Welcome to Nereid’s documentation. This documentation is divided into different parts. We recommend that you get started with *Installation* and then head over to the *Quickstart*. Besides the quickstart, there is also a more detailed *Tutorial* that shows how to create a complete (albeit small) application with Nereid. If you’d rather dive into the internals of Nereid, check out the *API Reference* documentation.

The main dependencies of Nereid are Flask and Tryton. In addition Nereid uses Babel for internationalisation, Speak Later for lazy strings, WTForms for form building and ccy for currency mapping.

The template engine used is Jinja2 and Flask is built over Werkzeug a WSGI toolkit. These libraries are not documented here. If you want to dive into their documentation, check out the following links:

- Flask Documentation
- Tryton Documentation
- Babel Documentation
- Speaklater Documentation
- WTForms Documentation
- CCY documentation
- Jinja2 Documentation

If this documentation reminds you of Flask documentation, it is not a coincidence, it has been designed so to ease up the learning curve for anyone who knows both Flask and Tryton.
This part of the documentation, which is mostly prose, begins with some background information about Nereid, then focuses on step-by-step instructions for web development with Nereid.

Familiarity with Tryton is assumed.

## 1.1 Foreword

Read this before you get started with Nereid. This hopefully answers some questions about the purpose and goals of the project, and when you should or should not be using it.

### 1.1.1 What is Nereid?

Nereid is a web framework built over Flask, with Tryton as a Backend.

### 1.1.2 What are the uses of Nereid?

Nereid can be used to build web applications, that could use Tryton’s ORM as a backend. While, there are no inherent limitations which prevent you from using nereid to build any kind of web application, the design decision that we made while building nereid itself are tailored to build application that extend the functionality of the ERP system, like e-commerce system, EDI systems, Customer/Supplier Portals etc.

### 1.1.3 Why Tryton as a backend?

Well, why not would be our question to you? It’s scalable, it’s flexible and offers the best approach we have seen so far into a declarative coding pattern for model design in any ORM. The unique way Tryton handles inheritance also makes it an excellent choice. In addition to the above, Tryton by default has several modules which make designing business applications faster in comparison to other frameworks.

Let’s say that you want to build a customer portal, (which is our example application), all that you need to do from your end is create a module which exposes the information that you want to, and leave other stuff like order management, account management etc to the existing Tryton modules.
1.1.4 Configuration and Conventions

Flask has many configuration values, with sensible defaults, and a few conventions when getting started. Nereid follows the same pattern of configuration values where it makes sense to use them. For configurations which may need to change by company or based on a specific website, the configuration is done from Tryton on the company or in website settings.

Each website that you plan to have needs to have a subdirectory within the application’s python source tree. By convention a static subdirectory with the static content like CSS and JS is also created within it. While this can be changed you usually don’t have to, especially when getting started.

Continue to Installation, the Quickstart, or the If you already use Flask.

1.2 If you already use Flask

If you are already used to programming in Flask you would be keen to know the main differences Nereid has with Flask. If you do not already use Flask, and is not familiar with Tryton this document might not make a lot of sense to you.

1.2.1 URL Routes

Flask applications use decorators to bind a function that should handle the request to a URL pattern. A basic example of this is:

```python
@app.route('/

    def index():
        return 'Index Page'

@app.route('/hello')
    def hello():
        return 'Hello World'
```

But, in the case of Nereid, the request handlers are methods in Model classes loaded in a Tryton Pool. In addition, to maintain the modularity of Tryton (where modules loaded are not modules in the python path, but the modules installed in a specific database), the use of decorators even using Flask blueprints is not possible. So for nereid, the URL rules are stored to a database:

```python
class Customer(ModelSQL, ModelView):
    
    A fictional customer model
    
    _name = "customer.customer"
    _description = __doc__

    def index(self):
        return 'Index Page'

    def hello(self):
        
        Render a profile page for the customer
        
        return 'Hello World'
```

and to create the URL rule, you could add them to an xml file
1.3 Installation

Nereid depends on a handful of Python libraries including Tryton.

