Modernização de Sistemas Legados para Disponibilização em Dispositivos Móveis com Arquitetura Baseada em *microservices*

Bruno C. Freitas[1], Roberto S. M. Barros[2], Daniel Patriota[1]

[1]Núcleo de Tecnologia da Informação - Universidade Federal de Pernambuco (UFPE)
Caixa Postal XXXXXX -- CEP -- Recife -- PE -- Brasil

[2]Departamento de Sistemas da Computação- Universidade Federal de Pernambuco (UFPE)
Caixa Postal XXXXXX -- CEP -- Recife -- PE -- Brasil

{bruno.cfreitas,daniel.patriota,roberto.smaior}@ufpe.br

Abstract. *The universal use of mobile computing devices, especially smartphones, encourages the organizations using information systems to adapt them to provide an adequate access through this computational tool. Legacy systems, however, can make this adaptation difficult, demanding the modernization of its architecture. In this context, the microservices architecture has been emerging. This work proposes a modernization process of legacy systems to a microservice-based architecture, using an intermediate step with REST wrapping technique in order to prioritize the availability of mobile access to these systems.*

Resumo. *O uso universal de dispositivos móveis computacionais, especialmente dos smartphones, impulsiona as organizações possuidoras de sistemas de informação a adaptá-los para um adequado acesso através deste veículo computacional. Os sistemas legados, no entanto, podem dificultar esta adaptação, demandando uma modernização de sua arquitetura. Neste contexto, a arquitetura de microservices tem se destacado. Este trabalho propõe um processo de modernização de sistemas legados para uma arquitetura baseada em microservices, usando uma etapa intermediária com a técnica de REST wrapping para priorizar a disponibilização em dispositivos móveis.*

1. Introdução

A popularidade atual dos dispositivos móveis, principalmente dos *smartphones*, é um fato incontestável. Apesar das limitações dos dispositivos móveis, a sua conveniência se sobressai, contribuindo para um uso cada vez mais universal destes dispositivos. Isto impulsiona a indústria e as organizações a criarem soluções de tecnologia de informação específicas para este novo mercado, chamadas soluções *mobile,* que podem ser classificadas em aplicações nativas, *mobile-sites* e aplicações híbridas [Wisniewski 2011]. Contudo, existem muitos sistemas que foram projetados em época passada, conhecidos por sistemas legados, que resistem significativamente a mudanças e evoluções para atender novos requisitos de negócio [Brodie e Stonebreaker 1995].

No contexto de modernização de sistemas legados, a migração para uma arquitetura de *microservices* tem se destacado [Fowler e Lewis 2014]. Nesta arquitetura, que é vista

como uma evolução de SOA - *Service Oriented Architecture* [Sharma 2016], o sistema é composto por diversos serviços independentes entre si, inclusive em nível de infraestrutura operacional, responsáveis por tarefas únicas, comunicando-se por mensagens para atingir objetivos de negócio. O processo de migração não é rápido, e pode ser mais difícil ainda se o sistema legado não estiver bem organizado de acordo com boas práticas de programação e princípios de engenharia de software, principalmente a alta coesão e o baixo acoplamento.

2. Método

Este trabalho propõe um processo genérico de alto nível para modernizar sistemas legados de forma incremental com o objetivo de integrá-los com dispositivos computacionais móveis por meio de uma arquitetura baseada em *microservices*. Contudo, em virtude do tempo necessário para migrar partes do sistema para a nova arquitetura, que demanda entendimento de código e regras de negócio implementadas, o processo faz uso de uma etapa intermediária com a técnica de REST *wrapping*, de maneira a priorizar a disponibilização do sistema em dispositivos móveis.

O referido processo foi aplicado em um estudo de caso, que consiste no sistema de gestão acadêmica SIGA da UFPE (Universidade Federal de Pernambuco), feito na plataforma J2EE 2 com mais de 15 anos de uso, para criação de uma aplicação de acesso a informações de discentes de pós-graduação, especificamente notícias cadastradas no sistema e conceitos obtidos pelo discente em disciplinas. As etapas do processo proposto estão dispostas na Figura 1, sendo detalhadas em sequência.

