

PATOS DE MINAS. Prefeitura Municipal. A cidade: negócios. Patos de Minas, Prefeitura de Patos de Minas. Apresenta dados sobre a economia do município. Disponível em <<http://www.patosdeminas.mg.gov.br/acidade/negocios.php>>. Acesso em: 4 mar. 2017.

REIS, Eduardo Sá dos. Implementando uma solução Business Intelligence com o Microsoft SQL Server 2005 – Parte 1. Edição 59, 2008.

SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHAN, S. Sistema de banco de dados. São Paulo: Pearson Makron Books, 1999.

TURBAN, Efraim. et al. Business intelligence: um enfoque gerencial para a inteligência do negócio. Trad. Fabiano Bruno Gonçalves. Porto Alegre: Bookman, 2009.

UTILIZAÇÃO DE MICROSSERVIÇOS E CLOUD COMPUTING NA PROTEÇÃO DOMICILIAR



Kassius Antonio Ferraz
Graduando em Sistemas de Informação pelo Centro
Universitário de Patos de Minas – UNIPAM.
kassius.f@hotmail.com



José Corrêa Viana
Especialista em Gestão de Tecnologia da Informação
pelo IGTI e Professor de Sistemas de Informação pelo
Centro Universitário de Patos de Minas – UNIPAM
jcorrea@unipam.edu.br



RESUMO

Este artigo demonstra, por meio de um exemplo prático, a junção de computação na nuvem, arquitetura em microsserviços e internet das coisas para criar uma plataforma de proteção domiciliar. Descreve-se como utilizar o Scrum em projetos de um único desenvolvedor e quais ferramentas é possível utilizar do framework. Ao longo do artigo, são abordados os sucessos e os fracassos obtidos durante o desenvolvimento da plataforma, bem como cada componente da arquitetura que foi desenvolvida e como foi utilizada a nuvem para armazenar os dados e executar cada componente.

PALAVRAS-CHAVE: *Microsserviços; Computação na Nuvem; Segurança Domiciliar.*

ABSTRACT

This paper demonstrates, through practical examples, the union of cloud computing, architecture in micro services and internet, so as to create a platform of home security. We described how to use the Scrum in projects of a single developer and which tools it is possible to use from framework. Along the article, we approach the successes and failures obtained during the development of the platform, as well as each component of the architecture that was developed and how the cloud was used to store the data and execute each component.

KEYWORDS: *micro services; cloud computing; home security*

UTILIZAÇÃO DE MICROSSERVIÇOS E CLOUD COMPUTING NA PROTEÇÃO DOMICILIAR

1 INTRODUÇÃO

A cada dia aparece um novo serviço ou produto na Internet. Netflix, Dropbox e Facebook são exemplos claros destes novos serviços que vêm sendo cada vez mais utilizados. Esses serviços precisam escalar à medida que seus números de usuários aumentam, mas, para isto, eles necessitam de recursos disponíveis praticamente no momento da execução.

Esse crescimento só é possível devido a essas empresas disponibilizarem seus serviços na Nuvem. Os termos "nuvem", "computação na nuvem" ou cloud computing são utilizados por profissionais na área para se referir a uma infraestrutura pela qual os recursos se tornam de fácil alcance, ficando disponíveis por vários data centers espalhados pelo mundo (TAURION, 2009).

Para melhor desempenho e escalabilidade na nuvem, está se tornando cada vez mais comum o uso da arquitetura de micro serviços, que são pequenas aplicações que na maioria das vezes disponibilizam interfaces capazes de realizar tarefas. A aplicação por completo seria o conjunto desses pequenos serviços. Como cada micro serviço consegue funcionar de forma independente, quando necessário, pode-se utilizar ferramentas para escalar e disponibilizar os serviços utilizando somente o que for necessário (TAURION, 2009).

