

**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

**Факультет компьютерных наук
Департамент программной инженерии**

Согласовано

Профессор департамента
программной инженерии факультета
компьютерных наук, канд. техн. наук

_____ С.М. Авдошин
” ” _____ 2019 г

Утверждаю

Академический руководитель
образовательной программы
«Программная инженерия»
профессор, канд. техн. наук
В. В. Шилов

_____ В. В. Шилов
” ” _____ 2019 г

**Криптографические алгоритмы и протоколы для распределенных
реестров**

Программа и методика испытаний

ЛИСТ УТВЕРЖДЕНИЯ

RU.17701729.04.01 51 01-1

Студент группы БПИ 151 НИУ ВШЭ

_____ Куприянов К. И.
” ” _____ 2019 г

Подп. и дата	
Инв. № дубл.	
Взам. инв. №	
Подп. и дата	
Инв. № подл.	

2019

УТВЕРЖДЕНО
RU.17701729.04.01 51 01-1

Криптографические алгоритмы и протоколы для распределенных реестров

Программа и методика испытаний

RU.17701729.04.01 51 01-1

Листов 95

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

2019

Содержание

1	Объект испытаний	81
1.1	Наименование программы	81
1.2	Краткая характеристика	81
2	Цель испытаний	82
3	Требования к программному изделию	83
3.1	Требования к функциональным характеристикам	83
3.1.1	Состав выполняемых функций компоновщика	83
3.1.2	Состав выполняемых функций реализации блокчейна	83
3.1.3	Требования к временным характеристикам компоновщика	83
3.1.4	Требования к интерфейсу компоновщика	84
3.1.5	Требования к интерфейсу реализации блокчейна	84
3.2	Требования к надежности	84
3.2.1	Обеспечение устойчивого функционирования компоновщика	84
3.2.2	Обеспечение устойчивого функционирования реализации блокчейна	84
4	Требования к программной документации	85
4.1	Предварительный состав программной документации	85
5	Средства и порядок испытаний	86
5.1	Параметры технических средств, используемых во время испытаний	86
5.2	Порядок проведения испытаний	86
5.3	Условия проведения испытаний	86
5.3.1	Требования к численности и квалификации персонала	86
6	Методы испытаний	87
6.1	Проверка требований к документации	87
6.2	Проверка требований к интерфейсу	87
6.3	Проверка требований к функциональным характеристикам	87
6.3.1	Проверка требований к компоновщику	87
6.3.2	Проверка требований к реализации блокчейна	90
6.4	Проверка требований к временным характеристикам	92

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

6.4.1 Проверка требований к компоновщику	92
6.4.2 Реализация блокчейна	92
7 Приложение 1. Терминология	93
7.1 Терминология	93
8 Приложение 2. Список используемой литературы	95

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

1. Объект испытаний

1.1. Наименование программы

Наименование программы на русском: “Криптографические алгоритмы и протоколы для распределенных реестров”.

Наименование на английском: “Cryptographic Algorithms and Protocols for Distributed Ledgers”.

1.2. Краткая характеристика

Программа предназначена для пользователей машин на семействе ОС GNU/Linux. Цель работы — создать удобное приложение для автоматизации программирования, которое генерировало бы готовый код блокчейна с использованием алгоритмов, выбранных пользователями.

данный продукт будет служить “инструментарием” для оператора или любого другого интересующегося криптографическими алгоритмами и протоколами, который имел бы потребность интегрировать блокчейн в своё приложение (регистрация гостей в отеле, социальную сеть, переводы, учёт документов). так же программа будет полезна людям, которые хотят узнать как работают современные распределённые реестры с рассмотренными аспектами. это позволит быстро получать необходимую техническую информацию, которую с трудом можно найти в общем доступе. программа должна предоставлять не только генерацию кода, но и дружелюбный интерфейс командной строки, в которой форматирование и подсветка не будут сбивать с толку неподготовленного пользователя.

главной чертой данного приложения является самоподдерживаемая система по работе с исходными кодами алгоритмов, расположенными удалённо. а так же лёгкая, быстрая масштабируемость и модульность программного кода.

Приложение состоит из двух компонент:

1. Позволяющей сгенерировать код блокчейна с использованием выбранных пользователем алгоритмов
2. Является выходом первой компоненты, и по своей сути обособленным приложением — блокчейном

В дальнейшем (1) будет именоваться **компоновщик**, а (2) — **реализация блокчейна**.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

2. Цель испытаний

Цель проведения испытаний заключается в проверке выполнения заявленных в техническом задании требований к программной документации и составу выполняемых функций программы, надежности и корректности ее работы, а также интерфейсу и внешнему виду приложения.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

3. Требования к программному изделию

3.1. Требования к функциональным характеристикам

3.1.1. Состав выполняемых функций компоновщика

1. Возможность вывода на консоль вариантов выбора по категориям:
 - (a) Отображение возможных структур реестра
 - (b) Отображение возможных типов открытости реестра
 - (c) Отображение возможных алгоритмов консенсуса
 - (d) Отображение возможных алгоритмов хэширования
 - (e) Отображение возможных алгоритмов генерации случайных чисел
 - (f) Отображение возможных алгоритмов цифровой подписи
2. Генерирование значений для выбора “по умолчанию”
3. Возможность записать выбора пользователя
4. Возможность установки загруженных алгоритмов на ФС машины без исключительных root прав
5. Возможность генерировать код по указанной директории
6. Возможность замера времени работы выбранных алгоритмов
7. Возможность просмотра справочной информации по остальным параметрам реестра
8. Вывод информации в цвете, обозначающий степень поддержки программой алгоритма