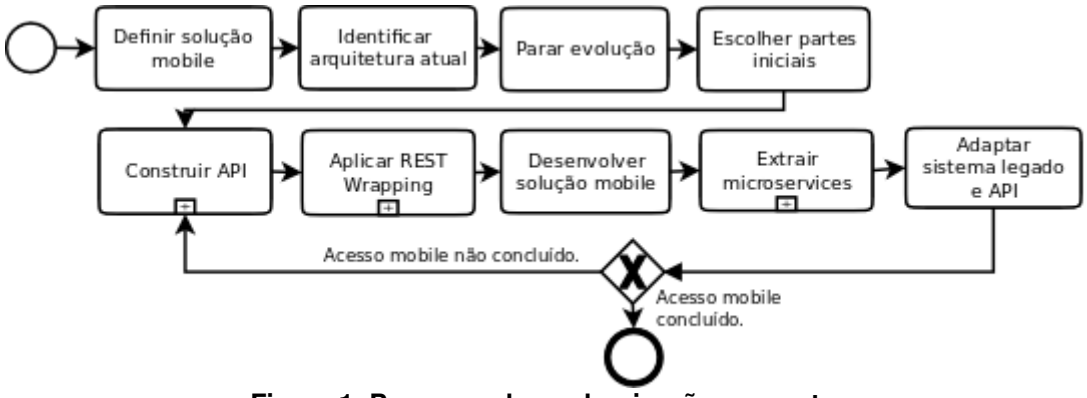

Figura 1: Processo de modernização proposto.

A definição sobre o tipo de solução *mobile* que será utilizada deve levar em consideração principalmente a diversidade de tipos de dispositivos que o público alvo do sistema possui e a intenção de alcance da organização. Pode-se realizar uma análise interna ou aplicar pesquisa com o público alvo do sistema para identificar qual tipo de solução é a mais adequada: aplicação nativa, *mobile-sites* ou híbridos. Mais de um tipo de solução pode ser utilizada de forma gradativa. No estudo de caso, optou-se por utilizar aplicação nativa para o sistema Android.

O entendimento da arquitetura existente é essencial para planejar e executar a modernização do sistema. Pode-se utilizar, como referências, documentações específicas sobre a arquitetura, caso existam, ou realizar uma análise do código legado. A arquitetura do SIGA dispõe de camadas de apresentação, fachada, cadastro e repositório. Parar a evolução do sistema existente nos moldes antigos também é crucial, uma vez

que, do contrário, aumentará o trabalho posterior de modernização.

A escolha das partes iniciais do sistema para começar a modernização deve priorizar: (1) objetivos de migração baseados na demanda de utilização por dispositivos móveis; (2) agrupamentos de funcionalidades afins e (3) restrições ocasionais em virtude de natureza transacional, que existem quando operações exigem um nível imediato de consistência de dados.

Em seguida o processo prevê a criação de uma *Web API REST* (*Representational State Transfer*) para o sistema, que receberá as requisições de acesso aos recursos, atuando de acordo com o padrão API Gateway [Richardson 2015]. As etapas para construção da API consistem em identificar recursos disponibilizados, definir as representações destes recursos, definir as URIs (*Uniform Resourse Identifiers*), identificar as ações que serão disponibilizadas sobre os recursos, especificar a API, prover segurança por meio *framework* OAuth 2.0 e por fim implementá-la em uma linguagem de programação.

A extração de partes do sistema em *microservices* pode ser muito demorada. Em virtude disto, e da emergente demanda por acesso *mobile*, é possível utilizar a técnica de REST *Wrapping* como etapa intermediária, possibilitando que a API consuma recursos diretamente do sistema legado, permitindo o imediato desenvolvimento de aplicações voltadas para os dispositivos móveis enquanto os *microservices* são construídos. A ideia geral da técnica é permitir que o sistema legado seja envolto por uma tecnologia que habilita acesso a suas funcionalidades ou dados por meio de uma API REST [Kangasaho 2016]. Neste ponto, o sistema legado provê serviços intermediados por uma API.

Com a demanda de acesso *mobile* satisfeita, os esforços para migração de partes do sistema para a arquitetura de *microservices* podem ser iniciados. É preciso conhecer detalhes da regras de negócio, para que partes do sistema legado dêem lugar a *microservices* independentes. A Figura 2 ilustra este subprocesso.

