.	Not used during import.	Used to identify the project to book the time on. The task „Result“ is ignored (compare screenshot below).	Times such as absences, training etc might go here. When the task name „Not assigned“ is found this column is used.	Project time should go here. This column is used by default for reading time.	

An example screenshot can be seen below.

	A	B	C	D	E	F	G
1	D564324	Barney Bear	Not assigned	22		22	
2	1123456	Fred Tester	Washing Dogs		8,4	8,4	
3	1123456	Fred Tester	HosuchProject		8,4	8,4	
4			Taxes in Lithuania		0,6	0,6	
5	1765421	Mickey Mouse	Result	1,8	20,2	22	
6			Messages		2,6	2,6	
7			Coiffeur training		17,7	17,7	
8	X345322	Chip F. Munk	Pizza for the world		17	17	
9	X879832	Goofy Bush	Pizza for the world		18	18	
10							
11							
12							

Import steps

Step 1 – Select file and booking period

The import starts with point (1) on the maintenance screen: Import BW data.

The first step is a file and date selection dialog. The date should be in the month you want to book the time for, in the format yyyy-mm-dd. The day does not matter. All data will be tagged with the current date, too, in order to identify the most current set of data for this month. The file should have the format described above.

Plan Report My Team Maintain

Import a BW file

Date for BW data (yyyy-mm-dd), dd can be any day

Select a CSV File :

Step 2 – Verification of employee and project data

















In the second import step employee and project data are verified:

- Does the employee exist in the database?
- Is the task linked to a project in the database?

All imported lines will be listed, with a column for employee and task that shows either a green check for „OK“ or an alert sign if there is a problem.

- If all data is OK there will be a button on the screen that allows you to proceed.
- If you see no data or get a rails error screen the file format is very likely wrong. In this case you should check the file or the server log.

Below, you can see two screenshots with and without errors.

Plan	Report	My Team	Maintain
Imported Data			
Reporting month: 2008-07-01			
Total number of lines with errors: 7			
Pernr	Name	Task	Days EE Prj
D564324	Barney Bear	Not assigned	22.0  
I123456	Fred Tester	Washing Dogs	8.4  
I123456	Fred Tester	NosuchProject	8.4  
I123456	Fred Tester	Taxes in Lithuania	0.6  
I765421	Mickey Mouse	Messages	2.6  
I765421	Mickey Mouse	Coiffeur training	17.7  
X345322	Chip F. Munk	Pizza for the world	17.0  
X879832	Goofy Bush	Pizza for the world	18.0  

Plan	Report	My Team	Maintain
Imported Data			
Reporting month: 2008-07-01			
Data OK.			
<input type="button" value="Proceed"/>			
Pernr	Name	Task	Days EE Prj
I123456	Fred Tester	Washing Dogs	8.4  
I123456	Fred Tester	World Domination	10.4  
D564324	Barney Bear	Washing Dogs	22.0  

An error on the project that results from a missing link can be resolved by creating this link via item (3) on the maintenance screen: BW link to project. This will list all currently open projects and offer the option to create or edit links. The task name to be entered is the one from the CSV that you see on the import screen in the „Task“ column.

Note: You can have several tasks linked to the same project.

Step 3 – Book days for teams and projects

The proceed button will bring you to the next verification step for team and project time booking. This verifies that:

- The employee is assigned to a team in the booking month.
- The booking date is in the timeframe of the project.

You will see a list with the converted data and a list of errors if there are any. Again, you will only be allowed to proceed if all data is correct.

Below you can see screenshots with and without errors.

[Plan](#) [Report](#) [My Team](#) [Maintain](#)

Conversion results

Reporting month: 2008-07-01

No team found in reporting period: Employee I123456

[Plan](#) [Report](#) [My Team](#) [Maintain](#)

Conversion results

Reporting month: 2008-09-01

Commit to database

Team	Project	Days
Red	Washing dogs	22.0
Blue	Washing dogs	8.4

If everything is OK you can commit the imported data to the database. All statistics and report for the booking month will now use that data set.

Changing the file format

The file format is defined in `ProjecttracksController::do_import`, in file `app/controllers/projecttracks_controller.rb`

Changing it is straightforward: This method only deals with reading the data from the columns, so you can easily change it without affecting the other import steps.

5.3 User Administration, Roles and Rights

Roles and Rights

The authorization concept of GSPlan is based on:

- Role(s) assigned to a user
- Right(s) assigned to a role, linking a right to a controller and action

A set of rights define a whitelist for a role: Any action not explicitly allowed is forbidden. Each right gives you access to one action in a controller. For ease of use we will note this as `controller/action` in the following paragraphs.



The predefined right `right_create_project` gives you access to `projects/create`, i.e. the create action in the project controller. In effect, everybody who has that right can now create a project. This right is part of the predefined role „Teamlead“. All users with the role „Teamlead“ will be able to create projects.

The special controller/action placeholder „*“ gives you right to any controller or action.