3.1.2. Состав выполняемых функций реализации блокчейна

1. Возможность генерации “адреса кошелька” — пары приватный + публичный ключ
2. Использование в качестве алгоритма цифровой подписи выбранный пользователем
3. Использование в качестве алгоритма хэширования выбранный пользователем
4. Возможность записи данных “адреса кошелька” на ФС машины
5. Возможность отправки от одного пользователя другому условное количество условной криптовалюты
6. Возможность получения всей цепочки транзакций, которые были проведены за текущую сессию путём вызова API функции майнера

3.1.3. Требования к временным характеристикам компоновщика

На машине с установленной ОС Ubuntu Linux (версия ядра 4.15.0-47), с процессором Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz, 16 Гб ОЗУ, текущей утилизацией диска 0.06%, load average за последнюю минуту 0.22, время выполнения программы не должно превышать 0.05 секунд (без учёта на установку пакетов алгоритмов в систему). Требования к временным характеристикам от значений сети не зависят.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

3.1.4. Требования к интерфейсу компоновщика

Поскольку требования к интерфейсу в настоящем Техническом Задании не предъявляются, соответствующие испытания проводятся не будут.

3.1.5. Требования к интерфейсу реализации блокчейна

Поскольку требования к интерфейсу в настоящем Техническом Задании не предъявляются, соответствующие испытания проводятся не будут.

3.2. Требования к надежности

3.2.1. Обеспечение устойчивого функционирования компоновщика

Для надежной работы программы требуется исполнение следующих требований:

1. Обеспечение поддержания заряда аккумуляторной батареи устройства (ноутбука) на уровне не ниже 20%, иначе обеспечить бесперебойную подзарядку оборудования
2. Обеспечение использования лицензионного программного обеспечения
3. Обеспечение защиты операционной системы и технических средств от вредоносного воздействия шпионских программ, компьютерных вирусов и сетевых червей
4. Обеспечение своевременного обновления программных составляющих устройства

3.2.2. Обеспечение устойчивого функционирования реализации блокчейна

Для надежной работы программы требуется исполнение следующих требований:

1. Обеспечение бесперебойного питания (UPS) оборудования; если же ожидается запланированное завершение выполнения программы в период времени, не превышающий ожидаемое время разрядки аккумулятора оборудования — поддержание уровня заряда 20%.
2. Обеспечение использования лицензионного программного обеспечения
3. Обеспечение защиты операционной системы и технических средств от вредоносного воздействия шпионских программ, компьютерных вирусов и сетевых червей
4. Обеспечение своевременного обновления программных составляющих устройства

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

4. Требования к программной документации

4.1. Предварительный состав программной документации

1. “Криптографические алгоритмы и протоколы для распределенных реестров. Техническое задание”
2. “Криптографические алгоритмы и протоколы для распределенных реестров. Пояснительная записка”
3. “Криптографические алгоритмы и протоколы для распределенных реестров. Руководство оператора”
4. “Криптографические алгоритмы и протоколы для распределенных реестров. Программа и методика испытаний”
5. “Криптографические алгоритмы и протоколы для распределенных реестров. Текст программы”

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

5. Средства и порядок испытаний

5.1. Параметры технических средств, используемых во время испытаний

Для испытания программы необходимо учесть следующие системные требования:

1. Персональный компьютер со следующими минимальными требованиями:
 - (a) Операционная GNU/Linux версии ядра 4.15.0-47-generic и выше
 - (b) 64-разрядный (x64) процессор
 - (c) 1ГБ оперативной памяти (ОЗУ)
 - (d) 100 МБ свободного места на внутреннем накопителе
2. Интерпретатор Python3.6.5 и выше

5.2. Порядок проведения испытаний

Испытания проводятся поэтапно, друг за другом, в следующем порядке:

1. Испытание выполнения требований к программной документации
2. Испытание выполнения требований к функциональным характеристикам программы, надежности и корректности ее работы
3. Испытание выполнения требований к временным характеристикам

5.3. Условия проведения испытаний

5.3.1. Требования к численности и квалификации персонала

Минимальное количество персонала, требуемого для работы программы: 1 оператор. Пользователь данного программного продукта должен разбираться в командной строке (shell) GNU/Linux, иметь базовые навыки в командах, уметь устанавливать и удалять программы, запускать их. Перед использованием программы пользователь должен быть заранее проинструктирован и уведомлен о составе выполняемых функций и других характеристиках приложения, а так же сопровождён необходимой технической документацией.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

6. Методы испытаний

Испытания представляют собой процесс установления соответствия программы и программной документации заданным требованиям.

6.1. Проверка требований к документации

Проверяется наличие всех документов перечисленных в пункте 4.1 данного документа и их соответствие ГОСТ.

6.2. Проверка требований к интерфейсу

Требования к интерфейсу не предъявлялись.

6.3. Проверка требований к функциональным характеристикам

6.3.1. Проверка требований к компоновщику

Проверка реализованного функционала продемонстрирована на скриншотах ниже.

