



Using Mobile Cloud Computing for Developing Context-Aware Multimedia Applications

A Case Study of the CAOS Framework

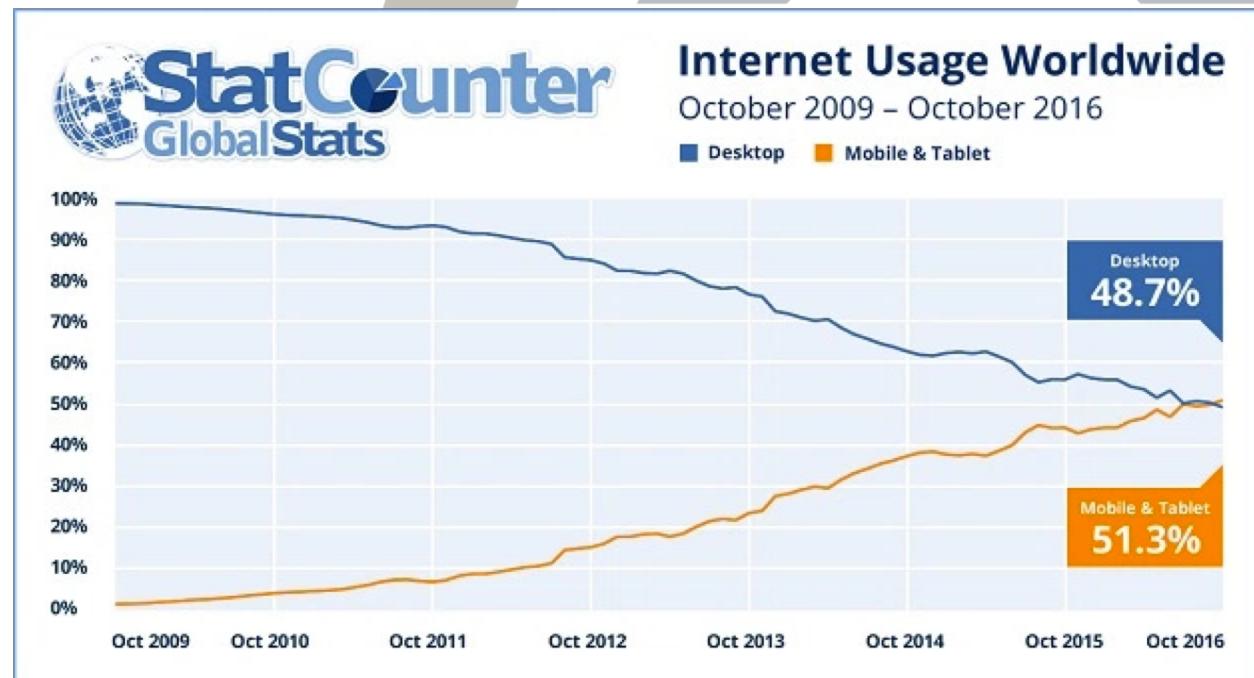
Fernando A. M. Trinta, Paulo A. L. Rego,
Francisco A. A. Gomes, Lincoln S. Rocha e José N. de Souza

Grupo de Redes de Computadores, Engenharia de Software e Sistemas (GREat)
Universidade Federal do Ceará (UFC)

Computação Móvel



- Paradigma cada vez mais presente



Alguns dados...

A screenshot of a web browser window displaying a news article from TecMundo. The title of the article is "Total de smartphones no Brasil supera PCs e chegará a 236 milhões em 2 anos". The page includes social sharing buttons for Facebook, Twitter, Google+, LinkedIn, and a count of 5 shares. Below the article is a large image of many smartphones.

Total de smartphones no Brasil supera PCs e chegará a 236 milhões em 2 anos

POR LEONARDO ROCHA | @leonardo.barreiros.rocha - EM PRODUTO - 25 OUT 2016 - 13H13

[COMPARTILHAR](#) [G+](#) [in](#) 3 5 compartilhamentos

Mais dados...

Android inches ahead of Windows as most popular OS

Microsoft no longer has a dominant operating system when you look at internet users worldwide, StatCounter finds.

BY MICHELLE MEYERS / APRIL 3, 2017 10:55 AM PDT

f t F e m

Ad closed by Google

Cenário atual

- Aumento da capacidade de processamento e armazenamento...
- Inclusão de sensores
- Melhoria das plataformas de desenvolvimento
- Como consequência:
 - Novas aplicações
 - Manipulação de grandes quantidades de dados
 - Dados complexos, processos de inferência mais complexos
 - Uso de informações contextuais



Waze

- Auxílio da informação de trânsito
- Dados dos usuários auxiliam a detecção de pontos de gargalo
 - Passivamente
 - Ativamente
- 65 milhões de usuários ativos
- Número de editores voluntários
 - 420K



Inloco media

- Marketing georeferenciado
- Sensores: acelerômetro, magnetômetro, wifi
- Precisão de 3mts indoor
- 70 milhões na américa latina
- Valor de mercado:
R\$ 100 Mi

05. IN LOCO O PODER DA LOCALIZAÇÃO

Através de nossa tecnologia exclusiva de localização, mapeamos mais de 5 milhões de estabelecimentos comerciais no Brasil, incluindo mapas internos como Shoppings e Aeroportos. Tudo para que você possa engajar sua audiência em todos os momentos de consumo, através das **segmentações exclusivas** da In Loco:



Antes: pre-targeting

Segmento o consumidor mais propenso a comprar seu produto através do seu comportamento online e offline.