Para exemplificar os benefícios que se tem ao desenvolver uma aplicação baseada em micro serviços rodando na nuvem, foi desenvolvido uma plataforma de segurança. Essa plataforma utilizará sensores do Microsoft Kinect para captar movimentos de pessoas que adentrem em um ambiente protegido. Os movimentos captados serão enviados como eventos para a plataforma na nuvem. Esses eventos serão processados, e se necessário, serão reportados para um aplicativo móvel.

O desenvolvimento de aplicações em uma arquitetura de micro serviços consegue agregar vários benefícios os quais estão ligados à melhoria na manutenção, à evolução e aplicação de novas tecnologias, à resiliência a erro, à facilidade na entrega, entre outros.

Quando se reúne o microsserviço à nuvem, é possível disponibilizar facilmente a aplicação,

independentemente da quantidade de usuários. À medida que um serviço recebe uma grande carga, é possível liberar mais recursos para ele, ou então subir uma segunda instância da aplicação, mantendo a alta disponibilidade.

Os ganhos de todos os benefícios da arquitetura em conjunto com os serviços da nuvem foram apresentados com o desenvolvimento de uma plataforma de segurança domiciliar.

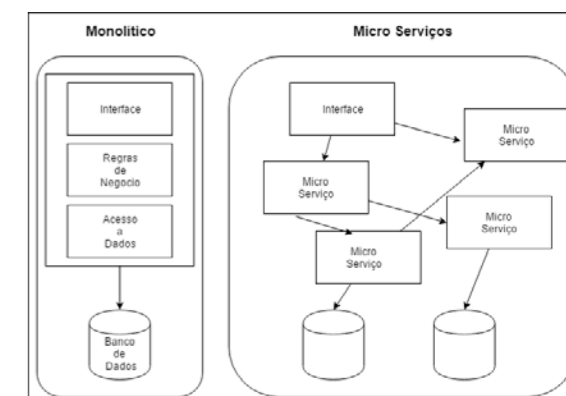
2 REFERENCIAL TEÓRICO

As conceituações dos temas abordados neste projeto serão apresentadas nesta seção.

2.1 ARQUITETURA BASEADA EM MICRO SERVIÇOS
Segundo Bonér (2016), a arquitetura baseada em microsserviços possui um conceito simples: defende a criação de um sistema a partir de uma coleção de serviços pequenos e isolados, cada um dos quais possui seus dados, escalável e resiliente ao fracasso. Os serviços se integram com outros serviços para formar um sistema coeso, que é muito mais flexível do que os sistemas empresariais típicos que são construídos atualmente.

Desta forma, pode-se afirmar, a partir da arquitetura monolítica, que cada microsserviço representaria um módulo interno do monolítico. Como apresentado na Figura 1, enquanto no monolítico todas as funcionalidades ficam integradas num mesmo projeto, na arquitetura de microsserviços, seriam disponibilizadas pequenas aplicações com suas funcionalidades através de requisições HTTP (Protocolo de Transferência de Hipertexto) ou de AMQP (Protocolo Avançado de Enfileiramento de Mensagens).

Figura 1. Diferenças entre arquitetura monolítica e micro serviços



Ao aplicar uma arquitetura baseada em microsserviços, é possível obter vários fatores positivos:

- **Escalabilidade:** quando necessário, é possível aumentar ou diminuir os recursos gastos para apenas um recurso dependendo de sua necessidade.
- **Resiliência:** caso ocorra um erro no serviço e o processo consiga continuar mesmo com a falha, o sistema continuará operante. Já no caso do monolítico, caso ocorra um erro, o sistema finalizará a execução do processo.
- **Heterogeneidade de Tecnologia:** cada serviço pode ser escrito e desenvolvido diferente dos demais, da mesma forma que a base de dados pode ser persistida em qualquer sistema de armazenamento. Por exemplo, é possível ter dois serviços, sendo um escrito em Java, utilizando o banco relacional MySQL, trocando mensagens com outro serviço, escrito em Node.JS, com banco documental MongoDB.
- **Fácil Entrega:** não é necessário parar todo o sistema para fazer a troca de versão de um serviço. Em alguns casos, o sistema continua operante mesmo se alguns serviços forem parados.
- **Substituição e Melhorias:** pelo fato de microsserviços terem um código relativamente pequeno, se necessário, reescrevê-lo ou adotar novas tecnologias terá um custo extremamente baixo.

