The Cclite Manual

Introduction

Cclite is software, written in Perl, for community currencies and local exchange trading systems (LETS). It allows multiple currencies within one trading group and exchange between different trading groups. It has multiple interfaces and (recently) an osCommerce™ (http://www.oscommerce.com/) gateway. This manual has (at least) two distinct purposes:

- Manual for potential and actual users of Cclite
- Explanation of philosophy, design decisions and directions

The current shape of the software probably reflects my view of alternative currencies. It isn't the only one and I've included a web resources chapter at the end, for those who want to read about origins, theory and controversies.

Thanks to Mary Fee (of Letslink Uk), Michael Linton (of LETS and openmoney), Saul Albert and Jo Walsh and all others with whom I've been discussing this sporadically.

Thanks to Sook-Yin Chow in whose appartment I settled down to start writing this. I have also been in the Da Lat in Belleville (opposite the metro) and 149 Avenue de Choisy eating pho bo tai, the MSG works wonders too. Now back in London, I've been eating next door to the Geffrye Museum, also. I'd also recommend Govinda's run by the Hare Krishna temple in Soho Street and 'Indian Veggie' at the far end of Chapel Market in Islington. If you want meat, then the Lahore in Umberston Street is pretty famous, why I just ate there last night. The next version of this will be restaurant guide rather than a manual.

Getting Help with Cclite

If you want to help with Cclite use Googlegroups: `http://groups.google.com/group/cclite` you may need to be patient. For other questions use the contact form at `http://www.hughbarnard.org`

Conventions Used

Words that are in `courier` are technical, names of scripts, modules, files etc.

Words in *italics* are example names and values.

Word in <words> angles are values to be filled.

In general, command line, program fragments etc. in the text are picked out with a shadowed border.

Words and phrases in **Bold** are usually names of menu tab and entries, for example: **Create Currency**

We don't want any of that BNF stuff (`http://en.wikipedia.org/wiki/Backus-Naur_form`) in this manual!

Buying this manual as a dead tree will support this work and it won't be popular enough to cause any significant deforestation either.

1. Installation

Environment and Installation Overview

The first part of this section deals with upgrading, and the system and software environment for Cclite. You need to read this to se whether you can install it within on the server or within the hosting account that you have chosen.

For installation itself, there are three parts:

- Unpacking and Installation
- Configuration
- Setting up a Registry (you need at least one)
- Setting up a Currency within a Registry

The software now has a web installer, which (I hope) makes some of the configuration easier and more accessible outside the command line. This is the third version of the installer and there are probably still some rough edges.

Within a registry; the registry manager must log on and set up (at least) one valid currency.

There's a complete review of changes in this version in `doc/release.txt` and at the end of this manual.

Upgrade to 0.5.0 from Previous Version

These are the main steps. If you have modified templates etc., you may need to save and restore them. Since the user table has been expanded, for example, it needs to use the 'new' templates.

1. Save existing registries before step 2!

2. Apply the contents of `registry_upgrade1.sql` and possibly `registry_upgrade.sql` (if you haven't upgraded for a while) to any existing registries

3. Install this version via `tar` etc., see Unpacking and Installation later in this section.

4. Remove or rename any existing `cclite.cf` (which contains out of date information) and create again using `ccinstall.cgi`

Environment Overview

The current version of Cclite was developed under its own (virtual server) web root on Redhat Linux 9 and more recently Fedora FC6. This means, for example, that the web directory unpacks as `public_html` rather than `html` (Suse).

It can probably be mixed with another application under the same web root but my preference would be subdomaining (which is also a possible to solution to community currency namespace and discovery), for example: `cclite.yourdomain.com`.

Requirements

You need to be able to install Perl modules from CPAN (http://www.cpan.org) on your server. However, the Apache configuration is a standard configuration and therefore should be OK in a hosted environment. Cclite needs this software:

- A webserver, it’s being developed using Apache 2.0
- Perl interpreter, probably at about 5.6.x or above
- MySQL at 4.0 or above; uses InnoDB tables (to allow transactions, later)
- The Perl modules listed in the following table

Perl Modules for Cclite

Soap::Lite	http://www.soaplite.com only if configured as multiregistry or using SMS collecter readsms_from_gammu.pl remotely
Digest	
Digest::SHA2	Both Soaplite and Digest need some support packages, if you use perl -MCPAN -e shell or Webmin, they will be suggested at install time.
Bit::Vector	
DBI	You need this for the MySQL interface, it's often supplied with commodity hosting.
If you want to try and use RSS, this is needed for Ccrss.pm:	
XML::RSS;	
XML::Simple;	
LWP::Simple;	I suspect this is dragged in via SOAP anyway!

From this version 0.4.0 on, if you configure as multiregistry=no, you don't need SOAP::Lite. But you may need it for readsms_from_gammu.pl

The easiest way to install the modules is using the Perl CPAN module and the shell command or apt-get install on Debian based systems such as Ubuntu:

```
perl -MCPAN -e shell

>install Soap::Lite
```

etc.

SMS Interface

The SMS interface also needs gammu, a Linux utility for handling mobile phones. This is described later. It replaces `gsmlib` because it handles more phone, it's a server and it's actively maintained.

MySQL Server

`MySQL` can be installed as an `rpm` or `deb` package on distributions that accept this package format.

REST Interface

If you use the REST interface you need `AllowOverride All` in the Apache configuration for the `public_html` directory., this will let the rewrite rules in `.htaccess` to work.

Additional Tools

During development, I used `webmin` and `phpmyadmin`. I use `webmin` on the test servers to set up payment mailboxes and `cron` jobs.

This is a matter of taste but a good choice for less command line oriented users. I don’t use `phpmyadmin` on test servers because, hopefully) there’s no need to change the databases around too much and this can be a major security threat to (at the very least) the databases.

Unpacking and Installation

Follow these steps:

1. Download `cclite.tar.gz` and put into a directory in `/tmp` or directly under the new home directory for the domain or subdomain. This is the root directory not the html directory!

2. `tar –xzvf cclite.tar.gz` to unpack the software

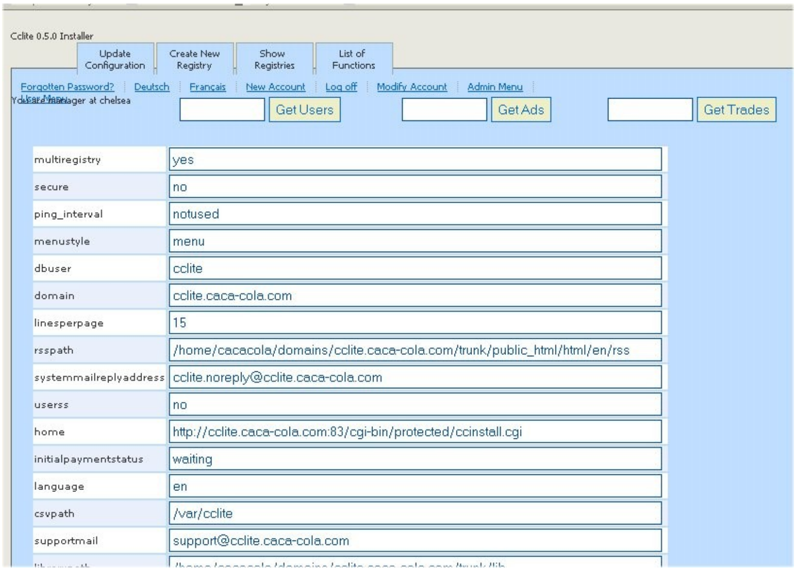
3. Move the directories or the contents of directories (if there’s something else within the web root or the web root is not `public_html`) to the web root.

4. Change all the `*.cgi` permissions to world read and world execute

Since there's now an experimental deb package and a tar file that's suitable for debian (therefore Ubuntu) style installions, please read the notes for versions marked 0.5.1

Configuration

5. Go to:
   `http://<yourdomain>/cgi-bin/protected/ccinstall.cgi`
   this gives error messages for missing libraries and common problems:



multiregistry	yes
secure	no
ping_interval	notused
menustyle	menu
dbuser	cclite
domain	cclite.caca-cola.com
linesperpage	15
rsspath	/home/cacacola/domains/cclite.caca-cola.com/trunk/public_html/html/en/rss
systemmailreplyaddress	cclite.noreply@cclite.caca-cola.com
userss	no
home	http://cclite.caca-cola.com:83/cgi-bin/protected/ccinstall.cgi
initialpaymentstatus	waiting
language	en
csvpath	/var/cclite
supportmail	support@cclite.caca-cola.com

The `config` subdirectory will need to be writable for this to create or update the configuration file (`cclite.cf`), otherwise you can cut and paste the resulting configuration with an editor, for example.