Durante: in loco

Influencie o seu consumidor em seu momento de decisão, dentro do seu ponto de venda, com até 1 metro de precisão.



Depois: retargeting

Atinja o consumidor que esteve em contato com sua marca recentemente. Não seja esquecido!



inlocimedia

Tendências

- Maior uso de aprendizado de máquina e inteligência artificial
 - Intelligent apps
- Mais BigData
- Crowdsensing, Crouwdsourcing
- Integração com dispositivos wearable
 - Internet of Things
- Maior demanda por recursos
- Maior uso de energia



Conhecem?!



+



Google Docs

Computação em Nuvem

"A Nuvem é um grande repositório de recursos virtualizados facilmente utilizáveis e acessíveis (como hardware, plataformas de desenvolvimento e/ou serviços). Esses recursos podem ser dinamicamente reconfigurados para ajustar a carga (escala) variável do sistema, permitindo também um uso ótimo dos recursos. Esse reservatório de recursos é geralmente explorado por um modelo pay-per-use (pagar para usar) no qual as garantias são oferecidas por um Provedor de Infraestrutura por meio de SLAs (Service Level Agreements - Acordo de Nível de Serviço)"

Vaquero, L.M. and Rodero-Merino, L. and Caceres, J. and Lindner, M. "A break in the clouds: towards a cloud definition" em ACM SIGCOMM Computer Communication Review, 2008

Na nuvem

- Aplicações como Serviços (Software como Serviço)
- Virtualização
- Elasticidade
- Pagamento por Uso



Cloud Computing X Mobile Computing

Computação em Nuvem

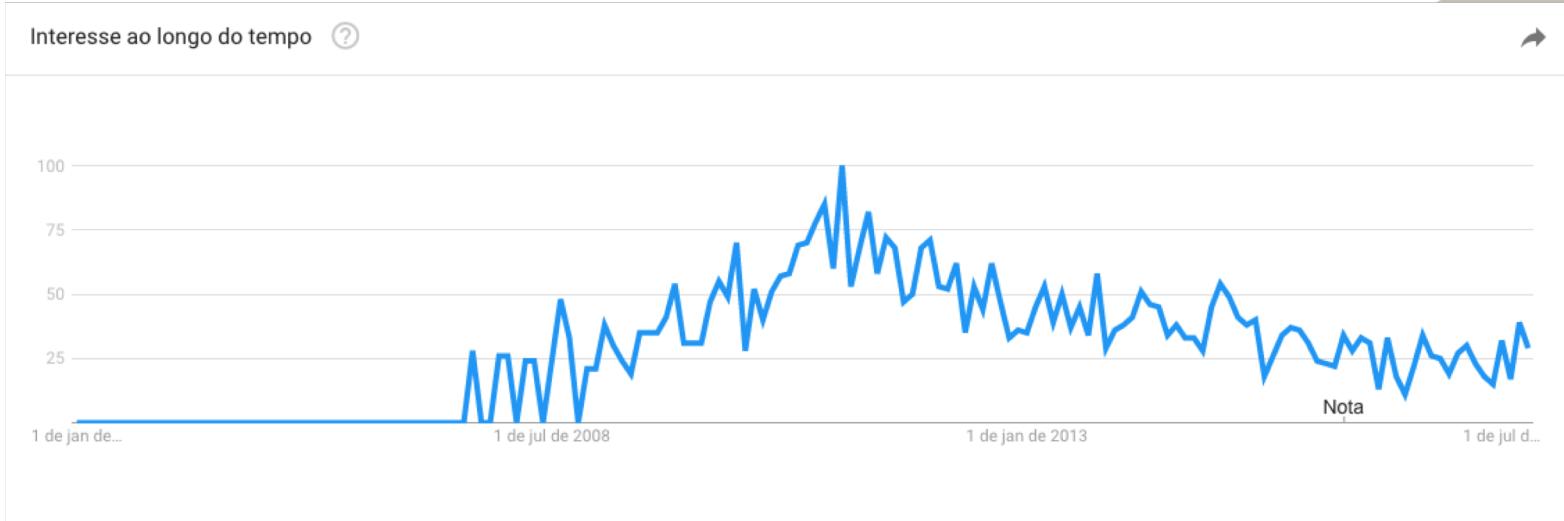
Recursos Infinitos
Elasticidade

Computação Móvel

Dispositivos mais limitados
Consumo de Energia

**Mobile Cloud
Computing**

Mobile Cloud Computing

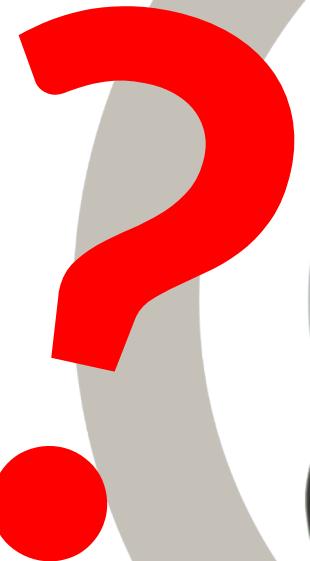


“... aiming to provide a range of services, equivalents to the cloud, adapted to the capacity of resource-constrained devices, besides performing improvements of telecommunications infrastructure in order to improve the service provisioning.”