So how do you get all that on your computer quickly? There are many ways you could do that, but the most kick-ass method is virtualenv. The Flask Documentation has a detailed section on using virtualenv to install Flask. You could refer to the same and then follow the instructions below.

1.3.1 virtualenv

virtualenvs are isolated Python environments.

If you are on Mac OS X or Linux, chances are that one of the following two commands will work for you in creating a virtualenv

$ sudo easy_install virtualenv

or even better

$ sudo pip install virtualenv

One of these will probably install virtualenv on your system. Maybe it’s even in your package manager. If you use Ubuntu, try

$ sudo apt-get install python-virtualenv

If you are on Windows and don’t have the easy_install command, you must install it first. Check the pip and distribute on Windows section for more information about how to do that. Once you have it installed, run the same commands as above, but without the sudo prefix.

Once you have virtualenv installed, just fire up a shell and create your own environment. I usually create a project folder and a venv folder within

$ mkdir myproject
$ cd myproject
$ virtualenv venv

New python executable in venv/bin/python
Installing distribute............done.
Now, whenever you want to work on a project, you only have to activate the corresponding environment. On OS X and Linux, do the following

```bash
$ . venv/bin/activate
```

If you are a Windows user, the following command is for you

```bash
$ venv\scripts\activate
```

Either way, you should now be using your virtualenv (notice how the prompt of your shell has changed to show the active environment).

Now you can just enter the following command to get Nereid activated in your virtualenv

```bash
$ pip install Nereid
```

A few seconds later and you are good to go.

### 1.3.2 System-Wide Installation

This is possible as well, though I do not recommend it. Just run `pip` with root privileges

```bash
$ sudo pip install Nereid
```

(On Windows systems, run it in a command-prompt window with administrator privileges, and leave out `sudo`.)

### 1.3.3 Living on the Edge

If you want to work with the latest version of Nereid, you can tell it to operate on a git checkout. Either way, virtualenv is recommended.

Get the git checkout in a new virtualenv and run in development mode

```bash
$ git clone http://github.com/openlabs/nereid.git
Initialized empty Git repository in ~/dev/nereid/.git/
$ cd nereid
$ virtualenv venv --distribute
New python executable in venv/bin/python
Installing distribute............done.
$ . venv/bin/activate
$ python setup.py develop
... 
Finished processing dependencies for Nereid 
```

This will pull in the dependencies and activate the git head as the current version inside the virtualenv. Then all you have to do is run `git pull origin` to update to the latest version.

### 1.3.4 pip and distribute on Windows

On Windows, installation of `easy_install` is a little bit trickier, but still quite easy. Read the section on pip and distribute on Windows on the Flask documentation for a better understanding.
1.3.5 Cloning for Development

If you are cloning the repository for development or updating the documentation, you also need to initialise the git submodules for the theme used in the documentation:

```bash
$ git clone http://github.com/openlabs/nereid.git
Initialized empty Git repository in ~/dev/nereid/.git/
$ cd nereid
$ git submodule init
Submodule 'docs/_themes' (git://github.com/openlabs/flask-sphinx-themes.git) registered for path 'docs/_themes'
$ git submodule update
Submodule path 'docs/_themes': checked out 'revision #'
```

1.4 Quickstart

Eager to get started? This page gives a good introduction to Nereid. It assumes you already have Nereid installed. If you do not, head over to the Installation section.

1.5 A minimal application

A minimal Nereid application first requires a Tryton database with the Nereid module installed. If you already have a database with Nereid installed head over to creating website.

1.5.1 Setting up a database

TODO

1.5.2 Creating a new website

Once the nereid module is installed in a Tryton database, open the Websites menu under Nereid/Configuration, and create a new website with the following settings.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>abcpartnerportal.com</td>
</tr>
<tr>
<td>URL Map</td>
<td>Choose Default</td>
</tr>
<tr>
<td>Company</td>
<td>Choose any</td>
</tr>
<tr>
<td>Default Language</td>
<td>English</td>
</tr>
<tr>
<td>Guest User</td>
<td>Create a new Nereid User</td>
</tr>
<tr>
<td>App User</td>
<td>Create or choose a User</td>
</tr>
</tbody>
</table>

Refer to the trytond_nereid.routing.WebSite for details on what each of the fields mean.

