ERLANGMS: Uma Plataforma em Erlang/OTP para Modernização de Sistemas Legados através de uma Abordagem Orientada a Serviços na UnB

Everton Agilar[1], **Alysson Ribeiro**[1], **Renato Ribeiro**[1]

[1]Universidade de Brasília – Campus Universitário Darcy Ribeiro
Caixa Postal – 70910-90 – Brasília – DF – Brasil

{evertonagilar,rcarauta,alyssondsr}@unb.br

Resumo. *Nos últimos anos, a modernização dos sistemas legados da Universidade de Brasília (UnB) tem sido prioridade para o CPD/UnB. A Arquitetura Orientada a Serviços (SOA) surge como uma maneira de solucionar este problema, disponibilizando uma abstração de alto nível entre as aplicações e a camada de negócio ou serviço. Este artigo aborda este tema e descreve alguns resultados obtidos com o uso da plataforma ERLANGMS desenvolvido pelo CPD/UnB sob uma abordagem orientada a serviços que compreende um processo de modernização e um barramento aderente ao estilo arquitetural Representational State Transfer (REST). Mais especificamente, são discutidos os principais resultados alcançados, as quais destacam-se um barramento de serviços orientado a contrato de serviços; um kit de desenvolvimento (SDK) para criação de serviços; um processo denominado SMSOC para guiar as atividades de migração de sistemas legados; e alguns serviços de apoio como um proxy LDAP para unificar o login de usuários e um serviço de autenticação/autorização OAuth2 para os serviços em REST.*

1. Introdução

Os sistemas legados correspondem às aplicações que sustentam o funcionamento negocial de uma Instituição e consolidam a maior parte das informações corporativas [Bennett 1995, Bisbal et al. 1999]. Na Universidade de Brasília (UnB), há uma gama considerável de sistemas legados desenvolvido ao longo dos últimos 30 anos pelo CPD/UnB que consistem em um arcabouço de regras de negócios que são de vital importância para o pleno funcionamento da Instituição. Entretanto, com as sucessivas revisões nas regras de negócios para mantê-los alinhados com as necessidades e a obsolescência tecnológica desses sistemas, tornaram-se rígidos e inflexíveis, a ponto de serem de difícil manutenção e evolução.

De forma geral, os sistemas da UnB dividem-se em três áreas de negócio: área acadêmica, administrativa e de pessoal. A maioria desses sistemas foram construídos em diferentes linguagens de programação, arquiteturas e plataformas que não conversam entre si, a não ser, por meio do banco de dados. Durante muitos anos, a linguagem de programação VB foi a predominante. Os dois sistemas mais importantes escritos em VB são o *Sistema Acadêmico (SIGRA)* e o *Sistema de Pessoal (SIPES)*, sendo os demais sistemas escritos em VB.Net, C#, PHP, ASP e Java (a plataforma atual).

Neste cenário, as tradicionais práticas de manutenção deixam de atender às organizações, que buscam formas de reduzir os custos com a manutenção, maximizar a

integração entre os sistemas, torná-los mais flexíveis às mudanças de forma para prolongar sua vida útil e facilitar a evolução desses sistemas [Bennett 1995, Bisbal et al. 1999, Comella-Dorda et al. 2000].

Ao conduzir a modernização dos sistemas legados na UnB, optou-se por experimentar com a arquitetura orientada a serviços, particularmente seguindo o estilo arquitetural *REST*, já adotado em muitas Instituições devido ao aproveitamento da infraestrutura web existente e a facilidade para invocar serviços REST a apartir de qualquer sistema [Fielding 2000, Kalin 2013]. Assim, foi criada a plataforma *ERLANGMS*, que compreende um barramento de serviços desenvolvido na linguagem *Erlang/OTP* para possibilitar a publicação de serviços; de um processo de modernização denominado *Software Modernization through Service Oriented Computing (SMSOC)*, para guiar os trabalhos de modernização e disponibilizar uma arquitetura de software padronizada para criação dos serviços.

Salienta-se que embora *SOA* seja um tema de crescente interesse por parte dos pesquisadores e da indústria, identificou-se a necessidade prévia de condução de um mapeamento sistemático para caracterizar a modernização de sistemas legados no contexto da manutenção de software [Agilar et al.]. Desse modo, muitas decisões de design da plataforma proposta foram definidas a partir deste estudo prévio onde verificou-se que a maior parte das contribuições na literatura estão relacionados aos aspectos gerenciais da modernização de software (55,88% das publicações) e há poucos relatos de contribuições que descrevem (ou validem) técnicas ou ferramentas de modernização de software.

