Engenharia de Software: Calculadora em 3 camadas - V3a

Breno Keller

Universidade Federal de Ouro Preto

kellerbrenons@gmail.com

Introdução

- Trabalho prático para exemplificar o desenvolvimento de uma aplicação Cliente/Servidor utilizando APIs do QT;
- Objetivo: Evoluir a arquitetura do projeto;
- Código fonte disponível em: `https://bitbucket.org/KellerBreno/calculadora/`
  - Versão 3a.

Arquitetura

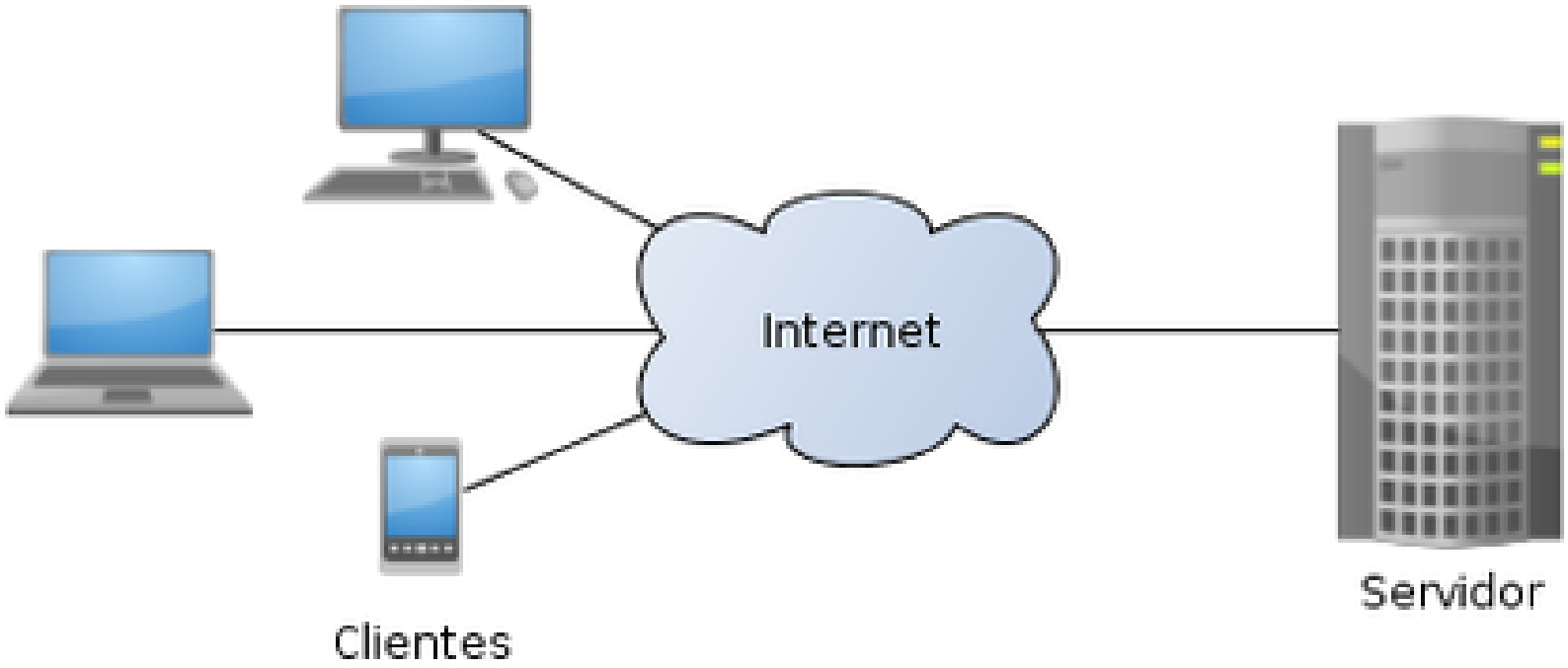

Figura: Arquitetura da Aplicação

Componente: Client até V3

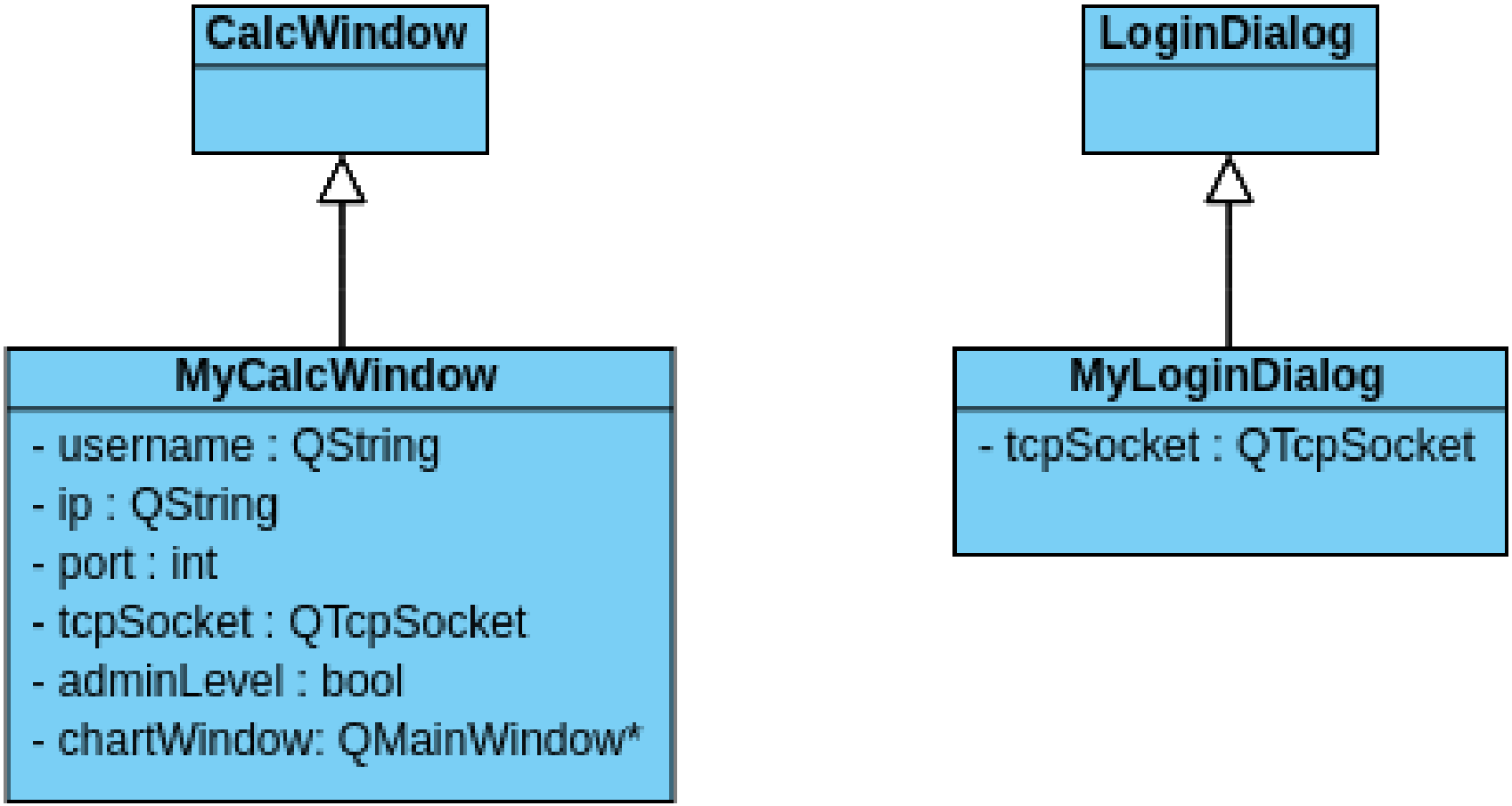