Tip: Since version 2.0.0.3 the name of the website is used by the WSGI dispatcher to identify the website that needs to be served. When you test the site locally, it is not usually possible to mimic your production url. This can be overcome by using a simple WSGI middleware which overwrite HTTP_HOST in the environ.
1.5.3 Creating the application and template

Once the website is created, a python script which loads nereid and runs the application needs to be written. This script is used to load Nereid, configure your application settings and also serves as an APP_MODULE if you plan to use WSGI HTTP servers like Gunicorn

```python
#!/usr/bin/env python
from nereid import Nereid

CONFIG = dict(

    DATABASE_NAME = 'nereid',
    STATIC_FILEROOT = 'static/',
    TRYTON_CONFIG = '../etc/trytond.conf',
    DEBUG = False,
    TEMPLATE_LOADER_CLASS = 'nereid.templating.FileSystemLoader',
    TEMPLATE_SEARCH_PATH = '.',
)

# Create a new application
app = Nereid()

# Update the configuration with the above config values
app.config.update(CONFIG)

# Initialise the app, connect to cache and backend
app.initialise()

class NereidHostChangeMiddleware(object):
    '''
    A middleware which alters the HTTP_HOST so that you can test
    the site locally. This middleware replaces the HTTP_HOST with
    the value you prove to the :attr: site
    :
    :param app: The application for which the middleware needs to work
    :param site: The value which should replace HTTP_HOST WSGI Environ
    '''
    def __init__(self, app, site):
        self.app = app
        self.site = site

    def __call__(self, environ, start_response):
        environ['HTTP_HOST'] = self.site
        return self.app(environ, start_response)
```
if __name__ == '__main__':
    # The name of the website
    site = 'abcpartnerportal.com'

    app.wsgi_app = NereidHostChangeMiddleware(app.wsgi_app, site)
    app.debug = True
    app.static_folder = '%s/static' % site
    app.run('0.0.0.0')

You can now test run the application

$ python application.py

The above command launches a single threaded HTTP Server for debugging purposes which listens to the port 5000. Point your browser to localhost:5000 and you should now be able to see a debug screen, with the ~jinja2.exceptions.TemplateNotFound Exception and its traceback. This is because you have not defined the template yet.

1.5.4 Defining the templates

For this quickstart section we will load the templates from the filesystem as we have used the FileSystemLoader as Template Loader in the application config. The template loader looks up templates in the folder for the site that is currently being rendered. In this case the template would be abcpartnerportal.com/home.jinja.

Create a template file home.jinja in the folder abcpartnerportal.com and fill in the following code

```html
<html>
<body>
  <h1>Welcome to Nereid</h1>
</body>
</html>
```

Run the application again and you should be able to see the rendered HTML on your browser at localhost:5000

1.6 Tutorial

TODO
If you are looking for information on a specific function, class or method, this part of the documentation is for you.

2.1 API Reference

2.1.1 Application Object

class nereid.Nereid(**config)

... Unlike typical web frameworks and their APIs, nereid depends more on configuration and not direct python modules written along the APIs. Most of the functional code will remain on the modules installed on Tryton, and the database configurations.

... add_ctx_processors_from_db()

- Adds template context processors registers with the model nereid.template.context_processor

- cache_default_timeout

  The default timeout to use if the timeout is not explicitly specified in the set or set many argument

- cache_dir

  The directory where cache files are stored if FileSystemCache is used

- cache_init_kwargs

  If a custom cache backend unknown to Nereid is used, then the arguments that are needed for the initialisation of the cache could be passed here as a dict

- cache_key_prefix

  A prefix that is added before all keys. This makes it possible to use the same memcached server for different applications. Applies for: MecachedCache, GAEMemcachedCache. If key_prefix is none the value of site is used as key

- cache_memcached_servers

  A list or tuple of server addresses or alternatively a memcache.Client or a compatible client.

