Desenvolvimento de Controlador de Experimentos OMF em Céu

Carlos Mattoso

Projeto Final de Graduação

Centro Técnico Científico
Departamento de Informática
Curso de Bachalerado em Engenharia de Computação

Orientadora: Prof.ª Noemi Rodriguez

Rio de Janeiro
Junho de 2016

Carlos Mattoso

Desenvolvimento de Controlador de Experimentos OMF em Céu

Relatório de Projeto Final, apresentado ao programa Engenharia de Computação da PUC-Rio como requisito parcial para a obtenção do Bacharel em Engenharia de Computação.

Prof.ª Noemi Rodriguez
Orientadora
Departamento de Informática — PUC–Rio

Rio de Janeiro, 21 de Junho de 2016

Resumo

Mattoso, Carlos; Rodriguez, Noemi. **Desenvolvimento de Controlador de Experimentos OMF em Céu**. Rio de Janeiro, 2016. 22p. Relatório de Projeto Final — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Neste trabalho será implementado um controlador de experimentos OMF na linguagem de programação Céu que possibilite o uso dos construtos nativos da linguagem para tratamento de eventos na descrição de experimentos destinados a ambientes de experimentação em rede. Um sub-produto deste trabalho será um *binding* em Céu para uma biblioteca de AMQP escrita na linguagem de programação C, que concilie as propriedades e métodos desta biblioteca ao paradigma de programação reativa e estruturada de Céu.

Palavras–chave

Ambientes de experimentação em rede. Programação orientada a eventos. Passagem de mensagens por filas.

Abstract

Mattoso, Carlos; Rodriguez, Noemi. **Development of OMF Experiment Controller in Céu**. Rio de Janeiro, 2016. 22p. PhD Thesis — Department of Informática, Pontifícia Universidade Católica do Rio de Janeiro.

In this project an OMF experiment controller will be implemented in the Céu programming language to enable the use of the languages's native event handling constructs for the description of experiments targeting networking testbeds. A byproduct to this work will be a Céu binding to an AMQP library written in C that combines the properties and methods of such library with the structured reactive programming paradigm of Céu.

Keywords

Networking Testbeds. Event-oriented Programming. Message Queuing.

Sumário

1
Introdução

A emergência de ambientes de experimentação, conhecidos como *testbeds*, para o amparo do desenvolvimento de agentes conectatos em rede, cria a necessidade por protocolos e ferramentas que possibilitem a descrição dos recursos, da configuração e dos processos que definem completamente um experimento reproduzível. Neste contexto, desenvolveu-se o *OMF* (1), um arcabouço para gerência de *testbeds* e controle de experimentos.

O objetivo do OMF é assegurar o nível apropriado de abstração, tanto do ponto de vista do operador de *testbeds* quanto do pesquisador (2). O operador tem acesso a um conjunto de serviços que facilitam a gerência do *testbed*. Por outro lado, o arcabouço oferece suporte a *scripts*, escritos pelo pesquisador em linguagem de alto nível, que descrevem o experimento a ser realizado, automatizando sua execução.

O LabLua desenvolveu, nos últimos anos, pesquisas sobre redes de sensores sem fio (RSSF) e participou do desenvolvimento do *testbed* CeuNa-Terra (3), implementado com o apoio da Rede Nacional de Ensino e Pesquisa. Também desenvolveu um conjunto de tecnologias que podem ser empregadas em aplicações destinadas a RSSF, como a linguagem de programação estruturada síncrona e reativa Céu (4) e o sistema de programação de RSSF Terra (5).

Neste contexto, surgiu a ideia de se expandir os casos de uso da linguagem Céu através do desenvolvimento nesta de uma implementação alternativa do arcabouço OMF. Isto se deve a experimentos OMF serem descritos em torno de eventos externos e internos aos agentes sob teste e a existência de construtos nativos na linguagem para definição e tratamento de eventos como entidades de primeira classe.