Desta forma, pode-se afirmar, a partir da arquitetura monolítica, que cada microsserviço representaria um módulo interno do monolítico.

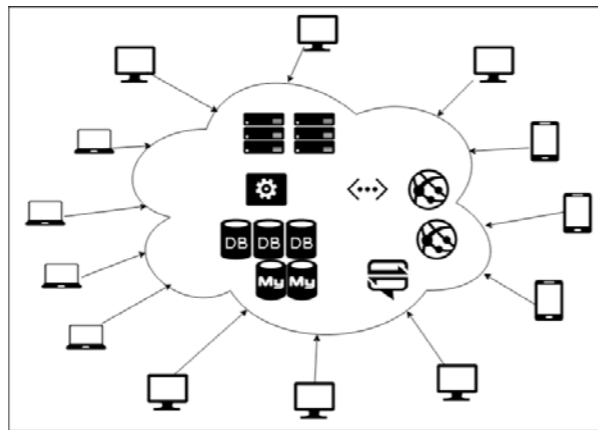
2.2 COMPUTAÇÃO EM NUVEM

A necessidade de provisionar recursos computacionais de forma fácil e rápida foi tema de uma palestra que ocorreu em 2006, feita pelo CEO do Google, Eric Schimdt, assim fazendo nascer a computação na nuvem.

Podemos dizer que a *Computação em nuvem* é um termo para descrever um ambiente de computação baseado em uma imensa rede de servidores, sejam estes virtuais ou físicos. Uma definição simples pode então ser um conjunto de recursos como capacidade de processamento, armazenamento, conectividade, plataformas, aplicações e serviços disponibilizados na Internet (TAURION, 2009, p. 2).

É possível dividir a computação na nuvem em dois grandes tipos: a nuvem privada e a nuvem pública. A nuvem privada tem como característica o seu funcionamento interno na empresa, onde os custos da manutenção e da expansão dos recursos são de total responsabilidade da empresa, sendo necessário uma equipe de infraestrutura para tal tarefa. Já em nuvens públicas não é necessário manter uma equipe, pois a manutenção dos serviços disponibilizados é por total conta da empresa fornecedora, ficando a cargo das empresas apenas provisionar e gerenciar os recursos que serão gastos.

Figura 2. Exemplo de serviços existentes na nuvem sendo acessados por diversos dispositivos.



Fonte: Adaptado de W3ii, 2016

Como exemplificado na Figura 1, é possível ter vários tipos de serviços sendo ofertados por empresas de nuvens públicas. Estes serviços podem ser gerenciados através de portais acessados pela Internet. Dentre esses serviços são disponibilizadas máquinas virtuais, bem como bancos de dados, hospedagem de aplicações, hubs de notificação para aparelhos móveis, entre outros.

A utilização de computação em nuvem no âmbito empresarial apresenta inúmeras vantagens:

- Não é necessário ter uma equipe de infraestrutura nem de servidores internos para manter os serviços funcionando.
- A escalabilidade de recursos somente ocorre quando houver necessidade.
- A cobrança é realizada somente por aquilo que é usado.
- Não há necessidade de adquirir recursos antecipadamente, pois as provedoras de serviço sempre terão o recuso disponível.

3 METODOLOGIA

Scrum é um framework ágil, simples e leve, utilizado para gestão do desenvolvimento de produtos complexos imersos em ambientes complexos. Scrum é embasado no empirismo e utiliza uma abordagem iterativa e incremental para entregar valor com frequência e, assim, reduzir os riscos do projeto (SABBAGH, 2013, p. 17).

O Scrum, por ser um framework, contém diversas ferramentas que podem ser utilizadas, e muitas delas são direcionadas a uma equipe de pessoas. Essas pessoas recebem papéis, e cada um deles tem por si suas responsabilidades. Estas ferramentas, que são atribuídas a grupos de pessoas ou direcionadas para cada pessoa, não foram utilizadas, pois não é possível uma única pessoa tomar para si todos os papéis.

