Whitepaper - WCS - published by: wcs:root : Sat Jan 18 16:40:07 CET 2020 # World Compensation System (WCS) **Keywords**: #2020, #Blockchain, #InternetOfValue, #RSK, #DeFi, #DeFiApp, #DFApp, #wcsDFApp

Classification: DeFi : EcoSystem : @rif_os : **Version**: 0.0.1 (2020-01-15) **Status**: *in work*

Purpose

Simple. Eliminate money invisibility.

User-story

> I dare you to throw out all your money, all your papers and coins and individual national currencies, and start over.
>
> Develop an international monetary system that is wide open, totally visible, immediately traceable, completely accountable. Establish a Worldwide Compensation System by which people would be given Credits for services rendered and products produced, and Debits for services used and products consumed.
>
> Under the new Worldwide Compensation System, WCS, the transfer of Debits and Credits would be immediate and totally visible. That is, anybody and everybody could inspect the account of any other person or organization at any time. Nothing would be kept secret, nothing would be 'private'
>
> Everything would be on the system of Credits and Debits. Returns on investments, inheritances, winnings of wagers, salaries and wages, tips and gratuities, everything. The WCS would deduct 10 percent of all earnings each year from the income of those *voluntarily requesting* such a deduction. Everyone in the society would be able to observe who was choosing to offer the 10 percent for the general good of all, and who was not. And everyone's records would be open to everyone else. And nothing could be purchased without Credits. There would be no other negotiable currency. ([source:#CWG, @realNealWealsh](http://ISBN))

Short-name: WCS, DeFiApp, wcsDFApp

Disclaimer: All quoted phrases are verbatim copies found at Conversations with God. Book Two (see Bibliography)

Bibliography: - Conversations with God (c) 1997 Neale Donald Walsch. ISBN 978 0 340 76544 9

Use-cases

- Typical Use-cases and User-workflow

Community

- [@WorldCompensationSystem (Twitter)](https://twitter.com/WorldCompensationSystem)

License

Code is under the The Unlicensed. Documentation is under the Creative Commons Attribution license.

Contributing

Please read our Contribution Guide and [Code of Conduct]

Donations

BTC-Address: | 0x123..0000 |

Whitepaper

Table-of-contents

../arch/dfs/dFSwcs - # Distributed File-System WCS

../arch - ## Architecture

../arch/fs - # File System

../users - # Users

../commands - ### Commands (status:*in-work*)

../apps - ### Applications

../services/wcsServer - # wcsServer - World Compensation System server

../services - ### Services

Operating (Eco-)System Concept

1. wcsO(E)S - WCS Operating Eco-System
    1. Platform
    2. Transactions
    3. Use-cases
        1. Notation
        2. Welcome home
        3. Help
            1. View description manual
        4. User management
        5. File-System (Minimum commands)
            1. Present working directory (pwd)
            2. Listing files (ls)
        6. Applications, Services, Commands and Tools
            1. Running local tool system-service
            2. Running local command system-service
            3. Running local user-application
            4. Running remote user-application
            5. Commands
            6. Tools
        7. Local Services
            1. Financial
        8. Value Operations
        9. Distributed Services
        10. Group citizenship
        11. Work – Get Idle Task (according to current citizenship)
        12. Donate
        13. Contribute to Nation's taxes
        14. Value Creation (out-of-thin-air)
        15. Credits
        16. Value Transfer - Request for service

17. Value Transfer - Investment
    1. Communities
18. Running DApplication in debug-mode
19. User-management

4. User-Application development
    1. DApplication development help
5. Returning home
6. Lend User-service
7. Claim lend User-service
8. Borrow service ## Application notes # wcsO(E)S - WCS Operating Eco-System

Platform

realisation platform - wcsOS Linux based. Minimal distribution

Transactions

```
homeland$send 2 user1
Sending 2 Credits to /users/user1
TX: Send pubkey
RX:
TX:
RX:
```

Use-cases

Keywords: Socratic thinking, design thinking, Agile methodologies, user stories

Notation

The following `code` sections provide the main use-case description. First step during implemenation consist of: 1. Copy Use-case description as /dst/test.expected 1. Replace old section and include new created file (/dst/test.expected) here instead 1. Implement required functionality 1. Implement test.sh 1. tool arg1 –arg-2 > test.result 1. diff test.result test.expected 1. Implement build.sh 1. build 1. /test.sh

Welcome home

```
** Welcome @homeland
homeland$
```

Help

View description manual `help`, `man` or check our documentation.

```
homeland$man
home
whoami
pwd
ls <folder>
ver
```

```
homeland$ver
wcsOS 0.0.1
```

User management

```
homeland$home
```

```
homeland$user create user1
New wallet created
Address: user1:0xc50...0000
```

