Manual de referência do OpenBus 2.0.0

Tecgraf

29 de junho de 2012

Sumário

1 Introdução

O OpenBus [7] é um projeto de um middleware para integração de sistemas computacionais heterogêneos, ou seja, desenvolvidos em diferentes linguagens de programação e plataformas computacionais. O OpenBus se baseia em duas tecnologias complementares para constituir uma infraestrutura básica de integração de sistemas. São elas:

CORBA *Common Object Requester Broker Architecture* [6] é um padrão da indústria que especifica um middleware para sistemas distribuídos heterogêneos orientados a objetos. CORBA é define o modelo básico de comunicação usado nas integrações de sistemas feitas com o OpenBus. CORBA tem suporte para inúmeras linguagens de programação e plataformas computacionais, que nos permite integrar com o OpenBus uma grande variedade de sistemas.

SCS *Software Component System* [1] é um modelo simples e flexível de componentes de software baseado em CORBA que permite estruturar sistemas usando uma arquitetura baseada em componentes. O SCS é usado no OpenBus tanto como um modelo arquitetural básico para a infraestrutura básica oferecida como também para estruturar a forma como as intregrações são feitas.

Em cima dessas duas tecnologias o OpenBus introduz duas novas extensões, que basiamente definem a infraestrutura especializada para integração de sistemas computacionais:

Barramento de Integração É o conceito central do OpenBus. O barramento que é o meio através do qual toda interação e comunicação entre os sistemas integrados é feita. O barramento é uma extensão de CORBA com suporte a controle de acesso, que basicamente consiste na autenticação de todo sistema que acessa o barramento, assim como a identificação de forma segura da origem de toda comunicação (chamadas CORBA) feita através desse meio.

Serviços de Apoio à Integração Juntamente ao barramento, o OpenBus também provê suporte para registro e descoberta de serviços ofertados pelos sistemas integrados, comunicação baseada em eventos, entre outras funcionalidades através de uma arquitetura orientada a serviços (*Service-Oriented Architecture*, ou SOA [2]). O objetivo desses serviços é oferecer funcionalidades básicas e essenciais que visem facilitar e agilizar o desenvolvimento da integração dos diferentes sistemas.

Este documento tem como objetivo apresentar o OpenBus 2.0.0 e seus conceitos principais. Nesta seção introdutória apresentaremos uma definição do projeto e os motivadores para o seu uso. Na seção 2 apresentamos a arquitetura do sistema, e entramos um pouco mais em detalhes de como ocorre a comunicação dentro do barramento na seção 3. Em seguida, na seção 4 falamos um pouco mais sobre o projeto de acordo com a visão de cada tipo de usuário, e, por fim, na seção 5 apresentamos um glossário com os conceitos principais do projeto.

1.1 Por que utilizar o OpenBus?

Quando há necessidade de integração entre aplicações, algumas soluções podem ser apresentadas conforme a complexidade do ambiente a ser integrado. Caso o problema se resuma a troca de dados entre as aplicações, uma primeira solução é a integração externa, que consiste na exportação do dado em um determinado formato padronizado que possa ser carregado pela aplicação de destino. Este processo é normalmente manual, deixando a cargo do usuário, ou do *job* que automatiza o processo, a responsabilidade pela sua execução. Um outro problema é a possibilidade de ocorrer perda de informação quando o formato padrão utilizado não consegue representar corretamente o dado de origem.

As opções de soluções disponíveis para a integração, podem ser ampliadas quando há a presença de aplicações de código aberto ou que ofereçam interfaces de programação. Para estes casos, uma integração programática pode ser realizada de forma direta, através de uma implementação que permita ao usuário comandar a transferência de dados. Este modo de interação agrega mais qualidade, diminuindo as chances de perda de informação durante a transferência do dado, pois os metadados das aplicações envolvidas podem ser considerados. Além da troca de dados, há a possibilidade de troca de mensagens entre as aplicações. O problema desta solução é que uma ponte deve ser desenvolvida para cada par de aplicações, o que resulta em um número quadrático de codificações de pontes em relação ao número de aplicações. Por exemplo, para integrar 10 aplicações precisamos implementar 45 pontes. Além disso, a troca de mensagens é naturalmente limitada entre cada par de aplicações.

Os problemas apresentados acima pela programática direta, podem ser eliminados com a adoção de uma plataforma comum de integração entre aplicações. Como por exemplo um *barramento*, em que cada aplicação se conecta a este, segundo um protocolo próprio deste barramento, e carrega ou exporta dados para outras aplicações também conectadas ao barramento. Essa arquitetura define uma linguagem comum entre todas as aplicações, e, induz o compartilhamento de interfaces bem definidas para os serviços comuns entre as aplicações envolvidas. Desta forma o esforço de codificação das pontes pode ser reduzido para um esforço linear, ou seja, para se integrar n aplicações, n pontes são necessárias. No mais, a troca de mensagens dinâmicas entre todas as aplicações passa a ser possível, possibilitando um ambiente de colaboração entre as aplicações.