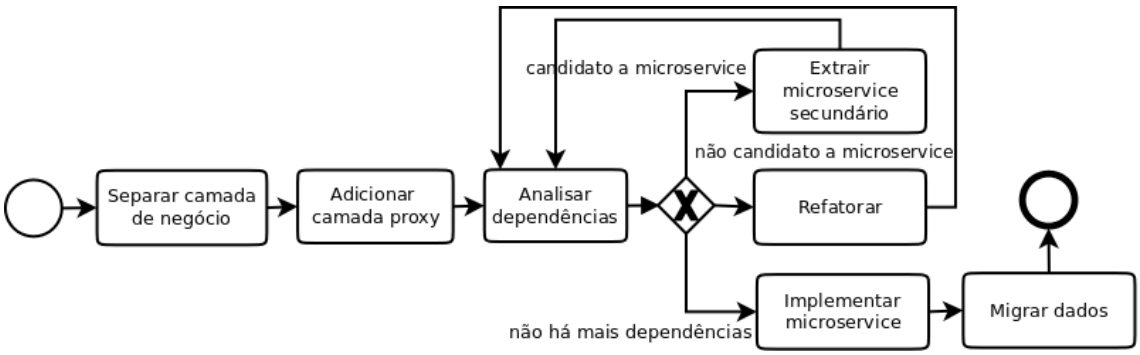

Figura 2: Subprocesso de extração dos microservices

As partes do sistema que serão extraídas não contemplarão a interface com o usuário, e sim as regras de negócio e persistência de dados. Por isso, é preciso estabelecer, caso já não exista no sistema legado, uma separação clara entre o código das camadas de apresentação e de regras de negócio. Para padronizar o local de alteração para invocar os *microservices* dentro do sistema, pode-se utilizar uma camada *proxy* que receberá as solicitações da camada de apresentação e poderá requisitar à camada de negócio da aplicação ou encaminhar para um *microservice*. Esta camada funcionará como uma fachada para os *microservices* dentro do sistema legado.

É possível também que existam dependências dentro do sistema tanto a nível de código

como a nível de banco de dados: outras partes do sistema que fazem acesso diretamente aos seus dados ou funções, além de restrições ou referências internas no próprio banco de dados, como chaves estrangeiras. Estas dependências precisam ser desfeitas. Cada dependência deve ser analisada se pode ser uma candidata a *microservice.* O resultado da análise poderá culminar em: (1) extração de *microservice* secundário - caso seja considerado que sua extração imediata trará valor de negócio no contexto da extração principal, aplicando o processo de forma recursiva; (2) refatoramento - não sendo considerada candidata imediata à extração, a dependência deverá ser eliminada por meio de refatoramento: as chamadas aos dados ou funções da parte que será extraída devem ser modificadas para utilizar o *proxy* da camada de negócio referente à parte extraída.

Esgotado o ciclo de análise de dependências, pode-se dar a implementação dos *microservices*. Toda a liberdade proporcionada por esta arquitetura pode ser posta em prática: a linguagem de programação mais adequada, o banco de dados preferido ou mais indicado para o contexto, dentre outros aspectos. Não há definitivamente nenhuma amarra à tecnologia do sistema legado. À medida que os *microservices* forem implementados, os seus respectivos dados também precisam ser migrados.

3. Resultados

Ao concluir a etapa intermediária que utiliza a técnica REST *Wrapping*, foi implementado aplicativo para Android intitulado "discentePos", que para obter autorização de acesso redireciona o usuário para o navegador padrão do dispositivo móvel e carrega a página de autenticação do servidor de autorização OAuth 2.0. A Figura 3 mostra o aplicativo em funcionamento.