По запуску приложения с параметром `init`, на консоль должно выводиться приветственное сообщение с краткой информацией по дальнейшим действиям. (пункт 1 на Рис. 1)

```

-----
~ # gsl --init --name myledger --path ~/tmp/gsl                                coldmind@coldmind-l
[2019-05-21 22:01:23] [goodsteel_ledger -> 19531 -> 140107680777984] [INFO] >> Start Goodsteel Ledger: a program for generatin
g distributed ledgers
[2019-05-21 22:01:23] [config -> 19531 -> 140107680777984] [INFO] >> Loading config from /etc/gsl/config.yaml
[2019-05-21 22:01:23] [config -> 19531 -> 140107680777984] [INFO] >> Configuration loaded

=====INITIALIZE LEDGER=====
=====
Name: myledger
Path: /home/coldmind/tmp/gsl

=====MAKE YOUR CHOISES=====
=====

THIS color indicates you will be provided with code or documentation for a particular algorithm BUT it will not be included in
YOUR ledger code!
THIS color indicates that GSL will generate a working code for your ledger using a particular algorithm

Choose type of concrete algorithm from which your blockchain will consist of:

Choose type of hashing of the ledger
1: SHA-256
2: SHA-512
3: Scrypt
4: KECCAK-256
5: KECCAK-512
6: Ethash
7: X11
8: X17
9: myr-groestl
10: Lyra2rev2
11: blake2s
12: blake2b
Enter num from 1 to 12, default [1]: 9

Choose type of digital signature of the ledger
1: ECDSA
2: DSA
3: GOST R 34.10-2012
Enter num from 1 to 3, default [1]: 3

```

Рис. 1: Начало работы компоновщика

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

Алгоритмы должны отображаться пронумерованно; по категориям (Рис. 1)

Вывод алгоритмов должен быть в цвете, обозначающим степень поддержки программой алгоритма (пункт 3 Рис. 1)

Должен поддерживаться выбор алгоритмов “по умолчанию”, и при отсутствии ввода определённого номера алгоритма, выбираться указанный в качестве алгоритма “по умолчанию” (пункт 2 Рис. 1)

Now, choose related themes for which you will be provided with relevant information (links, web sites, etc.)

```
Option: structure of the ledger
1: Blockchain
2: DAG
3: Hashgraph
4: Holochain
5: Tempo
Enter num from 1 to 5, default [1]: 4
```

```
Option: openness of the ledger
1: Public
2: Private
Enter num from 1 to 2, default [1]: 1
```

```
Option: consensus of the ledger
1: PoW
2: PoS
3: DPoS
4: PoA
5: PoWeight
6: BFT
Enter num from 1 to 6, default [1]: 5
```

```
Option: random of the ledger
1: DRBG
2: CPRNG
Enter num from 1 to 2, default [1]: 2
```

The following config is to be set:

Рис. 2: Вывод опций по которым будет дана справочная информация

После выбора алгоритмов хэширования и цифровой подписи, пользователю должны показываться свойства/структура/другие алгоритмы распределённых реестров, по которым можно получить справочную информацию (Рис. 2)

The following config is to be set:

```
structure:
- Holochain
openness:
- Public
consensus:
- PoWeight
hashing:
- X17
random:
- CPRNG
digital signature:
- GOST R 34.10-2012
```