H. T. Dinh, C. Lee, D. Niyato, and P. Wang. A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless Communications and Mobile Computing*, 2011

Sendo assim...

Como a Computação Móvel
em Nuvem pode trazer
benefícios para o tratamento
de aplicações móveis sensíveis
a contexto?



Chances de Integração...

- Migrar processamento da inferência contextual
 - Melhoria de Desempenho
 - Economia de Energia
- Cenários
 - Crowdsensing
 - Health Care
 - IoT
 - Virtual Reality



Exemplo

Google's translation headphones are here, and they're going to start a war

Nigel Kendall

Douglas Adams' in-ear translator from Hitchhiker's Guide is becoming a reality. But knowing what people are really saying could be disastrous

9295 801

Friday 6 October 2017
14.00 BST



Advertisement

Google Cloud Platform

Receba US\$300 de crédito para testar hoje mesmo.

TESTE GRÁTIS

Google Cloud

https://adclick.g.doubleclick.net/pcs/click?xai=AKAOjstPsvf4SBK-sHjabcPUGzNg5qKwlVnWyVEJuPCKAjGfEhEmUS24-2br2EX8SwtDuhGhPyhP_zDWuqeXjVHJo280qXrYbU23XHLVrC7wG0GV89qpsEAA&sig=Cg0ArKJSzBjWOFkJtUCH&urifix=..

Minicurso

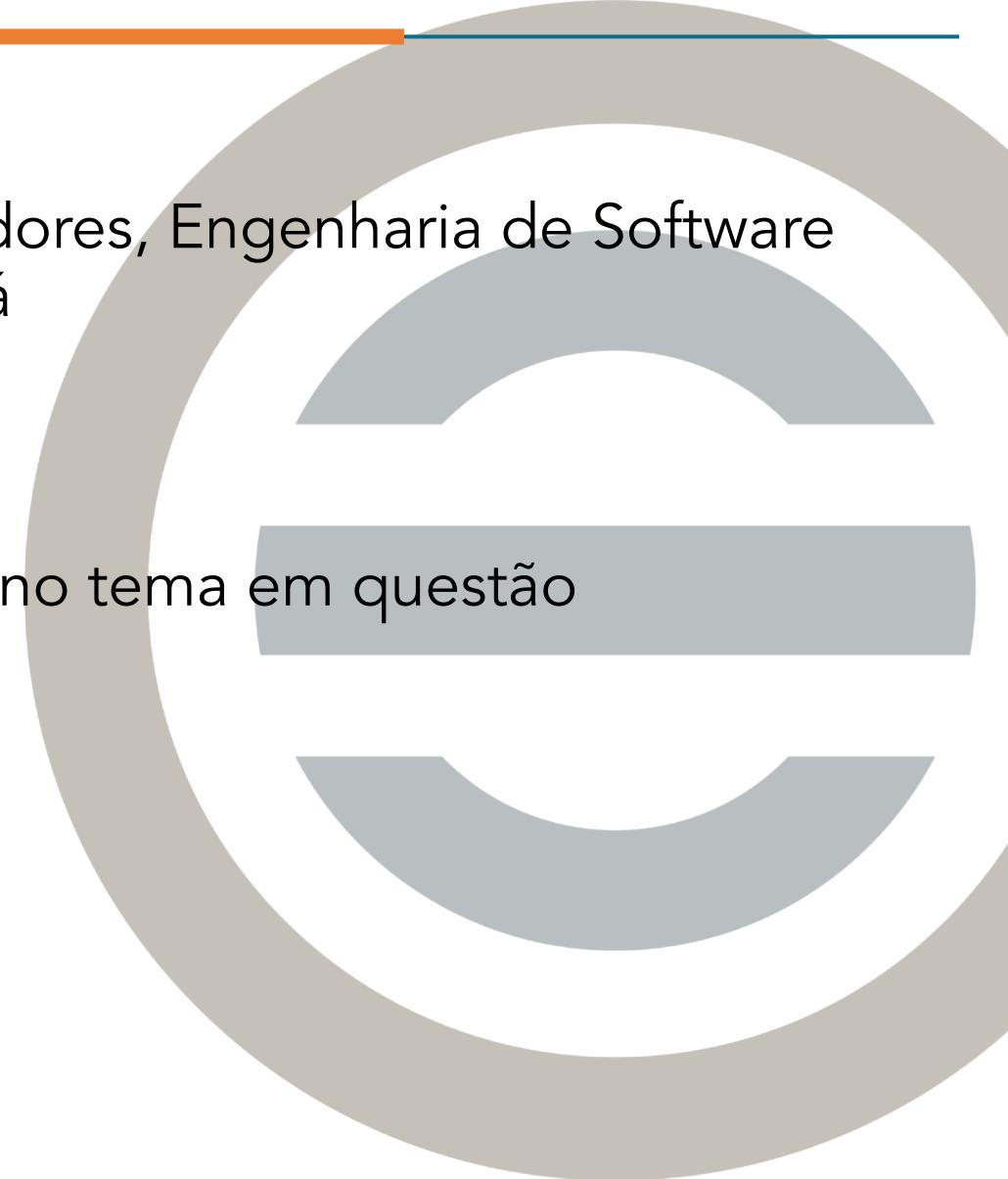
- Apresentar conceitos básicos de Mobile Cloud Computing (MCC) e Context-Aware Mobile Computing (CAM)
 - Context-Aware Mobile Computing
 - Mobile Cloud Computing
- Integração MCC e CAM
- Proposta CAOS
 - Hands on
- Oportunidades de Pesquisa