Figura: Diagrama de Classes: Client em V3

Componente: Client em V3a

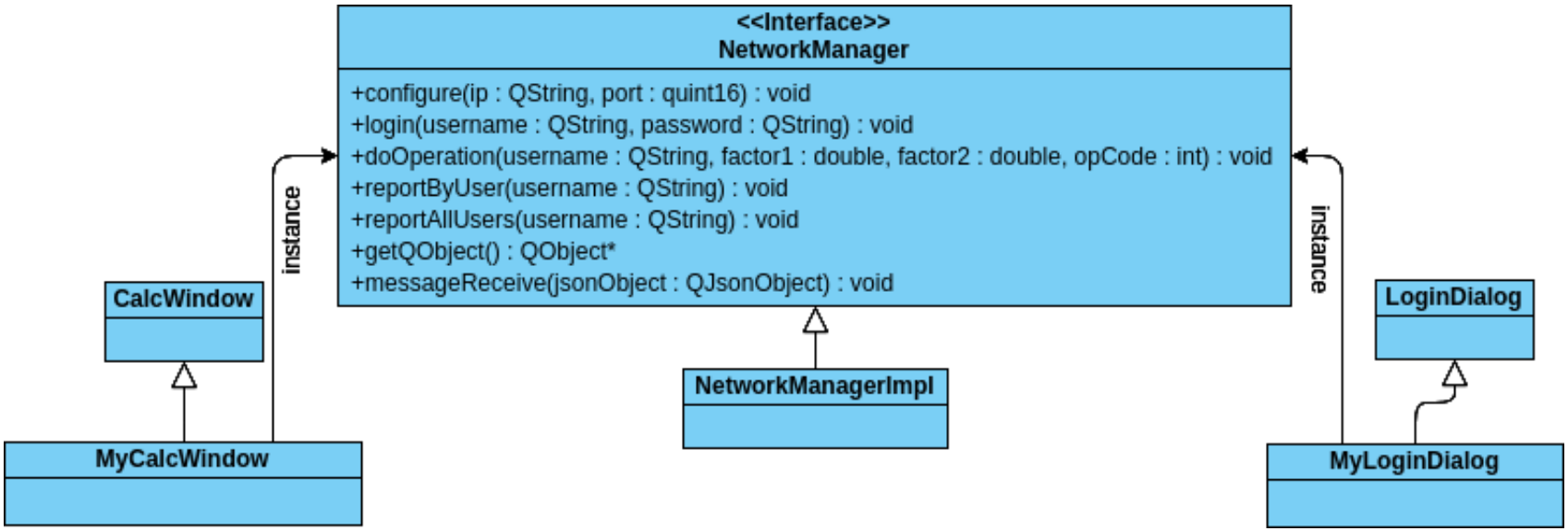

Figura: Diagrama de Classes: Client em V3a

Componente: Server até V2

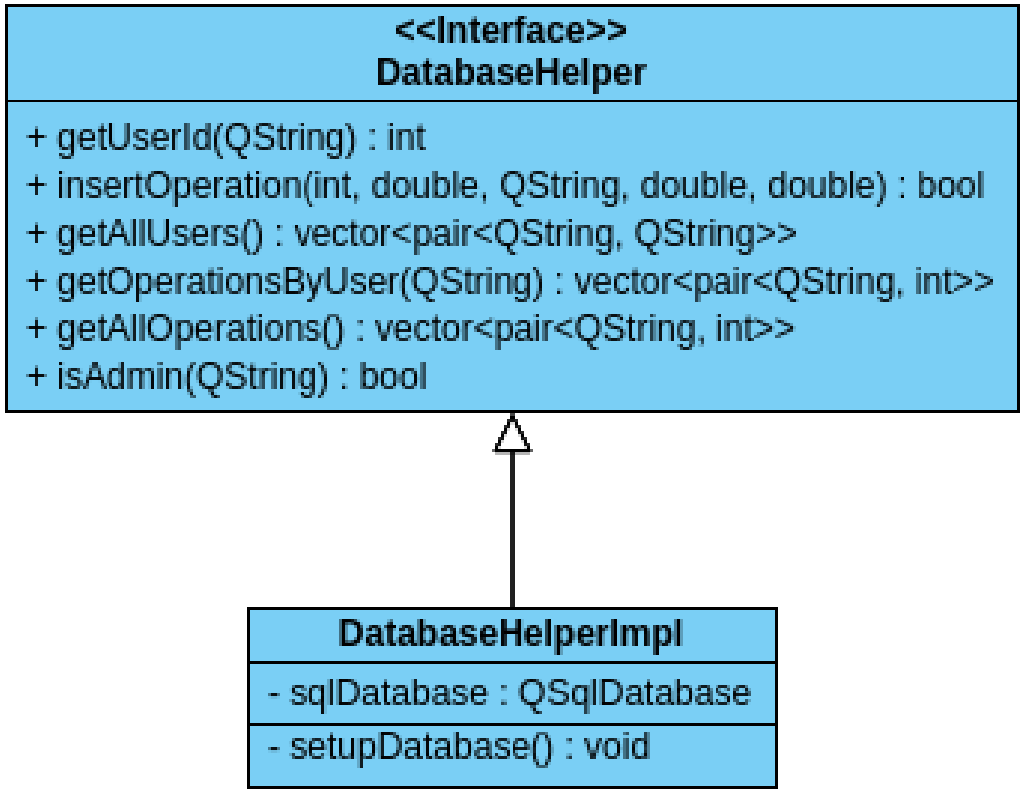

Figura: Diagrama de Classes: Database Helper

Componente: Server até V2

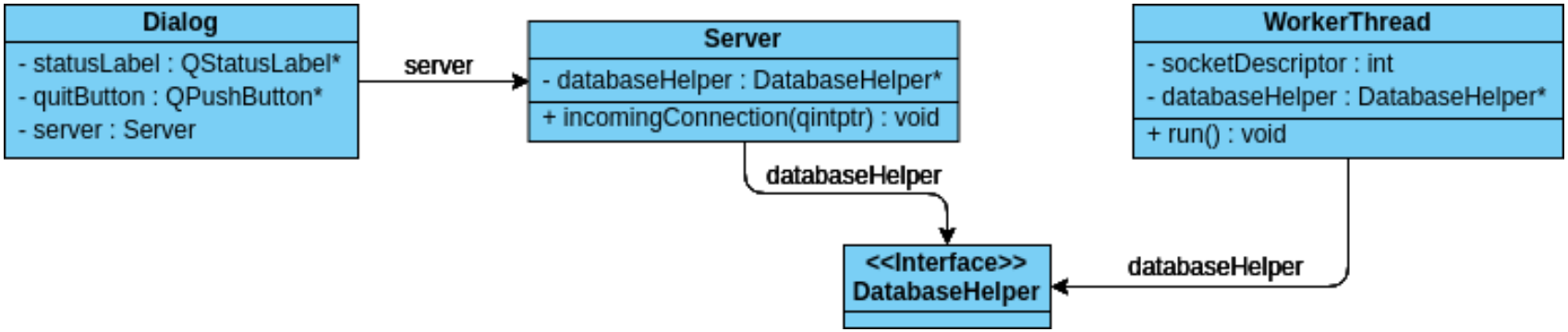

Figura: Diagrama de Classes: Server

Componente: Server a partir de V3

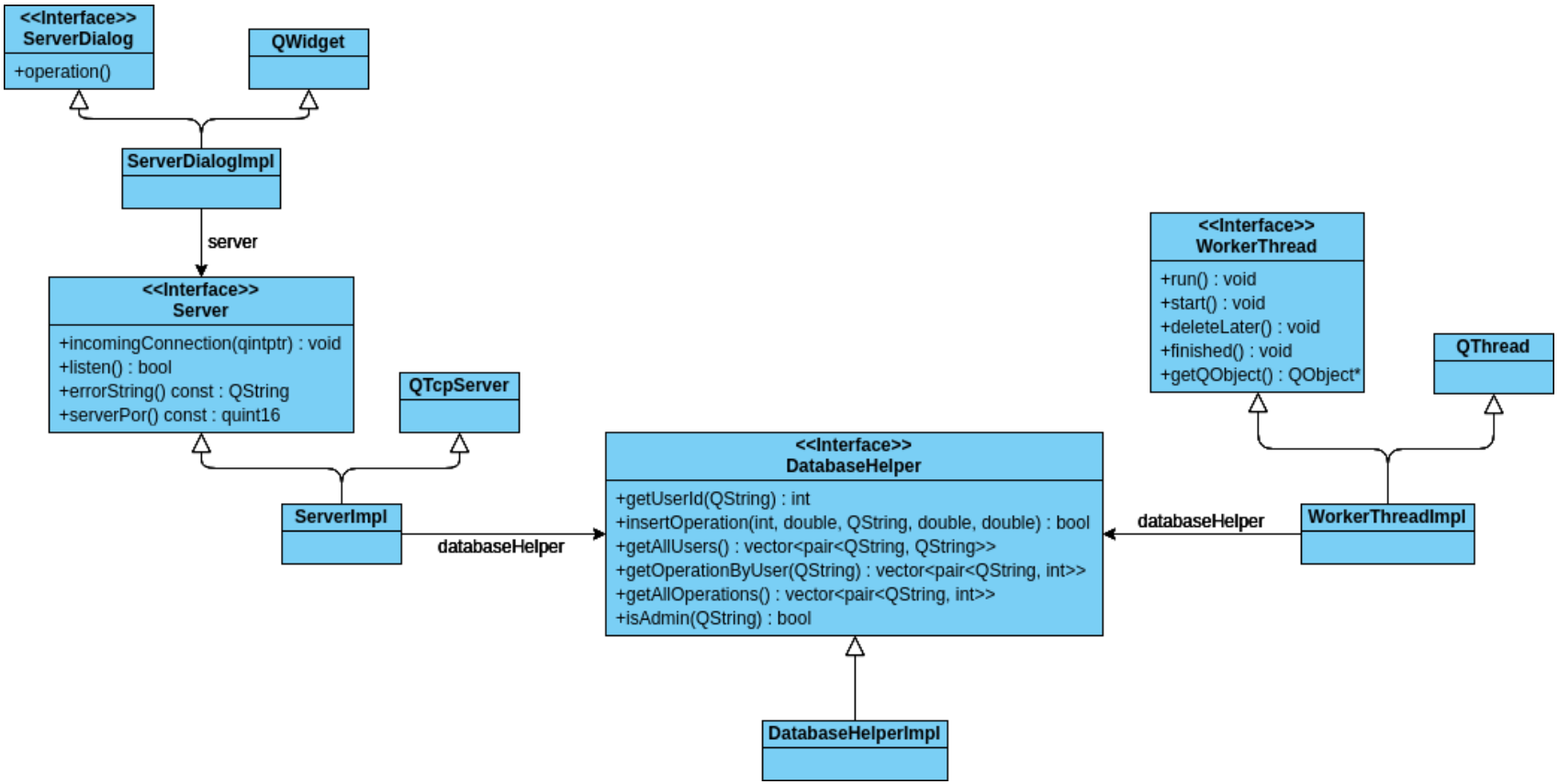

Figura: Diagrama de Classe

Componente: Server a partir da V3