Proceed with this config? [YES]/NO: YES

Рис. 3: Подтверждение выбранных опций

Должен выводиться итоговый выбор пользователя с вопросом о намерении принять

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

изменения и продолжить дальнейшее выполнение программы (Рис. 3)

```
[2019-05-21 22:37:44] [goodsteel_ledger -> 30332 -> 13966628647040] [INFO] >> Start getting your ledger's algorithms
Looking in indexes: https://pypi.python.org/simple/
Requirement already satisfied: x17_hash in /home/coldmind/ local/lib/python3.6/site-packages (1.5)
running install
running build_ext
running egg_info
writing pygost.egg-info/PKG-INFO
writing dependency_links to pygost.egg-info/dependency_links.txt
writing top-level names to pygost.egg-info/top_level.txt
reading manifest file 'pygost.egg-info/SOURCES.txt'
reading manifest template 'MANIFEST.in'
warning: no files found matching '*' py' under directory 'tests'
warning: no files found matching '*' py' under directory 'tests'
writing manifest file 'pygost.egg-info/SOURCES.txt'
installing library code to build/bdist.linux-x86_64/egg
running install_lib
running build.py
creating build/bdist.linux-x86_64/egg
creating build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/Face.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_gost341194.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_gost3412.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_gost341194.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_gost3410_vko.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_gost3412.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_gost34112012.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_gost34112012256.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_gost28147_mac.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_mac.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_cms.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/utills.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_pfx.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_gost28147_mac.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_gost28147.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_gost3410_vko.py -> build/bdist.linux-x86_64/egg/pygost
copying build/lib/pygost/test_gost3410.py -> build/bdist.linux-x86_64/egg/pygost
creating build/bdist.linux-x86_64/egg/pygost/asnschemas
creating build/lib/pygost/asnschemas/x509.py -> build/bdist.linux-x86_64/egg/pygost/asnschemas
creating build/lib/pygost/asnschemas/pfx.py -> build/bdist.linux-x86_64/egg/pygost/asnschemas
copying build/lib/pygost/asnschemas/cms.py -> build/bdist.linux-x86_64/egg/pygost/asnschemas
copying build/lib/pygost/asnschemas/_init_.py -> build/bdist.linux-x86_64/egg/pygost/asnschemas
copying build/lib/pygost/test_gost3412.py -> build/bdist.linux-x86_64/egg/pygost
creating build/bdist.linux-x86_64/egg/pygost/stubs
creating build/bdist.linux-x86_64/egg/pygost/stubs/pygost
copying build/lib/pygost/stubs/pygost/test_gost34112012512.py -> build/bdist.linux-x86_64/egg/pygost/stubs/pygost
copying build/lib/pygost/stubs/pygost/test_gost341194.py -> build/bdist.linux-x86_64/egg/pygost/stubs/pygost
copying build/lib/pygost/stubs/pygost/test_gost3412.py -> build/bdist.linux-x86_64/egg/pygost/stubs/pygost
copying build/lib/pygost/stubs/pygost/test_gost3412.py -> build/bdist.linux-x86_64/egg/pygost/stubs/pygost
copying build/lib/pygost/stubs/pygost/test_gost28147_mac.py -> build/bdist.linux-x86_64/egg/pygost/stubs/pygost
copying build/lib/pygost/stubs/pygost/test_gost34112012256.py -> build/bdist.linux-x86_64/egg/pygost/stubs/pygost
copying build/lib/pygost/stubs/pygost/test_gost3410_vko.py -> build/bdist.linux-x86_64/egg/pygost/stubs/pygost
copying build/lib/pygost/stubs/pygost/_init_.py -> build/bdist.linux-x86_64/egg/pygost/stubs/pygost
copying build/lib/pygost/stubs/pygost/test_gost28147.py -> build/bdist.linux-x86_64/egg/pygost/stubs/pygost
copying build/lib/pygost/stubs/pygost/test_gost3410_vko.py -> build/bdist.linux-x86_64/egg/pygost/stubs/pygost
copying build/lib/pygost/stubs/pygost/test_gost3410.py -> build/bdist.linux-x86_64/egg/pygost/stubs/pygost
copying build/lib/pygost/stubs/pygost/test_gost34112012.py -> build/bdist.linux-x86_64/egg/pygost/stubs/pygost
byte-compiling build/bdist.linux-x86_64/egg/pygost/test_gost34112012.py to test_gost34112012.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/test_gost34112012256.py to test_gost34112012256.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/test_gost28147_mac.py to test_gost28147_mac.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/test_cms.py to test_cms.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/test_pfx.py to test_pfx.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/test_gost28147_mac.py to test_gost28147_mac.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/test_gost3410_vko.py to test_gost3410_vko.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/test_gost3410.py to test_gost3410.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/asnschemas/x509.py to x509.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/asnschemas/pfx.py to pfx.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/asnschemas/cms.py to cms.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/asnschemas/_init_.py to _init_.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/test_gost3412.py to test_gost3412.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/test_gost3410_vko.py to test_gost3410_vko.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/test_gost3410.py to test_gost3410.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/test_gost28147.py to test_gost28147.cpython-36.pyc
byte-compiling build/bdist.linux-x86_64/egg/pygost/test_wrap.py to test_wrap.cpython-36.pyc
installing package data to build/bdist.linux-x86_64/egg
running install_data
copying AUTHORS -> build/bdist.linux-x86_64/egg/
copying COPYING -> build/bdist.linux-x86_64/egg/
copying INSTALL -> build/bdist.linux-x86_64/egg/
copying NEWS -> build/bdist.linux-x86_64/egg/
copying README -> build/bdist.linux-x86_64/egg/
copying SECURITY -> build/bdist.linux-x86_64/egg/
copying THIRDS -> build/bdist.linux-x86_64/egg/
copying VERSION -> build/bdist.linux-x86_64/egg/
creating build/bdist.linux-x86_64/egg/EGG-INFO
copying pygost.egg-info/PKG-INFO -> build/bdist.linux-x86_64/egg/EGG-INFO
copying pygost.egg-info/dependency_links.txt -> build/bdist.linux-x86_64/egg/EGG-INFO
copying pygost.egg-info/top_level.txt -> build/bdist.linux-x86_64/egg/EGG-INFO
zip_safe flag not set; analyzing archive contents
creating 'dist/pygost-3.15-py3.6.egg' and adding 'build/bdist.linux-x86_64/egg' to it
removing 'build/bdist.linux-x86_64/egg' (and everything under it)
Processing pygost-3.15-py3.6.egg
Removing /home/coldmind/ local/lib/python3.6/site-packages/pygost-3.15-py3.6.egg
Copying pygost-3.15-py3.6.egg to /home/coldmind/ local/lib/python3.6/site-packages
pygost 3.15 is already the active version in easy-install.pth
Installed /home/coldmind/ local/lib/python3.6/site-packages/pygost-3.15-py3.6.egg
Processing dependencies for pygost==3.15
Finished processing dependencies for pygost==3.15
```

Рис. 4: Процесс установки выбранных алгоритмов

Рис. 5: Завершение установки выбранных алгоритмов

При подтверждении выбора набора алгоритмов пути до исходных кодов алгоритмов должны быть получены из хранилища, и по ним должна произойти установка в систему (Рис. 4 - 5).

```
Holochain:
- https://github.com/holochain/holochain-rust
Public:
- Depends on your implementation: https://masterthecrypto.com/public-vs-private-blockchain-whats-the-difference/
PoKweight:
- Read https://filecoin.io/filecoin.pdf
X17:
- https://pypi.org/project/x17_hash/
CPRNG:
- https://riptutorial.com/python/example/3857/create-cryptographically-secure-random-numbers
GOST R 34.10-2012:
- https://pypi.org/project/pygost/
```