File-System (Minimum commands)

Present working directory (pwd)

```
homeland$pwd
/node/homeland --> {nodeuuid}
```

Listing files (ls)

```
homeland$ls
local services
app1 -> /node/homeland/apps/app1
cmd1 -> /node/homeland/commands/cmd1
tool1 -> /node/homeland/tools/tool1

remote services
dapp1 -> /node/lapland/dapps/dapp1

local-users
me -> /users/me

remote-users
```

Applications, Services, Commands and Tools

Running local tool system-service

```
homeland$stat
connection OK
0 services running
```

Running local command system-service

```
homeland$cmd1 --verbose
running cmd1
```

Running local user-application

```
homeland$app1 --verbose
running app1@localhost (127.0.0.1)
```

Running remote user-application

```
homeland$dapp1 --verbose
running dapp1@lapland
address: 0xc5..000

no local credits (use command: credits)
no local debts (use command: debts)

use '$commands' for a listing of available operation commands
use '$apps' for a listing of currently available user-services
```

Commands

```
homeland$commands
operations
credits - credits are gained by certain proof-of-work
debts - debts are credits debited to the users (to be payed later in time or as part of a lo
loans - list of available loans (request for asset-transfer (value transfer)
offers - list of published offers (request for service)
assets - show local assets (including applications,commands,services,tools and monetary valu


actions
debit -
loan -
value - set 'own' market-value (local cost of service-unit) -- analog to BTC-Network-Fee or
```

```
cost - update 'own' local (production) cost (fixed costs (including internet + electricity))
income - show current regular income
work - produce a work-product (costing local energy (computational power))

transactions
credit - credit user for consumed-service (online time + service computer power)
accept/service - accept offer
send - transfer asset to otheruser
reputation - get/set reputation-value (quadratic-voting) to given asset
```

Tools

```
homeland$tools
2 tools

telnet - (0 credit:government:culture)
ping - (1 credit:foundation:ibm)
telnet - (1 credit:institute:fraunhofer)
hash - (1 credit:university:berlin)

homeland$apps           ($ls /apps)
3 apps, 1 local

tetris - (16 times:company:gameco:*)
doomclone - (872355 times:user:girx34:*****)

xyz - (4 times:local)
```

Local Services

Financial

```
homeland$wallet
Address: 0xc50..000
0 Credits
0 Debits
```

Value Operations

```
homeland$credits
0 credits

homeland$debts
0 debts
```

```
homeland$assets
2 assetts, 1 leasing, 1 licensed, 0 invested

leasing
tetris - (2 Leasing Credits left:16:company:gameco:*)

licensed
doomclone - (34 Credits licensed:872355:user:girx34:*****)

invested 0
```

```
homeland$credit tetris
no neighbors found
```

Distributed Services

```
homeland$neighborhood
0 neighbors
```

```
homeland$discover
discovering users
34 neighbor users found
3 communities found
```

```
homeland$connect
connecting with local neighborhood
1. peer-to-peer connection established. Hello node34
2. peer-to-peer connection established. Hello
3. remote-connection established. Hello server78.google.com
OK service-connection established. Hello homequarters (@WCS00.org) time:34.251 ms
```

```
homeland$ls
local services
app1 -> /node/homeland/apps/app1
cmd1 -> /node/homeland/commands/cmd1
tool1 -> /node/homeland/tools/tool1

remote services
dapp1 -> /node/lapland/dapps/dapp1

local-users
me -> /users/me

neighborhood
user1 -> /neighborhood/nod35/user1
```

Group citizenship

```
homeland$ls nations
local-nation
nation1 -> /federation/ethereum/ethnation1
bitcoin -> /bitcoin/BTC
rsk -> /rif_os/rsk/Rootstock
```

```
homeland$citizen
connecting with local nation
1. remote-connection established. Hello finance.gov
2. remote-connection established. Hello congress.EU
OK service-connection established. Hello homequarters (@WCS00.org)

0 taxes
2 messages
1 requests
10 offers
```

```
homeland$citizen federation1
connecting with local federation1
1. remote-connection established. Hello finance.gov
OK service-connection established. Hello homequarters (@WCS00.org)

0 taxes
2 messages
1 requests
```

Work – Get Idle Task (according to current citizenship)

```
homeland$idle federation1
scientific.phsychedelics (945 Users)
```

Donate

```
homeland$donate -idle neighborhood
thank you
```

Contribute to Nation's taxes

```
homeland$tax 8
contribute to local nation 8hrs full-time
completed (8 Credits)
```

Value Creation (out-of-thin-air)

```
homeland$offer -idle federation1
offering local idle-time for federation1

homeland$offer 8
offering local-resources for 8hrs nation (default)
rejected (not enough resources)

homeland$offer 6
offering local-resources for 6hrs nation (default)
accepted
completed. thank you (6 Credits granted)
```