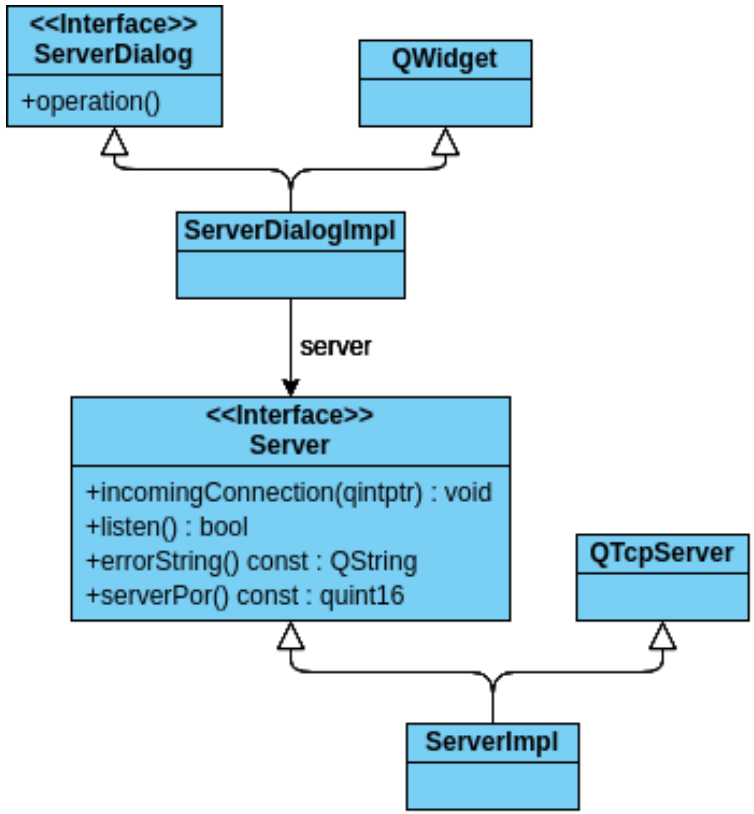

Figura: Diagrama de Classes: Server e ServerDialog

Componente: Server a partir da V3

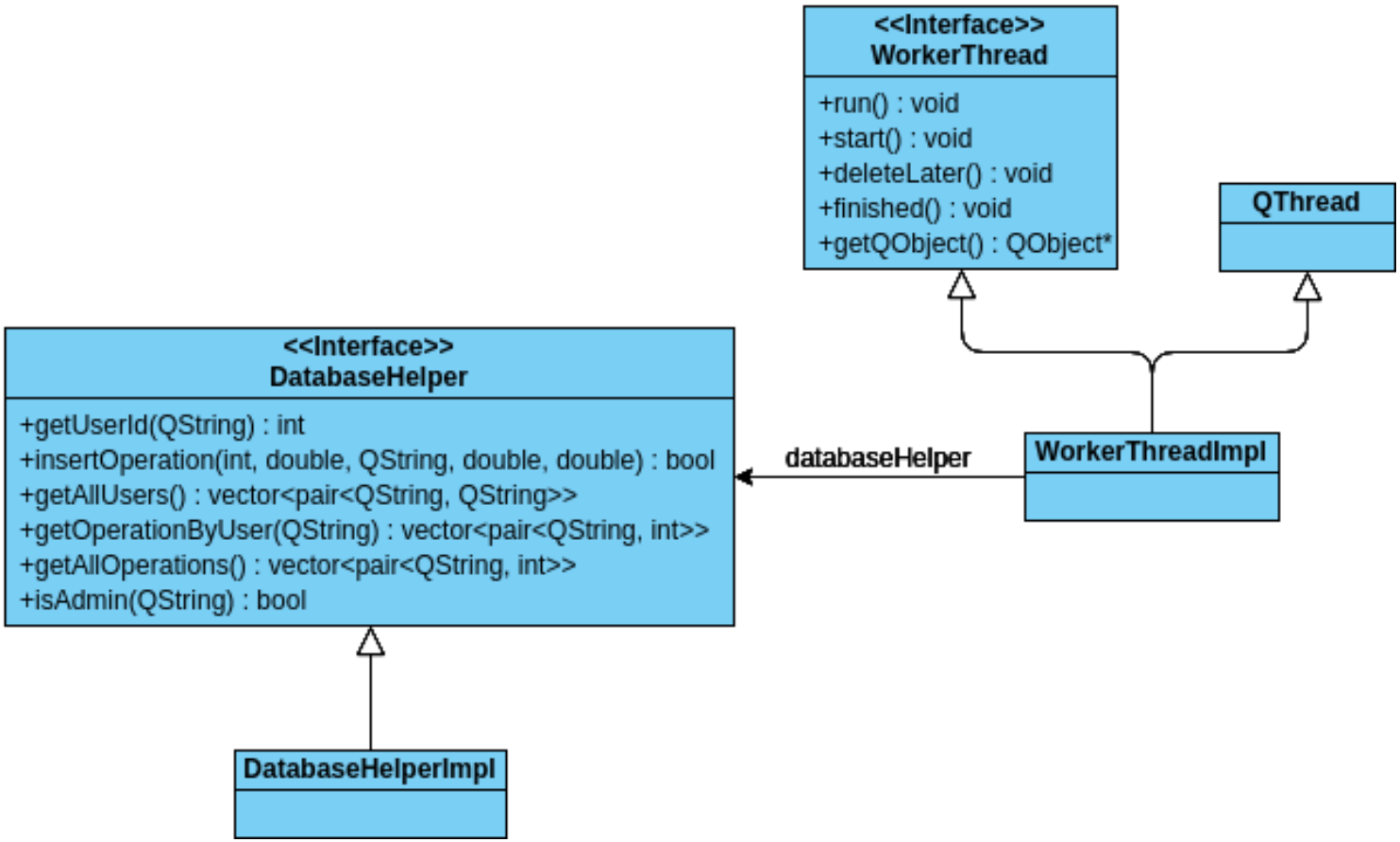

Figura: Diagrama de Classes: WorkerThread e DatabaseHelper

Diagrama de Sequência até V2

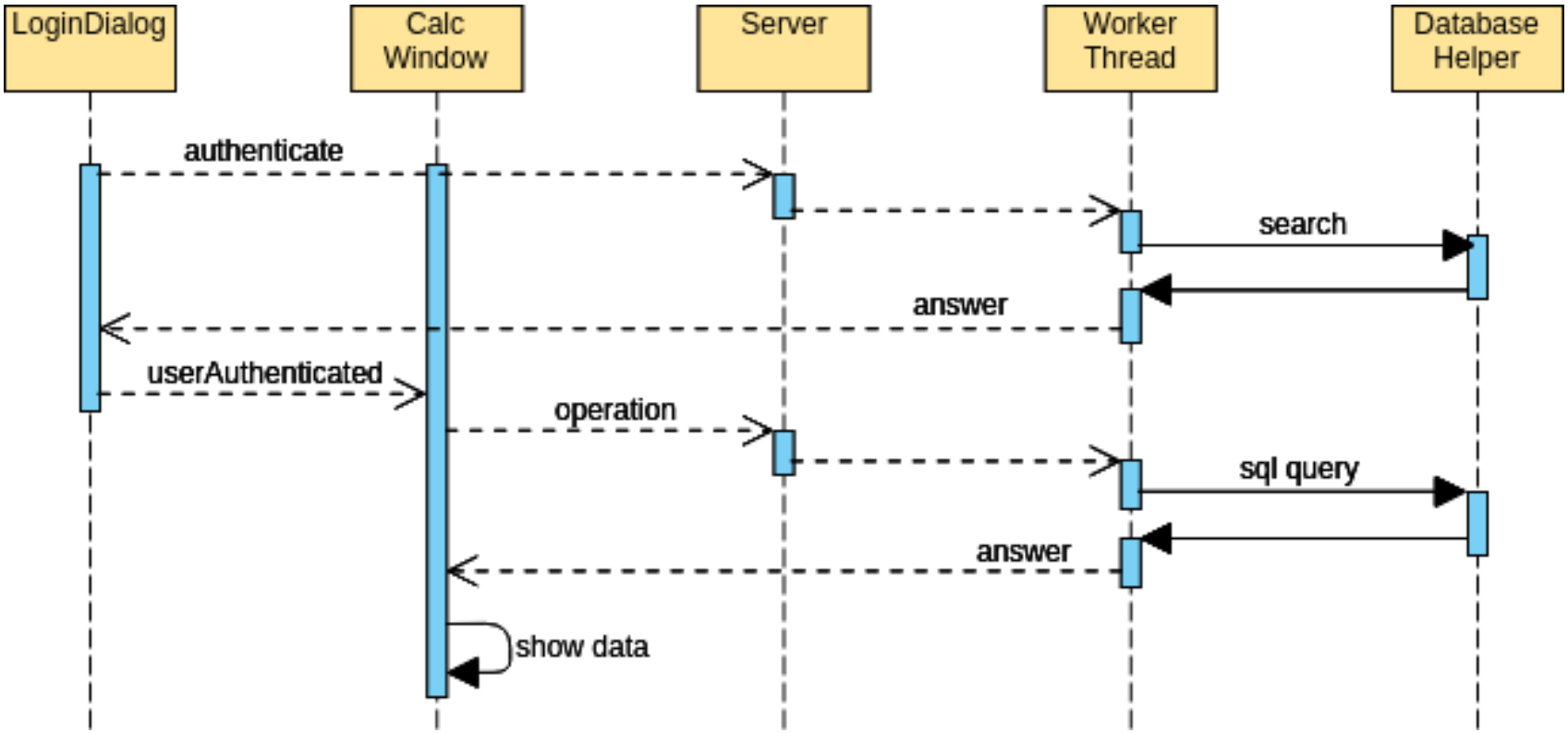

Figura: Diagrama de Sequência

Diagrama de Sequência em V3

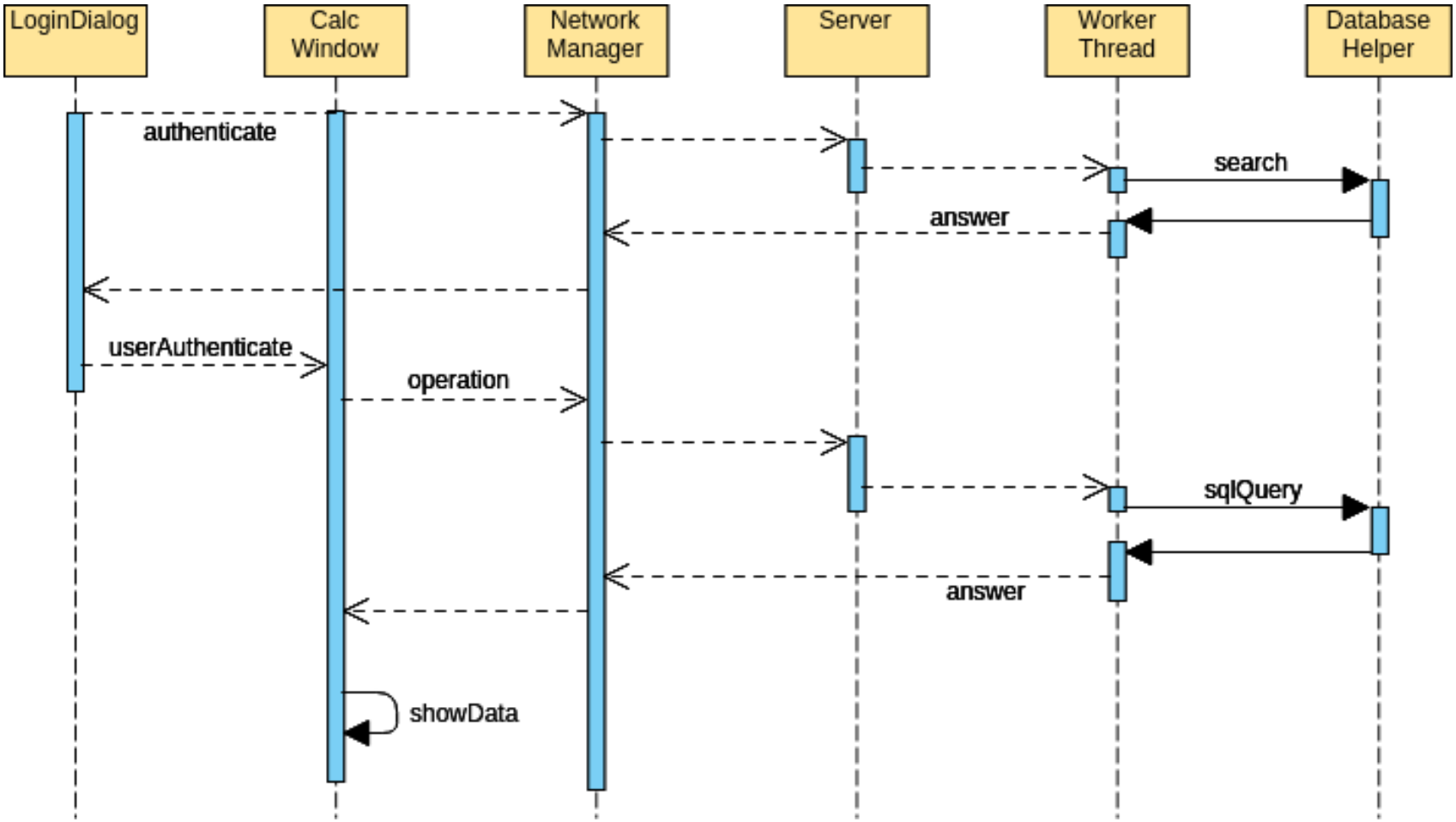

Figura: Diagrama de Sequência

Modelo de Dados

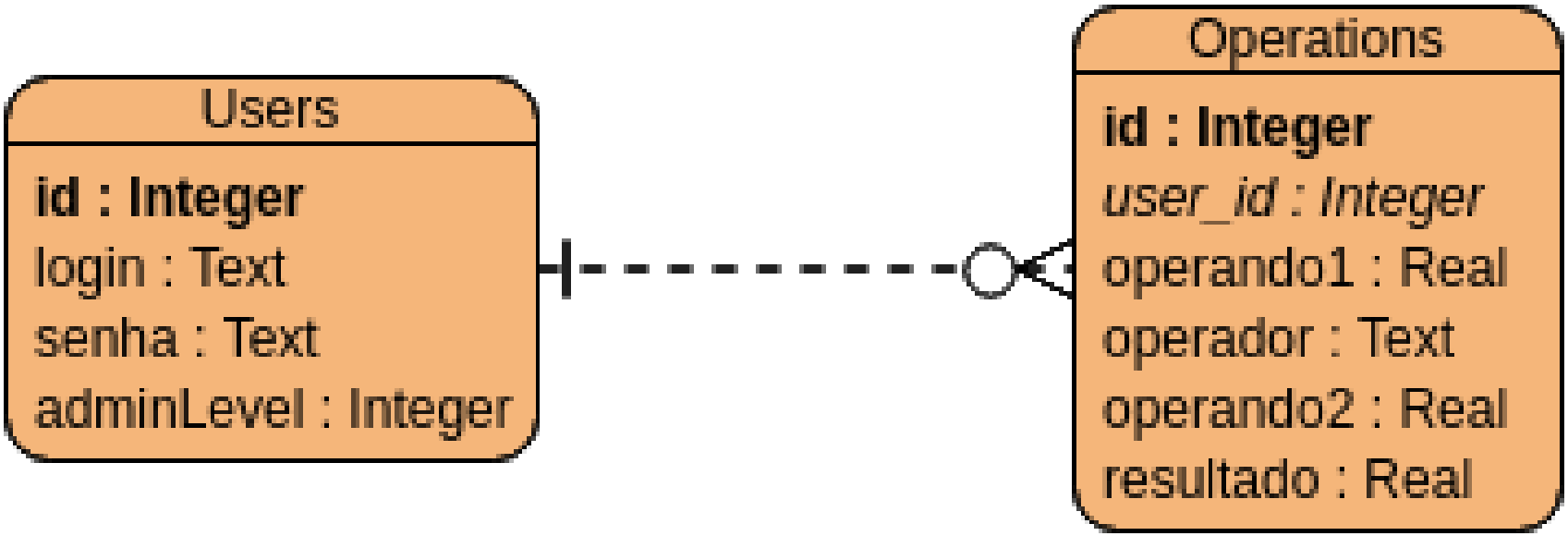

Figura: Modelo de dados utilizado

Validação de Dados