- cache_threshold

  The maximum number of items the cache stores before it starts deleting some items. Applies for: SimpleCache, FileSystemCache
**cache_type**

The type of cache to use. The type must be a full specification of the module so that an import can be made. Examples for werkzeug backends are given below

```
```

**create_jinja_environment()**

Extend the default jinja environment that is created. Also the environment returned here should be specific to the current website.

**create_url_adapter**(request)

Return the URL adapter for the website instead of the standard operation of just binding the environ to the url_map

**database**

Return connection to Database backend of tryton

**database_name**

The name of the database to connect to on initialisation

**dispatch_request**()

Does the request dispatching. Matches the URL and returns the return value of the view or error handler. This does not have to be a response object.

**get_method**(model_method)

Get the object from pool and fetch the method from it

model_method is expected to be ‘<model>.<method>’. The returned function/method object can be stored in the endpoint map for a faster lookup and response rather than looking it up at request time.

**initialise()**

The application needs initialisation to load the database connection etc. In previous versions this was done with the initialisation of the class in the __init__ method. This is now separated into this function.

**initialised = False**

boolean attribute to indicate if the initialisation of backend connection and other nereid support features are loaded. The application can work only after the initialisation is done. It is not advisable to set this manually, instead call the initialise()

**jinja_loader**

Creates the loader for the Jinja2 Environment

**load_backend**()

This method loads the configuration file if specified and also connects to the backend, initialising the pool on the go

**load_cache**()

Load the cache and assign the Cache interface to

**load_websites**()

Load the websites and build a map of the website names to the ID in database for quick connection to the website

**pool**

A proxy to the _pool

**request_class**

The class that is used for request objects. See Request for more information.
alias of Request

**response_class**
The class that is used for response objects. See Response for more information.

alias of Response

**root_transaction**
Allows the use of the transaction as a context manager with the root user.

**select_jinja_autoescape(filename)**
Returns True if autoescaping should be active for the given template name.

**session_interface = <nereid.sessions.NereidSessionInterface object at 0x33d3810>**
the session interface to use. By default an instance of NereidSessionInterface is used here.

**transaction(http_host)**
Allows the use of the transaction as a context manager. The transaction created loads the user from the
known websites which is identified through the http_host

**translations_path**
The location where the translations of the template are stored

**tryton_configfile**
Configuration file for Tryton. The path to the configuration file can be specified and will be loaded when
the application is initialised

**wsgi_app(envirron, start_response)**
The actual WSGI application. This is not implemented in __call__ so that middlewares can be applied
without losing a reference to the class. So instead of doing this:

```python
app = MyMiddleware(app)
```

It’s a better idea to do this instead:

```python
app.wsgi_app = MyMiddleware(app.wsgi_app)
```

Then you still have the original application object around and can continue to call methods on it.

In Nereid the transaction is introduced after the request_context is loaded.

**Parameters**
- environ – a WSGI environment
- start_response – a callable accepting a status code, a list of headers and an optional ex-
  ception context to start the response

### 2.1.2 Templating

**class nereid.templating.TrytonTemplateLoader(app)**
Loaders are responsible for loading templates from a resource. In this case the template is loaded from Tryton
Database

**Parameters** app – the application instance

**get_source(environment, template)**
The environment provides a get_template method that calls the loader’s load method to get the Template
object. Hence, to override and implement a custom loading mechanism this method has to be sublassed.

It’s passed the environment and template name and has to return a tuple in the form (source, filename, 
uptodate)
source - source of the template as unicode string/ASCII bytestring filename - None (as not loaded from filesystem) uptodate - A function to check if template has changed

```
nered.templating.render_template(template_name, _prefix_website=True, **context)
```

Renders a template and by default adds the current website name as a prefix to the template name thereby allowing automatic name conversions from nereid base template to application specific templates

### 2.1.3 Backend - Tryton Connection

**nered.backend**

Backed - Tryton specific features

**copyright**

3. 2010-2012 by Openlabs Technologies & Consulting (P) Ltd.

