Hyperledger Mentorship Project Presentation

November 2022

Implement iroha-cpp library for Hyperledger Iroha 1

- **Name**: Andrzej Gruntowski
- **Location:** Poland, Kraków/Cracow
- **Mentor(s):** Grzegorz Bazior
- **Hyperledger Project**: Hyperledger Iroha 1.x

Project Description:

Hyperledger Iroha 1 is blockchain implemented in C++. To interact with Irohas' nodes (perform commands and queries) there are few client libraries: iroha-python, iroha-javascript, iroha-java, iroha-ios. Despite the fact that Iroha has implemented iroha-cli in C++ there is no client library in C++. In short the project is about implementing the iroha-cpp library.

Project Objectives

- Familiarize with Iroha codebase
- Learn and modify (if necessary) protobufs
- Implement iroha-cpp library for Hyperledger Iroha 1
- Get to know architecture of Iroha 1.x. Check how it works?
- Documentation
- Examples

Project Deliverables

- Implement iroha-cpp library for Hyperledger Iroha 1
- Deliver necessary documentation
- Add usage examples

Project Execution & Accomplishments

- Library was implemented
- Provided examples on how to use the library
- Documentation was updated

Let’s jump into the code

1) Iroha-cli module
2) Iroha-lib module

Demo

Run TxExample.cpp

Recommendations for future work

- accommodate iroha-lib to iroha-cli
- refactor iroha-cli module (too many wrappers, generators, serializes, converters)
- iroha-cli → too long main.cpp file. Consider splitting into more smaller functions. Now it’s hard to read or even understand
- define one consistent way of using namespaces

  protocol::Transaction

  Transaction

  model::Transaction
- rename class names (filenames should correspond to it’s class names IMHO), filenames, field names

Recommendations for future work

- consistent field names account_id or user_name
- SOLID principles → open for extension, but closed for modification (Tx.cpp could be designed in a smarter way)

```
def f(**kwargs):
    for k, v in kwargs.items():
        print('k {} == v {}'.format(k, v))
f(first='Iroha', second='1.5', third='open source')
```

Recommendations for future work

- Consider removing fields like

```
MenuContext current_context_;
MenuPoints commands_menu_;
MenuPoints result_menu_;
```

from InteractiveTransactionCli class and moving them to separate classes

Recommendations for future work

- Protobufs objects should be updated. There are not up to date

    /build-iroha-Imported_Kit-Debug/schema

Recommendations for future work

- This is not good practice

```
ToriiResponse GrpcClient::getTxStatus(const std::string& tx_hash)
{
    TxStatusRequest statusRequest;
    statusRequest.set_tx_hash(tx_hash);
    ToriiResponse toriiResponse;
    grpc::ClientContext context;
    command_stub_->Status(
                &context,
                statusRequest,
                &toriiResponse);
    return toriiResponse;
}
```

Recommendations for future work

- Update documentation?

  Provide a video on how to configure, build and run Irohad from scratch for the new joiners

How to execute Iroha-lib samples?

0)
```
docker ps -a
docker rm <container id>
```

1)
```
docker run --name some-postgres \
-e POSTGRES_USER=postgres \
-e POSTGRES_PASSWORD=mysecretpassword \
-p 5432:5432 \
--network=host \
-d postgres:9.5
```

How to execute Iroha-lib samples?

3)

```
./irohad --config /home/laptop/qt-workspace/iroha/example/config.postgres.sample --genesis_block
/home/laptop/qt-workspace/iroha/example/genesis.block --keypair_name
/home/laptop/qt-workspace/iroha/example/node0 --drop_state –overwrite_ledger
```

4)

Run selected example

Tools and Technologies used

C++

Cmake

Docker

Blockchain

Qt creator gcc 6.3.0

Protobufs

Grpc

Git

Qt creator

Things I'm happy with

```
return iroha_lib::Tx(
          account_name,
          keypair)
      .createDomain(
          domain_id,
          user_default_role)
      .createAsset(
          asset_name,
          domain_id,
          0)
      .signAndAddSignature();
```

Project Output or Results:

- Wiki

https://wiki.hyperledger.org/display/INTERN/Project+Plan+-+Implement+iroha-cpp+library+for+Hyperledger+Iroha+1

- Main PR

https://github.com/hyperledger/iroha/pull/2660

- Other PR(s)

https://github.com/hyperledger/iroha/pull/2935

Insights Gained

- Keep it simple stupid
- Don’t reinvent the wheel

How to (re)generate C++ protobuf(s)?

```
sudo apt-get install autoconf automake libtool curl make g++ unzip
wget https://github.com/protocolbuffers/pr...
tar -xvzf protobuf-cpp-3.11.0.tar.gz
cd protobuf-cpp-3.11.0
./configure
make
make install
sudo ldconfig

hello.proto
protoc --cpp_out=. hello.proto

g++ -std=c++17 hi.cpp hello.pb.cc -o abc `pkg-config --cflags --libs protobuf`
./abc
```

HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS

How to (re)generate C++ protobuf(s)?

```cpp
// hi.cpp
#include<iostream>
#include "hello.pb.h"

int main() {
    candy::Person p;
    p.set_name("ruchi");
    p.set_id(24);
    p.set_email("rk@gmail.com");
    std::cout << "\n Name is:" << p.name();
    std::cout << "\n id=" << p.id();
    std::cout << "\n email=" << p.email();
    return 0;
}
```

```proto
// hello.proto
syntax ="proto3";
package candy;

message Person {
    string name=1;
    int32 id=2;
    string email=3;
}
```

THANK YOU
Andrzej Gruntowski