- Validar Dados na GUI;
- Validar Dados no Control;
- Validar Dados no GUI/Control.

Validar na GUI

Vantagem:

- Evitar requisições desnecessárias.

Desvantagem:

- Não é simples a troca de GUI.

Validar no Control

Vantagem:

- Validação centralizada;
- Segurança.

Desvantagem:

- Se torna o gargalo.

Validar no GUI/Control

Vantagem:

- Equilíbrio nas requisições, somente existem requisições válidas.

Desvantagem:

- Segurança.

Login até V3

Até a V3 do código, todo o controle estava na interface.

Login até V3

```
// Client

QString username = user_input_line->text();
QString password = password_input_line->text();
QString ip = ip_input_line->text();
QString port = port_input_line->text();

if(username.isEmpty() || password.isEmpty() ||
    ip.isEmpty() || port.isEmpty()){
    QMessageBox::critical(this, "Login",
        "Por favor preencha todos campos acima",
        QMessageBox::Ok);
    return;
}
```

Login até V3

```
// Client

tcpSocket.connectToHost(ip, port.toInt());

QJsonObject jsonObject;
jsonObject.insert("operationType", 1);
jsonObject.insert("username", username);
jsonObject.insert("password", password);
QJsonDocument jsonDocument(jsonObject);
QString jsonString(jsonDocument.toJson(
    QJsonDocument::Compact));
QByteArray jsonData = jsonString.toUtf8();

tcpSocket.write(jsonData);
```

Login até V3

Em V3a, o controle foi movido para uma classe especializada de controle.

Login a partir de V3a

```
// Client

QString username = user_input_line->text();
QString password = password_input_line->text();
QString ip = ip_input_line->text();
QString port = port_input_line->text();

if(username.isEmpty() || password.isEmpty() ||
        ip.isEmpty() || port.isEmpty()){
    QMessageBox::critical(this, "Login",
            "Por favor preencha todos campos acima",
            QMessageBox::Ok);
    return;
}
```

Login a partir de V3a

```
// Client

NetworkManager *networkManager =
        NetworkManager::getInstance();
networkManager->configure(ip,port.toInt());
connect(networkManager->getQObject(),
        SIGNAL(messageReceive(QJsonObject)), this,
        SLOT(readMessage(QJsonObject)));
networkManager->login(username,password);
```

Login a partir de V3a

```
// Control

if(ip.isEmpty() || port == 0){
    // Throw expection
    return;
}

tcpSocket.connectToHost(ip, port);

QJsonObject jsonObject;
jsonObject.insert("operationType", 1);
jsonObject.insert("username", username);
jsonObject.insert("password", password);
```

Login a partir de V3a

```
// Control

QJsonDocument jsonDocument(jsonObject);
QString jsonString(jsonDocument.toJson(
        QJsonDocument::Compact));
QByteArray jsonData = jsonString.toUtf8();

tcpSocket.write(jsonData);
```

Fim