2 Arquitetura

O OpenBus pode ser essencialmente divido em três partes principais: o barramento, os serviços núcleo, e os serviços extras. A figura 1, apresenta a arquitetura do OpenBus e suas partes principais. Entraremos

em detalhes sobre as partes principais do OpenBus nas subseções a seguir.

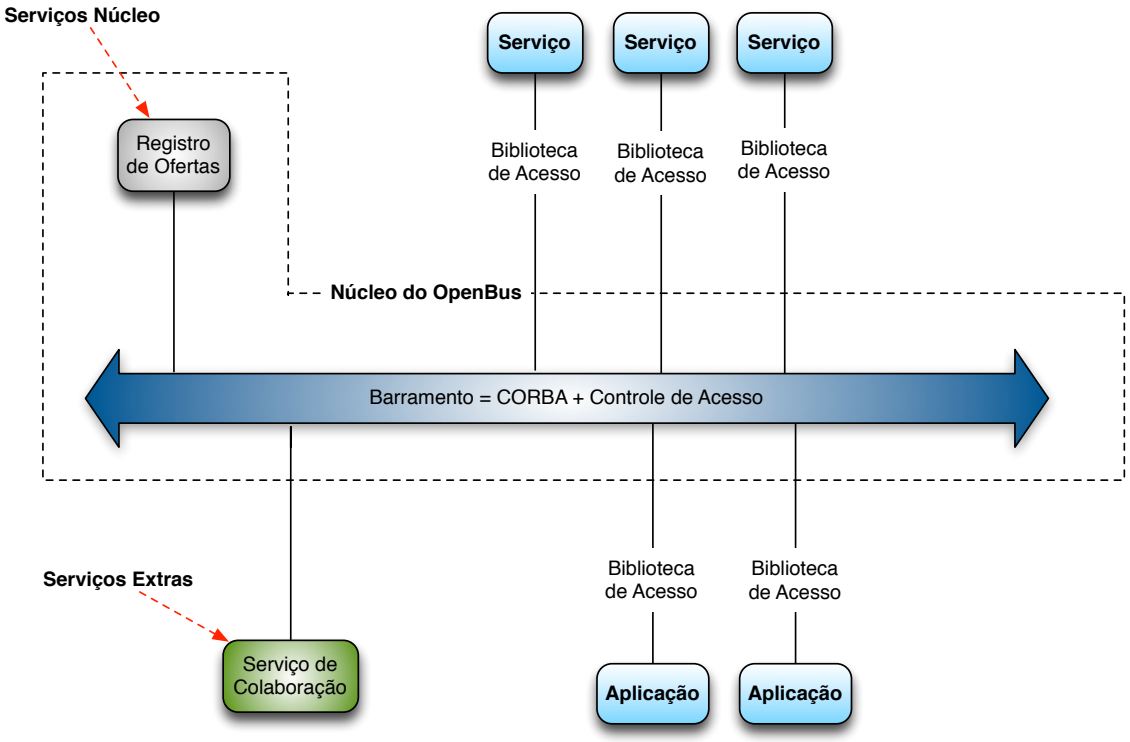

Figura 1: Arquitetura do OpenBus

2.1 Barramento

O barramento é o meio de comunicação entre os sistemas integrados. Toda comunicação feita pelos sistemas integrados através do barramento consiste de chamadas de objetos distribuídos usando o padrão CORBA. Contudo, ao contrário das chamadas de CORBA comuns, nas chamadas feitas através do barramento, é imposto um rigoroso controle de acesso que só permite chamadas por sistemas autenticados, assim como a identificação do sistema que originou cada chamada.

O barramento é composto por um serviço de Controle de Acesso e pelo uso do *middleware* CORBA, com os quais especifica um protocolo de acesso.

O Controle de Acesso serve como ponto de entrada do barramento, sendo responsável por autenticar, renovar e gerenciar os logins de serviços e aplicações ao barramento. A autenticação de login é uma verificação realizada no estabelecimento de um login que permite associá-lo a uma dada entidade. A autenticação pode ser feita através de uma senha (login por senha), certificado previamente registrado no barramento (login por certificado), ou através do compartilhamento de uma autorização de um login previamente estabelecido (login por single sign-on).