Рис. 6: Справочная информация в конце выполнения компоновщика

По завершении выполнения программа должна выводить справочную информацию по выбранным свойствам распределённого реестра (Рис. 6).

```
~/tmp/gsl/myledger > ll
total 32K
-rw-rw-r-- 1 coldmind coldmind 14K May 21 22:37 miner.py
-rw-rw-r-- 1 coldmind coldmind 2.3K May 21 22:37 mydss.py
-rw-rw-r-- 1 coldmind coldmind 291 May 21 22:37 myhashing.py
-rw-rw-r-- 1 coldmind coldmind 6.5K May 21 22:37 wallet.py
```

Рис. 7: Директория со сгенерированным кодом

После завершения работы программы по указанной директории должны располагаться модули wallet.py и miner.py вместе с выбранными алгоритмами хэширования и электронной подписи (Рис. 7).

Должно производиться автообновление алгоритмов каждый день в 21:00. Каждый день в репозитории проекта должен быть коммит в 21:00 с обновлениями алгоритмов,

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

если таковые имеются.

6.3.2. Проверка требований к реализации блокчейна

Проверка реализованного функционала продемонстрирована на скриншотах ниже. Модуль wallet.py будем называть **кошелёк**, а miner.py — **майнер**.

```
~/tmp/gsl/myledger » python3 wallet.py                                coldmind@coldmind-l

Which action would you like to take?
1. Generate new wallet
2. Send coins to another wallet
3. View transactions

□
```

Рис. 8: Возможности кошелька

Возможности кошелька должны включать в себя 3 функции: генерирования нового адреса (пары публичный и приватный ключ), отправки средств с одного адреса на другой, а так же просмотр блокчейна (Рис. 8).

```
-----
~/tmp/gsl/myledger » python3 wallet.py                                coldmind@coldmind-l

Which action would you like to take?
1. Generate new wallet
2. Send coins to another wallet
3. View transactions

1
=====

IMPORTANT: save this credentials or you won't be able to recover your wallet
=====

Write the name of your new address: kirill
Your new address and private key are now in the file kirill.txt
Repeat? Would you like one more action? (Y/[N])
Exiting..
-----
```

Рис. 9: Генерация адреса kirill

При выборе первой опции должен отображаться диалог с требованием ввести имя, и дальнейшей генерацией адреса кошелька (пары публичный-приватный ключи) (Рис. 9)

```
-----
~/tmp/gsl/myledger » cat kirill.txt                                    coldmind@coldmind-l
Private key: 20878886106463861047567562769948029178950056671838418434664116114810111370736
Wallet address / Public key: xiY4YpAcz7IPpQWNDvRn5dLsjXusDxi+Sv5vLwp6dYR50BLTFpF0E+hZIXu0SGauM7ujRmjqqT10RQyVY9U7+NSzwjQk12MZ
ulCk7V/T1qGqvfsxuCyxblS8lIE351cTERIk+4cuyH2S1rS0w50kmm+7PW0ykzN1cTejKBsR8=

~/tmp/gsl/myledger » cat julia.txt                                    coldmind@coldmind-l
Private key: 36576356659659993861862725112889947658242081595918339788301050076371118315257
Wallet address / Public key: Ld/aIVziUKTZ2rzVnuzAW14RxQMDtYdzURY0V0lf5b1dxrGZ286dxr6rmM0XdfBNeWLSJZHzeMhxnHKb4L8zk7shUXl0j0o/
D0iJ0VeQq0K0UULiLS92AAWKPCBURf3Br/X4I8M0E0LWr+ctJGBoRAiBdGcDpV6xQz4CP0cfE=
```

Рис. 10: Просмотр сгенерированных адресов кошельков

Адрес кошелька должен записываться в файл с расширением .txt и указанным именем в названии (Рис. 10).

При запуске майнера, должен вестись лог о проведённых транзакциях и их валидациях (Рис. 11 - Рис. 12).

