

Estudo de modelos de Edge Computing utilizando arquitetura em nuvem

Vinicius Gomes de Lima

Orientador: Raul Ikeda Gomes da Silva

Inspira, 2018/2019

Sumário

1	Introdução.....	2
2	Vertentes de edge computing.....	4
	2.1 Cloudlet.....	4
	2.2 Fog computing.....	6
	2.3 Mobile edge computing.....	7
3	Comparação das vertentes.....	8
	3.1 Gerenciamento de request.....	8
	3.1.1 Fog computing.....	8
	3.1.2 Mobile Edge computing.....	8
	3.1.3 Cloudlet.....	9
4	Aplicações.....	9
	4.1 Proposta de uma nova arquitetura.....	9
5	Metodologia.....	10
6	Resultados finais.....	11
	6.1 Tutorial de instalação do kubernetes.....	12
7	Conclusão.....	12
8	Bibliografía.....	12

1. Introdução

A evolução do *Cloud Computing* já é bem nítida nesses últimos anos, e com essa evolução surgiram necessidades que não conseguiram ser supridas com a atual arquitetura.

Essas necessidades aparecem devido ao rápido crescimento dos dispositivos *IoT*(*Internet of Things*) e no aumento de dispositivos móveis. É estimado pela Cisco que o número desses dispositivos *IoT* conectados seja de 50 bilhões em 2030(Ai. Yuan, 2018), assim como o aumento de remessas de celulares e tablets cresceu exponencialmente em 2016 chegando a 1.9 bilhões de unidades globais de acordo com *International Data Corporation*(*IDC*) (T. H. Noor, 2016) .

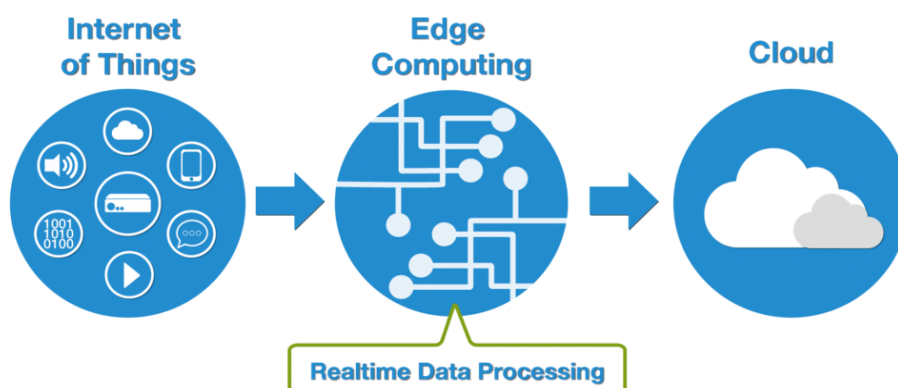
Com esse aumento, a rede se torna congestionada, o que dificulta a transferência de dados, ou seja, uma maior lentidão na internet. Além de que alguns desses dispositivos precisam de uma resposta rápida, e com uma rede mais congestionada, o tempo de enviar as informações para a nuvem, esperar eles serem processados e enviados de volta, ou seja a latência da rede, torna inviável a utilização de tecnologias como realidade aumentada.

O *Edge computing* pode ser a solução para esses problemas com a promessa de trazer os serviços de nuvem cada vez mais perto das fontes de dados, dispositivos *IoT*, fazendo com que os dados sejam processados antes de mesmo de “saírem” para a nuvem que temos hoje em dia. A Figura 1 representa o fluxo de uma arquitetura em *edge*, onde os dispositivos *IoT* enviam dados à serem processados no servidor *edge*, nele os dados serão processados e mandados de volta sem precisar se conectar à um servidor externo, porém caso seja necessário alguns desses dados podem ir para a nuvem.