Cclite will try and guess most of the parameters based on the domain and the physical path where the software is installed. Here's a key to the values:

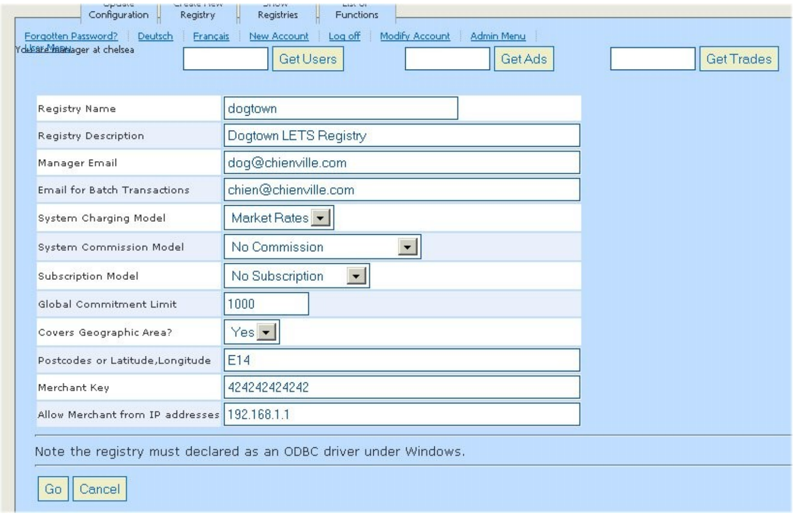
Name	**Values**	**Comments**
multiregistry	yes|no	Soap::Lite is not needed if you say no. But in this case there's no multiregistry transfer
secure	no	not used, at present
linesperpage	15	no of items in table listings
systemmailreplyaddress	cclite.noreply@cclite.caca-cola.com	no reply address for mail messages
initialpaymentstatus	waiting|accepted	Status of payment. If you put this to accepted, there's no accept step.
dbpassword		Mysql user password
literalspath	/home/cacacola/domains/cclite.caca-cola.com/literals	Path to literals files
htmlpath		Path to html root. Unused at present.
initialuserstatus	unconfirmed|active	Initial status of user. If you want email confirmation, set to unconfirmed
templates	/home/<directory>../templates/html	Path to templates. Each language has a subdirectory. Above web root
ping_interval	notused	Not used currently.
menustyle	menu	Leave this setting.
domain	cclite.<yourdomain>.com	Your domain
dbuser	cclite	Mysql user
home	http://<yourdomain>/cgi-bin/cclite.cgi	Home for user access.
language	en	Default language
librarypath	/home/<directory>../lib	Path to libraries. Above web root

supportmail	support@cclite.caca-cola.com	Support email, currently unused
systemmailaddress	cclite@cclite.caca-cola.com	System email, currently unused.
csvpath	/var/cclite for example	path for storing and processing comma separated variable files as batch.. See batch payments section.
userss	no	If set to 'yes' you can generate rss and will need the XML::RSS modules.
registrypublickey	`notused`	Public key for registry, Currently unused (and probably in the wrong place.
smspath		Where gammu will store sms message, must be the same as the gammu.cf inbox

6. Modify any configuration parameters and update. Leave any `notused` entries, as the form checks that everything has a value.

Set up a Registry

7. Use the **New Registry** entry to create a new registry. Here's an example, there's more in the Administration section:



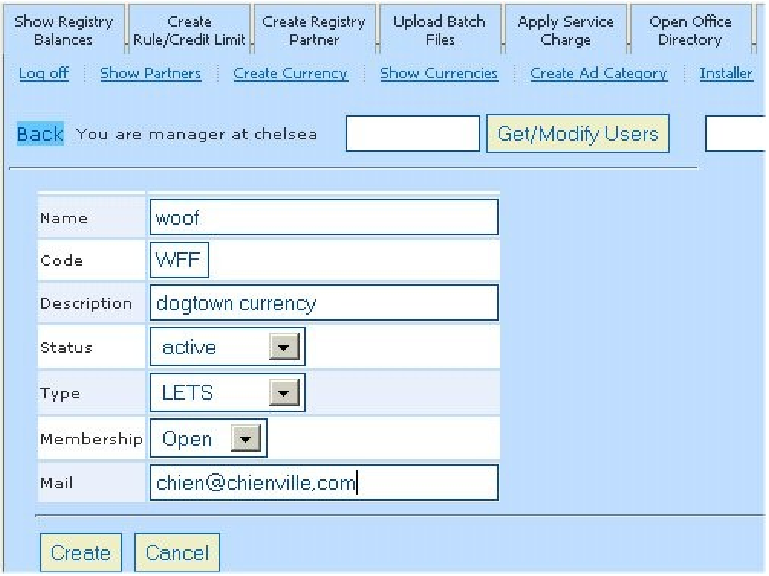
You are manager at chelsea

Registry Name	dogtown
Registry Description	Dogtown LETS Registry
Manager Email	dog@chienville.com
Email for Batch Transactions	chien@chienville.com
System Charging Model	Market Rates
System Commission Model	No Commission
Subscription Model	No Subscription
Global Commitment Limit	1000
Covers Geographic Area?	Yes
Postcodes or Latitude,Longitude	E14
Merchant Key	424242424242
Allow Merchant from IP addresses	192.168.1.1

Note the registry must declared as an ODBC driver under Windows.

Set Up a Currency

8. Go to `http://<yourdomain>/cgi-bin/cclite.cgi` login to the registry as manager, password '`manager`' and choose **Admin Menu.**
9. In the **Admin Menu**, choose **Create Currency** to create a new currency within your registry, here's an example:



Name	woof
Code	WFF
Description	dogtown currency
Status	active
Type	LETS
Membership	Open
Mail	chien@chienville.com

10. Please change the manager password and off-line `ccinstall.cgi` when you've finished installing. The web interface will display an error message if this is not done.
11. If using `rss` make sure the subdirectory used to write the feeds is writable by the server.

Advanced Install Topics

This describes the optional scripts that are non-web. Some require additional Perl modules too. They are mail payment, SMS payment and `rss` offers and wants. There is also (as of March 2006) a very primitive module for converting the offers/wants into Open Office format.

All these scripts are batch scripts and need a `cron` job to run. The `cron` job frequency depends on how much activity you expect in the hosted registries. These batch components are set up as follows:

- Configuration, currently at the top of the script
- Test batch file on command line, if possible
- Set up `cron` job
- Test complete tool chain

In the near future, the configuration will probably become a little more rational.

Installing Mail Payment

I don’t recommend this at the moment, it’s very primitive and insecure, but…You’ll need a valid mailbox for each active registry. These mailboxes are of form

`<registryname>.cclite@domain.com`.

What to Change

These are the items that to change at the top of `readmail.pl`.

```
# The library path needs to be absolute for the cron to
work...
# this path is correct for the debian package
# change this!:
use lib "/usr/share/cclite/lib";    # find the modules
#------------------------------------------------------
-
# change these two, if necessary
 my $mail_dir = "/var/spool/mail" ; # mail directory
 my $registry = "dalston" ; # change this to your
registry
#------------------------------------------------------
-
```

Both mail and SMS payments have the same format and use the main transaction routine in Cclite.pm. They both need to parse and translate the incoming textual data, which they do in a similar way.

The payment format for mail and SMS is:

```
     Send x <currency> to <username>
at <registryname> for <reason phrase>
```

For example:

```
     Send 4 duckets to ddawg at
dalston for barking lessons
```

Both mail and SMS only work within one registry, at present. The from registry and to registry is assumed to be the same.

Mail payments will be recorded with a status of `waiting`, if this is not changed in the configuration file, `cclite.cf`. SMS payments are always recorded as waiting at present.

The person making the payment is recognised by the `From:` email address (and that's one of a number of insecurities). Therefore the `From:` address must correspond to the address in a valid, active user for that registry.

Currently the mailboxes for the registries need to be added to the list at the top of the `readmail.pl` script. This will probably become configurable, shortly.

It's useful to run the script from the command line with some test email payments, to detect problems, before setting the cron.

Lastly a `cron` job to run the script needs to be setup, either via Webmin or directly via `crontab`.

Installing RSS Small Ads

This is also pretty immature at the moment. It has the same architecture as the other batch components. It needs userss=yes in `cclite.cf` which processes a require for `XML::Rss` . I've done this to keep the 'minimum' install somewhat simpler.

It needs extra Perl modules to handle the RDF and XML parts. Modify the top of the script `writerss.pl` to include the registries that have rdf feeds.

What to change

Here's what to change in `writerss.pl`:

```
# The library path needs to be absolute for the cron to work...
# change this!:
 use lib "/home/yourdomain/domains/cclite.yourdomain.com/lib" ;

# simply loop around each registry creating an rdf file
# for the type of advert for each one
#-------------------------------------------------------
# here, for example, all adverts in dalston, only offered
adverts in stokie
#
 my %registries = qw(dalston all stokie offered) ;
#-------------------------------------------------------
# this is the path where the rss files are written
# needs to be writable by the server
 my $path =
"/home/bigwaveheuristics.com/domains/cclite.bigwaveheuristics.c
om/public_html/html/en/rdf" ;
```

Test on the command line, if possible by using:

```
./writerss.pl
```

and then set up and test a `cron`, to automate the feed process.

Installing SMS payment

Mail and SMS payments have the same format and use the main transaction routine in Cclite.pm. They both parse and translate the incoming text data in a similar way.

readsms_from_gammu.pl requires, gammu (see http://www.gammu.org/wiki/index.php?title=Gammu:SMSD), a compatible telephone (Nokia is often OK) and the right cable.

Gammu runs as a server via gammu.sh (in the batch directory) to pick up the messages and readsms_from_gammu.pl puts them into the transaction process. readsms_from_gammu can use SOAP and isn't necessarily on the same system as the cclite core.

The top of the script may need modification for the library and path to gsmlib, depending on where gsmlib is found on your system.

What to Change

These are readsms.pl lines to change.

```
#-----------------------------------------------------    our
$configfile =
      '/usr/share/cclite/config/cclite.cf';
#-----------------------------------------------------
```

and:

```
# -----------------------change these if
necessary-----------------------------------

my $local = 0

  ; # set local = 1 if this script is on the same comput-
er as the rest of cclite

my $domain = '<put.remote.domain.here>';    # remote do-
main if the script is not local

my $sleep = 360 ; # sleep for three minutes between pass-
es..

my $sms_dir = $configuration{smspath};    #  sms inbox
directory

my $debug = 1 ; # give various displays including when it
'runs'

#--------------------------------------------------------
----------------------------
```

may need to be changed too.

As before, set up and test `readsms_from_gammu.pl` on the command line and then install, it's got a sleep loop in it, it doesn't need cron. At present, it’s only been tested with one incoming telephone number; against one registry. There’s an argument for multiple numbers, since this would reduce the cost of incoming texts by sorting them by phone provider.

SMS payments are recorded with a status of `waiting`. SMS payments also have a standard title field at present.

If you stop using the phone, turn off gammu and readsms_from_gammu.pl !

Installing Batch payment

`readcsv.pl` provides processing for comma separated variable files (CSV(which can be uploaded via the web. These files can be used for manually recorded off-line payments or transfer of data from spreadsheets or other systems, for example.

As with SMS and mail, this is run as a cron job. Change the location of the configuration in `batch/readcsv.pl`:

```
# change the location of the configuration
file here.
# needs to be hard coded because cron is run
from, for example, libexec
#
my %configuration ;
BEGIN {
%configuration =
readconfiguration('/home/yourdomain/domains/c
clite.yourdomain.com/trunk/config/cclite.cf')
;
```

Read about uploading the CSV file to the server in Batch Payment, that's how the data is supplied.