Logo, foram utilizados somente certos artefatos e eventos para apoiar o desenvolvimento do projeto. Foram escritas histórias de usuário, as quais vão apresentar cada requisito do sistema de forma simples. Junto a elas também foram escritos os critérios de aceitação, que demonstram a quais regras de negócio a história deverá obedecer.

Cada história também recebeu seu peso de dificuldade, baseado na sequência de Fibonacci, para a qual quanto maior o número, mais complexa e difícil a história será para ser desenvolvida. Todas as histórias descritas formaram o Product Backlog, que é uma lista de histórias que foram desenvolvidas para a entrega do projeto.

Para definir quando uma história foi finalizada, foi criada uma definição de pronto (Definition of Done), que demonstra quais regras todas as histórias deverão seguir para serem concluídas.

Sprints são períodos de tempos que variam entre duas e quatro semanas, conforme cada aplicação. Antes de iniciá-la, é necessário fazer o planejamento para definir quais histórias comporão o sprint backlog. Geralmente, as histórias que entregam mais valor ao interessado ou histórias que são dependências de outras são realizadas nas primeiras sprints. Com o sprint backlog pronto, as histórias foram quebradas em tarefas menores que, quando realizadas, farão com que todos os requisitos da história estejam desenvolvidos.

3.1 FERRAMENTAS

Foram utilizadas durante a construção do projeto as seguintes ferramentas:

- **Visual Studio:** ambiente de desenvolvimento

integrado desenvolvido pela Microsoft, usado para desenvolvimento de projetos de várias plataformas.

- **Visual Studio Team Service:** plataforma de planejamento, versionamento de código, testes e implantação de sistemas.
- **Xamarin:** framework de desenvolvimento nativo de aplicações móveis multiplataforma.
- **Azure e Google Cloud Platform:** plataformas que disponibilizam soluções baseadas em nuvem, dentre elas: máquinas virtuais, hospedagem de aplicações web, banco de dados e armazenamento de arquivos.
- **.Net Core:** conjunto de ferramentas que permitem a criação de aplicações multiplataforma, desde aplicações que rodem em linha de comando até aplicações web, que foram totalmente reescritas, melhorando seu desempenho e sua usabilidade. Com elas é possível utilizar várias linguagens de programação de alto nível que serão usadas, no caso o C#.
- **Node.js:** conjunto de ferramentas que permite a criação de aplicações utilizando a linguagem JavaScript.
- **MongoDB:** banco de dados não relacional para armazenamento de objetos Javascript.
- **MySQL:** banco de dados relacional para armazenamento de dados estruturados.

4 DESENVOLVIMENTO E RESULTADOS

Para controle das funções do software, foram desenvolvidas histórias de usuário para acompanhar a evolução do projeto, utilizando-se a ferramenta de gerência Team Foundation Service. Foram totalizadas 12 histórias de usuário. A Figura 3 discrimina estas histórias junto com sua pontuação de dificuldade estimada.

Figura 3. Lista de histórias de usuário e suas respectivas pontuações de dificuldades