O resultado da autenticação é a criação de um login, que representa uma autenticação de uma entidade junto ao barramento para poder acessá-lo. Todo login possui um tempo de validade (lease), no qual permanecerá válido sem necessidade de renovação, mas deve ser renovada para que não expire. Contudo, um login pode ser invalidado explicitamente por um administrador antes do tempo de lease.

O *middleware* CORBA (*Common Object Request Broker Architecture*) é a especificação de um padrão de ORB sobre a qual o OpenBus é definido e implementado, e é a tecnologia adotada para prover suporte à comunicação de objetos distribuídos em múltiplas linguagens. O OpenBus pode ser visto como uma

extensão do padrão CORBA.

Por sua vez, o protocolo de acesso é o conjunto de regras para realização de chamadas CORBA através de um barramento OpenBus, que foi introduzido na versão 2.0.0 do OpenBus. O principal objetivo desse protocolo é garantir um nível adequado de segurança das chamadas que permita assegurar a identidade da entidade que iniciou cada chamada realizada através do barramento. Maiores informações sobre o protocolo podem ser encontradas no documento de especificação do protocolo [10].

Para facilitar o trabalho de integração de serviços e aplicações no barramento, oferecemos uma biblioteca de acesso [11, 12, 13, 14], que implementa o protocolo de acesso e exporta uma interface de programação (API). A biblioteca de acesso é implementada nas linguagens: C++, C#, Java e Lua.

2.2 Serviços Núcleo

Serviços Núcleo são os serviços oferecidos pelo próprio barramento, tipicamente acessados através da API (*Application Program Interface*), que é a interface de programação oferecida às aplicações que precisam acessar o barramento, seja para fazer ou receber chamadas através do barramento, ou acessar serviços oferecidos pelo barramento.

Atualmente o barramento possui apenas um serviço núcleo, que é o Registro de Ofertas. Através dele é possível registrar e buscar serviços oferecidos no barramento, além de permitir a inscrição observadores de registros de ofertas.

2.2.1 Registro de Ofertas

O Registro de Ofertas é um serviço núcleo do barramento, que permite que entidades publiquem e busquem ofertas de serviços no barramento. Uma oferta de serviço corresponde a um cadastro no Registro de Ofertas de um conjunto de propriedades e um componente SCS [1], que implementa um conjunto de interfaces que representam um dado serviço a ser utilizado através do barramento.

Uma propriedade da oferta corresponde a um par de nome e valor, que, ao registrar uma oferta de serviço no barramento, a entidade pode definir um conjunto de propriedades correspondentes à oferta, que poderão ser utilizadas como critério de filtro no momento de buscar por ofertas de serviço no Registro de Ofertas. Toda oferta publicada no barramento possui um conjunto de propriedades geradas automaticamente, como a data do registro, a entidade que publicou a oferta, a versão do componente, entre outras.

O Registro de Ofertas também disponibiliza um mecanismo de observação de publicação, atualização e remoção de ofertas. Maiores detalhes sobre a especificação e o uso do serviço de Registro de Ofertas podem ser encontrados nos manuais de uso das bibliotecas de acesso [11, 12, 13, 14]

2.3 Serviços Extras

A categoria de Serviços Extras define os serviços que acrescentam funcionalidades para auxiliar a integração entre serviços e aplicações, mas que não são parte do núcleo do barramento. Não é obrigatória a presença desses serviços em uma instância do barramento.

Nesta versão atual do OpenBus disponibilizamos apenas um serviço extra: o Serviço de Colaboração. Esse serviço tem como principais funcionalidades:

- criar e compartilhar uma sessão de colaboração
- oferecer um canal de comunicação para enviar eventos para todos os membros da sessão

Não entraremos em maiores detalhes sobre as funcionalidades e o uso do Serviço de Colaboração neste documento. Para mais informações, consulte o manual do próprio serviço [9].

3 Comunicação no OpenBus

Dado que o barramento serve como uma solução de integração, isso quer dizer que todas as chamadas entre serviços e aplicações integradas serão interceptadas pelo barramento? A resposta é não!

Vamos explicar então como ocorre a comunicação dentro do barramento, utilizando como exemplo um cenário onde temos um serviço para publicar no barramento e uma aplicação que vai utilizar o serviço publicado. Iremos dividir esse cenário em duas fases, explicando a seqüência de interação que ocorre dentro das mesmas.

A primeira fase, ilustrada pela figura 2, corresponde ao processo de realização do login, publicação e busca do serviço publicado no barramento. Primeiro o serviço realiza o login no barramento, e com o sucesso no login, ele pode publicar o serviço a ser ofertado no Registro de Ofertas. Então a aplicação realiza também o seu login no barramento, e pode buscar pelo serviço no barramento, através do Registro de Ofertas. Caso as propriedades especificadas na busca correspondam às propriedades da oferta, ela é retornada pela busca e com isso a aplicação consegue obter uma referência para o serviço desejado.