2
Situação Atual

O arcabouço OMF pode ser discretizado em três planos: controle, medição e gerenciamento (1). O primeiro engloba as ferramentas e diretivas que possibilitam ao pesquisador definir e por em execução um experimento; aqui destaca-se o *controlador de experimentos (CE)*. O segundo é constituído das tecnologias que possibilitam ao pesquisador coletar métricas dos recursos sob teste para avaliação posterior; a coleta de métricas é definida através da *OMF Measurement Library (OML)*, não abordada neste trabalho. Finalmente, a camada de gerenciamento diz respeito a infraestrutura e *testbed* nos quais o experimento é executado, destacando-se neste plano os recursos e seus controladores.

Os recursos na camada de gerenciamento são os alvos do experimento. Para possibilitar o desenvolvimento de tais recursos como módulos coesos e independentes, o arcabouço OMF define entidades denominadas *controladores de recursos (CRs)*, que são responsáveis por intermediar a comunicação entre o CE e os recursos sob seu controle. A distribuição oficial do OMF disponibiliza uma biblioteca em *Ruby*, chamada de *omf_rc* (8), que permite a descrição e uso de controladores de recursos. Contudo, controladores de recursos podem ser implementados sem depender desta biblioteca, desde que lidem corretamente com os protocolos de comunicação.

O CE na camada de controle é o aplicativo que executa o experimento, sendo sua implementação oficial chamada de *omf_ec* (7). Para executar um experimento, o pesquisador deve fornecer como entrada para o CE um arquivo denominado *descrição de experimento (DE)* que especifique completamente os recursos sob teste, sua configuração inicial e as ações a serem executadas sobre os mesmos durante o experimento. Com base neste arquivo, o CE envia pedidos aos CRs dos recursos sob teste e coleta as respostas enviadas por estes, exibindo-as ao usuário caso este especifique tal comportamento. Na distribuição oficial do OMF, tais arquivos devem ser elaborados na linguagem de programação *OMF Experiment Description Language (OEDL)* (6). Esta linguagem é, na verdade, uma extensão de *Ruby* que provê uma série de comandos específicos ao domínio de definição e orquestração de experimentos.

Deste modo, o pesquisador pode facilmente descrever seus experimentos, mas também tem acesso as funcionalidade nativas de *Ruby*, sendo possível realizar lógicas mais complexa em seu DE.

A comunicação entre o CE e os CRs não se dá de forma direta. O arcabouço OMF exige o emprego de um servidor que suporte o paradigma de troca de mensagens *pub/sub*. Utilizando-se este servidor, ambas as partes trocam mensagens através de tópicos definidos ao longo da execução do experimento, segundo o protocolo *Federated Resource Control Protocol (FRCP)* (9). Este protocolo especifica o formato do *payload* das mensagens intercambiadas e uma série de regras sobre como o intercâmbio destas deve ocorrer, segundo seu tipo e parâmetros. A implementação oficial de OMF suporta dois tipos de protocolos de troca de mensagens: *Extensible Messaging and Presence Protocol (XMPP)* e *Advanced Message Queuing Protocol (AMQP)* (A), este mais recomendado.

AMQP especifica que a troca de mensagens entre aplicações deve ocorrer por intermédio de filas de mensagens atreladas a *exchanges*, ambos definidos e gerenciados por um servidor central, chamado de *broker*. Aplicações produtoras de mensagens as enviam para um *exchange* que, com base em seu tipo e em uma chave de roteamento possivelmente definida no cabeçalho de cada mensagem, roteia cada uma para o subconjunto apropriado de filas ou para o descarte, caso nenhuma fila atrelada ao *exchange* esperasse por mensagens com a chave especificada. Aplicações consumidoras, por sua vez, increvem-se em filas e consomem mensagens destas segundo um critério de *round-robin*. No contexto de OMF deve-se utilizar um servidor que implemente o protocolo AMQP, através do qual o CE e CRs realizam sua troca de mensagens.