Como a maior parte dos dispositivos *IoT* possuem uma limitação de hardware, é possível utilizar contêineres, que são um conjunto de processos organizados isoladamente do sistema permitindo utilizar esses dispositivos com maior eficiência. O *kubernetes* faz o papel de orquestrar esses contêineres, monitorando o uso de memória, cpu, tempo de vida, prevendo possíveis falhas e automatiza procedimentos importantes para manter o sistema com o menor número possível de

falhas, reativando contêineres falhos, escalando processos com alta demanda e desescalando processos com baixa.

Figura 1: Representação do *edge computing*. Mais a esquerda, Internet das coisas, no centro, *Edge computing* e a direita Nuvem.



Fonte : https://www.talari.com/glossary_faq/edge-computing/

Pode-se dividir em três arquiteturas diferentes: *cloudlet*, *fog computing* e *mobile edge computing*. Essas, possuem a mesma essência, diminuir a distância da fonte para diminuir a latência e congestionamento na rede, porém a diferença do *workflow* é o diferencial e é o que permite o uso dessa tecnologia em diversos cenários.

Usando como analogia um bairro, o *cloudlet* seria *datacenters* compactos, um computador ligado à internet com poder computacional ocioso, ou até um cluster de máquinas.

No *mobile edge computing*, os servidores estão em movimento, celulares, dispositivos *IoT*, notebooks em hibernação ou até carros autônomos. Além do próprio poder computacional, esses poderiam utilizar do processamento ocioso desses outros dispositivos conectados entre si.

Já com o *fog computing*, ainda utilizando um bairro como exemplo, no centro geográfico existiria um servidor principal com alto poder computacional e alguns servidores menores seriam distribuídos no bairro. Ao enviar dados para serem processados, esses dados passam por cada um desses nós, havendo uma

granularidade do processamento, ou seja, cada um possui uma responsabilidade diferente, como filtrar ou organizar dados por exemplo, até chegar no servidor principal.

2. Arquiteturas de *edge computing*

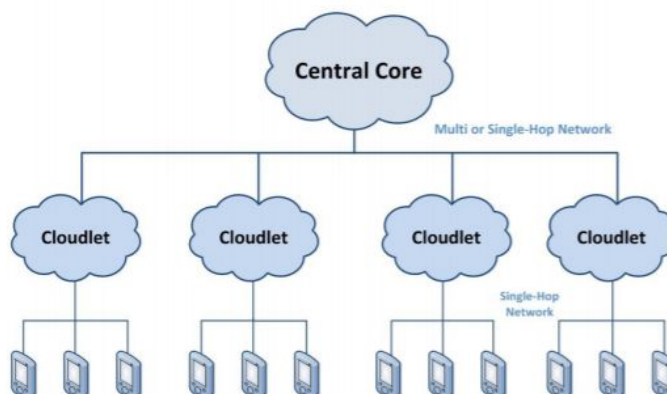
Essa seção irá detalhar melhor as três principais vertentes de *edge computing*, em relação aos equipamentos e funcionamento da rede.

2.1 Cloudlet

Cloudlets é uma pequena nuvem localizada próxima ao usuário *mobile*. Ele é instalado em um servidor localizável, *stateless* e que está rodando uma ou mais máquinas virtuais, em que os dispositivos podem enviar os dados a serem processados. É composto por computadores confiáveis, um bom *hardware* com uma boa conexão a internet e que esteja disponível para os dispositivos próximos se conectarem (B. Aleksandar , 2013).

Como é possível ver na figura 4, eles representam a camada intermediária da hierarquia da arquitetura (Camada do dispositivo, camada cloudlet e a nuvem) para chegar em um tempo de resposta menor (Ai. Yuan, 2018).