Este artigo apresenta a plataforma ERLANGMS e discute as experiências obtidas na implantação do barramento de serviços no CPD/UnB. Mais especificamente, são descritos os principais resultados alcançados, as quais destacam-se:

- Desenvolvimento de vários serviços web na linguagem Java para atender as demandas da UnB com o uso do processo de modernização SMSOC e o design de arquitetura subjacente, denominado de kit de desenvolvimento (SDK).
- Desenvolvimento de um serviço proxy LDAP ou Lightweight Directory Access Protocol para unificar o acesso dos usuários ao oferecer um ”logon único” para os sistemas da UnB, como o Redmine, Webmail ou SEI, que se utilizam do protocolo LDAP para autenticação de usuários;
- Desenvolvimento de um serviço de autenticação OAuth2, integrado ao Sistema de Controle de Acesso (SCA) da UnB, que permite que os serviços possam ser consumidos pelos clientes utilizando a mesma infraestrutura de segurança já existente.

ATÉ AQUI

2. Metodos

Atualmente, os sistemas da Universidade de Brasília (UNB) são protegidos por meio de um sistema de controle de acesso (SCA). Esse sistema proporciona a criação de permissões atráves de usuários que tem perfis e estes tem transações. Um dos problemas enfrentados com essa abordagem é o fato de que as aplicações da UNB estão vinculadas a pessoa, ou seja, uma pessoa pode ter diversos usuários diferentes. Outro problema enfrentado é o fato desse sistema, por ter sido implementado de forma proprietária, não

está em conformidade com os padrões do mercado no que se refere a autenticação e autorização, o que impossibilita que sistemas externos a universidade possam compartilhar dessas informações.

A implementação de um framework padrão de mercado se torna necessária para a resolução dos diversos problemas mencionados. Com isso foi escolhido o oauth 2 por ser um framework autamente utilizado pelas maiores empresas de informatica e por ser hoje um padrão de mercado.

O oauth 2 é um framework que veio com a ideia de se padronizar a maneira como as aplicões cliente acessam conteúdo restrito. O oauth 2 se baseia em tokens de acesso. Ao se autenticar, uma aplicação cliente recebe do servidor de autorização um token de acesso. A aplicação cliente pode acessar os recursos restritos, basta apenas enviar o token recebido pelo servidor e este valida ou não o acesso aos dados requisitados.

Para a implantação do oauth 2 é necessário seguir uma série de passos preconizados pela rfc. Uma importante caracteristica, que faz com que o oauth 2 seja amplamente utilizado é sua obrigatoriedade de uso junto com o HTTS.

Diversos serviços já estão disponibilizados como: a declaração de aluno, o catálogo de aluno, os cursos dos alunos, dentre outros. Todos os serviços ultilizam o barramento como comunicação entre as aplicações cliente e os dados armazenados no servidor.

Os serviços seguem um padrão, chamado de catálogos, que fornecem a url de acesso, qual a linguagem do serviço, onde está o código do serviço, seu método HTTP de acesso e a forma de autorização.

Segue abaixo um exemplo de um dos serviços já disponibilizados.

3. Resultados

3.1. Solução de Autorização

3.1.1. Requisitos

Os requisitos da solução de autorização propostos neste trabalho, são:

Req1 Autorização. O protocolo deverá conceder autorizações apenas aos recursos que o cliente autenticado tenha direito. As autorizações serão válidas por um período de tempo finito.

Req2 Administrador. O administrador do recurso deve ser capaz de revogar autorizações concedidas a qualquer momento.

Req3 Integridade. Deve ser possível detectar alterações ilícitas nas mensagens durante o processo de autenticação e autorização.

Req4 Privacidade. Dados restritos trocados durante o processo de autorização devem ser mantidos em sigilo de pessoas e serviços não autorizados;

Req5 Compatibilidade com o barramento ErlangMS. O protocolo deve usar a arquitetura REST e o formato JSON para troca de mensagens.

As soluções encontradas no Mapeamento Sistemático apresentado em [de Sousa Ribeiro and Canedo 2016] e os requisitos definidos para a solução de

segurança que será utilizada no barramento ErlangMS serviram de apoio para a escolha do protocolo de autorização.

Desta forma, foi implementado no protocolo de autorização Oauth 2 como um serviço do ErlangMS. A implementação do OAuth foi facilitada pelo uso das bibliotecas disponibilizadas no barramento e na linguagem Erlang para tratar requisições HTTP/REST. O ErlangMS já possui suporte ao uso de TLS, desta forma, o OAuth 2 foi implementado respeitando os requisitos de integridade e privacidade.

3.1.2. OAuth 2

O OAuth (*Open Authorization Protocol*) é um protocolo de autorização que permite que um aplicativo de terceiros possa obter acesso limitado a um serviço através de trocas de mensagens e *tokens* de acesso [Torroglosa-García et al. 2013]. O protocolo utiliza a arquitetura REST e mensagens JSON [Noureddine and Bashroush 2013].

A versão do Oauth 2.0 traz um protocolo completamente novo em relação a versão 1.0, publicado em dezembro de 2007, não sendo compatível com o mesmo [Hammer-Lahav 2010, IETF 2010]. O OAuth 2.0 define quatro entidades:

- *Resource Owner*: é o usuário que autoriza uma aplicação a acessar seus dados;
- *Client*: aplicação que deseja acessar os dados do usuário;
- *Resource Server*: Servidor que hospeda os recursos protegidos, recebe e responde às solicitações de acesso aos recursos com os *tokens* do protocolo ;
- *Authorization Server*: Servidor que concede o *token* de acesso após verificar a identidade do usuário e a concessão da autorização ao cliente.