O ciclo de vida de um experimento inicia-se com a elaboração de uma descrição de experimento que é então passada como entrada a um controlador de experimentos. Este, por sua vez, inicia uma sequência de troca de mensagens com os controladores de recursos especificados na DE. Esta troca de mensagens ocorre através de um *broker* AMQP, sendo a definição dos tópicos, empacotamento das mensagens e roteamento destas realizados segundo as regras estipuladas pelo FRCP. Tais processos são executados por trás dos panos pelo CE, sem a necessidade de conhecimento ou intervenção do pesquisador sobre os mesmos. Ao fim do experimento, o estado no servidor de mensagens é limpo e possíveis métricas são armazenadas automaticamente.

Embora OMF seja um arcabouço robusto, uma limitação inerente a *Ruby* é a ausência de eventos como entidades de primeira classe. Isto acarreta em definições de eventos e diretivas para seu tratamento não tão naturais, constituindo uma barreira para a introdução de pesquisadores a este ambiente.

Para facilitar o uso de *testbeds* OMF, pretendemos oferecer a possibilidade do pesquisador escrever a descrição de seu experimento em Céu, uma linguagem desenvolvida para lidar com sistemas orientados a eventos.

3
Proposta e Objetivos do Trabalho

O principal produto deste trabalho será a implementação de um CE alternativo em Céu que apresente paridade de funcionalidades frente ao oficial e seja capaz de processar DEs escritas em Céu. A escrita de DEs em Céu tornará possível a definição e tratamento de eventos como entidades de primeira classe, simplificando a estrutura desses arquivos. Além disso,introduzirá um novo caso de uso de Céu sob o domínio de definição e orquestração de experimentos em *testbeds*.

É necessária a adoção de uma biblioteca de AMQP para que o CE Céu seja capaz de se comunicar com os CRs segundo o processo delineado no capítulo anterior. Sua implementação em Céu como um módulo independente justifica-se por tornar mais conveniente seu emprego em futuras aplicações Céu. Esta biblioteca será implementada com base na biblioteca C *rabbitmq-c* (B), expondo as funcionalidades essenciais do AMQP por esta suportadas. Contudo, devido a natureza bloqueante da implementação de certas funcionalidades na biblioteca C, propriedade esta contrária ao paradigma síncrono reativo de Céu, serão impostas limitações ao uso de tais funcionalidades para que se ofereça uma biblioteca de mais alto nível em Céu.

As bibliotecas desenvolvidas adotarão de forma o mais rigorosa quanto possível as imposições de Céu, em detrimento aos conceitos dos protocolos AMQP e OMF. Por exemplo, ao invés de se definir uma variável de estado em uma chamada AMQP que destrói uma entidade quando do término de uma conexão, será imposto ao programador que a destruição da entidade ocorra através do término do escopo no qual sua representação em Céu reside. Deste modo, impera o conceito de programação estruturada de Céu: o fluxo de execução do programa se dá segundo reações a eventos, as quais implicam a ativação ou terminação de escopos que, por sua vez, levam a criação e execução ou terminação de entidades (e.g. filas AMQP).

Para validação dos produtos desenvolvidos, será disponibilizada uma série de testes que garanta a corretude das funcionalidades da biblioteca AMQP e do CE OMF Céu. Finalmente, se houver disponibilidade de tempo, tentar-se-á também neste trabalho apresentar uma demonstração de um experimento que

execute sobre o *testbed* CeuNaTerra através de comandos publicados pelo CE desenvolvido em Céu. Ainda, para fins de comparação, seria também feita uma demonstração equivalente utilizando o CE oficial.

Este trabalho não irá oferecer uma integração com OML, mas o código final do CE deverá ser o mais claro e extensível possível para que tal integração possa vir a ser adicionada em trabalhos futuros. Além disso, não será entregue uma biblioteca em Céu para implementação de controladores de recursos.