Figura 2: Hierarquia da arquitetura: Dispositivo, cloudlet, Cloud



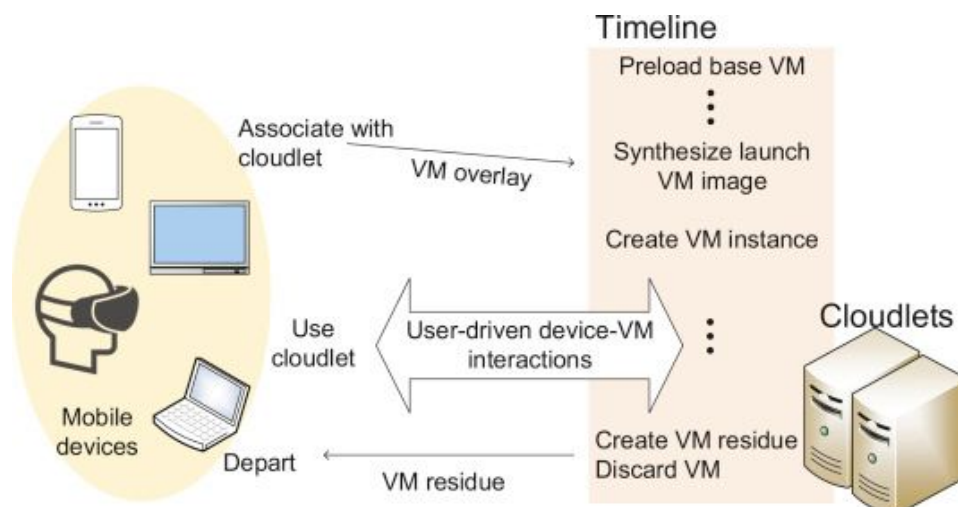
Fonte: Cloudlet Challenges, 2013

Um método para a transferência de dados usado é a síntese de VM (*Virtual machine*), na Figura 3 há um modelo mostrando algumas etapas dessa síntese com a conexão de um dispositivo com o cloudlet. Nesse caso, uma sobreposição do aplicativo é transferida do dispositivo móvel para o cloudlet.

Na síntese de VM a aplicação envia uma sobreposição de VM para o cloudlet, que já possui um VM base. O *cloudlet* descomprime essa sobreposição, aplica na VM base para iniciá-la, então cria uma instância da VM. Assim o dispositivo pode começar a fazer as operações de *offloading* nesta instância. No fim da sessão, essa instância é destruída, porém a VM iniciada pode ficar guardada no *cache* para futuras sessões, o cloudlet, então, gera um resíduo da VM que é mandado de volta para o dispositivo para ser incorporado na aplicação. (Ai. Yuan, 2018)

A síntese de VM é útil em ambientes caracterizados por redes não confiáveis, perda de conexão constantes e necessidade de um rápido *deployment*. Por exemplo, um usuário precisa executar uma aplicação que necessita um intenso processamento configurado para ser executado no *cloudlet*. No tempo de execução, a aplicação encontra o *cloudlet* mais próximo e faz o *offloading*. Porém, devido a perda de conexão com a rede, ou falta de energia no *cloudlet*, a aplicação perde a conexão com o servidor. Então, a aplicação pode procurar por um novo *cloudlet* e executar aplicação sem precisar configurá-la ou o *cloudlet*. Isso é uma solução flexível que permite o uso de recursos disponíveis na rede, substitui recursos desconectados e customiza dinamicamente novos recursos. (A. Bahtovski, 2014)

Figura 3: Síntese de VM



Fonte : Edge computing technologies for Internet of Things: a primer, 2018

2.2 Fog Computing

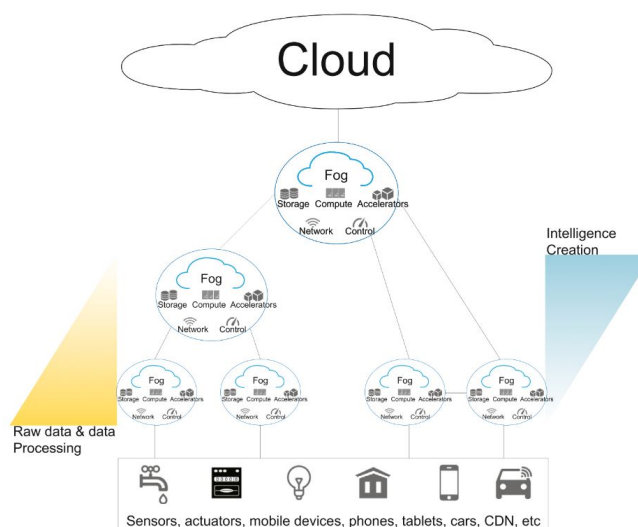
Fog computing é um modelo de distribuição que funciona na camada intermediária entre o servidor *cloud* e o dispositivo/sensor IoT. Esse provém conectividade, serviços de armazenamento e processamento com o objetivo de trazer um serviço de baseado em *cloud* para mais perto do dispositivo IoT (T. Shletami, 2018).