Installing/Enabling REST

An experimental REST interface has been added to this release (March 2006, April 2007). This is described in

It needs `AllowOverride` http://httpd.apache.org/docs/2.2/mod/core.html#allowoverride within the host or virtual host configuration usually `httpd.conf` if you're using Apache.

There’s enough to be useful but it's not complete (when is anything, ever complete?). The interface uses Apache rewrite rules in the web root (`public_html`) in the `.htaccess` file. Here's the example rule for paying somebody:

```
# experimental REST style rewrite for Cclite
RewriteEngine on

# /pay/ddawg/dalston/23/duckets
RewriteCond   %{HTTP_COOKIE}
      userLogin=(\w+).*token=(\S+)\b
RewriteRule   pay/(\w+)/(\w+)/(\w+)/(\w+)    /cgi-
bin/cclite.cgi?
action=transaction&subaction=om_trades&fromregistry=
$2&tradeSource=
%1&tradeDestination=$1&toregistry=$2&tradeCurrency=$
4&tradeAmount=$3&token=%2  [R]
```

This translates into an URL such as:

```
https://cclite.yourdomain.com/pay/ddawg/dalston/23/duck
et
```

Also, the user must be logged in before any operation can be done. The login and logoff operation is also in this interface.

Installation Troubleshooting

Here are some of the commoner problems and their solutions.

Problem	**Possible Solution**
Internal Server Error	The scripts need to be world executable `chmod a+x` or ask your web space provider about this.
REST interface doesn't work	The Apache configuration file `httpd.conf` may need to be changed to allow rewrite rules in `.htaccess`
Database upgrade to 4.0 or above needed	Your current MySQL installation needs to be at least 4.0
Database password or user name problem	There’s probably a user or password problem when connecting to the database
Ccinstall: Cclite installer Can't find or work out library path: does not exist or unreadable?	You need to check the path and permissions for the `lib` subdirectory against the display. All the scripts use the libraries in the `lib` subdirectory so nothing will work until this is fixed.
Ccinstall: Cclite install Can't find install template directory: $dir/templates/html/ $language/install does not	Can’t find the templates, these are needed for display. You need to check the

exist or unreadable?	path and permissions for the `templates` subdirectory against the display.lay.

2. Administration

This part describes some of the initial and day-to-day administration for a Cclite installation. It mainly concerns web administration.

Overview

Cclite is mainly for use by users, on the web (with payment via email and SMS). However, some communities may not have the web or use it partially, therefore a central administrator can do all the currency transfers as well as the administration.

Whatever the administration and process model, the set up order is:

- Set up registry or registries
- Set up currency (ies) within registry
- Set up users

Use by a central administrator

This is the more unusual case and requires a few modifications (these will probably get put into the set up, in a while):

- Remove the new user screen and link from public view
- Make all users set up as `active`, this can be done with `ccinstall.cgi` to change the configuration
- Make all payments `accepted`, , this can be done with `ccinstall.cgi` to change the configuration
- Remove all the user modification screens from public view
- Remove all the payment screens from public view
- Decide what to do about the yellow pages!

This gives a system where users can check their (and other peoples) balances on the web and the central administrator does the rest (add users, transfer currency, modify users etc:).

There’s also an intermediate option that users set themselves up and a central administrator activates them after induction; an identity check and other bits of process, for example.

Use by all Users

This is the expected mainstream use. Users set themselves up, confirm their email addresses to activate themselves and make their own transfers and create their own small ads.

The central administrator only sets up registries (usually one registry for a given scheme, which they administer), currencies and partner registries, for example.

The central administrator also applies the service charge, if it exists, since this isn't automatic currently.

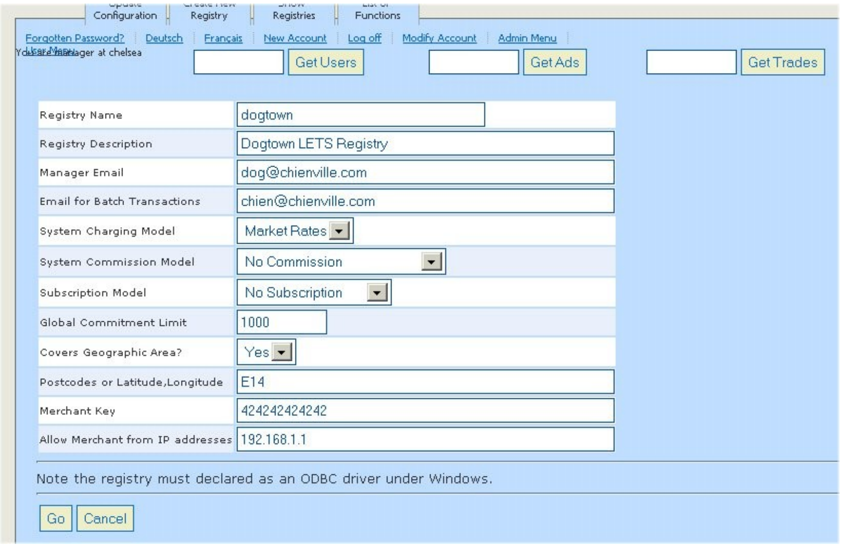
Create Registry

This is the first operation after software installation. Normally; a registry is named after the scheme that it supports. For example, registry *dogtown* is for the LETS scheme in Dogtown.

This is done via `ccinstall.cgi`, a separate script. It’s possible just to use `sql/registry.sql` via `mysql` for example.

> If you’re only planning to use one registry, remove the script afterwards. Take away execute permissions, if you leave it in place.

Look at http://yourdomain.com/`cgi-bin/protected/ccinstall.cgi`, link to the registry install:



Registry Name	dogtown
Registry Description	Dogtown LETS Registry
Manager Email	dog@chienville.com
Email for Batch Transactions	chien@chienville.com
System Charging Model	Market Rates
System Commission Model	No Commission
Subscription Model	No Subscription
Global Commitment Limit	1000
Covers Geographic Area?	Yes
Postcodes or Latitude,Longitude	E14
Merchant Key	424242424242
Allow Merchant from IP addresses	192.168.1.1

Note the registry must declared as an ODBC driver under Windows.

The global commitment limit is the maximum amount (currency independent) that a user can 'owe' in this registry. If present, it is tested before each transaction.

The postcodes or Latitude, Longitude field should contain comma separated lists.

The merchant's key is for use with osCommerce and needs to correspond to the merchant's key set up in the Cclite osCommerce gateway. This key permits straight through transactions from the shop and needs to be kept safely and changed if compromised.

Dropping a Registry

There’s no current way of doing this in Cclite. However, a registry is one database, so dropping the database will remove all traces of that registry.

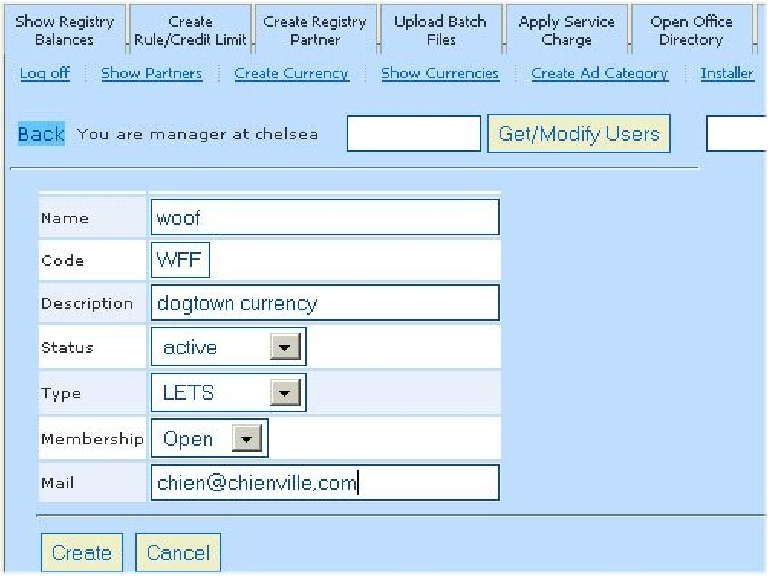
I’m unlikely to make this an ‘easy’ operation, even if it’s included (because these records will almost always need to be saved, for example, for tax purposes). I’ll probably just provide a database name change to something that’s agreed to be an inactive name.

Creating a Currency

First, a note, currencies *do not exist independently* from registries. They are part of a registry. This is a questionable decision for database structure but (I feel) a valid decision from the point of view of ‘currency ideology’.

The scheme or trading group (a group of *people*, as expressed by the registry component) is primordial and that gives rise to and contains a currency or currencies. Enough, already!

A currency is set up via the **Create Currency** link.



You are manager at chelsea

Name	woof
Code	WFF
Description	dogtown currency
Status	active
Type	LETS
Membership	Open
Mail	chien@chienville.com

A given currency is currently available to **all** members of a registry. There is a flag available to ‘close’ a currency, which will remove that currency from the drop down lists. It's then up to the administrator to produce 'private' transfer forms for the currency.

This debate is also linked to map-type representations of groups, currencies, offers, wants etc. that could be very handy. I’ll see which way this argument goes. I'm planning to add a latitude and longitude field to the registry records, so that they can be represented as presence on a map.

Setting Up Users

Users set themselves up via the **New Account** screen in the general menu. A central administrator could also use this function, if the scheme is administered from a central point.

Setting up users is described later in the manual.