4
Atividades Realizadas

4.1
Estudos Preliminares

O aluno tinha prévia experiência com Céu, devido a participação em projeto de pesquisa para desenvolvimento de algoritmos distribuídos destinados a ambiente de redes de sensores sem fio. Contudo, tal pesquisa ocorreu ao longo de 2012 e a linguagem sofreu profundas mudanças desde então. Durante esta pesquisa o aluno também foi exposto ao conceito de *testbeds*, mas não chegou a utilizá-las.

Por outro lado, neste trabalho o aluno foi exposto pela primeira vez aos conceitos, técnicas e ferramentas dos domínios de envio de mensagens por fila em sistemas distribuídos, empregados no AMQP, e orquestragem de ambientes de experimentação, empregados no OMF.

4.2
Estudos Conceituais e da Tecnologia

Em razão da exposição limitada do aluno aos domínios dos protocolos AMQP e OMF, foi necessário um estudo aprofundado de suas documentações. Neste processo, estudou-se também o servidor *RabbitMQ*, uma instância de um *broker* AMQP, e as implementações oficiais em *Ruby* de um controlador de experimentos OMF e de uma biblioteca para desenvolvimento de controladores de recursos OMF.

Através do estudo do material relacionado, o aluno familiarizou-se com as entidades primitivas de AMQP, aprendeu a forma como são expostas através do *RabbitMQ* e as propriedades deste para ser capaz de ajustar suas configurações a fim de realizar os testes necessários. Por fim, analisou bibliotecas de AMQP em *JavaScript*, *Python* e *C*, a fim de ter uma boa base para projetar e implementar a versão em Céu.

No contexto de OMF, estudou-se a arquitetura de um projeto OMF, a fim de se entender bem o encaixe de cada componente da mesma, focando-se principalmente nos papéis do controlador de experimentos e do servidor de mensagens utilizado para a comunicação entre os controladores de recursos e

o controlador de experimentos. Ainda quanto ao processo de comunicação, o aluno dominou os conceitos e regras estipulados pelo FRCP, que dita como a comunicação deve ocorrer.

Por fim, foi necessário ao aluno ler o manual e estudar os tutoriais de Céu, a fim de se refamiliarizar com a linguagem. Neste sentido, também foi importante adaptar-se ao estilo de programação reativa e estruturada imposto por Céu, que difere significativamente de linguagens as quais estava mais acostumado, como *C*, *Java* e *JavaScript*.

4.3 Testes e Protótipos para Aprendizado e Demonstração

O principal desenvolvimento se deu na frente de AMQP. Implementou-se uma versão inicial de uma biblioteca de AMQP em Céu (C), que disponibiliza módulos equivalentes às entidades primitivas de AMQP além de auxiliares para certas operações, como o envio de mensagens. Para cada módulo desenvolveram-se testes simples que validam sua funcionalidade. Por fim, implementaram-se dois exemplos empregando-se todos os módulos, a fim de se realizar um teste de integração completo. Em um exemplo enviou-se uma mensagem para um consumidor de mensagens desenvolvido na biblioteca de *Python*. No outro fez-se o caminho contrário: recebeu-se uma mensagem de um publicador escrito em *Python*. Assim, a elaboração desta biblioteca seguindo boas práticas de Céu comprovou a viabilidade da linguagem para sua aplicação sob este domínio.

Sob o escopo de OMF, além dos estudos realizados, executou-se um experimento simples sobre um controlador de recursos simulado utilizando-se a implementação em *Ruby* de um controlador de experimentos. Este teste possibilitou ao aluno aprender o funcionamento da ferramenta, observar o fluxo da execução de um experimento e compreender a troca de mensagens que ocorre entre os dois pontos ao longo deste processo.

4.4 Método

O desenvolvimento de cada parte do projeto, isto é, os componentes AMQP e OMF, se deu em duas etapas: primeramente, um estudo do problema a ser atacado e, posteriormente, a implementação da solução.