Um domínio de *fog computing* é feito com dispositivos normais de rede, como roteadores, *switches*, servidores, *base stations*(BS), etc. E pode ser implementado perto dos sensores IoT. Portanto, as partes das redes permitem o *fog computing* de criar uma expansiva transição dos serviços baseados em *cloud* (T. Shletami, 2018).

Uma arquitetura hierárquica típica baseada em *fog computing* é mostrada na Figura 5. No ponto de vista funcional, um *node* apresenta várias funcionalidades, incluindo, conexão a uma rede, processamento, aceleração, armazenamento e controle. Esses nodes podem comunicar-se entre si através de cabos ou *wireless* (Ai. Yuan, 2018)

Nodes feitos para análise aprimorada precisam configurar módulos de aceleração como unidades de processamento gráfico, arranjos de portas programáveis em campo (*Field Programmable Gate Array, FPGA*) e processadores de sinal digital para fornecer rendimento computacional suplementar (Ai. Yuan, 2018).

Figura 4: Modelo de fog computing.



Fonte : Edge computing technologies for Internet of Things: a primer, 2018

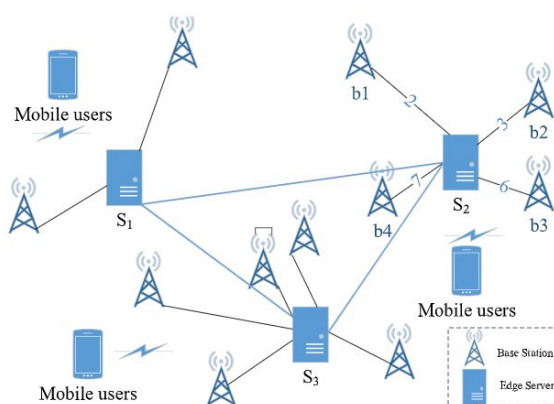
2.3 Mobile edge computing

Mobile edge computing (MEC) traz o processamento e armazenamento de dados para RAN (*Radio Access Network*). Esses servidores rodam *MEC hosts*, permitindo a virtualização dos processos dentro das *Base Stations* (BS). Os servidores oferecem informações da rede em tempo real incluindo a capacidade da rede e o quanto está sendo utilizado, assim como informações do usuário, como sua localização. (D. Koustabh, 2017)

Como é possível ver na Figura 6, as BS ficam distribuídas em áreas estratégicas a fim de se conectar com os *edge computing servers* (ECSs), permitindo mais áreas com pontos de conexão para os usuários usarem os serviços.

Porém, os locais de distribuição dos servidores *edge* nessa vertente é um desafio. A localização é um ponto crítico ao tempo de acesso dos usuários móveis assim como a utilização de recursos, especialmente em cidades inteligentes que incluem muitas BSs. Com uma rede muito grande, servidores *edge* mal posicionados resultarão em maior tempo de resposta e o *workflow* ficará desbalanceado dentro da rede, causando algum dos servidores ficarem sobrecarregados enquanto outros ficaram inativos. (W. Shangguang, 2017)

Figura 5: Exemplo de MEC com os distribuição dos servers



Fonte: Edge server placement in mobile edge computing, 2019

3. Comparação das vertentes

Nessa seção será feita a comparação das três vertentes, como mecanismo de comunicação, localização, assim como qual o melhor caso para utilizá-los.