Goods/Services

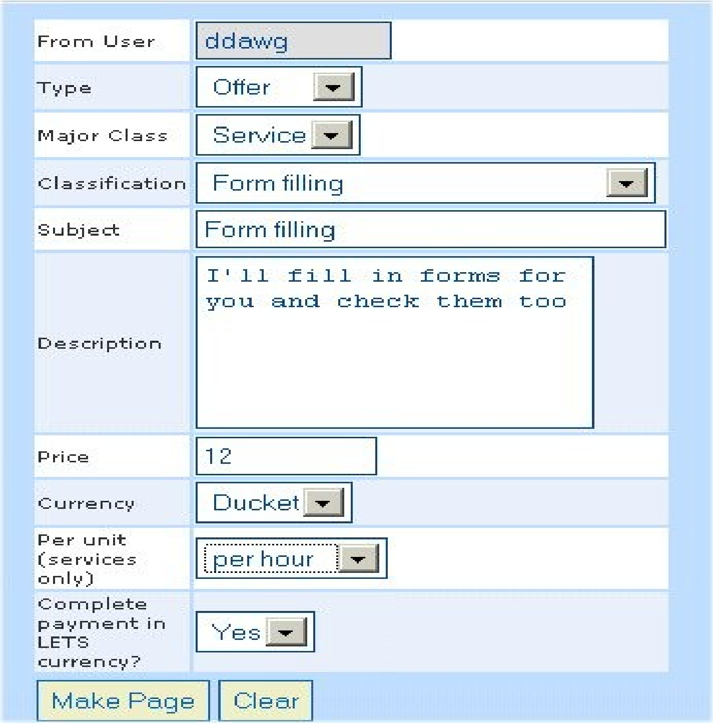
This is the least developed part of the software, currently. However, it’s convenient to have ‘everything’ necessary for a scheme within one package. In version 0.5.0 there's some development of the directory listing code. There are several development directions, if it stays:

- Generation of personal pages via cron, once per day. This is a heavy operation and also needs to be made multilingual.
- Addition of an interface to give pdf and OpenOffice (and therefore paper) publication. I’ll probably do this with perl pdf modules. The OpenOffice script partially exists as `ccdirectory.cgi` already.
- Standard classifications in the categories, so that there can be multi-registry searching (SOAP property anyway).
- The Strohalm foundation is also currently (summer 2006) thinking about classification standards.

Adding and Offer or Want

Click on the **Place Ad** tab and fill in the form. If the offer or want is completely payable in an alternative currency, choose the **True Lets** option.

Adverts are added to the users personal pages in chronological order. Possibly a cron job to clear them, every three months (say) might also be useful.

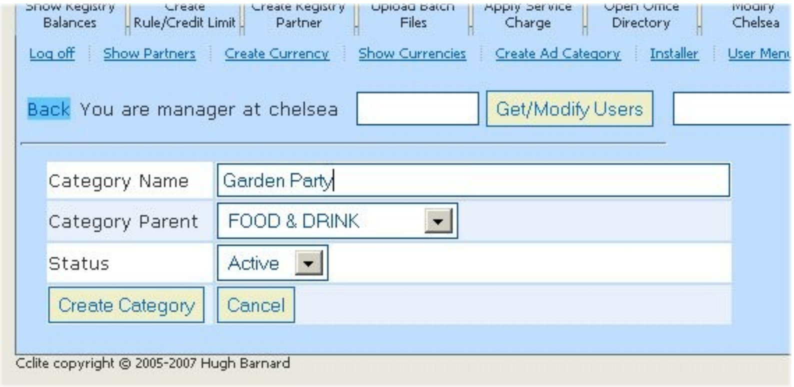


From User	ddawg
Type	Offer
Major Class	Service
Classification	Form filling
Subject	Form filling
Description	I'll fill in forms for you and check them too
Price	12
Currency	Ducket
Per unit (services only)	per hour
Complete payment in LETS currency?	Yes

Adding a Category

This is currently fairly flaky. Just click on the **Create Ad Category** link and fill in the form.

The category should then appear in the drop down for adding adverts. Don’t create millions of categories that only contain one thing!



You are manager at chelsea

Category Name	Garden Party
Category Parent	FOOD & DRINK
Status	Active

Cclite copyright © 2005-2007 Hugh Barnard

Linking Registries

This describes how to link Cclite registries so that payment can be made from an account in one registry to another. Thus, for example, someone from registry *islington* could transfer currency to registry *newham*.

For this; the two registries must be mutually linked (see below) and must share a common currency (this is simply a currency with exactly the same name, at present, there are no deeper semantics at present).

In the alternative currency universe, trading between registries or schemes (sometimes called intertrading) is a subject that is full of controversy. Sometimes it's disallowed, sometimes it's allowed via a single system transfer account etc. etc. In Cclite, it could be done via a system account (because that is the simplest case of 'many' accounts) but also can be done between individual accounts. There's no *need* to use this, at all!

I've built this into Cclite for example:

- It's nearly impossible to add as an afterthought
- It needn't be used in the majority of cases
- It needs experimentation, to check the computer science, usability, architectures etc.
- Reed's law and expanding number of trading systems
- I'm a believer in multiple differently scoped (regional, local) complementary currencies, this therefore fits somewhat with my personal model

I, personally, believe that the conventional currency problems are more linked to fractional reserve banking, derivatives, usury and money as a 'first class object' and not to this aspect. It’s also fair to say that *nobody knows*, since no-one has a *really big* alternative currency system in operation. Also, should any individual scheme be or become *really big* anyway?

Limitations and Comments

It’s clear that this mutual linking won’t scale into a large system. The next step, if it becomes popular is to provide a directory (could be a simple file, could be LDAP) of mutually accessible registries and currencies for a given region (say).

I am unsure about this, because it may lead to central control by whoever is ‘running’ the directory. The computer science is not hard, the problems of governance, politics and control are more so.

On practical problem with linked systems is that they begin to display tidal effects of conventional money. That is, affluence washes from system to system enriching one and impoverishing another.

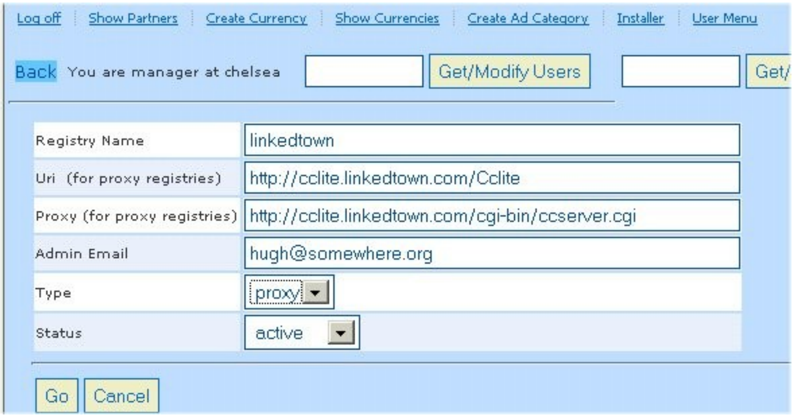
There's probably arguments for adding features for managing intersystem trade balances, perhaps via specialised 'exchangers', in future developments. I'm currently (November 2008) looking at Apache MQ and Net::Stomp but we'll see...

Partner Registry

To enable this type of transfer, a registry needs a partner registry. The software must be configured as `multiregistry` in both cases.

Each registry must have its opposite number as partner (*islington* needs *newham* **and** *newham* needs *islington*). This also corresponds somewhat to the deontology (go on, surf now, you know you want to: http://en.wikipedia.org/wiki/Deontological_ethics !) of the situation.

This is set up in the administration as follows. Currently it’s fairly insecure and both https and basic authentication for the SOAP exchanges can be used fairly easily, read the SOAP cookbook `http://cookbook.soaplite.com/` or contact me. If this gets used, further versions will probably have improved security as standard.



You are manager at chelsea

Registry Name	linkedtown
Uri (for proxy registries)	http://cclite.linkedtown.com/Cclite
Proxy (for proxy registries)	http://cclite.linkedtown.com/cgi-bin/ccserver.cgi
Admin Email	hugh@somewhere.org
Type	proxy
Status	active

Shared Currency

Both registries need a currency with exactly the same name (this is the only thing that makes the two currencies equivalent (be careful!)). This is set up in the administration screens using **Create Currency** as usual .

SOAP aspects

The SOAP transactions are transported via `http` (or `https`) in this version.

This is the simplest because it doesn’t require extra ports or create firewalling difficulties.

I've also recently made some experimental PHP clients for Cclite SOAP and discovered some difficulties associated with message format and passing hashes (these could just be my inexperience, but it's clear that it's 'hard' and therefore a bit inaccessible to people who 'just want it to work').

Therefore, with release 0.4.0, I've started to work on a REST (Representational State Transfer see

http://www.xfront.com/REST-Web-Services.html which gives a quick summary of this). See the section on REST payment etc. in the rest of the manual to see how this is progressing.

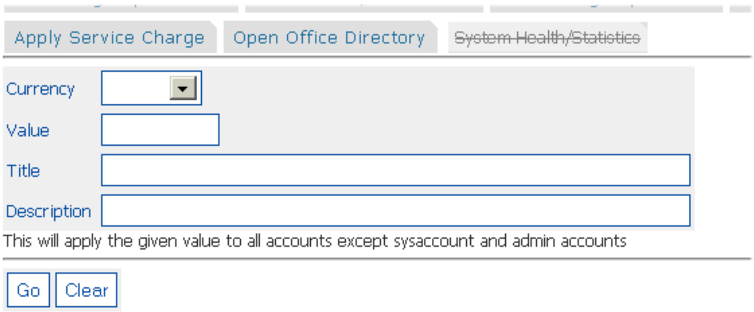
For Amazon, about 85% of ‘interfacers’ use REST see http://www.oreillynet.com/pub/wlg/3005 suggesting that it’s easier to understand and implement.

Apply Service Charge

This is a new function (as of June 2006). It will apply the service charge quoted to every non-manager non-sysaccount user within the given registry.

It puts a balancing transaction for each into `sysaccount` in the registry. Remember that `manager` is used for software management, currency creation etc. and should not participate in day-to-day transactions.

It operates as a standard transaction using the standard routine. There may be a need to condense the `sysaccount` entries into one single entry, for display convenience.



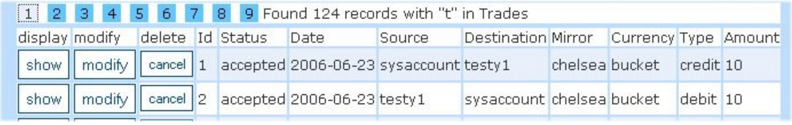
This will apply the given value to all accounts except sysaccount and admin accounts

There is also a `servicechargelimit` value in the configuration that can be used to prevent very large values being applied to the registry.