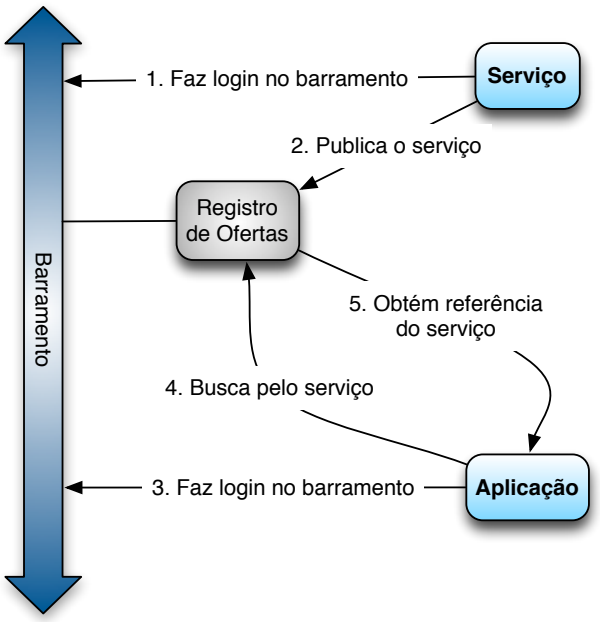

Figura 2: Esquema demonstrando como ocorre o início de uma integração entre duas entidades através do barramento.

Começa então a segunda fase, ilustrada pela figura 3, que é caracterizada pela comunicação direta entre a aplicação e o serviço publicado no barramento. Neste cenário, a aplicação atua como o cliente, e o serviço ofertado como o servidor. O cliente (aplicação) faz uma requisição para o servidor (serviço), e a primeira coisa que o servidor irá fazer é verificar junto ao barramento se o cliente que faz a requisição possui um login válido. Se o login for válido, o servidor então pode atender a requisição.

Essa descrição que apresentamos da segunda fase é uma visão bem simplificada do que realmente acontece, dado que o usuário que faz uso da biblioteca de acesso [11, 12, 13, 14] não precisa se preocupar com esses detalhes, pois as bibliotecas de acesso se encarregam de respeitar e implementar todo o protocolo [10] necessário para que a comunicação seja segura e correta. Na próxima vez que o mesmo cliente falar com o mesmo servidor, o servidor não precisará necessariamente realizar uma chamada remota para o barramento, caso as informações necessárias para validar a requisição ainda estejam válidas e presentes nos caches das bibliotecas.

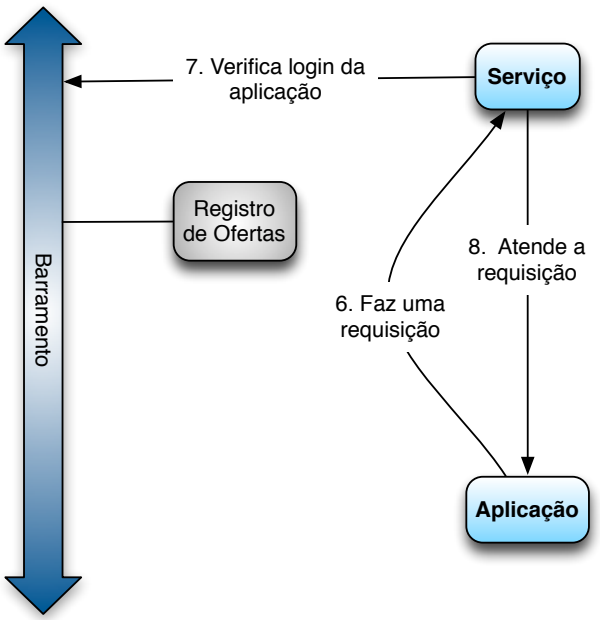

Figura 3: Esquema demostrando como ocorre a comunicação entre duas entidades no barramento.

4 Perspectivas

Agora que já apresentamos a arquitetura e os conceitos gerais do OpenBus, iremos falar um pouco sobre algumas características específicas do OpenBus, que dependem do papel que o usuário tem no barramento.

O OpenBus não traz para a sua implementação uma distinção entre administrador e gerente do barramento. Ambos desempenham papéis de administração. Porém, entendemos que nem sempre o administrador do barramento também irá gerenciar o barramento. Logo, vamos separar esses dois conceitos e apresentar os conceitos do barramento que competem a cada um desses perfis.