A Tabela 1 mostra uma dessas diferenças na implementação das vertentes.

Tabela 1 : Comparação das vertentes

	Fog Computing	Mobile-Edge Computing	Cloudlet Computing
Node devices	Routers, Switches, Access Points, Gateways	Servers running in base stations	Data Center in a box
Node location	Varying between End Devices and Cloud	Radio Network Controller/Macro Base Station	Local/Outdoor installation
Software Architecture	Fog Abstraction Layer based	Mobile Orchestrator based	Cloudlet Agent based
Context awareness	Medium	High	Low
Proximity	One or Multiple Hops	One Hop	One Hop
Access Mechanisms	Bluetooth, Wi-Fi, Mobile Networks	Mobile Networks	Wi-Fi
Internode Communication	Supported	Partial	Partial

fonte : Comparison of Edge Computing Implementations:Fog Computing, Cloudlet and Mobile Edge Computing, 2017

3.1 Gerenciamento de *requests*

3.1.1 Fog Computing : Nessa arquitetura há um orquestrador (*Fog Orchestrator*) atuando por cima dos *Fog Nodes* se comunicando através de funções expostas pela camada de abstração Fog (*Fog abstraction layer*). O usuário se comunica com esse orquestrador, enviando uma *request* com requerimentos específicos a ser utilizado pela rede. Estes requerimentos podem ser *Quality of service(QoS)*, configuração mínima do *Fog Node*, *load balance* entre outros.

O orquestrador compara os requerimentos da *requests* com os serviços expostos por cada *node*, retornando uma lista ordenada dos disponíveis, e assim escolhendo qual a melhor opção a ser utilizada. (D. Koustabh, 2017)

3.1.2 Mobile Edge computing : Assim como no *Fog computing*, existe um orquestrador na arquitetura, *Mobile edge orchestrator*, esse também recebe os *requests* dos usuários. Nesse orquestrador é mantido todas as aplicações da rede e é constantemente sendo atualizado pela *ME Platform Manager*. Com o recebimento do *request*, o orquestrador verifica se há uma aplicação desse tipo já sendo utilizada, se sim, o request é redirecionado para ele aplicação, caso contrário, se a aplicação necessária for suportada pela rede, então essa é instanciada, porém caso essa não

esteja rodando e não seja suportada, o processo é redirecionado para um servidor de *Cloud*. (D. Koustabh, 2017)

3.1.3 Cloudlet : O cloudlet ao receber o *request*, esse é analisado pelo *Cloudlet agent*, agente responsável por um grupo de *nodes* dentro de uma proximidade, que se comunica com o *Node agent*, agente responsável por um ou mais *execution environments (EE)*. Assim é possível uma escolha mais otimizada para *requests* mais complexas e direcioná-las para *nodes* com maior poder computacional. (D. Koustabh, 2017)

4. Aplicações

Nessa seção será discutido onde *edge computing* está sendo usado e onde poderia ser aplicado.

O ramo de pesquisa e desenvolvimento de *edge* ainda está no começo, ou seja, as aplicações são muito mais teóricas do que práticas, mas alguns cenários que estão utilizando essa tecnologia são no setor industrial, para processamento dos dados dos sensores dos maquinários para monitoramento e revisão em tempo real. Nas cidades inteligentes no controle de tráfego, onde os semáforos possuem sensores e câmeras para tomada de decisão em relação a temporalização. E para orquestração de vídeo, com instâncias perto do usuário prontas para fazer *download* de *streaming*, principalmente usado para eventos esportivos.

E alguns cenários, onde seria possível implementar a tecnologia, na agricultura, com tratores e máquinas autônomos, realidade aumentada e virtual (RA e RV, respectivamente).

4.1. Proposta de uma nova arquitetura:

O aumento de dispositivos conectados, IoT, assim como aplicações que em tempo-real, o desenvolvimento e pesquisa na área de *edge computing* se torna promissor e essencial para tecnologias futuras, considerando isso, uma possível nova arquitetura foi pensada utilizando o conceito do *mobile edge computing*.