The service charge is applied per currency, rather than globally.

Cancelling Transactions

As of version 0.5.0, a manager can now search for any transaction (via the search box) and cancel the transaction:



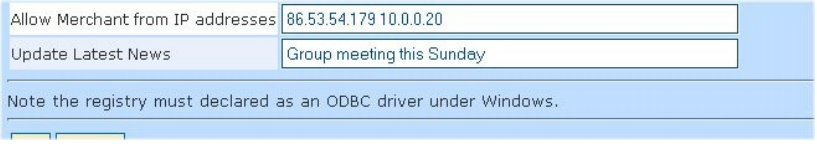
1 2 3 4 5 6 7 8 9 Found 124 records with "t" in Trades

display	modify	delete	Id	Status	Date	Source	Destination	Mirror	Currency	Type	Amount
show	modify	cancel	1	accepted	2006-06-23	sysaccount	testy1	chelsea	bucket	credit	10
show	modify	cancel	2	accepted	2006-06-23	testy1	sysaccount	chelsea	bucket	debit	10

This will modify the transaction status to cancelled and it will not be counted in volumes, balances etc. However, in the future, I believe that a registry with 'lots' (however we define that) of cancelled transactions should be regarded as dysfunctional.

Displaying News

This gives a one line text-only news display, use the field in your registry:



Allow Merchant from IP addresses	86.53.54.179 10.0.0.20
Update Latest News	Group meeting this Sunday

Note the registry must declared as an ODBC driver under Windows.

To remove the current news, blank out the line.

3. Payment

This, of course, is the central function for Cclite. Payment can be web, mail, SMS or batch. Payment can be made within a registry or to a partner registry that shares a currency. To set up partner registries, see the Administration chapter.

Payment can take place successfully or can fail with an explanatory message. Amongst the reason for failure are:

- Non-existent payee, check the spelling, or use auto-suggest for local transactions
- Over commitment limit, if there is one
- Problem at the remote registry, if using linked registries
- Various problems filling in the payment form

Of these, the problem with a remote registry may not lead to a diagnostic message. This is another weak area.

If payment is successful, a transaction reference will be returned. This is a long list of letters and numbers that is unique to the transaction.

For the technically minded, it's a hash of the transaction data using the SHA2. It uses SHA2 since MD5 and SHA1 are suspected of producing collisions (things will the same reference). On screen, it looks like this:

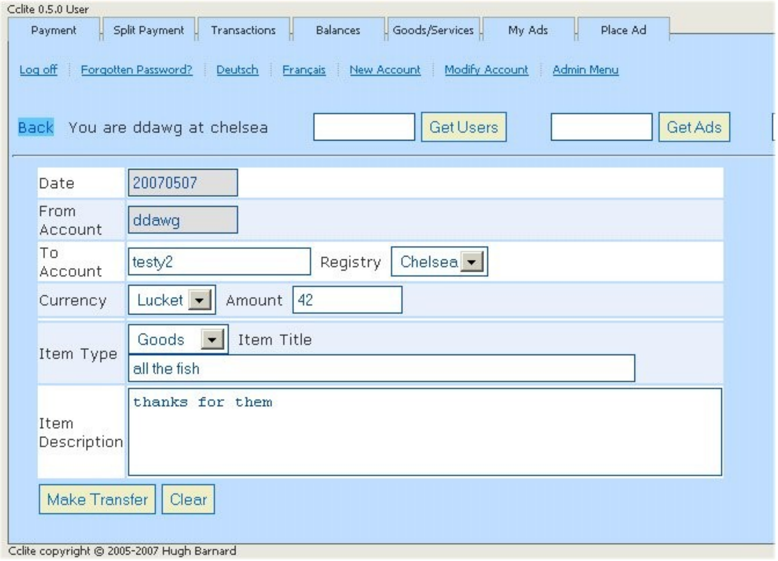
```
Transaction Accepted
Ref: YmH8Y7msvfZaPVWDCrljjmqF47bLcCiKV1zI4C7CEjHCm
XkGQIH6b1jq1rkD8lbaplObJ6lMcohApyR8Rc3pHQ
```

Web payment

From version 0.5.0 there's a drop down list of suggestions for the destination account, if you're using a local registry. Just type the first one or two letters and wait a moment. If you don't want this, remove or change the permissions of `ccsuggest.cgi`

The web payment form is for the logged in user. Registries and currencies that are available are presented in two drop downs. At present the registries and currencies are independent lists (otherwise there must be an intermediate transaction, when the registry is chosen), therefore a transaction can still fail because a currency is not supported in a given registry.

If payment is made to a distant registry, there may be a pause whilst the SOAP transaction completes (depending on the general environment for the distant registry too).



You are ddawg at chelsea

Date	20070507		
From Account	ddawg		
To Account	testy2	Registry	Chelsea
Currency	Lucket	Amount	42
Item Type	Goods	Item Title	all the fish
Item Description	thanks for them		

Cclite copyright © 2005-2007 Hugh Barnard

The Registry menu will list all registries that are set up as partners of the current registry. The Currency menu will list all 'open' currencies.

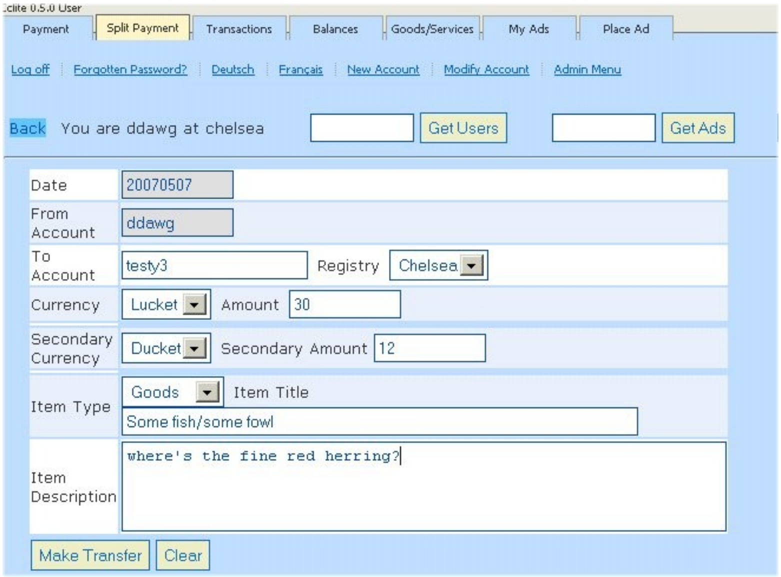
Split Web payment

This function is designed to aid systems and schemes that want to make payment for a single product or service in two different currencies. The most obvious example, is partial payment in national currency and partial payment in a community currency.

Both are recorded, within a registry using the same transaction reference (this is the transaction reference of the primary currency transaction).

The split web payment form is for the logged in user. The transaction is very similar to web payment (including the auto-suggest), but there are two drop downs for currencies and quantities. At present the registries and currencies are independent lists, therefore a transaction can still fail because a currency is not supported in a given registry.

If payment is made to a distant registry, there may be a pause whilst the SOAP transaction completes.



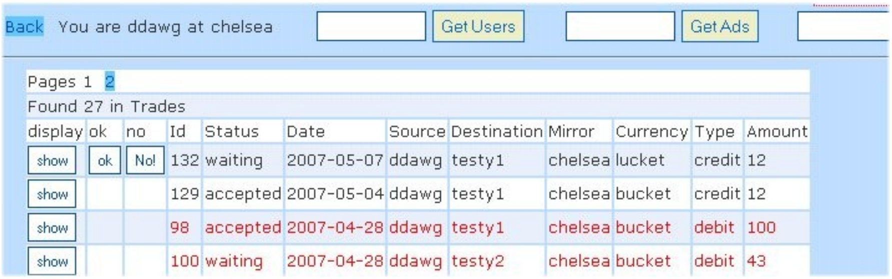
You are ddawg at chelsea

Date	20070507		
From Account	ddawg		
To Account	testy3	Registry	Chelsea
Currency	Lucket	Amount	30
Secondary Currency	Ducket	Secondary Amount	12
Item Type	Goods	Item Title	Some fish/some fowl
Item Description	where's the fine red herring?		

The **Registry** menu lists all registries that are set up as partners of the current registry. The **Currency** menu lists all 'open' currencies.

Waiting for approval

By default, a completed payment appears in the partner account as a `waiting` transaction. The receiving user clicks on the `OK button` in the transaction list to complete the transaction (or declines it if not wishing to receive payment).



Pages 1 2

Found 27 in Trades

display	ok	no	Id	Status	Date	Source	Destination	Mirror	Currency	Type	Amount
show	ok	No!	132	waiting	2007-05-07	ddawg	testy1	chelsea	lucket	credit	12
show			129	accepted	2007-05-04	ddawg	testy1	chelsea	bucket	credit	12
show			98	accepted	2007-04-28	ddawg	testy1	chelsea	bucket	debit	100
show			100	waiting	2007-04-28	ddawg	testy2	chelsea	bucket	debit	43

This may not be useful in centrally run systems. To switch this off, see the installation chapter, it needs a change in `cclite.cf`: `initialpaymentstatus=waiting` to `accepted` (can be done using the installer). This means that a transaction is accepted by default.

Also if a registry has 'lots' of declined transactions that should be regarded as a problem for the registry.

Mail payment

Mail payment is made towards a queue belonging to the user’s home registry, for example *cclite.dalston@cclite.bigwavheuristics.com*

This mailbox is set up by hand, at present. The payment is from the mail address in the user’s record, otherwise it fails. The interface is insecure and experimental, at present. The payment format for mail and SMS is:

```
     Send x <currency> to <username>
at <registryname> for <reason phrase>
```

For example:

```
send 4 duckets to ddawg at dalston
for barking lessons
```

Mail payments also have a standard title fields, at present.

SMS Payment

SMS payment is made towards a gateway mobile number corresponding to the home registry (and perhaps to the user’s mobile provider, this reduces the payment cost).