4.1 OpenBus para Administradores

Esta seção apresenta alguns conceitos que são importantes para os administradores do barramento, onde por administradores consideramos as pessoas que possuem acesso a máquina onde o barramento será executado e são os responsáveis por levantar e parar o barramento.

O barramento e os serviços núcleo são distribuídos em um mesmo programa, o *busservice*. Logo, para disparar o barramento, basta executar o programa passando as suas configurações por linha de comando ou por arquivo de configuração. Se uma configuração não for explicitada, será utilizado o seu valor padrão.

As opções de configuração são:

-host Endereço de rede usado pelo barramento.
Valor padrão é "*", que significa que vai ouvir em todos os IPs da máquina.

-port Número da porta usada pelo barramento.
Valor padrão é 2089.

-database Arquivo de dados do barramento.
Valor padrão é "openbus.db".

-privatekey Arquivo com chave privada do barramento.
Valor padrão é "openbus.key"

-leasetime Tempo em segundos de lease dos logins de acesso.
Valor padrão é 180 segundos.

-expirationgap Tempo em segundos que os logins ficam válidos após o lease.
Valor padrão é 10 segundos.

-admin Usuário com privilégio de administração.
Não é definido um valor padrão.

-validator Nome de pacote de validação de login.
Não é definido um valor padrão.

-loglevel Nível de log gerado pelo barramento.
O valor padrão é 3. Os níveis vão de 0 a 5, onde 5 é o nível com mais detalhes, e o 0 desativa o log.

-logfile Arquivo de log gerado pelo barramento.
Não é definido um valor padrão.

-oilloglevel Nível de log gerado pelo OiL [5, 4] (debug).
O valor padrão é 0. Os níveis vão de 0 a 5, onde 5 é o nível com mais detalhes, e o 0 desativa o log.

-oillogfile Arquivo de log gerado pelo OiL (debug).
Não é definido um valor padrão.

-noauthorizations Desativa o suporte a autorizações de oferta.

-nolegacy Desativa o suporte à versão antiga do barramento.

-configs Arquivo de configurações adicionais do barramento.
O valor padrão é "openbus.cfg".

As opções "admin"e "validator"podem receber mais de um valor. Quando essas opções são configuradas por linha de comando, basta repetir a opção o número de vezes desejadas. Quando são configuradas através do arquivo de configuração, é necessário informar a lista de valores desejados, no formato de uma tabela Lua [3]. O código 1 apresenta um exemplo de arquivo do configuração.

Código 1: Exemplo de arquivo de configuração.

```
1 validator = {"openbus.test.core.services.TesterUserValidator"}
2 loglevel = 5
3 admin = {"admin1", "admin2"}
4 leasetime = 30
```

Existem duas formas de se definir o arquivo de configuração que o programa irá utilizar: utilizando a opção "config" por linha de comando, ou definindo um valor para a variável de ambiente *OPENBUS_CONFIG*. O *busservices* foi pensado para que as opções passadas por linha de comando tenham prioridade sobre as opções definidas no arquivo de configuração.

Através da opção "validator" informa-se o módulo do validador de senhas que será utilizado pelo barramento. Disponibilizamos em `openbus.core.services.passwordvalidator.LDAP` um validador LDAP, que precisa ser configurado com a lista de servidores utilizados na autenticação. Para maiores informações como configurar o validador LDAP, consulte [8].

O barramento possui algumas características importantes e devem ser mencionadas. Uma delas, é que o barramento procura persistir todo o seu estado no diretório de dados (definido através da opção "database"). Dessa forma, sempre que ele é iniciado, e o mesmo já possui uma base de dados populada, esses dados serão recarregados e farão parte do estado inicial do barramento.

Outra característica importante é que o barramento mantém compatibilidade com a API da versão imediatamente anterior. Ou seja, mesmo que a versão do barramento evolua, os clientes (serviços e aplicações)

ainda conseguem acessar o barramento utilizando bibliotecas de acesso de uma versão anterior. O OpenBus utiliza um esquema de versionamento com quatro números, no formato *A.B.C_D*, onde:

- Campos **A** e **B** significam a versão da IDL e deve ser igual em todos os pacotes que são compatíveis: versão IDL, barramento e bibliotecas de acesso.
- Campo **C** é incrementado quando ocorre alguma modificação na API interna.
- Campo **D** representa uma versão apenas com correções de falhas.