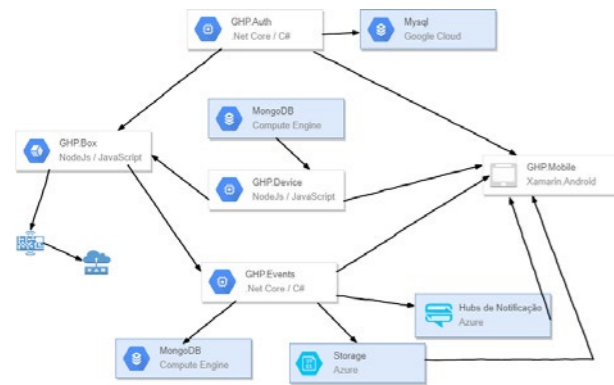
Order	Title	Story
1	Como usuário quero me autenticar para poder usar o sistema.	3
2	Como aplicação quero conectar ao sensor para poder monitorar o ambiente.	5
3	Como usuário quero ativar e desativar a proteção para eu poder escolher quando utilizar.	5
4	Como sensor quero gerar eventos para poder notificar o usuário dos acontecimentos.	8
5	Como sensor quero obter ordens remotas para que possa ser executado pelo usuário remotamente.	8
6	Como usuário quero autenticar usuário para poder realizar atividades na plataforma.	3
7	Como usuário quero registrar um novo usuário para poder realizar atividades na plataforma.	3
8	Como serviço quero receber um comando remoto para que possa transmitir-lo para o sensor.	5
9	Como serviço quero enviar o comando recebido para o sensor para que ele possa executar alguma ação.	5
10	Como serviço quero notificar o usuário para que ele possa ficar informado sobre acontecimentos da plataforma.	5
11	Como usuário quero receber push notifications para que possa receber eventos que ocorrem nos sensores.	12
12	Como usuário quero enviar comandos remotos para o serviço para que eu possa controlá-lo caso necessário remotamente.	5

Fonte: Elaborado pelo autor.

O projeto totalizou 74 pontos de dificuldade, o que resultou na formação de 6 sprints de 15 dias. Não foram estimadas as horas de desenvolvimento durante a execução do projeto. Para atender às histórias de usuários e seus critérios de aceitação, foi arquitetada e desenvolvida a arquitetura apresentada na Figura 4.

imagens retiradas pelos sensores foi utilizado o serviço da Azure de armazenamento de dados, o Azure Blob Storage. Este serviço foi escrito em .Net Core(C#) para facilitar o envio de dados para o armazenamento de dados. Este serviço também utiliza o Hub de notificações da Azure para enviar notificações para a aplicação mobile.

Figura 4. Arquitetura desenvolvida para a solução



Fonte: Elaborado pelo autor.

O modelo arquitetural apresenta três microsserviços:

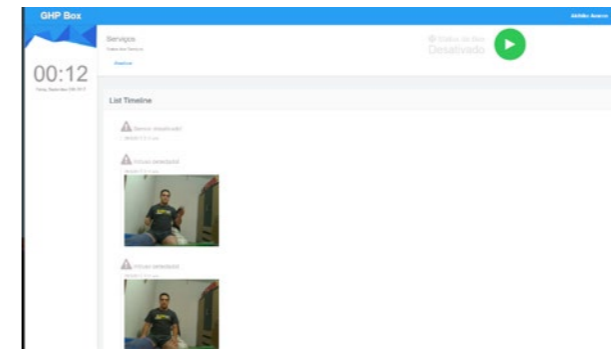
- *GHP.Auth: este serviço tem como responsabilidade manipular as informações dos usuários de suas credenciais de acesso. Esse serviço foi escrito em .Net Core(C#), e essa tecnologia foi escolhida, pois é necessário manipular dados estruturados. Esse serviço utiliza MySQL como base de dados relacional.*
- *GHP.Device: esse serviço tem como responsabilidade manipular as informações relacionadas aos dispositivos conectados a plataforma. É por ele que os comandos remotos são enviados para o dispositivo que controla os sensores e a câmera. Esse serviço foi escrito em Javascript (Node.js), por ser fácil realizar implementações de sockets e a manipulação de dados não estruturados. Neste caso, foi utilizado MongoDB por ele também não manter dados não estruturados.*
- *GHP.Events: esse serviço tem como responsabilidade armazenar e notificar o usuário dos novos eventos gerados pelos sensores. Estes eventos também ficam disponíveis para acesso posterior ou para auditorias futuras. Foi utilizado o MongoDB, pois o volume de dados esperado para ser armazenado é alto. Para armazenar as*

Estes três serviços estão rodando atualmente em um serviço de hospedagem da Azure. Ao utilizar os serviços da Azure para hospedagem, é possível escalar em determinados cenários cada serviço separado. Este escalonamento pode ser feito tanto verticalmente (mais recursos para instância) quanto horizontalmente (novas instâncias da aplicação), além de garantir fácil aplicação de balanceamento de carga e alta disponibilidade dos serviços.