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

```
~/tmp/gsl/myledger * python3 miner.py
* Serving Flask app "miner" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [17/May/2019 12:50:06] "GET /blocks?update=q3nf394hgg-random-miner-address-34nf3i4nflkn3oi HTTP/1.1" 200 -
{"index": 1, "timestamp": "1558086606.8309455", "data": {"proof-of-work": 71271, "transactions": [{"from": "network", "to": "q3nf394hgg-random-miner-address-34nf3i4nflkn3oi", "amount": 1}], "hash": "820539ad1af5001742702ea099cb7e33e30e122b61c108fcc102bbc10ccc0301"}
127.0.0.1 - - [17/May/2019 12:50:06] "GET /blocks?update=q3nf394hgg-random-miner-address-34nf3i4nflkn3oi HTTP/1.1" 200 -
127.0.0.1 - - [17/May/2019 12:50:06] "GET /txion?update=q3nf394hgg-random-miner-address-34nf3i4nflkn3oi HTTP/1.1" 200 -
{"index": 2, "timestamp": "1558086606.8755133", "data": {"proof-of-work": 142542, "transactions": [{"from": "network", "to": "q3nf394hgg-random-miner-address-34nf3i4nflkn3oi", "amount": 1}], "hash": "bf531e792069f890996969d7f870eb9ed49bd167c6ec35c1a75f79355139161"}
127.0.0.1 - - [17/May/2019 12:50:06] "GET /blocks?update=q3nf394hgg-random-miner-address-34nf3i4nflkn3oi HTTP/1.1" 200 -
127.0.0.1 - - [17/May/2019 12:50:06] "GET /txion?update=q3nf394hgg-random-miner-address-34nf3i4nflkn3oi HTTP/1.1" 200 -
{"index": 3, "timestamp": "1558086606.9611478", "data": {"proof-of-work": 285084, "transactions": [{"from": "network", "to": "q3nf394hgg-random-miner-address-34nf3i4nflkn3oi", "amount": 1}], "hash": "2cd997bd4563ba997e140dd38a4c4020cdc832abc1cdd5acd3a605f8a70737b8"}
127.0.0.1 - - [17/May/2019 12:50:06] "GET /blocks?update=q3nf394hgg-random-miner-address-34nf3i4nflkn3oi HTTP/1.1" 200 -
127.0.0.1 - - [17/May/2019 12:50:07] "GET /txion?update=q3nf394hgg-random-miner-address-34nf3i4nflkn3oi HTTP/1.1" 200 -
{"index": 4, "timestamp": "1558086607.1272025", "data": {"proof-of-work": 570168, "transactions": [{"from": "network", "to": "q3nf394hgg-random-miner-address-34nf3i4nflkn3oi", "amount": 1}], "hash": "54f0d961ed8a2043d91716647e5108467cdd6e11ed502da33bdc734d1ff66c3a"}
127.0.0.1 - - [17/May/2019 12:50:07] "GET /blocks?update=q3nf394hgg-random-miner-address-34nf3i4nflkn3oi HTTP/1.1" 200 -
127.0.0.1 - - [17/May/2019 12:50:07] "GET /txion?update=q3nf394hgg-random-miner-address-34nf3i4nflkn3oi HTTP/1.1" 200 -
{"index": 5, "timestamp": "1558086607.4534817", "data": {"proof-of-work": 1140336, "transactions": [{"from": "network", "to": "q3nf394hgg-random-miner-address-34nf3i4nflkn3oi", "amount": 1}], "hash": "1ae23f6dc5419a0579aec2ff8f2e09c92f6cfe1f49cfe2693deaab01024c731"}
127.0.0.1 - - [17/May/2019 12:50:07] "GET /blocks?update=q3nf394hgg-random-miner-address-34nf3i4nflkn3oi HTTP/1.1" 200 -
127.0.0.1 - - [17/May/2019 12:50:08] "GET /txion?update=q3nf394hgg-random-miner-address-34nf3i4nflkn3oi HTTP/1.1" 200 -
{"index": 6, "timestamp": "1558086608.120859", "data": {"proof-of-work": 2280672, "transactions": [{"from": "network", "to": "q3nf394hgg-random-miner-address-34nf3i4nflkn3oi", "amount": 1}], "hash": "bc2e393cb4ffe5d12e832ad879153266e98fe468767d6dc964f95c045afb11c"}
coldmnd@coldmnd-l
```

Рис. 11: Лог работы майнера

```
127.0.0.1 - - [21/May/2019 23:57:47] "GET /blocks?update=k40df238gn-random-dkf13-address-k394rb
gfGKe392f HTTP/1.1" 200 -
b'\xc6&8b\x90\x1c\xcf\xb2\x0f>\xa5\x8d\x0e\xef4g\xe5\xd2\xec\x8d{\xac\x0f\x18\xbeJ\xfeo\x96,)\xe
9\xd6\x11\xe4\xe0KM\xcf3\xdf\xd0\xa1d\x8cn9!\x9a\xb8\xce\xee\x8d\x19\xa3\xaa\x04\xef59\x142U\x8f
T\xef\xe3R\xcf\x08\xd0\x93]\x8cf\xe9B\x93\xb5\x7f0Z\x86\xaa\xef7\xec\x06\xe0\xb2\xc5\xb9R\xef2R\x
04\xdf\x9d\\LDH\x93\xee\x1c\xbb!\xf6J-kH\xec9:I\xa6\xfb\xb3\xd6\xd3)34\x87\x13z2\x81I\x1f'
<class 'bytes'>
New transaction
FROM: xiY4YpAcz7IPPqWNDvRn5dLsjXusDxi+Sv5vliwp6dYR50BLTfPf0E+hZIXu0SGauM7ujRmjggT10RQyVY9U7+NSz
wjQk12MZulCk7V/T1qGqvfsxuCyxbLS8lIE351cTERIk+4cuyH2Si1rS0w50kmm+7PW0ykzNIcTejKBSR8=
TO: Ld/aIVziUKT22rzVnuAW14RxQMDtYdzURYOV0if5b1dxrGZ286dxxr6rmM0XdfBNwLSJZHzeMhxnHKb4L8zk7shUX
10j0o/D0ij0VeQq0KOUULiLS92AAWKPCBURf3Br/X4I8M0E0Lwr+ctJGB0Ra1BdGcDpV6xQz4CP0cfe=
AMOUNT: 13
127.0.0.1 - - [22/May/2019 00:29:02] "POST /mycoin HTTP/1.1" 200 -
```

Рис. 12: Лог регистрации новой транзакции

В кошельке при отправке условных средств с одного счёта на другой, должны требоваться публичный и приватный адреса отправителя, а так же публичный ключ получателя (Рис. 13).