**license** GPLv3, see LICENSE for more details

### 2.1.4 Testing Helpers

**nered.testing**

Implements test support helpers. This module is lazily imported and usually not used in production environments.

**copyright**

3. 2010-2013 by Openlabs Technologies & Consulting (P) Ltd.

**copyright**

3. 2010 by Armin Ronacher.

**license** GPLv3, see LICENSE for more details.

```
class nered.testing.NereidTestApp(**config)

A Nereid app which works by removing transaction handling around the wsgi app

```

**jinla_loader**

The jinja loader needs to be implemented by the test client and provided by the test case.

**load_backend()**

Just reuse the pool and DB already loaded by the tryton test loader

**root_transaction**

There is no need of a separate root transaction as everything could be loaded in the transaction context provided in the test case

**wsgi_app (environ, start_response)**

The actual WSGI application. This is not implemented in `__call__` so that middlewares can be applied without losing a reference to the class. So instead of doing this:

```
app = MyMiddleware(app)
```

It’s a better idea to do this instead:

```
app.wsgi_app = MyMiddleware(app.wsgi_app)
```

Then you still have the original application object around and can continue to call methods on it.

In Nereid the transaction is introduced after the request_context is loaded.
Parameters

- `environ` – a WSGI environment
- `start_response` – a callable accepting a status code, a list of headers and an optional exception context to start the response

### 2.2 Tryton Module API

#### 2.2.1 Website Object
Additional Notes

Design notes, legal information and changelog are here for the interested.

3.1 Version Management

The Version number of Nereid and its module has two parts. The first part indicates the version of Tryton the module is written for and the second part indicates the revision of the module for that series of Tryton.

Let us consider the version number 2.0.0.3, 2.0 indicates that the module works with the Version 2.0 of Tryton and that the module is in the 0.3 revision for the 2.0 series of Tryton.

When a newer version of Tryton is stable and released by the maintainers, nereid will also be updated to be compatible for the new version. When this is done, the previous version of the module is pushed to maintenance branch.

When updating to a newer series of Tryton, the minor revision begins again from 0.1. For example, if 2.4.0.6 is the last stable version of a nereid module in the 2.4 series and an update to 2.6 series is made, the version of the module would be 2.6.0.1.

3.1.1 Development Versions

To differentiate between a stable version and a development version Tryton uses BSD style even and odd numbers. For example 2.3 indicates a development release of tryton while 2.4 would be the stable version resulting from such development. However, it is not feasible to use the same with Nereid since, we would begin to migrate modules only after a stable version of Tryton would be released.

Therefore, it is necessary to identify if the revision you see on the repository is a stable tip or a development one. For the same, we suffix the version number with dev. Note that the development version 2.4.0.1dev is a version before 2.4.0.1:

```python
>>> from pkg_resources import parse_version
>>> parse_version('2.4.0.1dev') < parse_version('2.4.0.1')
True
```

3.1.2 Maintenance Branch

Maintenance branches of current revisions are created whenever a newer version of Tryton is available and the module is updated to the newer version. As of this writing the latest available stable release of Tryton is 2.4 and the upcoming 2.6 release has a major change to include the active record design pattern. When the 2.6 version is released the current
master branch will be checked out to a new branch called 2.4-maintenance and no further active development will be done on it.

At any point of time, the number of previous releases maintained would be the same as that of Tryton. For example if Tryton release management team stops maintaining version 2.0, the same would happen with Nereid.

**Note:** Critical bug fixes and security updates will be performed on maintenance releases

### 3.2 GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc.  [<http://fsf.org>](http://fsf.org) Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

#### 3.2.1 Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program—to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers’ and authors’ protection, the GPL clearly explains that there is no warranty for this free software. For both users’ and authors’ sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users’ freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to make the program non-free.
The precise terms and conditions for copying, distribution and modification follow.

### 3.2.2 TERMS AND CONDITIONS

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.