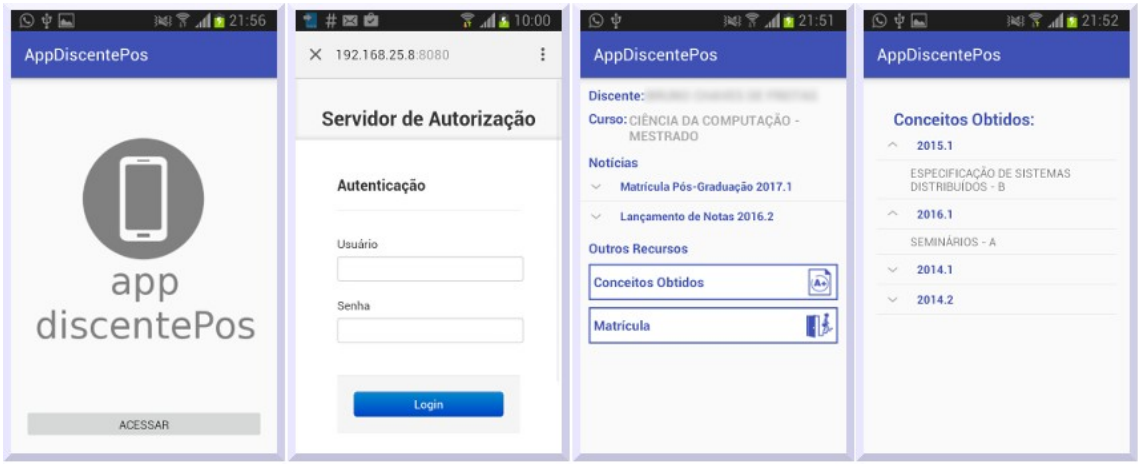

Figura 3: Aplicativo discentePos

No contexto da extração dos *microservices*, a parte de cadastro de notícias não apresentou nenhuma dependência, o que facilitou a extração. Contudo, o cadastro de conceitos obtidos apresentou um total de 171 dependências distribuídas em diversas partes do sistema. Uma destas dependências era a parte do sistema que tratava de dispensas de disciplinas, para a qual o processo foi aplicado recursivamente e 7 dependências foram encontradas e refatoradas. Basicamente os refatoramentos envolveram eliminar referências diretas à tabela de dados e substituir por chamadas à camada de cadastro adequada. Foram implementados os *microservices* de notícias e o de dispensa de disciplinas na linguagem JavaScript sob a plataforma NodeJS, com banco

de dados MySQL. A Figura 4 ilustra a arquitetura geral do sistema após a aplicação do processo.

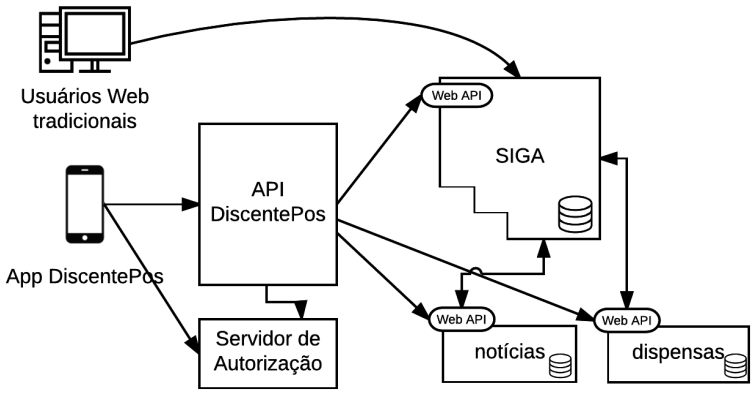

Figura 4: Visão geral da arquitetura após aplicação do processo.

4. Conclusão

Prover o adequado acesso a sistemas de informação por meio de dispositivos móveis é mandatório: uma demanda criada pelos próprios usuários. Os sistemas legados podem não estar aptos a esta tendência em virtude de sua tecnologia defasada. O processo de habilitar estes sistemas ao acesso via dispositivos móveis pode ser uma oportunidade de modernização. Neste contexto, a arquitetura de *microservices* é uma tendência para reestruturar sistemas legados, quebrando amarras tecnológicas e abrindo portas para novas demandas de evolução atuais e futuras.

O processo de modernização proposto abordou aspectos essenciais para habilitar o acesso mobile e quebrar o sistema legado em *microservices*, contudo, não contemplou a possibilidade de interrupção da extração em virtude das possíveis dificuldades encontradas relacionadas a dependências existentes. Como trabalho futuro, o processo poderia abordar aspectos no sentido de criar um mapeamento das dependências e estabelecer critérios que auxiliem na decisão por migrar ou não uma determinada parte do sistema para a nova arquitetura.

5. Referências

Brodie, M. L.; Stonebraker, M. Legacy Information Systems Migration: Gateways, Interfaces, and the Incremental Approach. San Francisco, CA, USA.

Fowler, M.; Lewis, J. Microservices. 2014. Disponível em: <http://martinfowler.com/articles/microservices.html>.

Kangasaho, M. Legacy application modernization with REST wrapping. Dissertação (Mestrado) — University of Helsinki, 2016.

Richardson, L.; Ruby, S.; Amundsen, M. RESTful Web APIs. O'Reilly Media, Inc., 2013. ISBN 9781449358068.

Sharma, S. Mastering microservices with java. Packt Publishing Limited, 2016. ISBN 9781785285172.

Wisniewski, J. Mobile that works for your library. Online. Academic OneFile.