Credits

```
homeland$credits
6 Credits
```

Value Transfer - Request for service

```
homeland$tetris
running tetris..
..
exiting tetris

homeland$credits
4 Credits
```

Value Transfer - Investment

```
homeland$invest kernel.org 2
2 Credits left

homeland$assets --all
3 assets

borrowed/leasing
tetris - (0 Leasing Credits left:16:company:gameco:*)

licensed
doomclone - (34 Credits licensed:1564355:user:girx34:****)

invested
kernel - (2 Credits invested:0.0000 earned:6463872355:community:linux.org:*****)
```

```
membership
```

Communities

```
homeland$ls communities
communities
community1

homeland$greetings community1 me --verbose
$greetings community1 me{pubkey:address:nodeuuid:useralias:mail}
>>me:greetings community1
>>me:credentials me{pubkey:address:nodeuuid:useralias:mail}
>>community1:greetings me
>>community1:here our credentials
>>{
>>  credentials: "community1{pubkey:address:nodeuuid:useralias:mail}"
>>}

homeland$offers
11 offers (1 miners)

homeland$accept 1
accepted top-priority offer (3 miners working in parallel)
completed (1 Credit granted)

homeland$credits
3 Credits

homeland$value dapp1
1 Credit
```

Running DApplication in debug-mode

```
homeland$dapp1 --verbose --debug
connection established (lapland @address: 0xc5..000:1234)

DEBUG: Exchange pubkey:me@homeland
DEBUG: 1 Debit Credit to me@homeland (-1 Debit Credit, 2 Remaining Credits in Total)
DEBUG: License key received (privatekey)
DEBUG: run dapp1@lapland using privatekey
running dapp1@lapland --key privatekey
...
exiting dapp1@lapland
DEBUG: 1 Credit granted to dapp1 (12463 in Total)
goodbye. connection closed (dapp1@lapland)
DEBUG: 2 Local Credits left (me@homeland)
```

User-management

```
homeland$login
homeland: me
Password: *** *

me@homeland$
```

User-Application development

DApplication development help

```
me@homeland$man app

json app(arg1, arg2){
 return json;
}
```

```
me@homeland$publish myapp1
{
 namespace : "me@homeland"
 name : "myapp1"
 return : "1"
}
```

Returning home

```
me@homeland$exit
homeland$
*** Welcome homeland

homeland$ls
local services
app1 -> /node/homeland/apps/app1
cmd1 -> /node/homeland/commands/cmd1
tool1 -> /node/homeland/tools/tool1

remote services
myapp1 -> /node/homeland/me:/myapp1
dapp1 -> /node/lapland/dapps/dapp1

local-users
me -> /users/me
```

```
neighborhood
user1 -> /neighborhood/nod35/user1 (light-consumer)
lapland -> /neighborhood/lapland/root (full-provider)

homeland$rate dapp1 ***
Awesome tool. Thank you!

homeland$SMS user1
Just tried '/homeland/dapp1' out. Is worth taking a look.

homeland$share dapp1 user1 2
Hey! Check '/homeland/dapp1' out. You are going to loooove it!
Timeout set to 2 hours
```

Lend User-service

```
homeland$lend dapp1
offering dapp1 -> /node/lapland/dapps/dapp1
waiting for acceptance
accepted (node75)
```

Claim lend User-service

```
homeland$claim dapp1
claming dapp1 -> /node/node75/dapps/dapp1
waiting for timeout
restored (homeland)
dapp1 -> /node/lapland/dapps/dapp1
```

Borrow service

Architecture

Concept

Create a World Compensation Ecosystem based on Decentralised Financial Applications.

Implementation: Operation System (including fs, dfs, time-shared applications)

wcsOS – linux based distribution

Layers: 1. Distributed peer-2-peer (P2P) Network (Blockchain based) 1. Distributed File system (dfsWcs) 1. Nodes are Servers 1. Servers 1. run System- and Users-services 1. route User- and System- interactions (transactions) 1. Users are Clients 1. Clients decide to participate or not (mounting/unmounting) as service suppliers in the network 1. Clients interact with other Clients 1. Clients request services from Servers (service suppliers) 1. Via Remote Procedure Call (RPC) returning values in JSON format 1. Clients transfer value-assets to single or multiple-users or services 1. Light-Clients connect and use the network only for short-time (SMS, PPP) 1. Value-assets are represented via Addresses in the Distributed File system 1. Clients and Servers interact via read/write file operations with eachother 1. Servers providing User-services are debted certain agreed amount per-use 1. Servers providing System-services are debted an agreed amount per-use, daily, monthly or yearly on donation basis