The right `projects/*` will allow access to all actions of the projects controller. Likewise the right `*/edit` will allow access to the edit action of all controllers. This is not normally a good idea.

The predefined role „Admin“ has the single right ***/***, allowing access to everything.

Adding your own roles and rights

Adding your own roles and rights (and assigning them to users) is straightforward, although there is no UI at this point.

First, start the ruby console (assuming your production environment):

```
RAILS_ENV=production ruby script/console
```

Execute all following commands in the console you just started.



Example 1 – Create a reviewer role, allow it to see the review list and create a review

```
role_reviewer = Role.create :name => "Reviewer"
right_list_rev = Right.create :name => "list_current_projects",
:controller => "reviews", :action => "current_projects"
right_create_rev = Right.create :name => "create_review", :controller =>
"reviews", :action => "create"
role_reviewer.rights << right_list_rev
role_reviewer.rights << right_create_rev
role_reviewer.save!
```



Example 2 – Add the right to see current project reviews to the teamlead role (assuming the right exists as defined above)

```
role_tl = Role.find_by_name(„Teamlead“)
right_list_rev = Right.find_by_name(„list_current_projects“)
role_tl << right_list_rev
role_tl.save!
```

For more examples on adding a role and assigning rights to it you can follow

[lib/tasks/db_populate_roles.rake](#)



In order to be useful a role will need some more rights than defined in the examples above. Typical rights to consider on all controllers are „show“, „create“, „new“, „delete“, „edit“, „update“. After assigning rights to a role you should always test the intended workflow. Please compare to the predefined teamlead role in [lib/tasks/db_populate_roles.rake](#)

Assigning a role to a user

While users can be defined and assigned to a team via the maintenance screen roles currently have to be assigned via the console. There are two rake tasks to help:

- Assign the team member role to all existing users: `rake db:make_all_teammembers`
- Assign a role to a user: `rake db:assign_role user=<user> role=<role>`

In order to make user Fred an administrator in the production system, you would either:



```
RAILS_ENV=production rake db:assign_role user=Fred role=Admin
```

Or



```
RAILS_ENV=production ruby script/console
fred = User.find_by_name(„Fred“)
role_admin = Role.find_by_name(„Admin“)
fred.roles << role_admin
fred.save!
```

Note: Users can have more than one role. The rake tasks do not check if a role is assigned twice but assume you know what you are doing.



As long as no users are defined in the system at all, all authority checks are switched off. When you define your first user the next step should be to assign it the administrator role.

5.4 Users vs. employees, how teams and countries work

Users are users of the application. This is a purely administrative concept.

Employees participate in projects. They are part of the planning and booking.

There will most likely be some overlap, depending on which rights you decide to give whom.

Each user should be assigned to a team: This makes the „My Team“ page work.

Each employee should be assigned to a team: This is used for the calculation of the team's capacity and booking time. Team members can be assigned on a temporary basis and with a certain percentage (relevant for capacity calculations).

Each team owns one or more countries (or functionalities, it really doesn't have to be a country).

5.5 Projects and work types

Each project is assigned to a country and has a type of work. You are free in the work types you define. The review workflow is defined per work type.

Each work type has the following relevant flags:

- Preload: If this type of work is part of your preload (i.e. must do, not negotiable) you should flag this. Currently the preload flag is visible in planning and reporting.
- Needs review: If a work type needs to be controlled through the review workflow, this flag must be set. E.g. development projects typically will be, whereas training probably won't.
- Continuous: If this work type is ongoing, such as maintenance, administration etc. this flag should be set. Project with these work types will be disconsidered for project reporting (PCT, Project Age, Project Times).

6 Reporting a problem

Problems can be reported via <http://janschrage.lighthouseapp.com/projects/15941>

Do **not** report **any** issues with the Internet Explorer. If you cannot reproduce an issue you have in IE on a browser that follows W3C standards such as Firefox, Opera, Konqueror or Safari report the problem to Microsoft, not to me. If you can reproduce a problem on a compliant browser, by all means report it. Overall GSPlan works on the Internet Explorer, but you may encounter quirks in the visualization. These problems are entirely due to the Internet Explorer not following W3C standards, in particular, CSS specifications.

7 License

Copyright (C) 2008 Jan Schrage [<jan@jschrage.de>](mailto:jan@jschrage.de)

This program 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 3 of the License, or (at your option) any later version.

This program 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 this program. If not, see [<http://www.gnu.org/licenses/>](http://www.gnu.org/licenses/).

8 3rd Party Components

This software uses the 3rd party components listed below.

- Icons from the Crystal icon set: <http://www.everaldo.com/crystal/>
- Icons from the Spheres icon set:
http://www.mouserunner.com/Spheres_Icons_Index.html
- Gruff chart plugin: <http://nubyonrails.com/pages/gruff>
- Aero Button Menu from the Dynamic Drive CSS library:
http://www.dynamicdrive.com/style/csslibrary/item/vista_aero_buttons_menu/
- Active Scaffold for some of the table maintenance: <http://activescaffold.com/>