The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.
The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.


No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work’s users, your or third parties’ legal rights to forbid circumvention of technological measures.


You may convey verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a The work must carry prominent notices stating that you modified it, and giving a relevant date.

b The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.

c You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the
Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make
them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature
extensions of the covered work, and which are not combined with it such as to form a larger program, in or
on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting
copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual
works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts
of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also
convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a Convey the object code in, or embodied in, a physical product (including a physical distribution
medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily
used for software interchange.

b Convey the object code in, or embodied in, a physical product (including a physical distribution
medium), accompanied by a written offer, valid for at least three years and valid for as long as you
offer spare parts or customer support for that product model, to give anyone who possesses the object
code either (1) a copy of the Corresponding Source for all the software in the product that is covered by
this License, on a durable physical medium customarily used for software interchange, for a price no
more than your reasonable cost of physically performing this conveying of source, or (2) access to copy
the Corresponding Source from a network server at no charge.

c Convey individual copies of the object code with a copy of the written offer to provide the Corresponding
Source. This alternative is allowed only occasionally and noncommercially, and only if you received the
object code with such an offer, in accord with subsection 6b.

d Convey the object code by offering access from a designated place (gratis or for a charge), and offer
equivalent access to the Corresponding Source in the same way through the same place at no further
charge. You need not require recipients to copy the Corresponding Source along with the object code. If
the place to copy the object code is a network server, the Corresponding Source may be on a different
server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain
clear directions next to the object code saying where to find the Corresponding Source. Regardless of what
server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as
needed to satisfy these requirements.

e Convey the object code using peer-to-peer transmission, provided you inform other peers where the ob-
ject code and Corresponding Source of the work are being offered to the general public at no charge under
subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a
System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is
normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation
into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in
favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or
common use of that class of product, regardless of the status of the particular user or of the way in which the
particular user actually uses, or expects or is expected to use, the product. A product is a consumer product
regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses
represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other
information required to install and execute modified versions of a covered work in that User Product from

3.2. GNU GENERAL PUBLIC LICENSE
a modified version of its Corresponding Source. The information must suffice to ensure that the continued 
functioning of the modified object code is in no case prevented or interfered with solely because modification 
has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and 
the conveying occurs as part of a transaction in which the right of possession and use of the User Product is 
transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), 
the Corresponding Source conveyed under this section must be accompanied by the Installation Information. 
But this requirement does not apply if neither you nor any third party retains the ability to install modified 
object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide sup-
port service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User 
Product in which it has been modified or installed. Access to a network may be denied when the modifica-
tion itself materially and adversely affects the operation of the network or violates the rules and protocols for 
communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in 
a format that is publicly documented (and with an implementation available to the public in source code form), 
and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one 
or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as 
though they were included in this License, to the extent that they are valid under applicable law. If additional 
permissions apply only to part of the Program, that part may be used separately under those permissions, but the 
entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from 
that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain 
cases when you modify the work.) You may place additional permissions on material, added by you to a covered 
work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if 
authorized by the copyright holders of that material) supplement the terms of this License with terms:

a Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; 
or

b Requiring preservation of specified reasonable legal notices or author attributions in that material or in 
the Appropriate Legal Notices displayed by works containing it; or

c Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such 
material be marked in reasonable ways as different from the original version; or

d Limiting the use for publicity purposes of names of licensors or authors of the material; or

e Declining to grant rights under trademark law for use of some trade names, trademarks, or service 
marks; or

f Requiring indemnification of licensors and authors of that material by anyone who conveys the material 
(or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that 
these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 
10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this 
License along with a term that is a further restriction, you may remove that term. If a license document contains 
a further restriction but permits relicensing or conveying under this License, you may add to a covered work 
material governed by the terms of that license document, provided that the further restriction does not survive 
such relicensing or conveying.
If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.


You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License,
of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.


If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.
The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS
n
nereid,??
nereid.backend,??
nereid.testing,??

t
trytond_nereid.routing,??