Essa nova arquitetura seria um compartilhamento de poder computacional *on-demand* e *real-time*, um exemplo para ficar mais claro seria dois carros autônomos, um modelo mais antigo(carro 1 para simplificar) e um mais novo(carro

2), caso o carro 1 esteja operando a 90 % da sua capacidade de processamento e a demanda tenda a aumentar, em algum momento algumas decisões falhando e um acidente ocorra. Nesse cenário, o carro 2 está próximo e com poder computacional ocioso, dentro dessa arquitetura, seria possível que o carro 1 use parte de processamento ocioso diminuindo o stress do sistema.

Isso entraria dentro das propostas originais de *edge computing*, haveria baixa latência, diminuição do tráfego de rede e a “ponta” da rede seriam todos os dispositivos conectados na rede com permissão de compartilhamento. Assim para uma pessoa poderia compartilhar um *GPU* que não está usando e cobrar um aluguel ou dentro de uma empresa, todos os dispositivos conectados a internet poderiam ser um nó, ou seja, o modelo utilizado antigamente, *BYOD (bring your device)*, poderia ser modificado por *BYODC (bring your device to cloud)*, pois a cada novo dispositivo, maior seria a capacidade da rede.

5. Prova de Conceito

Esse estudo teve como objetivo avaliar a implementação do modelo *edge*, comparando o tempo de resposta, viabilidade do projeto e um modelo de cobrança com serviços atuais de nuvem pública.

Uma prova de conceito (*Proof of concept - PoC*) foi feita para validar as hipóteses e a montagem do sistema utilizando um cluster.

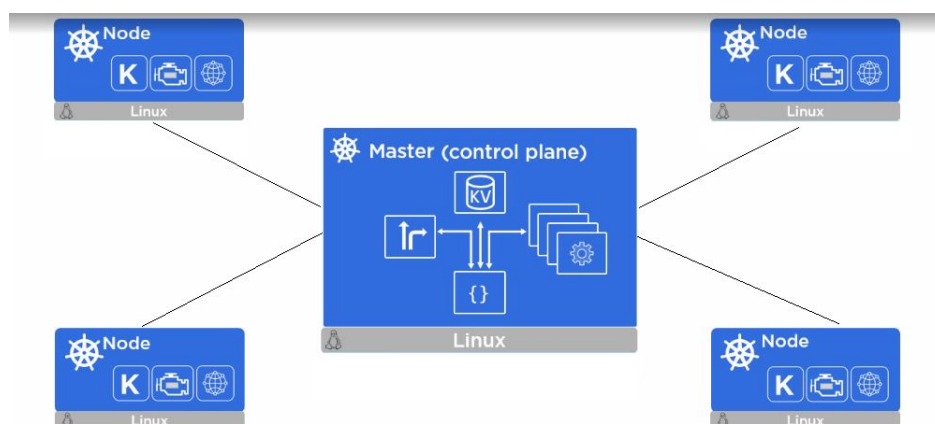
Através de estudos de casos e pequenos protótipos construídos com IoT, foi avaliado modelos de serviços de *Cloud* rodando em modelo *Edge Computing*. Tanto os custos operacionais quanto de projeto foram comparados com outros serviços disponíveis em nuvens públicas.

O protótipo foi feito utilizando computadores *raspberry pi* como pontos de rede, o sistema operacional usado é o *Hyprion*, ele permite conectar-se com o *Docker*, o que facilita a implementação do *kubernetes*, esse é o sistema utilizado para o gerenciamento e escalonamento dos contêineres.

Como é possível ver na Figura 6, há cinco máquinas, a central representando o *master* e em volta os *workers*. O master é responsável por monitorar toda a atividade dentro do cluster, essas atividades consistem em: uso de *cpu*, memória,

número de *workers*, réplicas, etc. Uma das tarefas mais importantes do *master*, é verificar se algum dos *workers* parou de funcionar ou se desconectou, a fim de criar outro e manter o sistema funcionando. Já o *worker* é responsável apenas em executar a aplicação que lhe foi atribuída.