A Figura 1 apresenta o fluxo abstrato do Oauth 2.0.

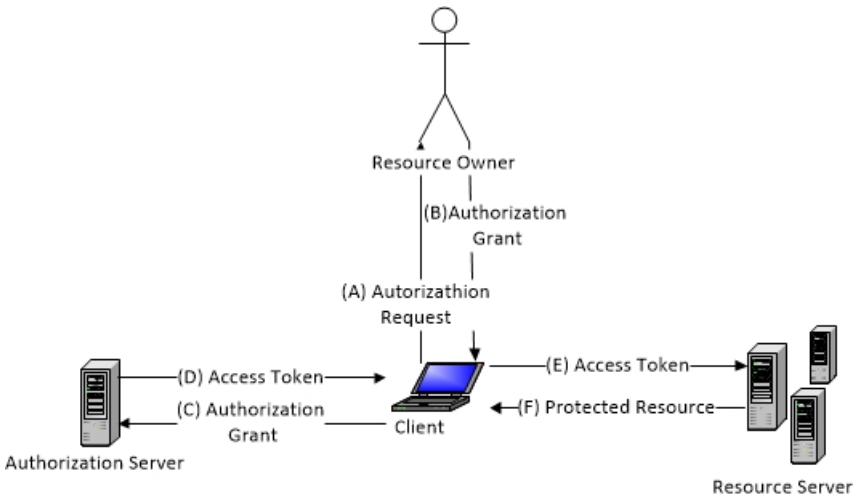

Figure 1. Abstract Flux of Oauth 2.0 [Hardt 2012]

As etapas do diagrama incluem:

(A) O cliente solicita uma autorização para acessar os recursos do usuário;
(B) O usuário autoriza o pedido e o cliente recebe a concessão da autorização;

(C) O cliente requisita o *token* de acesso apresentando a concessão da autorização e suas credenciais ao servidor de autorização;
(D) O servidor de autorização autentica o cliente, valida a concessão da autorização e emite o *token* de acesso;
(E) O cliente requisita o recurso protegido e apresenta o *token* de acesso;
(F) O servidor do recurso valida o *token* de acesso e concede o acesso aos recursos solicitados.

3.2. SCA

Para a integração do (SCA) com o protocolo oauth 2, o trabalho realizado teve como objetivo a criação de uma autenticação única e integrada para cada pessoa. Atualmente os perfis de acesso estão vinculados a um usuário, e vários usuários estão ligados a uma única pessoa. Para se ter um controle de acesso único e integrado é necessário associar todos os usuários a uma pessoa. Os seguintes passos foram feitos para padronização desse cenário.

Foi criado uma consulta que retornasse todos os perfis e usuários da pessoa. O resultado da consulta foi armazenado em um banco de dados interno do barramento para proporcionar maior velocidade na autenticação. Foi desenvolvida um sistema cliente de autenticação para o acesso único e integrado aos sistemas da UNB.

Os passos para a autenticação entre a aplicação cliente e o servidor de autorização seguem o seguinte fluxo. O usuário se autentica em uma aplciação cliente, que envia os dados de login e senha de acesso, por meio do SSL, ao receber o login e a senha o servidor autentica o usuário caso o login seja válido e retorna o token de acesso criptografado, que é armazenado pela aplicação cliente. Esse modelo permite que o usuário se autentique apenas uma vez e tenha acesso a todos os serviços que os perfis de acesso permitem.

References

Agilar, E., de Almeida, R. B., and Canedo, E. D. A systematic mapping study on legacy system modernization.

Bennett, K. (1995). Legacy systems: coping with success. *Software, IEEE*, 12(1):19–23.

Bisbal, J., Lawless, D., Wu, B., and Grimson, J. (1999). Legacy information systems: Issues and directions. *IEEE software*, 1(5):103–111.

Comella-Dorda, S., Wallnau, K., Seacord, R., and Robert, J. (2000). A survey of legacy system modernization approaches. Technical Report CMU/SEI-2000-TN-003, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

de Sousa Ribeiro, A. and Canedo, E. D. (2016). Solutions analysis of authentication and authorization for service oriented architectures. In *2016 11th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6.

Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, Irvine.

Hammer-Lahav, E. (2010). Introducing oauth 2.0. *Hueniverse, May*.

Hardt, D. (2012). The oauth 2.0 authorization framework.

IETF (2010). The oauth 1.0 protocol. *RFC*.

Kalin, M. (2013). *Java web services: up and running*. " O'Reilly Media, Inc.".

Noureddine, M. and Bashroush, R. (2013). An authentication model towards cloud federation in the enterprise. *Journal of Systems and Software*, 86(9):2269–2275.

Torroglosa-García, E., Pérez-Morales, A. D., Martinez-Julia, P., and Lopez, D. R. (2013). Integration of the oauth and web service family security standards. *Computer networks*, 57(10):2233–2249.