O estudo dos conceitos, técnicas e ferramentas dos domínios de AMQP e OMF foi feito de forma bastante aprofundada, tendo sido os pontos mais relevantes documentados e apresentados para a orientadora. Os resumos da

documentação foram cuidadosamente registrados para servirem de material de referência durante o desenvolvimento dos códigos do projeto.

O desenvolvimento dos códigos foi um processo bastante iterativo. Conforme os módulos foram desenvolvidos, consultou-se o autor da linguagem Céu para sua análise e crítica, com base na qual os módulos passaram pela refatoração necessária. Deste modo, foi possível o desenvolvimento de módulos fiéis ao paradigma de programação reativa e estruturada imposto pela linguagem e nos quais empregam-se as funcionalidades mais recentes de Céu, servindo assim também de um teste para as mesmas.

5
Plano de Ação

5.1
Proposta Original

Fez-se um estudo entre os meses de Janeiro e Março focado nas partes fundamentais do trabalho. Primeiramente, estudou-se AMQP e RabbitMQ (uma implementação popular de *broker* AMQP), focando-se tanto no projeto de arquitetura quanto em sessões práticas segundo tutorais. Posteriormente, estudou-se o arcabouço OMF, de modo a também se entender o projeto de arquitetura deste e como se encaixam, especificamente, o controlador de experimentos, os controladores de recursos e o FRCP. Com base nestes fundamentos dar-se-á partida no desenvolvimento do projeto ao longo do ano.

No que tange a biblioteca de AMQP em Céu, foi proposta a implementação dos métodos definidos pelo protocolo e dos respectivos testes que validassem seu correto funcionamento. Além disso, propôs-se um estudo de demais bibliotecas AMQP mirando-se paridade de funcionalidades. Quanto ao controlador de experimentos OMF, seria primeiramente estudada em detalhes sua implementação oficial em *Ruby*. Com base nisto, seria desenvolvida uma série de módulos e testes associados, a fim de se compor o CE OMF Céu. Assim, a implementação em Céu proposta seria o mais robusta e extensível possível.

Na primeira quinzena de Maio, deveria ter sido entregue a implementação de uma versão inicial da biblioteca de AMQP em Céu, na qual estariam implementados os principais métodos AMQP com respetivos testes de unidade. Esta biblioteca seria continuamente aprimorada ao longo do ano, principalmente através da correção de *bugs* e melhorias no código, mas também, possivelmente, através da adição de extensões do RabbitMQ.

Ao longo de Maio pretendia-se desenvolver a base do CE OMF em Céu, especificamente os componentes para definição de propriedades e aplicações de um experimento. Além disso, também deveria ter sido produzido um módulo para o FRCP. Em Junho, teria-se estendido o CE com a adição de grupos e eventos. Ao final de Junho, deveria ser possível ao pesquisador executar

um experimento utilizando todos os recursos aqui listados, mesmo que ainda estivessem presentes limitações.

5.2 Realizações em Projeto Final I

Ao longo do primeiro período constatou-se que a proposta original fora um tanto ambiciosa, dadas as limitações de tempo impostas ao aluno devido a participação em um elevado número de disciplinas. Contudo, produziu-se uma ampla documentação sobre os conceitos, regras e implementações das tecnologias AMQP e OMF e desenvolveu-se parte significativa da implementação da biblioteca Céu para AMQP.

A principal realização foi definitivamente o desenvolvimento da biblioteca AMQP em Céu. Isto exigiu uma familiarização as peculiaridades da linguagem e uma reflexão sobre como melhor conciliar os conceitos de AMQP com o modelo de programação estruturada que impera em Céu. Por ser ainda uma linguagem em desenvolvimento, a implementação de alguns módulos esbarrou em *bugs* do compilador; estes, todavia, foram prontamente corrigidos pelo autor da linguagem.