Network

Topology: Flower or Tree-of-Life (sacred geometry star 1:N, N:=6) * https://en.wikipedia.org/wiki/Overlapping_circles_grid#Modern_usage

Nodes:

```
Full-nodes: store the complete history of command-blocks (analog to batch-files (a.k.a trans
Light-nodes: store, validate and reconstruct environment from all nodes in local network (or
```

Local File-system

```
RK1.0
/ - WCS root Ecosystem
/commands
/dbin/ - Decentralised System services
/users/ - connected user addresses {publickey:addresshash:alias:inbox} (analog to /mnt)

RK1.1
/apps
/lang - implementation language specific files
/tools - Utility tools

RK1.2
/dapp/ - Decentralised User or Third-Party Applications (executable -- analog to /usr/bin)
```

Remote (distributed) File-system

/arch/dfs

```
RK2
/dapp/DeFi/ - Decentralised Financial Apps
```

Realisation

TCP/IP Server :port UNIX's "Everything is a File" -> (name:Address) - Network (distributed) File-System

Support Tools

```
neo4.js - Graph Database
```

Common Use-cases

- transfer value-assets

Network Startup

```
$wcsStart &
World Compensation System server (wcss) running
Listening on port:280182

$wcsStatus
Status: OK
```

Command, Services and Tools

```
mount - loads/unloads foreign WCS Networks
ls - show current Users in the Ecosystem
mv - allocate users in different sub-network
whoami -
ps - process
ping - ping
telnet - establish connection and echo server
mailto - use mail-alies to transfer transfer value-assets in the network
```

Application Notes

```
$whoami
{0x123456:mstcroix:mstcroix@protonmail.com}

$pwd
```

```
/

$ls
/apps
/commands
/dapps
/services
/users

$ls /users
/users/{0x123456:mstcroix:mstcroix@protonmail.com}
/users/{0x423456:mstcroix:none}
/users/{0x223456:anonymous:none}
/users/{0x723456:mstcroix:mstcroix@protonmail.com}

$ps
0 applications running

$/apps/App1 &
$ps
/apps/App1 running. 5 users connected
```

50fc328aad939c00fb848432a94943c9 ../arch/README.md 838c67d9e0f81417746e220fe56d6434 README.md d3777eb628218cf79d50e576d5c95bbd customer.md ca8f6611e7334b5878a412f6908fab36 platform.md 2860295f71c884833cec61f44f4c9ccf wcsOES.md 70a1947487f1741ee64cb109b8cddc82 whitepaper.md ## 2020 (CC) Creative Common License 2860295f71c884833cec61f44f4c9ccf wcsOES.md

```
bc9b4920af19ea249c5e51730a986a9a  ../tools/README.md
f92f6755c1f6c83858630cb4d4c419aa  ../networking/README.md
3582056e21f163e556a92a29f26da4bc  ../GLOSSARY.md
1d4ba0b4f97b65cb239ac157fa453df6  ../dapps/README.md
ca8f6611e7334b5878a412f6908fab36  ../whitepaper/platform.md
2860295f71c884833cec61f44f4c9ccf  ../whitepaper/wcsOES.md
d3777eb628218cf79d50e576d5c95bbd  ../whitepaper/customer.md
9e0f1913e90406c26246475461d291ff  ../whitepaper/README.md
70a1947487f1741ee64cb109b8cddc82  ../whitepaper/whitepaper.md
b85f6f905757b8a0d3f75430e13c47ad  ../operations/README.md
4e57eedfde6cb02c52aec8be79b015c9  ../README.md
94063115eb82858ccfd15ef5a3b21814  ../project/integration.md
68f05ceb68281268217108fb55876082  ../project/deployment.md
5cd4aa50a1a9f8d1b46b0b63c9d82e27  ../project/CONTRIBUTING.md
9c060f1741bc37163838ead55b73c8ab  ../project/workproducts.md
f16bab90fe5bf837c86b04e89f7dbb86  ../lang/README.md
2eaaf2bbe0e2dae25cbc17345d4ba75a  ../lang/c/README.md
5fe7603d97b3315406ce7c051f273a3e  ../arch/dfs/README.md
b6a84991d4f8957e69ccfd6d3e935e02  ../arch/dfs/dFSwcs/README.md
```

```
50fc328aad939c00fb848432a94943c9  ../arch/README.md
4ae120d33503361b35768677302e8c75  ../arch/fs/README.md
1c309bf14fbd49d5afcfac8da0635b5b  ../users/README.md
41990a8f6b22e6e1b72009ca47b6ffc7  ../commands/README.md
15fd8a06e94ecf5f079451536356171c  ../apps/README.md
a3c6c1e9fbc0dd9e6723f73f7402b08a  ../services/wcsServer/README.md
06ae6a9d35733170f372c50e1e6ed749  ../services/README.md
```