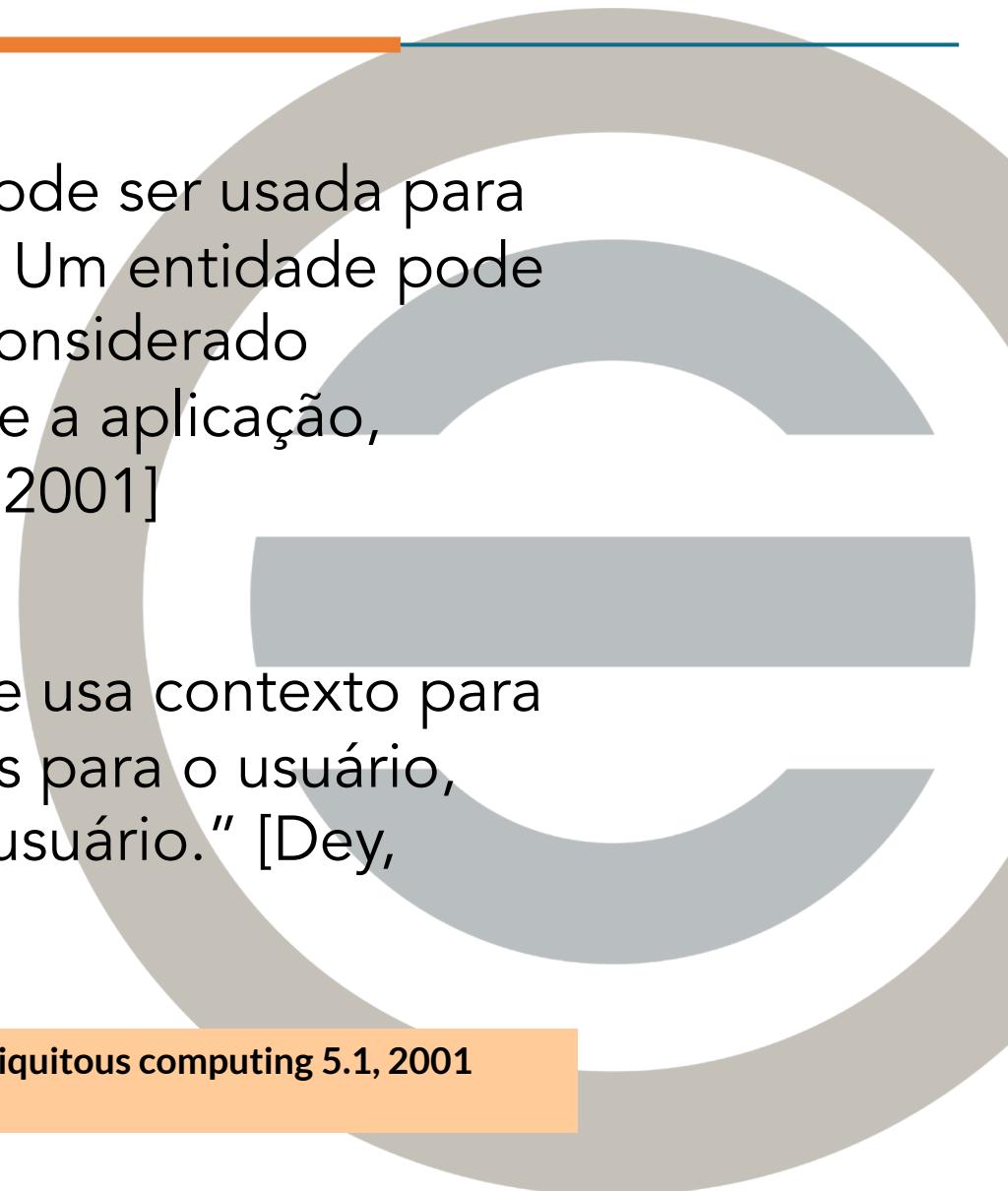
Arquivos do Minicurso:
<https://goo.gl/JtzG99>

Quem somos?

- GREat/UFC (great.ufc.br)
 - Grupo de Pesquisa em Redes de Computadores, Engenharia de Software e Sistemas – Universidade Federal do Ceará
- Pesquisas em MCC desde 2014
 - Artigos, dissertações e teses de doutorado no tema em questão
- Três membros no minicurso:
 - Fernando Trinta
 - Paulo Antonio Leal Rego
 - Francisco Anderson Almada



Context-Aware Mobile Computing (CAM)



“Contexto é qualquer informação que pode ser usada para caracterizar a situação de uma entidade. Um entidade pode ser uma pessoa, lugar ou objeto que é considerado relevante para interação entre o usuário e a aplicação, incluindo o usuário e a aplicação.” [Dey, 2001]

“Um sistema é sensível ao contexto se ele usa contexto para prover informação relevante e/ou serviços para o usuário, onde a relevância depende da tarefa do usuário.” [Dey, 2001]

Dey, Anind K. "Understanding and using context." Personal and ubiquitous computing 5.1, 2001