Quanto ao OMF, embora fora planejado o desenvolvimento de um módulo limitado para controle de experimentos em Céu, isto não aconteceu. Contudo, estendeu-se o estudo do protocolo realizado no começo do ano através de uma análise da implementação em *Ruby* do controlador de experimentos, o que possibilitou a compreensão da maneira como grupos e aplicações são representados, elucidando, por exemplo, a necessidade de gerenciar diferentes instanciações de uma mesma aplicação em um grupo; além disso, este estudo possibilitou uma melhor compreensão do encaixe do FRCP sob o escopo do controlador de experimentos.

5.3 Projeto Final II

Em resumo, para o segundo período, os objetivo principais são: continuar o desenvolvimento do módulo Céu de AMQP, implementar o controlador de experimentos OMF em Céu, desenvolver uma demonstração interessante para validação do projeto, eliminação de *bugs* e refatoramento do código onde vantajoso e, por fim, elaboração de um detalhado relatório final.

Para a biblioteca Céu de AMQP, faltam as seguintes pendências: automação dos testes de cada módulo; refatoração do módulo de filas para que as mensagens sejam recebidas por clientes da biblioteca através de instanciações de tal módulo; desenvolvimento dos exemplos oficiais utilizando o módulo Céu

para facilitar seu aprendizado, focando em explicitar as diferenças de projeto relativas a bibliotecas AMQP escritas em outras linguagens, devido a imposição de conceitos e técnicas de programação estruturada em Céu.

Contudo, o controlador de experimentos OMF em Céu é a prioridade. Primeiramente, será desenvolvido um módulo base que permita a definição das entidades básicas: grupos e aplicações. Para isto, utilizar-se-á a integração de Céu a Lua, possibilitando a definição de tabelas para especificação das entidades. Estendendo-se este módulo base, serão implementadas as etapas iniciais definidas pelo FRCP, com base no estudo deste realizado ao longo de Projeto Final I, para que se verifique a troca inicial de mensagens ocorrida em um experimento entre um controlador de recursos e o controlador de experimentos OMF em Céu.

Feito isto, há de se estender o controlador de experimentos para aceitar um arquivo Céu mais elaborado que descreva as operações a serem executadas, como, por exemplo, a operação de inicialização das aplicações instanciadas dentro de um grupo. Nesta etapa, há de se implementar uma versão em Céu de exemplos oficiais a fim de se verificar a corretude do módulo desenvolvido.

Por fim, tentar-se-á atacar o problema de integração do controlador de experimentos ao *testbed* CeuNaTerra, pelo menos a nível de estudo. Se possível, sua implementação será planejada e executada ao longo da disciplina de Projeto Final II.

6
Cronogramas

6.1
Cronograma Original

O cronograma abaixo resume o que havia sido especificado na Proposta:

- Janeiro a Março: estudo de Céu, AMQP e OMF.
- Abril: desenvolvimento da biblioteca de AMQP em Céu.
- Maio: fim do desenvolvimento da biblioteca de AMQP e início do desenvolvimento da base do controlador de experimentos OMF em Céu.
- Junho: fim do desenvolvimento de uma versão limitada do controlador de experimentos e documentação dos trabalhos.
- Julho a Setembro: melhorias pontuais ao que fora desenvolvido.
- Outubro: continuidade da implementação do controlador de experimentos, desenvolvimento de testes e início da integração ao *testbed* CeuNaTerra.
- Novembro: fim da implementação do controlador de experimentos, aprimoramentos aos códigos produzidos e fim de integração ao CeuNaTerra
- Dezembro: escrita do relatório final.

6.2
Cronograma Revisado

O cronograma que segue especifica o que foi de fato desenvolvido e o trabalho que há de ser feito ao longo da disciplina de Projeto Final II:

- Janeiro e Fevereiro: estudo de Céu, AMQP e OMF.
- Março: estudo de FRCP e de biblioteca em *Ruby* para criação de controladores de recursos OMF.
- Abril e Maio: desenvolvimento da biblioteca de AMQP e estudo da implementação em *Ruby* do controlador de experimentos OMF.
- Junho: documentação dos trabalhos desenvolvidos ao longo do período.