Os bancos de dados citados e demonstrados na Figura 3 estão rodando em ambiente de nuvem. O banco relaciona o MySQL que está sendo ofertado como SaaS (Software como Serviço), em que cabe ao provedor contratado manter, atualizar e fornecer backup. Já no caso dos bancos MongoDB, eles foram instalados em uma máquina virtual fornecida pelo provedor de nuvem IaaS (Infraestrutura como serviço), em que cabe ao administrador manter, atualizar e realizar backup dos dados.

Para captar os dados do ambiente, foi desenvolvido um dispositivo utilizando-se Raspberry, Arduino, sensores de presença e uma câmera para Raspberry. Esse dispositivo foi nomeado como GHP.Box e executa um servidor Node.js, que é responsável por controlar todos os dispositivos. O serviço interno faz a leitura do sensor: caso o dispositivo esteja ativado, a câmera irá disparar uma foto do ambiente. Esses dados recolhidos são enviados para o serviço GHP.Events descrito anteriormente. O custo total dos dispositivos eletrônicos necessários para o projeto foi de R\$ 385,00 reais. A Figura 5 apresenta a tela desenvolvida para acessar as informações do servidor embarcado junto aos sensores.

Figura 5. Interface de controle embarcada junto ao sensor

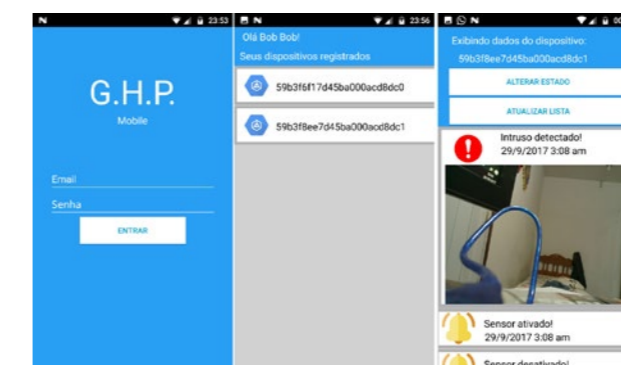


Fonte: Elaborado pelo autor.

Conforme apresentado, é possível notar algumas funcionalidades desenvolvidas como a atualização do formulário para atender a novos eventos: a alteração do estado do sensor e o acesso aos eventos do dispositivo.

A aplicação Mobile foi desenvolvida utilizando-se Xamarin.Android. Cabe ao aplicativo dar acesso remoto aos eventos gerados pelo dispositivo citado anteriormente. Ele acessa todos os serviços desenvolvidos obtendo os dados e apresentando-os na tela. Cabe a ele também permitir acesso remoto ao GHP.Box. As notificações geradas pelo GHP.Events são enviadas para ele, notificando o usuário dos eventos ocorridos no ambiente monitorado. A Figura 6 apresenta as telas que compõem a aplicação móvel.

Figura 6. Interfaces da aplicação mobile. Em (a) a tela de autenticação, em (b) os dispositivos visíveis ao usuário e (c) os eventos do dispositivo



Fonte: Elaborado pelo autor.

A imagem (a) prevê o acesso ao aplicativo, em que o usuário informa suas credenciais de acesso. Caso suas credenciais sejam válidas, ele é redirecionado

para a tela de dispositivos a que o usuário tem acesso, apresentada na imagem (b). Quando o usuário seleciona um dos dispositivos, ele é redirecionado para a tela apresentada na imagem (c), que apresenta todos os eventos gerados de forma decrescente a partir do último evento gerado até o mais novo. Esta tela também dá o acesso aos comandos remotos que estão disponíveis.