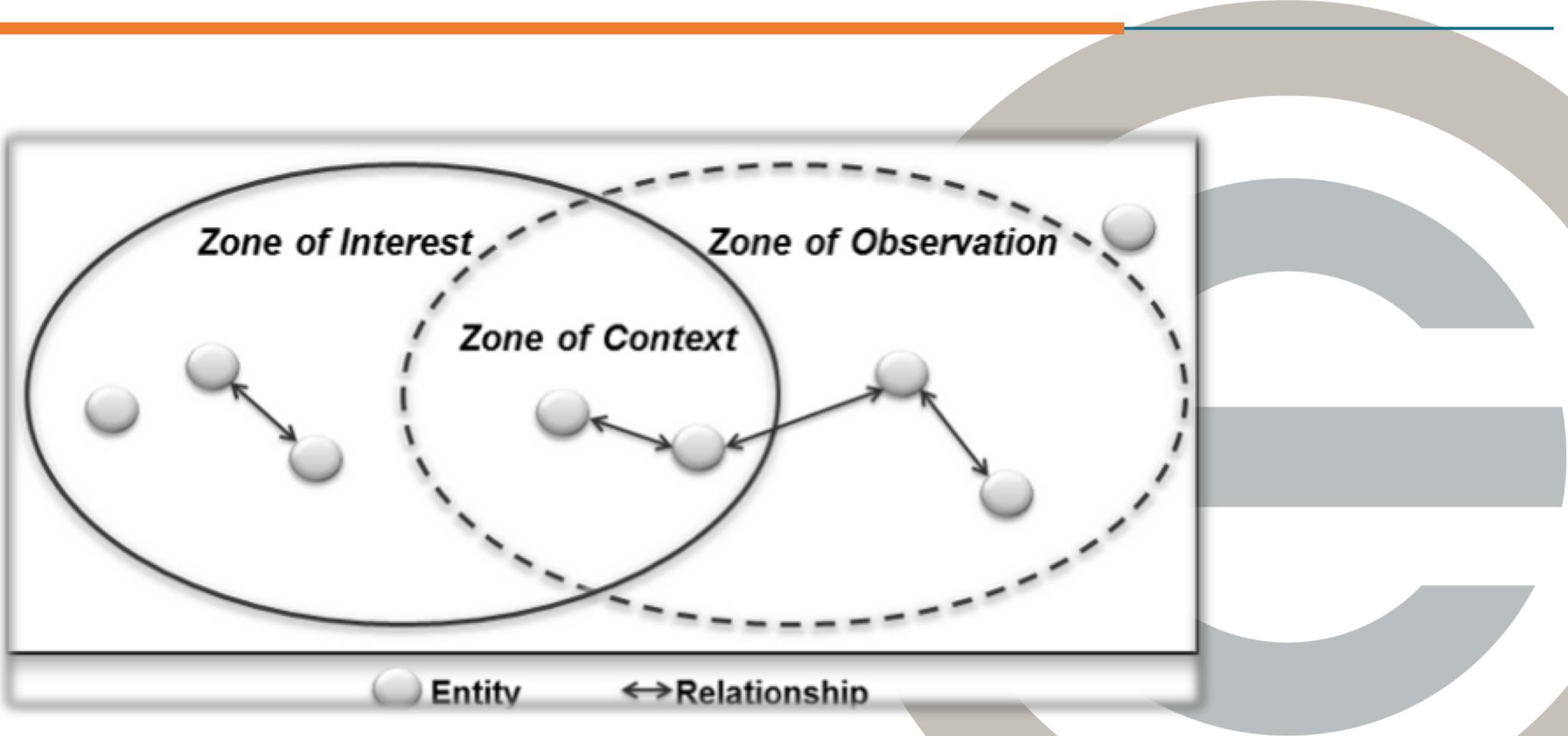
Context-Aware Mobile Computing (CAM)

“Os elementos que compõem o contexto dependem da sua relevância para o sistema, de ter consciência deles e da possibilidade de observá-los.” [Viana, 2011]

W. Viana, A. D. Miron, B. Moisuc, J. Gensel, M. Villanova-Oliver, and H. Martin, “Towards the semantic and context-aware management of mobile multimedia” *Multimedia Tools Appl.*, vol. 53, no. 2, pp. 391–429, Jun. 2011