При выборе третьей опции в кошельке, должен отобразиться полная цепочка транзакций (блокчейн) (Рис. 14).

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

```
-----
~/tmp/gsl/myledger » python3 wallet.py                                coldmind@coldmind-l

Which action would you like to take?
1. Generate new wallet
2. Send coins to another wallet
3. View transactions

2
From: introduce your wallet address (public key)
x1Y4YpAcz7IPPqWNDvRn5dLsjXusDxi+Sv5vLwlp6dYR50BLTfPf0E+hZIXu0SGauM7uJrmjggT10RQyVY9U7+NSzwjQk12MZulCk7V/T1qGqvfsxuCyxbLS8lIE35
1cTERIk+4cuyH2Si1rS0w50kmm+7PW0yKzNlCteJkBSR8=
Introduce your private key
20878886106463861047567562769948029178950056671838418434664116114810111370736
To: introduce destination wallet address
Ld/aIVzLlUKTZ2rzVnuzAW14RxQDMdtYdzURYOV0lf5b1dxrGZ286dxr6rmM0XdfBNeWLSJZHzeMhxnHkb4L8zk7shUXl0j0o/D0tj0VeQq0K0UULiLS92AAWKPCBU
Rf3Br/X4I8M0E0LWr+ctJGB0RAiBdGcDpV6xQz4CP0cfE=
Amount: number stating how much do you want to send
13
=====

Is everything correct?

From: x1Y4YpAcz7IPPqWNDvRn5dLsjXusDxi+Sv5vLwlp6dYR50BLTfPf0E+hZIXu0SGauM7uJrmjggT10RQyVY9U7+NSzwjQk12MZulCk7V/T1qGqvfsxuCyxbLS
8lIE351cTERIk+4cuyH2Si1rS0w50kmm+7PW0yKzNlCteJkBSR8=
Private Key: 20878886106463861047567562769948029178950056671838418434664116114810111370736
To: Ld/aIVzLlUKTZ2rzVnuzAW14RxQDMdtYdzURYOV0lf5b1dxrGZ286dxr6rmM0XdfBNeWLSJZHzeMhxnHkb4L8zk7shUXl0j0o/D0tj0VeQq0K0UULiLS92AAWK
PCBURF3Br/X4I8M0E0LWr+ctJGB0RAiBdGcDpV6xQz4CP0cfE=
Amount: 13

y/n
y
Transaction submission successful

Repeat? Would you like one more action? (Y/[N])
Exiting..
-----
~/tmp/gsl/myledger » 
```

Рис. 13: Процесс отправки средств

6.4. Проверка требований к временным характеристикам

6.4.1. Проверка требований к компоновщику

Время запуска приложения не должно превышать 1.05 секунд (Рис. 15)

Все временные требования должны быть соблюдены.

6.4.2. Реализация блокчейна

Временные требования к работе реализации блокчейна (майнера и кошелька) не предъявлялись.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

~/tmp/gsl/myledger » python3 wallet.py

coldmind@coldmind-1

```
Which action would you like to take?
1. Generate new wallet
2. Send coins to another wallet
3. View transactions

3
[{"index": "0", "timestamp": "1558468861.6067605", "data": "{ 'proof-of-work': 9, 'transactions': None}", "hash": "0b9cde7957adfce769df95dabb02a2ba0fe08c772c2ce49a7c65c77f5c9e161"}, {"index": "1", "timestamp": "1558468861.6604264", "data": "{ 'proof-of-work': 71271, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "640841869b994c0b01f077bdeba4a05ba17829219bf220b0fa21b7ff4122eff"}, {"index": "2", "timestamp": "1558468861.7079172", "data": "{ 'proof-of-work': 142542, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "e3f92d89609ba95a67a6cf08fe21941acb4f6e92b36ad2c458d29a472a53f41f"}, {"index": "3", "timestamp": "1558468861.7935686", "data": "{ 'proof-of-work': 285084, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "5cb2451bb5a9c095e762a7bdf8c410f412cb90bba08110adee88f75d4b8b29b8"}, {"index": "4", "timestamp": "1558468861.9601207", "data": "{ 'proof-of-work': 570168, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "015f83ca2fcf2502f5b4dc9eb36f0e9acc05c2c1d51daa3cdc2df54f9da4b4f0"}, {"index": "5", "timestamp": "1558468862.2882986", "data": "{ 'proof-of-work': 1140336, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "3ac6afffcaab7f049c7865f720f344537659508afb3582c642243154dc929eb09"}, {"index": "6", "timestamp": "1558468862.9600003", "data": "{ 'proof-of-work': 2280672, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "2fd58ac1580945a0695f47be4485d54f0e1db3c4d0d71a3d89f47930195311c"}, {"index": "7", "timestamp": "1558468864.1199691", "data": "{ 'proof-of-work': 4561344, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "5c257e176bb399c57a644b5cd38118d8c48fdb7046b9cc95bedb61f84518a47"}, {"index": "8", "timestamp": "1558468865.952813", "data": "{ 'proof-of-work': 9122688, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "d5303b53bcb7ec433dae6101569bba83097a31740f720a96df7f220466d155b"}, {"index": "9", "timestamp": "1558468869.1114795", "data": "{ 'proof-of-work': 18245376, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "ff27888d7e4df6a553ca1c5099c4fcbcb8d33696c0c68af192f73b511872a988"}, {"index": "10", "timestamp": "1558468875.0238767", "data": "{ 'proof-of-work': 36490752, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "6358863a0819dd6a6bb2e6cc6e24ce18e188c033491ccc9920d5c9cc4574c574"}, {"index": "11", "timestamp": "1558468886.5762112", "data": "{ 'proof-of-work': 72981504, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "74bab54e0b8682f81c1c2cc428534ce6edb4e0427dela7115404a97960de363"}, {"index": "12", "timestamp": "1558468909.7597685", "data": "{ 'proof-of-work': 145963008, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "67d61bb8ea2dd49c91724affb0ea56c99bb2f32764efa5bcfffe8068bf12ee8b5"}, {"index": "13", "timestamp": "1558468955.0765815", "data": "{ 'proof-of-work': 291926016, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "49f6b3c4ec5b9679c245da0fb092e9277896a5cc3d77c001031ee3439ad8f629"}, {"index": "14", "timestamp": "1558469049.0494485", "data": "{ 'proof-of-work': 583852032, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "34a97447dd030dd42f945d6a11435f0bbbaeac45e344796113aeee0b14bee66"}, {"index": "15", "timestamp": "1558469245.1669443", "data": "{ 'proof-of-work': 1167704064, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "a78903fc2ab59c634ea1a8418436f5d1529702182adee71c18d9602053c9b271"}, {"index": "16", "timestamp": "1558469669.0417874", "data": "{ 'proof-of-work': 2335408128, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "696cfb1d98910b6ad6cd1c5a1a0eabf24848596d76c8ab57605cdd093bcf8abd"}, {"index": "17", "timestamp": "1558470528.4890237", "data": "{ 'proof-of-work': 4670816256, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "f4f46c9bc8b4771acf94d68c66c9aed212e487eca4115e69c79f0cf871fe9b58"}, {"index": "18", "timestamp": "1558472267.4413888", "data": "{ 'proof-of-work': 9341632512, 'transactions': [{ 'from': 'network', 'to': 'k40df238gn-random-dkfi3-address-k394rbgfGKe392f', 'amount': 1}]", "hash": "7cee2cb4fd3580dd201426201ba93b7a951a72385fe912f6bcff6cf38a433657"}]
Repeat? Would you like one more action? (Y/[N])
Exiting..
```