Ao longo do desenvolvimento do projeto, seu escopo sofreu alterações, e uma das mais impactantes foi a troca do sensor Kinect do Xbox One para o Raspberry e Arduino. Existe a possibilidade de embarcar uma solução desenvolvida para a plataforma do Xbox One, porém esta não foi realizada durante o desenvolvimento do projeto, devido ao fato de que as bibliotecas, para obter determinados dados do Kinect, não estão disponíveis ainda para a plataforma universal do Windows.

5 CONCLUSÃO

Este artigo teve como objetivo apresentar os ganhos da utilização de microsserviços e computação na nuvem, com o desenvolvimento de uma plataforma de proteção domiciliar aplicando boas práticas de utilização de microsserviços e computação na nuvem. Logo, o objetivo de desenvolver a plataforma de segurança domiciliar foi alcançado. Ao longo do processo de desenvolvimento foi observado que houve uma heterogeneidade das tecnologias aplicadas ao processo de desenvolvimento, em que foram aplicadas somente tecnologias atualizadas e bem definidas para cada operação. Foi possível visualizar na prática todos os benefícios que são obtidos, ao se utilizar arquiteturas baseadas em microsserviços.

Ao longo do processo, foi identificado que o Kinect não era viável e ele foi logo substituído por outro hardware capaz de realizar tal processo. Entretanto, foi verificado que é possível embarcar uma aplicação dentro do Xbox One através da Plataforma Universal do Windows.

A utilização de computação na nuvem facilitou grande parte do processo de configuração de máquinas virtuais e dos serviços de hospedagem das aplicações, demonstrando a facilidade de tais plataformas.

Em versões futuras, será possível através de serviços cognitivos a detecção de faces e comandos de voz para controlar ações dos dispositivos da plataforma. Serão desenvolvidos novos dispositivos integrados

a plataforma, como sensores de temperatura, sensores de porta e alarmes. Outra funcionalidade que será capaz de ser utilizada é a utilização de Beacons como chaves de segurança presencial, assim não sendo mais necessária a utilização de aplicativos ou acesso a redes para ativar e desativar sensores.

REFERÊNCIAS

BONÉR, Jonas. Reative Microservices Architecture: Design Principles for Distributed Systems. Sebastopol: O'Reilly Media, 2016.

CARDOSO, Gabriel Schade. Microsoft Kinect: criando aplicações interativas com o Microsoft Kinect. São Paulo: Editora Casa do Código, 2013.

CUSTÓDIO, Thiago. Azure: Coloque suas plataformas e serviços no cloud. São Paulo: Editora Casa do Código, 2016.

NEWMAN, Sam. Building Microservices. Sebastopol: O'Reilly Media, 2015.

SABBAGH, Rafael. Scrum: Gestão ágil para projetos de sucesso. São Paulo: Editora Casa do Código, 2014.

SUTHERLAND, Jeff. Scrum: a arte de fazer o dobro do trabalho na metade do tempo. São Paulo: LeYa, 2014.

TAURION, Cezar Cloud computing: computação em nuvem: transformando o mundo da tecnologia da informação. Rio de Janeiro: Brasport, 2009

SOUSA, Carlos E. L. de; CAVALCANTI JUNIOR, Paulo C. A.; CARNEIRO, Thiago F. Simplifica! Burndown Chart. Disponível em: < <https://goo.gl/Mbi66i> />. Acesso em: 07 mar. 2017.

SLOYER, Jeff. Microservices in Bluemix. Disponível em: < <https://goo.gl/g6XloS> >. Acesso em: 07 mar. 2017.

W3II. Cloud Computing Overview. Disponível em: < <https://goo.gl/yhEkMy> >. Acesso em: 07 mar. 2017.

VIABILIDADE ECONÔMICA DO SISTEMA DE INTEGRAÇÃO LAVOURA-PECUÁRIA



Leonardo Costa Dias
Estudante de Graduação 8º período do Curso de Administração do UNIPAM
leonardo_costadias@hotmail.com



Gustavo Magalhães Vieira
Professor do Curso de Administração do UNIPAM
gustavomv@unipam.edu.br