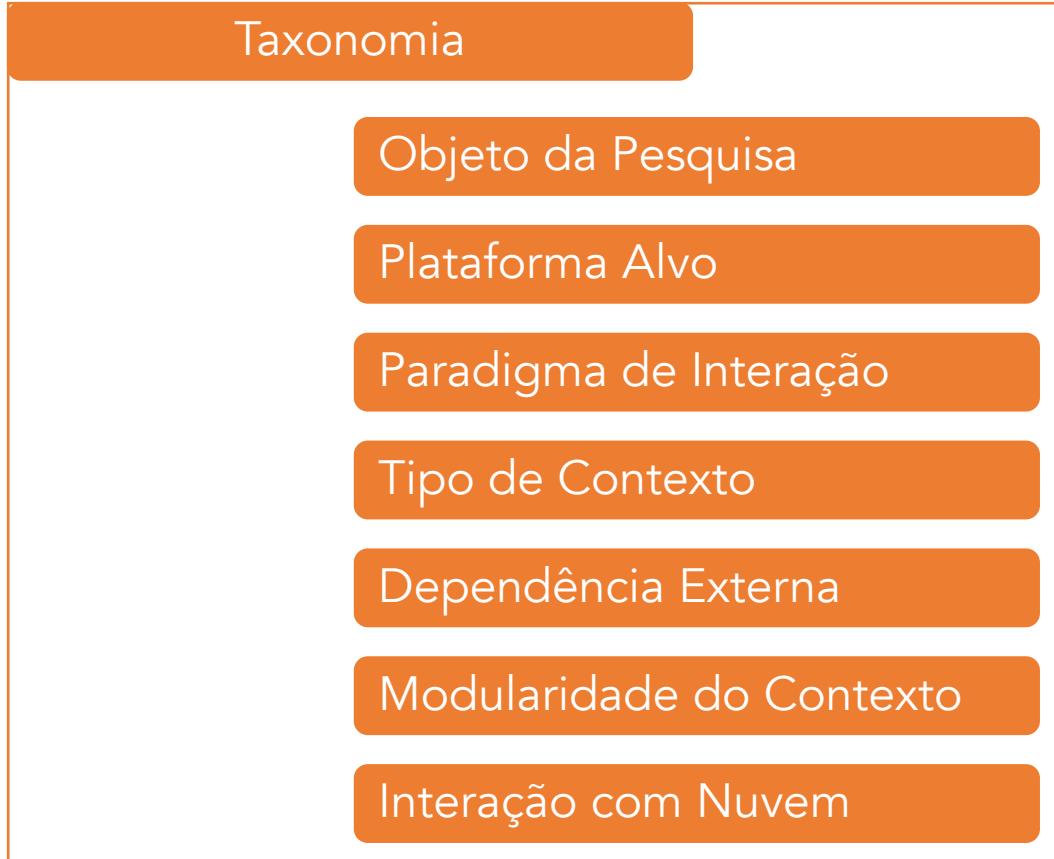
- De forma geral, a sensibilidade ao contexto pode ser entendida como a habilidade do sistema perceber e reagir a mudanças no seu ambiente de acordo com o contexto em que está inserido.

Definição de Contexto [Viana, 2011]



W. Viana, A. D. Miron, B. Moisuc, J. Gensel, M. Villanova-Oliver, and H. Martin, "Towards the semantic and context-aware management of mobile multimedia" *Multimedia Tools Appl.*, vol. 53, no. 2, pp. 391–429, Jun. 2011