Figura 6: Diagrama de funcionamento de um cluster kubernetes.



Fonte: <https://dzone.com/storage/temp/8742780-kubernetes-architecture.png>

Essa arquitetura permite distribuir os computadores *raspberry* a fim de manter o ponto de acesso o mais próximo possível das áreas com maior demanda. Permitindo qualquer um que esteja no raio de alcance desse ponto da rede se conecte e utilize seus serviços.

Com a rede sendo monitorada pelo *master*, em áreas com mais acesso as máquinas serão escaladas automaticamente para suprir a demanda.

6. Resultados finais

Primeiro foi feito a instalação do sistema operacional Hyprion nos computadores raspberries, após finalizado, foi preciso fazer algumas configurações em relação a certificados e *proxies*.

Com as configurações feitas, foi feito a instalação do *kubernetes* no sistema. Foi necessário fazer mais alguns ajustes antes de ser possível iniciar a plataforma na máquina.

Com essas configurações feitas, a plataforma se iniciou com sucesso como um *master*, assim foi possível conectar outro raspberry, como todas as configurações mencionadas acima, porém dessa vez não como *master*, mas como worker.

Mesmo com essas configurações prévias e sucesso, o sistema se mostrou instável, não iniciando novamente em outras fases de teste.

6.1 Tutorial de instalação do kubernetes:

- Baixar e configurar um cartão sd com a imagem do *HyriotOS*.
- Dentro do sistema operacional (SO), atualizar o sistema.
- Instalar o kubectl e o kubeadm.
- Inicializar o kubeadm como *master*.
- Em outro computador com *Hyriot* e kubectl, juntar a maquina no cluster do *master*.

7. Conclusão

Essa pesquisa introduz as três vertentes de edge computing, *fog computing*, *mobile edge computing* e *cloudlet*, explicando um pouco da arquitetura, métodos de *offloading*, assim como comparando como são gerenciados os *requests* e alguma aplicação de cada uma. Junto a isso, uma PoC seria utilizada para validar se realmente há uma melhora no que se refere à latência e integridade da rede ao trazê-la mais próxima à fonte. Anotações foram feitas para que futuras iterações tenham uma possibilidade maior de sucesso.

Uma possível nova arquitetura também foi proposta que se encaixa dentro dos conceitos iniciais de *edge computing*, possibilitando uma nova área de pesquisa e desenvolvimento nessa área, que já é um paradigma dentro de *cloud computing*.

8. Bibliografia

Y. Ai, M. Peng, K. Zhang, Y. **Edge computing technologies for Internet of Things: a primer**, Key Laboratory of Universal Wireless Communications for Ministry of Education, Beijing Uni

A. Bahtovski, M. Gusev. **Cloudlet Challenges**, Ss. Cyril and Methodius University Faculty of Information Sciences and Computer Engineering 1000 Skopje, Macedonia

T. H. Noor, S. Zeadally, A. Alfazi, Q. Z. Sheng. **Mobile cloud computing: Challenges and future research directions**, *Department of Computer System & Technology, University of Malaya, 50603 Kuala Lumpur, Malaysia.*

T. R. Sheltami, E. Q. Shahra, E. M. Shakshuki. **Fog Computing: Data Streaming Services for Mobile End-Users**. College of Computer Sciences and Engineering King Fahd University of Petroleum and Minerals (KFUPM), Dhahran, Saudi Arabia 31261.

W. Shangguang, Z. Yali, X. Jinlinag, Y. Jie, H. Ching-Hsien, **Edge server placement in mobile edge computing**, State Key Laboratory of Networking and Switching Technology ,Engineering Beijing University of Posts and Telecommunications , Beijing 100876, China.

D. Koustabh, K. D. Soumya, **Comparison of Edge Computing Implementations: Fog Computing, Cloudlet and Mobile Edge Computing**, 1-6. 10.1109/GIOTS.2017.8016213