Sendo assim, o barramento 2.0 permite a realização de acesso utilizando as bibliotecas de acesso de versão 1.5.x, onde *x* pode assumir qualquer valor de *C_D*. Porém, é importante deixar claro que versão 2.0 traz muitas melhorias no quesito de segurança, e essas melhorias não serão usufruídas por clientes que acessam o barramento com versões antigas das bibliotecas de acesso. O barramento permite que clientes 1.5 se comuniquem com clientes 2.0, e vice-e-versa, porém essas comunicações se realizam com o mesmo nível de segurança que existia na versão 1.5 do barramento. Esse suporte de compatibilidade com a versão anterior pode ser desabilitado através da opção "nolegacy".

4.2 OpenBus para Gerentes do Barramento

Consideramos gerentes do barramento aqueles que irão desempenhar o papel de organizar e acompanhar o que é publicado no barramento. Para desempenhar esse papel, disponibilizamos a ferramenta *busadmin*, que permite que se realize atividades de governança (administração) sobre o barramento.

Antes de entrar em detalhes sobre o uso e as funcionalidades do *busadmin*, vamos agora apresentar alguns conceitos importantes de governança dentro do OpenBus. Existem os conceitos de:

Categoria Representa uma categoria de entidades no barramento. Categorias de entidade são agrupamentos usados exclusivamente para facilitar a gerência das diversas entidades cadastradas no barramento pelo administrador.

Entidade Representa uma entidade do barramento registrada. Entidade é tudo aquilo que pode fazer login no barramento e usufruir dos recursos do barramento. Em particular, tanto usuários humanos como implantações de sistema são considerados entidades. Entidades podem ou não ser cadastradas no serviço, mas apenas entidades cadastradas podem ser autorizadas a ofertar serviços.

Certificado Chave pública que pode ser usado para autenticar uma dada entidade no barramento. É possível adicionar certificados para entidades não cadastradas no barramento.

Interface Definição de uma interface IDL de um serviço que pode ser ofertado no barramento.

Autorização Associação de uma interface IDL a uma entidade, indicando que processos conectados como essa entidade possam oferecer serviços que implementem essa interface no Registro de Ofertas do barramento.

Para utilizar a ferramenta, deve-se executar:

```
busadmin [opções] --login=<usuário> <comando>
```

Para adicionar e remover categorias, entidades, certificados, interfaces e autorizações, aconselhamos que se defina um arquivo de script Lua com o formato especificado no código 2, porque ele serve de entrada para os comandos "script" e "undo-script" da ferramenta *busadmin*, que executa ou desfaz a execução do lote de comandos especificados pelo arquivo de script.

O *busadmin* também permite realizar esses cadastros e descadastros manualmente, além de realizar consultas e outras atividades de gerência do barramento. A listagem completa dos comandos disponíveis é apresentada a seguir:

- **Configuração:**

  --host=<endereço> Informa o endereço do Barramento.
  Valor padrão é 127.0.0.1.

  --port=<porta> Informa a porta do Barramento.
  Valor padrão é 2089.

  --verbose=<nível> Aciona o verbose da API OpenBus.
  Valor padrão é 0. Os níveis vão de 0 a 5, onde 5 é o nível com mais detalhes, e o 0 desativa o log.

  --oilverbose=<nível> Aciona o verbose do OiL.
  Valor padrão é 0. Os níveis vão de 0 a 5, onde 5 é o nível com mais detalhes, e o 0 desativa o log.

- **Controle de Categoria:**

  --add-category=<id> --name=<nome> Adiciona uma categoria com o identificador e nome descritivo especificado.

  --del-category=<id> Remove a categoria com o identificador especificado.

  --set-category=<id> --name=<nome> Altera o nome descritivo da categoria com o identificador especificado.

  --list-category Mostra as informações de todas as categorias cadastradas.

  --list-category=<id> Mostra informações da categoria especificada.

- **Controle de Entidade:**

  --add-entity=<id> --category=<id_categoria> --name=<nome> Adiciona uma entidade com o identificador, categoria e nome descritivo especificado.

  --del-entity=<id> Remove a entidade com o identificador especificado.

  --set-entity=<id> --name=<nome> Altera o nome descritivo da entidade com o identificador especificado.

  --list-entity Mostra as informações de todas as entidades cadastradas.

  --list-entity=<id> Mostra informações da entidade especificada.

  --list-entity --category=<id_categoria> Mostra informações das entidades pertencentes a categoria especificada.

- **Controle de Certificado:**

  --add-certificate=<id_entidade> --certificate=<certificado> Cadastra o certificado para a entidade especificada.

  --del-certificate=<id_entidade> Remove o certificado da entidade especificada.

- **Controle de Interface:**

  --add-interface=<interface> Adiciona a interface na lista de interfaces permitidas no barramento.

  --del-interface=<interface> Remove a interface da lista de interfaces permitidas no barramento.

  --list-interface=<interface> Mostra as interfaces permitidas no barramento.