The payment must come from the mobile phone number that the user has registered in their user record. Registration in the user record is in international format, for example *44777755555* (UK number) or *33999911111* (French number). This is the format for the number received in the SMS message.

```
     Send x <currency> to <username>
at <registryname> for <reason phrase>
```

For example:

```
send 4 duckets to ddawg at dalston
for barking lessons
```

SMS only works within one registry and these payments also have a standard title field at present.

Payment via REST

To pay someone some amount, provide a sequence of URLs as the example below:

```
http://cclite.yourdomain.com/logon/ddawg/chelsea/<password>

http://cclite.yourdomain.com/pay/mmouse/chelsea/23/ducket

http://cclite.yourdomain.com/logoff
```

Note that the currency name is the exact name (in this case singular). If accepted, it will reply with a hash of the transaction as receipt reference:

```
Transaction Accepted
Ref: YmH8Y7msvfZaPVWDCrljjmqF47bLcCiK
V1zI4C7CEjHCmXkGQIH6b1jq1rkD8lbapl0bJ
6lMcohApyR8Rc3pHQ
```

Otherwise the transaction can be rejected for all the same reasons (non existent destination, currency etc.) as a form based or any other transaction.

This can form the basis for Perl LWP and PHP libcurl (for example) remote client programs. As of version 0.5.0 there is a an example (but rudimentary and minimally tested!) `curlclient.php` in the php subdirectory.

Batch Payments

Batch payments are processed via a batch file.

The contents of the batch file should be as below (lines starting with # are skipped, space lines are skipped).Please note the date format, also:

```
#date,from,to,fromreg,toreg,
currency,type,taxflag,value,desc

04/10/2005,"ddawg","manager","dalston","dalston",
"ducket","credit","N",345,"test1"

05/10/2005,"ddawg","manager","dalston","dalston",
"ducket","credit","N",456,"test2"

06/10/2005,"ddawg","manager","dalston","dalston",
"ducket","credit","N",42,"test3"
```

This file is uploaded into into the csv directory (defined by `csvpath=` , in cclite.cf) using the **Upload Batch Files** function in the **Admin Menu**.

It is processed by the `readcsv.pl` cron and a file `xxxx.csv.out` is written into `public_html/out/` containing the processing results (so that the results are available within the web root (this may or may not be desirable!). The input file is renamed to `xxxxx.csv.done` so it is not reprocessed.

This is the first version, so there are probably some rough edges.

4. OsCommerce Gateway

You ***DO*** need to know a little bit about OsCommerce™ itself to use this!

Philosophy and Structure

This describes the Cclite payment gateway for OsCommerce™. OsCommerce™ needs to be used in a slightly different way to support a social credit system or a closed barter group.

For example, there will be many producers and many consumers and they will probably be the same people. Also, a payment may not be to one producer but may be split up amongst many (for example, a rural group, A supplies eggs, B supplies bread but all this is accumulated into one ‘order’).

Therefore the gateway uses the `manufacturers_name` field in the `manufacturers` table in OsCommerce™ to identify ‘producers’ who are to be paid within the corresponding Cclite system.

The manufacturer name is the same as the `userLogin` name in the `om_users` table in Cclite. Sooner (or later, or you can do this!) I’ll supply a script to create manufacturers for every registered Cclite user (or should that be the other way around?).

The gateway is a standard OsCommerce™ gateway, written in PHP and installs using the install within OsCommerce™.

Restrictions

This is an alpha version of the gateway.

At present, one Cclite registry is connected with one shop front and only one currency is allowed per shop. OsCommerce™ handles conventional money where all the different currencies are linked via exchange rates.

For many alternative money theorists, exchange rates and conversion are often undesirable (because they leads to pure monetary operations, currency futures and derivatives, for example), so I don't plan to work in this area of the software at present. Currencies are independent in Cclite, so it doesn't make to much sense to alter the philosophy here.

Also, I did not (and do not) want to produce a specialised and forked version of OsCommerce™ (this usually makes life more difficult for everyone, OsCommerce folks, myself and the rest of the universe).

Installation Overview

To install and use the gateway, do the following steps:

1. Generate a **Merchant Key** for the Cclite registry that is to be connected.

2. Currently, use http://www.fourmilab.ch/hotbits/generate.html to provide a key. This needs to be the same on the Cclite side and on the Oscommerce side. This is used as a password between the shop and the registry. It corresponds to the API key in a standard credit card gateway.

3. Install and configure the Cclite gateway within OsCommerce administration. This is a standard OsCommerce gateway install.

4. Put the **Merchant Key** into Cclite registry that is to be connected, using the **Modify <name-of-registry>** in the **Admin Menu**.

5. Set up `manufacturers` corresponding to Cclite `users` with the OsCommerce shop administration software.

6. Set up products for each `manufacturer`.

Generating a Registry Merchant Key

NOTE: This will soon require `openssl` or gpg to be installed on the Cclite machine, the hotbits key is an alpha-version compromise.

You can just choose a ‘weak’ key (1234, for example!) if you are testing or experimenting.

Install and Configure the Cclite Gateway

This is a manual install for the present.

1. Copy `cclite.php` from the gateways subdirectory of Cclite into `catalog/includes/modules/payments` within OsCommerce

> NOTE: Yes, there are two files called `cclite.php`, one in `modules` and one in `languages`!

2. Copy `cclite.php` from the gateways subdirectory of Cclite into `catalog/includes/language/payments` within OsCommerce
3. Go to the administration tool and click on **Payments**.
4. Choose **Cclite** and click on **Install**, see below.
5. Edit any of the fields:

Enable CCLITE Module	True	Needs to be true, otherwise this won't be offered as a payment method
Transaction Currency	Only ducket	Name of Currency to be accepted
Currency Symbol	DCK	This needs to correspond to the currency symbol used in Cclite
Payment Zone	--none--	Not used at present
Set Order Status	Default	Not used at present
Sort order of display.	0	Sort order for these parameters, not used

Merchant Key	1234	Shop key that is generated in Cclite or user chosen key. Attention: a ‘weak’ key will mean that others may be able input transactions. You should change this key fairly frequently.
Gateway URL	https://yourdomain.com/cgi-bin/cclite.cgi	URL of the Cclite installation that is being used to collect payments
Default Registry	chelsea	Registry used to collect transactions
Enable Payments	live	Can be ‘live’ or ‘test’. With test, Cclite will display the data it’s receiving without finishing the transaction.

Testing the Gateway

To see whether the shop is communicating with the gateway, set **Enable Payments** to `test` and check the output from Cclite.

Payments are shown by manufacturer and each one will appear as a separate cgi value on the Cclite screen.

When testing is finished, re-configure the gateway to `live`.

Removing the Gateway

Just click **Remove** and the gateway will be removed from the current shop but can be reinstalled at any time.

5. REST Interface

See the Install chapter for the configuration required for REST. This section is a stub at present. I currently think that this type of interface will be easier to use than the pure SOAP interface (which requires much tighter coupling between the client and server). The experimental REST operations that are currently supported are:

Logon

You'll need to logon (and store the cookies somewhere within your client) before doing a payment operation, for example.

```
http://cclite.yourdomain.com/logon/ddawg/chelsea/password
```

Logoff

If you've logged on, please log off as well:

```
http://cclite.yourdomain.com/logoff
```

Payment via Rest

Pay someone some amount, the logged in user will be debited:

```
http://cclite.yourdomain.com/pay/manager/chelsea/23/ducket
```

This will reply with accepted and a hash of the transaction as reference:

```
Transaction Accepted
Ref: YmH8Y7msvfZaPVWDCrljjmqF47bLcCiK
V1zI4C7CEjHCmXkGQIH6b1jq1rkD8lbaplObJ
6lMcohApyR8Rc3pHQ
```

This is also described in the Payment Chapter.

Trading Summary

This will currently only deliver a summary for the logged in user, but the syntax allows extensions to queries by other users.

http://cclite.yourdomain.com/ddawg/summary

Detailed Transactions

This is not ideal at present because it’ll just deliver the first page of a paged set of transactions:

http://cclite.yourdomain.com/ddawg/transactions

PHP Client for REST

There's a php client called `curlclient.`php for REST in the php subdirectory. It's been very minimally tested. You'll need to change at least the following lines, to try it out:

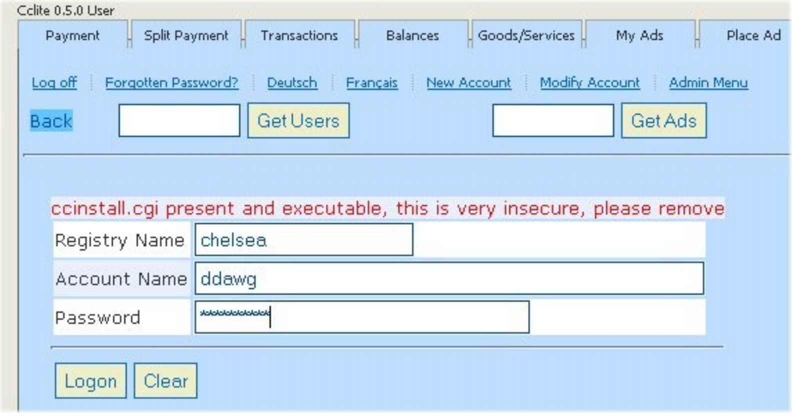
```
/* logon and store the logon cookies */

/* configuration variables */
$domain          = "yourdomain.com" ;
$cookiefile      = "/tmp/cclitecookies" ;
$user            = "youruser" ;
$registry        = "yourregistry" ;
$password        = "passwordofyouruser" ;
$currency        = "lucket" ;
$amount          = "youramount" ;
$payto           = "usertobepaid" ;
```

6. Logging In and User Information

Logging In

This is how the logon screen should look after a successful install.



ccinstall.cgi present and executable, this is very insecure, please remove

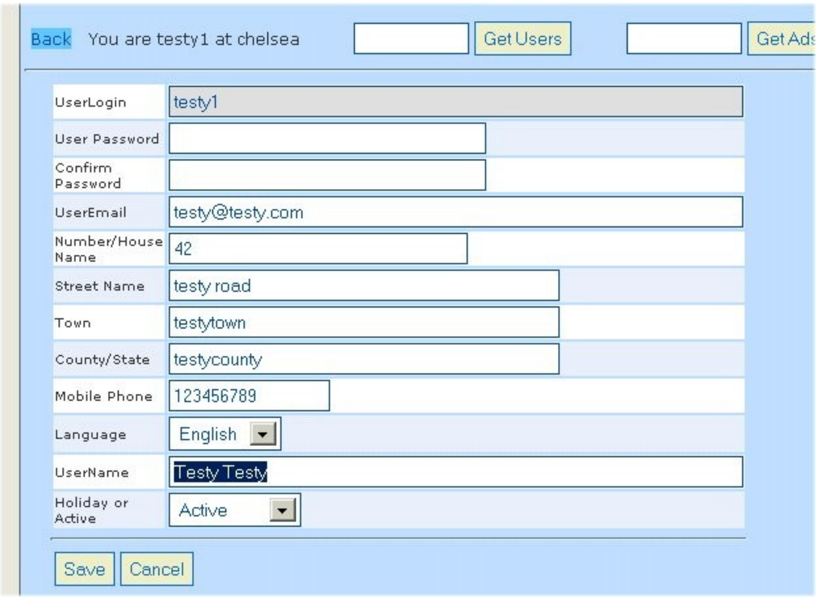
Registry Name	chelsea
Account Name	ddawg
Password	**********

You can change this screen from the standard to give a drop down (multiregistry) or fixed value for the registry name by changing `logon.html` in the `templates` subdirectory.