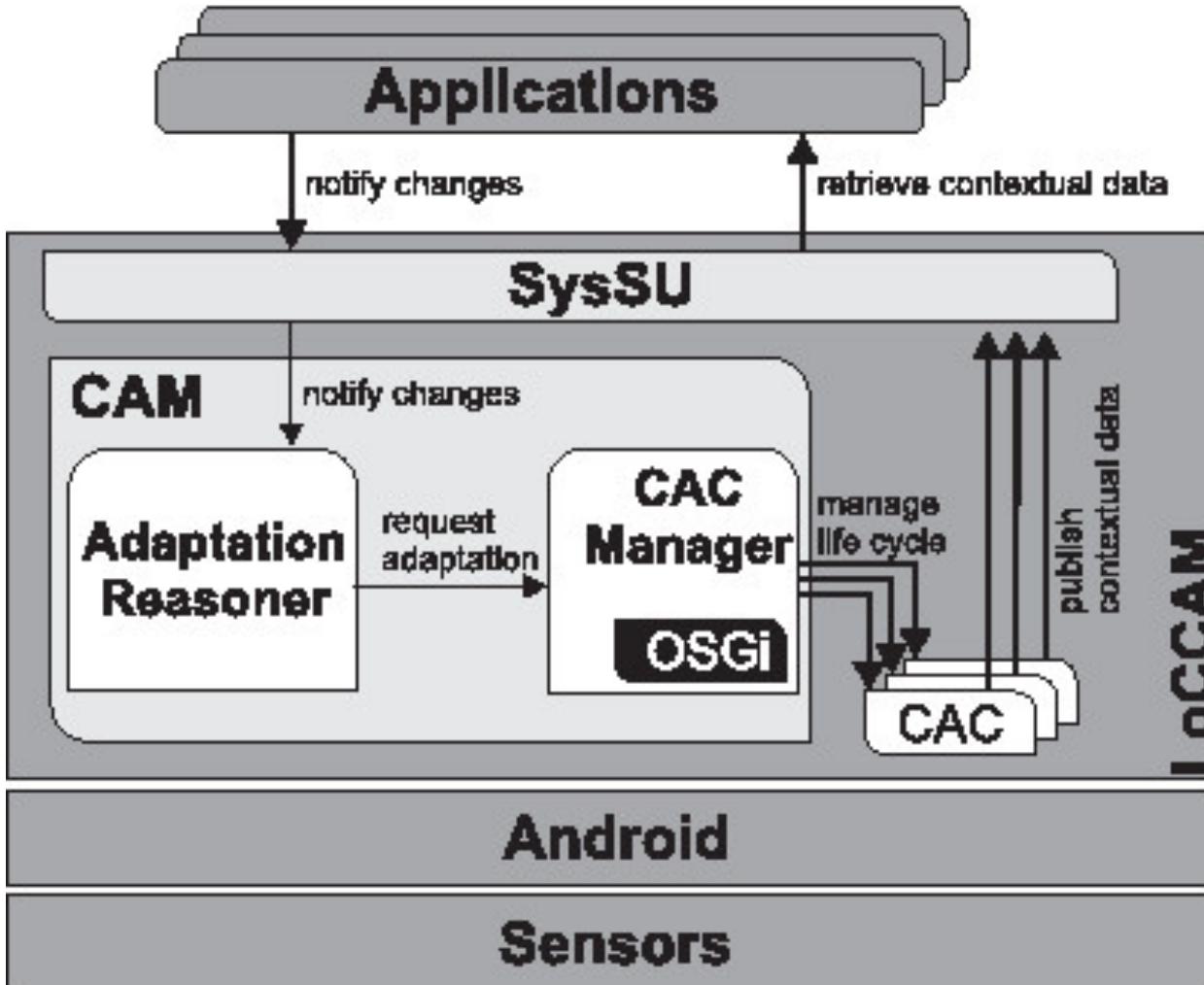
Soluções Existentes: Taxonomia



Algumas Soluções Existentes

Artigo	Objeto da Pesquisa	Plataforma Alvo	Paradigma de Interação	Tipo de Contexto	Dependência	Modularidade	Nuvem
[Mane and Surve 2016]	Middleware	Android	Requisição/Resposta	Dispositivo Físico Usuário Temporal	-	Sim	Não
[Da et al. 2014a]	Framework	Android Java SE	Requisição/Resposta Publish/Subscribe	Dispositivo Físico Usuário Temporal	-	Sim	Não
[Da et al. 2014b]	Middleware	Android Java SE	Requisição/Resposta Publish/Subscribe	Dispositivo Físico Usuário Temporal	-	Sim	Não
[Williams and Gray 2014]	Framework	Android	Publish/Subscribe	Dispositivo	-	Sim	Não
[Chihani et al. 2013]	Framework	RESTful	Requisição/Resposta Publish/Subscribe	Físico	-	Sim	Não
[Buthpitiya et al. 2012]	Framework	Android	Espaço de Tuplas	Físico	-	Sim	Sim
[Carlson and Schrader 2012]	Framework	Android	Publish/Subscribe	Dispositivo Físico Usuário Temporal	OSGI	Sim	Não

LoCCAM (Loosely Coupled Context Acquisition Middleware)



- CAM
 - Adaptation Reasoner
 - CAC Manager
 - OSGi
 - CACs
 - Físico
 - Lógico
 - Virtual
- SysSU
 - Espaço de Tuplas
 - Request/Response e Pub/Sub
 - Interoperabilidade

LoCCAM

- Modelo de Contexto
 - ContextKey: Corresponde ao tipo de informação contextual
 - Source: Informa a fonte da informação
 - Values: O valor da informação contextual
 - Accuracy: A precisão do sensoreamento
 - Timestamp: O instante em que a informação contextual foi capturada



LoCCAM

- Exemplo
 - ContextKey: context.device.location
 - Source: Physic
 - Values: [-35.456,45.678]
 - Accuracy: 10.0
 - Timestamp: 84586594022123