- **Controle de Autorização:**

  --set-authorization=<id_entidade> --grant=<interface> Autoriza a entidade a publicar a interface especificada.

--set-authorization=<id_entidade> --revoke=<interface> Remove a autorização da entidade de publicar a interface especificada.

--list-authorization Mostra todas as autorizações concedidas no barramento.

--list-authorization=<id_entidade> Mostra todas as autorizações da entidade especificada.

--list-authorization --interface="<interface1> <interface1>...<interfaceN>" Mostra todas as autorizações contendo as interfaces especificadas.

- **Controle de Ofertas:**

--del-offer Lista todas as ofertas publicadas no barramento e aguarda a escolha de uma oferta para ser removida.

--del-offer --entity=<id> Lista todas as ofertas publicadas pela entidade especificada e aguarda a escolha de uma oferta para ser removida.

--list-offer Lista todas as ofertas publicadas no barramento.

--list-offer=<id> Lista todas as ofertas publicadas pela entidade especificada.

--list-props Lista todas as ofertas publicadas no barramento e aguarda a escolha de uma oferta para listar todas as suas propriedades.

--list-props=<id> Lista todas as ofertas publicadas pela entidade especificada e aguarda a escolha de uma oferta para listar todas as suas propriedades.

- **Controle de Logins:**

--del-login=<id> Remove o login especificado.

--list-login Mostra todos os logins ativos no barramento.

--list-login --entity=<id> Mostra todos os logins ativos da entidade especificada.

- **Script:**

--script Executa um script Lua com um lote de comandos.

--undo-script Desfaz a execução de um script Lua com um lote de comandos.

- **Relatório:**

--report Constrói um relatório sobre o estado atual do barramento.

4.3 OpenBus para Desenvolvedores

Como desenvolvedores, nos referimos aos responsáveis por implementar a integração de seus respectivos serviços e aplicações ao barramento. Como informando na seção 2.1, oferecemos uma biblioteca de acesso, que implementa o protocolo e exporta uma API de programação, que oferece alguns facilitadores e auxilia a interação com o barramento. Para maiores informações, consulte o manual de uso da biblioteca de acesso, que é disponibilizada em C++ [11], C# [12], Java [13] e Lua [14].

Código 2: Exemplo de script para a ferramenta *busadmin*.

```
-- Definição de uma categoria
-- * comando: Category
-- * parâmetros:
--    * id = identificador da categoria
--    * name = descrição da categoria
Category {
  id = "TEST_Category",
  name = "Descrição da categoria",
}
-- Definição de uma entidade
-- * comando: Entity
-- * parâmetros:
--    * id = identificador da entidade
--    * category = identificador da categoria que a entidade pertence
--    * name = descrição da entidade
Entity {
  id = "TEST_Entity",
  category = "TEST_Category",
  name = "Descrição da entidade",
}
-- Definição de um certificado
-- * comando: Certificate
-- * parâmetros:
--    * id = identificador da entidade
--    * certificate = caminho para arquivo de certificado associado a entidade
Certificate {
  id = "TEST_Entity",
  certificate = "teste.crt",
}
-- Definição de uma interface
-- * comando: Interface
-- * parâmetros:
--    * id = repID da interface
Interface {
  id = "IDL:script/Test:1.0"
}
-- Conceder autorização
-- * comando: Grant
-- * parâmetros:
--    * id = identificador da entidade a ser autorizada
--    * interfaces = lista de interfaces a serem autorizadas.
Grant {
  id = "TEST_Entity",
  interfaces = {
    "IDL:script/Test:1.0",
  }
}
-- Remover autorização
-- * comando: Revoke
-- * parâmetros:
--    * id = identificador da entidade
--    * interfaces = lista de interfaces a serem desautorizadas.
Revoke {
  id = "TEST_Entity",
  interfaces = {
    "IDL:script/Test:1.0",
  }
}
```

5 Glossário

A

API *Application Program Interface*. Interface de programação oferecida às aplicações que precisam acessar o barramento, seja para fazer ou receber chamadas através do barramento, ou acessar serviços oferecidos pelo barramento.

Autorização Associação de uma interface IDL a uma entidade, indicando que processos logados como essa entidade possam oferecer serviços que implementem essa interface no Registro de Ofertas do barramento.

B

Barramento Toda infraestrutura oferecida pelo OpenBus que permite fazer chamadas CORBA com a identificação das entidades com que os processos que originaram a chamada foram autenticados.

Biblioteca de Acesso Biblioteca de programação que implementa a API do OpenBus. O projeto OpenBus oferece implementações dessa biblioteca nas linguagens Lua, Java, C++ e C#.

busadmin Ferramenta que permite realizar operações de adminstração do barramento (governaça).

busservices Programa que implementa o barramento e os serviços núcleo do barramento OpenBus.