Remember to remove or disable the installer to make the message go away.

Changing User Details

This is the current user modification screen. An administrator has control of more options, suspension, pre-delete etc. but cannot change the user password. If the password is corrupted, change with an email reset.



You are testy1 at chelsea

UserLogin	testy1
User Password	
Confirm Password	
UserEmail	testy@testy.com
Number/House Name	42
Street Name	testy road
Town	testytown
County/State	testycounty
Mobile Phone	123456789
Language	English
UserName	Testy Testy
Holiday or Active	Active

7. Technical Description

This section describes the organisation and contents of files. It also provides some guidance on changing things. Cclite is still under active development, some caution is needed.

Architecture

The architecture is pretty much model/view/controller. The controllers are `cclite.cgi`, `ccadmin.cgi`, `ccserver.cgi` (the SOAP controller) and `ccinstall.cgi`. The views are provided by the templates but this is currently imperfect (there’s still quite a bit of presentation generated in `Cclite.pm`; one of the main modules, for example).

`ccsuggest.cgi` contains the logic for making automatic suggestions for user names, transaction destinations and searches. The code is new and untidy. If you don't want this feature, remove it or remove its execute permissions.

`Ccdirectory.pm` now contains most of the user adverts and yellow pages logic. This is to enable users or groups to write their own version of this. There's work being done on the worst of this, as of version 0.5.0.

The 'business' logic is provided in the `Ccxxxx.pm` series modules.

Also the database code is in `Cclitedb.pm` and therefore, in principle, a new database or file store can be implemented without impacting the rest. Some of the non-standard SQL is provided as-is using `sqlraw`.

`Cchooks.pm` is included and will contain a programming interface for the cautious, routines that allow pre and post transaction code without modifying the main code base, for example. I haven't done very much with this to date (March 2006).

Subdirectory Layout

This is a summary description of software contents. If the implementation is web-only; it doesn’t currently need anything in `/batch`.

Subdirectory name	**Contents**	**Notes**

`cgi-bin`	`cclite.cgi`
	`ccserver.cgi`
	`ccsuggest.cgi`
		Main controller for cclite
		Server for SOAP web services
		auto-suggest for main scripts

`cgi-bin/protected`	`ccadmin,cgi`
	`ccinstall.cgi`
		Administration controller
		New registry creation. `ccinstall.cgi` should be removed or disabled when the install is finished

`lib`	Cclite Perl library modules also including `Simple::Template`	Controllers need to find this directory
`doc`		Manual and other documentation. Now includes html pages documenting the program code in doc/html
`sql`	Registry creation template	This contains SQL for a registry database including manager login and a sysaccount. Also sql for upgrading from previous versions.
`template/html/en`	English language templates	More templates can be added in template/<language code> for example
`public_html`	Intial setup page: `index.html`	**index.html needs to be removed after setup**
`php`	curlclient.php	Experimental curl based client for the REST interface

batch	readsms_from_gammu.pl readmail.pl writerss.pl gammu.sh	Read sms payment messages harvested by gammu Read mail payment messages Write rss offers and wants batch script for running gammu
config	**<script name>.cf** **gammu.cf**	Will be used for active configuration files. Only cclite.cf the main configuration file is used at present. gammu configuration file

On-line Function Documentation

I'm removing this as all controller and library documentation is provided as html in doc/html in a clearer format (thank you, perltidy).

Customisation

This section deals with other languages, adding, removing and changing things.

Adding Languages and translating

Cclite uses `Simple::Template` for its screens and thus can be translated at about 90% without great difficulty.

It isn’t perfectly multi-lingual ready yet. I’m not sure what would happen with multi-byte languages, either.

The second part of the language specific items are in `literals/literals.<language-code>`. These would also need to be translated, for example `literals.fr` would be the French version

To translate screens; copy out the English ones into another subdirectory in templates, for example `templates/html/it`, and translate. Then, an extra drop down item can be added to the user preferences part and to the main menu.

If you do this, please tell me via the contact form on my website and I will make them available generally, if possible.

Changing current screens

The screens are style sheet controlled via `styles/cc.css`. This is the best method for changing the appearance of things (colours, font sizes etc.) globally.

When changing individual screens, beware of the template tags, for example

```
$$fieldsref{name}
```

and

`!$$fieldsref{name}`   *this one is the name of a file must be on its own line*

If you want to make major modifications to the screens, read the `Simple::Template` documentation at http://www.cpan.org

Adding and Removing Functions

If you want to remove a function, remove it from the appropriate screen *AND* comment it out in the appropriate controller (otherwise it is still reachable via URL hacking, for example!). There’s more about this in the general programming section too.

Also, if you don’t want to install any more registries or make currencies, remove `ccinstall.cgi` after using it, for example. It's a *very good idea* to off-line, take away execute permissions or remove it between modifications anyway.

If you want to add a function:

- Specify the function (always a good idea!)
- Add the call to the function in the appropriate controller
- Code and test the function (which is usually added to `Cclite.pm`, the main logic part or `Ccadmin.pm` if it's an administration function
- Contact me, if you want, to put the new function in the main distribution *but remember Cclite has a GPL licence anyway*.

See also, the explanation of `Cchooks.pm` for adding pre-processing and post-processing to transactions. This is currently rather underdeveloped as of July 2006.

Adding and Removing Batch Operations

Preferably, use one of the existing batch operations as a model. A new batch operation requires a new script in /batch and perhaps a new `cron` job.

If it's some type of transaction processing; it should provide data to `sub transaction` in `Cclite.pm`. This means that there's a unified interface for all transaction operations (and that anything in `Cchooks.pm` will also be called).

Implementing Demurrage and Commissions

For example, intuitively, demurrage operations might be a once-a-month batch operation on accounts that are defined as inactive (within the money system that is implemented, this will vary from place to place). Commissions to a central account could also be implemented in this way.

As of July 2006, you can use and amend the processing that does the service charge.

Changing directory layout

I don’t really recommend this. Mail me and tell me why you want to do it and I’ll have a think about the reasoning though.

For example, I previously had the templates within the web root. But they shouldn't be accessed as web pages (since they are not complete, autonomous web pages themselves), so I moved them 'above' the root.

The new standard layout is Debian (and therefore Ubuntu) oriented because that is a popular distribution as of late 2008. I'm hoping to set up a few branches for each within my repository, but we'll see whether there's time for such foolishness.

Cchooks.pm Specialised transactions

This is under development, currently. It’s a somewhat empty module where cautious coders can put pre and post processing code for new transactions and users etc. it should have all input data available for the given type of operation.

It will enable transaction commissions for example.

Adding and Modifying REST Operations

To do this, add or modify the rewrite rules within:
`public_html/.htaccess`

The ones in there are pretty much 'test' quality rather than production quality.

> The security level for the REST interface is low at present and will probably get built up later.

SOAP based customisation

Theoretically, most of `Cclite.pm` (for example) can be accessed remotely from other pieces of software, written in other languages. This is a SOAP property.

It was certainly my view that `Cclite` should be able to exchange data with other software based registries. It can currently exchange data with itself using SOAP, I'll probably try either Python or Ruby, when I get time. If you do this, please tell me whether it works.

I also see that it may be useful to build and allow federated searches for goods and services within a region, for example. SMS payments also have a standard title field at present, The goods and services aspect can also be catered for, to some extent, by federating `rss` feeds from various registries in the region. As far as I'm concerned, convenience, utility and adoption are more useful than theory; so I'm waiting and seeing.

The SOAP accessible and federated aspects (of which payment is the most obvious and the original one) can also be used to build geography based displays for items, trading health and schemes. I may try and build a demonstration based on the `Google Maps API`, since that connects to other work that I am doing.

Finally, SOAP based federation from a single point (giving star shaped graphs) may not be optimal. I've been thinking about agents that could visit accessible registries and cumulate offers and wants, for example. Some vague intuitions about graph theory (especially the concept of a Hamiltonian, see http://en.wikipedia.org/wiki/Hamiltonian_path, for example) suggest that this might be more efficient. The computer science would be fun as well, an important point.

8. Web and Other Resources for Alternative Money

This list is small, partial and not in any particular order. I'd also currently recommend,

Healthy Money, Healthy Planet Developing Sustainability through New Money Systems by *Deirdre Kent*. This is a little New Zealand centric but is a good round up of many of the problems with conventional money and reasons for doing other kinds of money:

http://www.craigpotton.co.nz/products/published/books/booksocial/healthymoneyhealthyplanet

If you want to comment about this list or lobby for something else to be included, write to me:

http://www.newciv.org/ncn/moneyteam.html
http://www.openmoney.org/
http://thetransitioner.org/wen/tiki-index.php?page=Open+Money
http://www.letslinkuk.org/
http://ccit.wji.com/tiki-directory_browse.php?parent=36
http://www.transaction.net/
http://www.reinventingmoney.com/riegel.php
http://twenteenthcentury.com/uo/index.php/FacultyEconometrics
http://en.wikipedia.org/wiki/LETS
http://en.wikipedia.org/wiki/Money
http://www.gdrc.org/icm/lets-faq.html
http://www.reinventingmoney.com/worglCurrDemurr.php

Other Complementary Currency Software

I've been keeping a list, see it at:

`http://twenteenthcentury.com/uo/index.php/FacultyEconometrics`

This list does not include commercial, application server provider based and non-open source packages.