- Julho a Setembro:
  - Desenvolvimento do módulo base de OMF, permitindo a declaração de grupos e aplicações.
  - Automação dos testes das biblioteca de AMQP em Céu.
  - Refatoração do módulo de filas.
  - Desenvolvimento dos exemplos oficiais do *RabbitMQ*.
- Outubro:
  - Continuidade da implementação do controlador de experimentos OMF em Céu.
  - Desenvolvimento de testes e versões em Céu dos exemplos oficiais de OMF.
  - Estudo da integração ao *testbed* CeuNaTerra.
- Novembro:
  - Fim do desenvolvimento do controlador de experimentos OMF em Céu.
  - Aprimoramentos a todos os códigos produzidos (AMQP e OMF).
  - Se possível, execução do projeto de integração ao *testbed* CeuNaTerra do controlador de experimentos OMF em Céu.
- Dezembro: escrita do relatório final e apresentação para a banca.

Referências Bibliográficas

[1] RAKOTOARIVELO, T.; OTT, M.; JOURJON, G. ; SESKAR, I.. **OMF: A Control and Management Framework for Networking Testbeds**. SIGOPS Oper. Syst. Rev., 43(4):54–59, Jan. 2010. 1, 2

[2] DWERTMANN, C.; RAKOTOARIVELO, T.. **About OMF - An Executive Summary**. https://omf.mytestbed.net/projects/omf/wiki/Introduction. 1

[3] ROSSETTO, S.; SILVESTRE, B.; RODRIGUEZ, N.; BRANCO, A.; KAPLAN, L.; LOPES, I.; BOING, M.; SANTANA, D.; LIMA, V.; GABRICH, R. ; OTHERS. **CeuNaTerra: testbed para espaços inteligentes**. http://sbrc2015.ufes.br/wp-content/uploads/b-139668_1.pdf, 2015. 1

[4] SANT'ANNA, F.; OTHERS. **Céu: Embedded, Safe, and Reactive Programming**. Technical Report 12/12, PUC-Rio, 2012. 1

[5] BRANCO, A.; SANT'ANNA, F.; IERUSALIMSCHY, R.; RODRIGUEZ, N. ; ROSSETTO, S.. **Terra: Flexibility and Safety in Wireless Sensor Networks**. ACM Trans. Sen. Netw., 11(4):59:1–59:27, Sept. 2015. 1

[6] RAKOTOARIVELO, T.; DWERTMANN, C. ; HONG, J.. **The OMF Experiment Description Language (OEDL)**. https://mytestbed.net/projects/omf6/wiki/OEDLOMF6. 2

[7] **Experiment Controller (EC)**. http://mytestbed.net/doc/omf/file.experiment_controller.html. 2

[8] **Resource Controller (RC)**. http://mytestbed.net/doc/omf/file.resource_controller.html. 2

[9] RAKOTOARIVELO, T.. **Federated Resource Control Protocol (FRCP)**. https://github.com/mytestbed/specification/blob/master/FRCP.md. 2

[A] OASIS. **Advanced Message Queuing Protocol (AMQP) Version 1.0**. http://docs.oasis-open.org/amqp/core/v1.0/amqp-core-complete-v1.0.pdf, 2012. 2

[B] ANTONUK, A.. **RabbitMQ C AMQP client library**. `https://github.com/alanxz/rabbitmq-c`, 2016. 3

[C] MATTOSO, C.. **Céu RabbitMQ - Céu API for RabbitMQ**. `https://github.com/calmattoso/ceu-rabbitmq`, 2016. 4.3

Aceitação do Relatório de Projeto Final I

Assinatura do Aluno	Data
Assinatura da Orientadora	Data