C

Categoria Representa uma categoria de entidades no barramento. Categorias de entidade são agrupamentos usados exclusivamente para facilitar a gerência das diversas entidades cadastradas no barramento pelo administrador.

Certificado Chave pública que pode ser usado para autenticar uma dada entidade no barramento.

Controle de Acesso Serve como ponto de entrada do barramento, sendo responsável por autenticar, renovar e gerenciar os logins de serviços e aplicações ao barramento.

CORBA *Common Object Request Broker Architecture*. Especificiação de um padrão de ORB sobre a qual o OpenBus é definido e implementado. O OpenBus pode ser visto como uma extensão do padrão CORBA.

E

Entidade Qualquer coisa que pode ser autorizada a acessar o barramento. Cada entidade possui um identificador único na forma de um nome. Entidades podem ser usuários humanos de sistemas ou mesmo instalações de serviços específicos de aplicações.

I

IDL *Interface Description Language*. Linguagem de descrição de tipos e interfaces de CORBA.

Interface Definição de uma interface IDL de um serviço que pode ser ofertado no barramento.

L

Lease Período de tempo pelo qual um login no barramento permanecerá válido sem necessidade de renovação. Contudo, um login pode ser invalidado explicitamente por um administrador antes do tempo de lease.

Login Representa uma autenticação de uma entidade junto ao barramento para poder acessá-lo.

Logout Processo no qual o criador de um login o invalida, fazendo com que esse login não possa mais ser usado para acessar o barramento.

O

Oferta de Serviço Cadastro no Registro de Ofertas de um componente SCS que implementa um conjunto de interfaces autorizadas que representam um dado serviço a ser utilizado através do barramento.

OpenBus Nome deste projeto que define um barramento baseado em CORBA para integração de aplicações coorporativas multilinguagem.

P

Propriedades da Oferta Conjunto de pares nome e valor que descrevem uma oferta de serviço.

Protocolo Conjunto de regras de comunicação para acesso ao barramento OpenBus.

R

Registro de Ofertas Serviço núcleo do barramento que permite registrar e buscar ofertas de serviços no barramento. Esse serviço é disponibilizado através da API.

S

SCS *Software Component System*. Modelo de componentes de software distribuído adotado pelo OpenBus. Todos serviços ofertados no barramento devem ser modelados como componentes SCS.

Serviços Núcleo Serviços oferecidos pelo próprio barramento, tipicamente acessados através da API.

Serviços Extras Serviços que acrescentam funcionalidades para auxiliar a integração entre serviços e aplicações, mas que não são parte do núcleo do barramento.

Referências

[1] C. Augusto, E. Fonseca, L. Marques, S. Correa, and R. Cerqueira. SCS: Software component system. `http://www.tecgraf.puc-rio.br/scs`, May 2006.

[2] T. Erl. *Service-oriented architecture: concepts, technology, and design.* Prentice Hall PTR, 2005.

[3] Roberto Ierusalimschy. The programming language lua. `http://www.lua.org/`, February 2008.

[4] Renato Maia. OiL: An object request broker in the Lua language. `http://www.tecgraf.puc-rio.br/~maia/oil/`, 2005.

[5] Renato Maia, Renato Cerqueira, and Ricardo Calheiros. OiL: An object request broker in the Lua language. In Mauro S. P. Fonseca, editor, *Proceedings of SBRC'06 - Salão de Ferramentas*, pages 1439–1446, Porto Alegre, Brazil, June 2006. SBC.

[6] Object Management Group. *The Common Object Request Broker Architecture (CORBA) Specification - Version 3.1*, January 2008. document: formal/2008-01-04.

[7] TecGraf. OpenBus - Enterprise Integration Application Middleware. `http://www.tecgraf.puc-rio.br/openbus`, 2006.

[8] TecGraf. *Manual de instalação do OpenBus 2.0.0.* TecGraf, 2012.

[9] TecGraf. *Manual de referência do Serviço de Colaboração 1.0.* TecGraf, 2012.

[10] TecGraf. *Protocolo de Acesso do OpenBus 2.0.* TecGraf, 2012.

[11] TecGraf. *Tutorial do SDK C++ 2.0.0.* TecGraf, 2012.

[12] TecGraf. *Tutorial do SDK C# 2.0.0.* TecGraf, 2012.

[13] TecGraf. *Tutorial do SDK Java 2.0.0.* TecGraf, 2012.

[14] TecGraf. *Tutorial do SDK Lua 2.0.0.* TecGraf, 2012.