Appendix A. Release Notes 1/12/2005

Release Notes 17/12/2005

1. Modified rules table to reflect likely use: pretransaction, posttransaction and subscription for example.

Entries can be keyed by user and currency. In theory (because this hasn't been used in anger yet) if they're not keyed in this way, then they're universal.

2. Testing is currently taking place for the application under mod_perl2. This is logical given the size of the application and response requirements.

3. Literals moved into literals file, loaded using readmessages() in Ccu.pm. This file and the templates means that cclite is pretty much multilingual-ready now. Just add templates and literals.<languagecode>

4. Split Ccdirectory.pm and Cclite.pm so that directory stuff is independent from a coding point of view. show_user is still in Cclite.pm

5. Javascript libraries being moved into application wx, behaviour etc. ajax and prompting etc. should appear by next release.

6. Full address now stored in om_users: means smallish schema change!!! 'holiday' added to userStatus list to be used for indicating user's holidays, logic added in modify user to deal with this...shown at top of user page.

7. Show balance and volume now just adds all transactions to give a volume, display tidied up

8. Web installer and configuration updater ccinstall.cgi now supplied, alpha quality. Installer tests for required modules and templates etc. Removed create currency from install, needs logged on registry administrator.

9. If you turn off multiregistry in cclite.cf the only external modules needed are Digest::SHA2 and DBI. This is to simplify the install when used as stand-alone (80% of cases?). This also means that only the current registry is displayed for transfers.

10. Credit limit per currency, per registry introduced into rules table. sketchy support so far.

11. Configuration file cclite.cf now operational and can be updated using the installer. dbuser and dbpassword stored here, rather than top of Cclite.db

12. registrypublickey field introduced into cclite.cf. Start of introduction of some kind of public key crypto, probably based on GnuPg.

13. Tabular output and html tidied and standardised in most cases. Code needs cleaning though.

14. tradeHash implemented as sha 512 base 64 hash of original transaction put into both sides of transaction. tradeHash field enlarged to 128 chars. This will probably be signed by a private key, later.

15. added sysaccount userLevel to om_users, this is the classical LETS system account for subscriptions etc. means small schema change...needs more associated logic.

16. Search fixed and extended for all items...a manager can modify other users for example. It's probably not a good idea, EVER, to modify transactions though. Some security issues to be dealt with around this.

17. Currencies that are closed or suspended are not displayed in drop-down Transaction added to admin for modifying certain fields on currency, in testing.

18. Say No! to a waiting transaction means that the status becomes 'declined'. This will need extra logic in volume calculations and total etc.

19. ccadmin.cgi?action=functiondoc (there's no link to this, do it in the URL) will give a listing of all functions and their files/pod headings. Not completely infallible but useful.

20. Configuration files for batch scripts commented out. Variables for batch processses are still currently in the scripts themselves.

21. userss=no default in the cclite.cf default, means no rss feeds and therefore ccadmin.cgi does not need XML::Rss. Change to yes, if you want rss

Appendix B. Release Notes 31/03/2006

Please note that this is an intermediate release. I am moving house AND working from temporary accomodation etc. etc. therefore I've decided to put this release out.

1. Runaway process problem for readsms.pl fixed, can now be run without attached phone (though this doesn't make much sense..)

2. ccdirectory.cgi added. Gives primitive version of yellow pages directory as Open Office sxw document. Brief description in manual: not finished

3. Tax flag added to transactions table. Requires small schema change. Used to keep one alphabetic character to indicate tax code. Default is N = Non-taxable. Other letters can be user defined (depends on tax system of country).

4. Public key field for a currency (and therefore per currency in each registry, this is a possible weakness, it's as weak as the weakest key used for currency x) added to currency table. Preparation for encrypted email/batch transfers. Requires small schema change.

5. Batch process readcsv.pl added to read comma separated batch input, usually put into /var/cclite/something.csv. Results written to something.csv.out to enable checking, resubmission etc.

6. Note for manual: readconfiguration() within cron jobs contains hardcoded configuration path. cron runs from libexec, so no autodiscovery

7. ccupload.cgi added for batch input upload to readcsv.pl. Integrated into administration.

8. Note for manual: ugly hack in readmessages() in Ccu.pm to find messages file when using from cron jobs...needs to be changed here so that cron jobs pick up multilingual messages file. Marked 'very ugly hack'.

10. Note for manual: date formatting in csv files...: done

11. Primitive REST interface started. This will probably take over from the proposed SOAP interface, PHP testing has shown that it's going to be hard for foreign systems to implement and use. No security currently on rest! Description of this interface in manual.

12. Today is default transaction date, if not supplied: multiple interface problem

13. Corrected bug: transaction with no source now rejected! REST interface problem

14. To be corrected: clear up confusion about token: should be calculated at logon: not done

15. To be done: check whether balance records included in total calculations. Take on problem: balances should be zero. check whether balances included in show transactions

16. Trade hash displayed as receipt,reference on transaction acceptance. This is part of the RESTful scheme too.

17. Rejected admin link bounces/loops corrected, just comes back to user menu with message: You are not an administrator

18. Reference to the rss path in %fields is now $fields{rsspath} rather than $fields{path}. The default directory is also changed for rdf to rss. rsspath is now included in the configuration file.

19. Default web interface 'reskinned' both to improve appearance (I hope) and to find out how easy it is to 'reskin' (pretty easy).

20. To be corrected: configuration says it updated but it didn't...when no config cclite bombs saying it can't find &error : not done

21. To be remembered: regenerate registry_sql for 4.0 version of mysql: done

22. Made display more understandable for source and destination, these are misnomers, should be 'Partner' really.

23. Public key and merchant key fields added to registry table. Public key field is meant for the 'general' public key for that registry. Merchant key is the key used by oscommerce over https, it's symmetric and should be longish.

24. Alpha version of OsCommerce gateway added. Description in Manual. merchant_key needs to be added manual in database!

25. Setup now creates manager and a single sysaccount, the sysaccount is used to receive service charges. The manager is not charged. sysaccount has a default password of 'manager'

26. The package will grumble if ccinstall.cgi is left around in it's default place and is executable.

27. There's a link to a 'news' page in the public root made from cclite.cf, this is provisional and designed for event news.

Appendix C. Release Notes 31/10/2006

This release is in preparation!

1. Split payment facility added
2. Service charge facility added
3. Manual revised and enriched
4. Printed version of manual available on www.lulu.com

Appendix D. Release Notes 30/06/2007

1. Merchant key field added to registry creation and update (correction to 0.4.1)

2. Allowed IP added to registry, this is for straight through os-commerce transactions. Needs small registry change (for future use)

3. Added latest news field to registry, one-line news item for front of interface. Needs small registry change.

4. Modify registry characteristics transaction introduced and working

5. Tidy up for use Digest::SHA2 now confined to Ccsecure.pm. This opens way to version with use of SHA1 only (but not recommended).

6. Trade Reference (SHA2 hash) now added to detail display for transaction Possibly, in the future the hash should not include the status field?

7. html documentation for all scripts and modules in ../doc/html via perltidy

8. Auto-suggest for destination trade field using scriptaculous + ccsuggest.cgi Simplifies filling in destination for large groups.

9. Auto-suggest for all search fields using scriptaculous + ccsuggest.cgi

10. Auto-suggest to indicate when user has chosen unique screen name Simplifies screen name choice for large groups.

11. Can now use templatename to display templates other than index.html

12. Perl tidy on all modules and scripts

13. Css and skin changed to tabs based on http://labs.silverorange.com/archives/2004/may/updatedsimple. This is simpler than the previous one

14. Search forms now require at least one character input

15. Display user bug corrected by using $ $fieldsref{duserLogin}

16. News item can be displayed on front user page via registry modification form. Currently text-only news items.

17. Simple curlclient.php added to examples, interfacing php to REST interface for a currency transaction.

18. Added a find/modify yellow page transaction to administration menu.

19. Transaction summary now sorts 'push down'(reverse chronological) most recent transaction at top.

20. Revised/simplified code for yellow pages, more craiglist and goodbye to expanding menus. Code for 'old' style retained for the moment.

21. (bug correction) Can only delete ads that belong to logged on user or via admin.

22. Striped display for all tables, easier reading for transaction lists etc. Various types and types of status are colour coded.

23. Manager can cancel transactions, they are preserved but have a 'cancelled' status in the database. In the future, many cancelled transactions will reduce the 'system health' index (as will many declined).

24. Improved (one hopes) yellow pages directory. The transaction that delivers this can be used anywhere, it currently just returns a table.

25. When the manager finds transactions the whole set is delivered not just those that belong to the manager. Should be any, anyway!

26. Tidied up transaction reference hash, only hashes core information for transaction, not all template fields etc.means that hash can be reproduced if necessary (to test the authenticity of the transaction). Also needs private server token in hash.

27. Balance and volume constrained to 'active' and 'waiting' transactions

28. tradeSource and tradeDestination updated correctly in database, meaning extensive changes to transaction retrieval: get_trades added to deal with this

29 There's now a googlegroups group: http://groups.google.co.uk/group/cclite for

support this will take over from the sourceforge forum, which was hardly used

30. fields{maxreport} for summary and fields{righthandside} for displaying stuff to right of main display added

31. Validation and reporting improved for transactions. Remote error codes translated locally. Error transactions recorded in local database with status 'rejected' and error messages in the transaction description.

32. Split transactions commented in title and displayed as orange. This is not fully multilingual though.

33. Create category somewhat improved but could do with further work

34. Update and some corrections in manual.

INDEX

Alphabetic Index