Рис. 14: Отображение полной цепочки транзакций

```
=====
RUNTIME IS 0.4196741580963135
=====
```

Рис. 15: Лог замера времени

7. Приложение 1. Терминология

7.1. Терминология

Распределённый реестр (Distributed Ledger) — В примитивной своей реализации это распределённая база данных между сетевыми узлами или вычислительными устройствами. Каждый из узлов может получать данные других, при этом храня полную копию реестра. Обновления этих узлов происходят независимо друг от друга.

Блокчейн — Постоянно растущий список записей, называемых блоками, которые связаны и защищены с помощью криптографии. Он копируется его пользователями и устойчив к модификации. Машина с рабочей копией называется узлом.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

DAG — Направленный ациклический граф. Это ориентированный граф с данными, использующий топологическую сортировку (от ранних узлов к более поздним).

Биткоин (Bitcoin) — Электронная пиринговая платёжная система, используемая в качестве финансовой единицы (криптовалюты) одноимённую сущность. Создателем биткоина выступает некто Satoshi Nakamoto [1].

Эфириум (Ethereum) — Открытая, общедоступная, вторая по популярности, распределённая вычислительная платформа на основе технологии блокчейн и операционная система с функциональностью смарт-контрактов [2]

Алгоритм консенсуса — Набор математических операций, которые необходимо выполнять для поддержания консистентности всей сети.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата

8. Приложение 2. Список используемой литературы

Список литературы

- [1] Satoshi Nakamoto. «Bitcoin: A Peer-to-Peer Electronic Cash System». Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. Consulted, 1–9.» В: *Journal for General Philosophy of Science* 1 (2008), с. 1–9. ISSN: 09254560. DOI: 10.1007/s10838-008-9062-0. arXiv: 43543534534v343453.
- [2] Vitalik Buterin. *On Public and Private Blockchains*. 2015. URL: <https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/> (дата обр. 22.04.2019).
- [3] Единая Система Программной Документации. *ГОСТ 19.101-77 Виды программ и программных документов*. ИПК Издательство стандартов, 2001.
- [4] Единая Система Программной Документации. *ГОСТ 19.102-77 Стадии разработки*. ИПК Издательство стандартов, 2001.
- [5] Единая Система Программной Документации. *ГОСТ 19.103-77 Обозначения программ и программных документов*. ИПК Издательство стандартов, 2001.
- [6] Единая Система Программной Документации. *ГОСТ 19.104-78 Основные надписи*. ИПК Издательство стандартов, 2001.
- [7] Единая Система Программной Документации. *ГОСТ 19.106-78 Требования к программным документам*. ИПК Издательство стандартов, 2001.
- [8] Единая Система Программной Документации. *ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению*. ИПК Издательство стандартов, 2001.
- [9] Единая Система Программной Документации. *ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению*. ИПК Издательство стандартов, 2001.
- [10] Единая Система Программной Документации. *ГОСТ 19.603-78 Общие правила внесения изменений*. ИПК Издательство стандартов, 2001.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01 51 01-1				
Инв. №подл.	Подп. и дата	Взам. инв. №	Инв. №дубл.	Подп. и дата