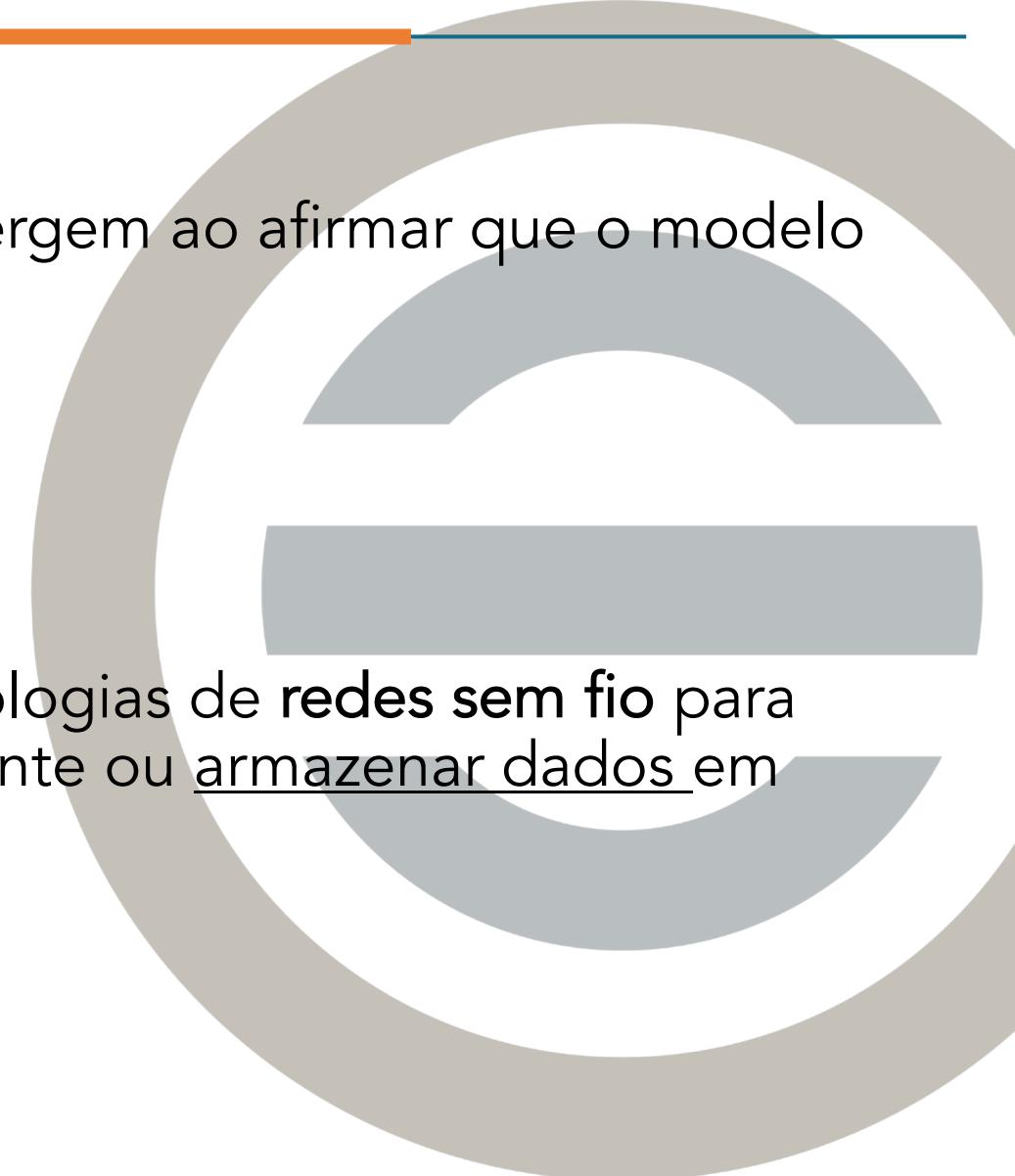
Mobile Cloud Computing

- Definição
 - É um novo paradigma que incorpora três tecnologias (**computação móvel, computação em nuvem e redes de comunicação**) e visa reduzir as limitações dos dispositivos móveis por meio do acesso ubíquo a recursos locais e da nuvem.
 - Tais recursos são utilizados para:
 - Melhorar o desempenho das aplicações
 - Conservar recursos dos dispositivos
 - Estender a capacidade de armazenamento
 - Melhorar a experiência do usuário



Mobile Cloud Computing

- Definição
 - Existem várias outras definições. Elas convergem ao afirmar que o modelo MCC é composto de:
 - Dispositivos móveis
 - Redes sem fio (wireless)
 - Servidores remotos
 - Onde os **dispositivos móveis** usam as tecnologias de **redes sem fio** para executar tarefas pesadas computacionalmente ou armazenar dados em **servidores remotos**.



Mobile Cloud Computing

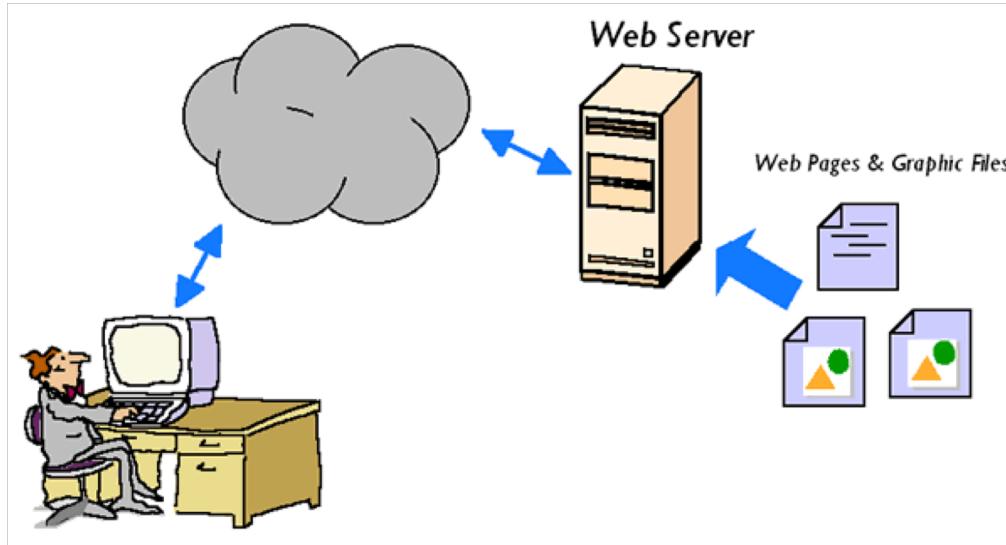
- Offloading
 - É a técnica utilizada em MCC para melhorar o desempenho das aplicações e reduzir o consumo de energia dos dispositivos móveis ao migrar processamento ou dados de um dispositivo móvel para outro dispositivos/infraestrutura com maior poder de computação e/ou armazenamento.
 - *Computation Offloading*
 - *Data Offloading*
 - O termo *Cyber Foraging* também é utilizado na literatura para falar de Computation Offloading

Satyanarayanan, Mahadev. "Pervasive computing: Vision and challenges." IEEE Personal communications 8.4, 2001.

Balan, Rajesh Krishna, and Jason Flinn. "Cyber Foraging: Fifteen Years Later." IEEE Pervasive Computing 16.3, 2017.

Mobile Cloud Computing

- Migrar computação para outra máquina não é uma ideia nova.



- Computation Offloading é diferente do modelo cliente-servidor.
 - Cliente-servidor: cliente **sempre** migra a computação para o servidor.
 - Offloading: dependendo da disponibilidade e qualidade da rede, offloading pode não ser possível ou vantajoso.

Mobile Cloud Computing

- O conceito de offloading é geralmente implementado por uma estrutura especial (e.g., design pattern) que permite que uma parte do código seja executada localmente no dispositivo móvel ou remotamente, **sem impactar na corretude da aplicação**.
- Tipos de aplicações móveis:
 - Offline ou nativas
 - Online
 - Híbrida



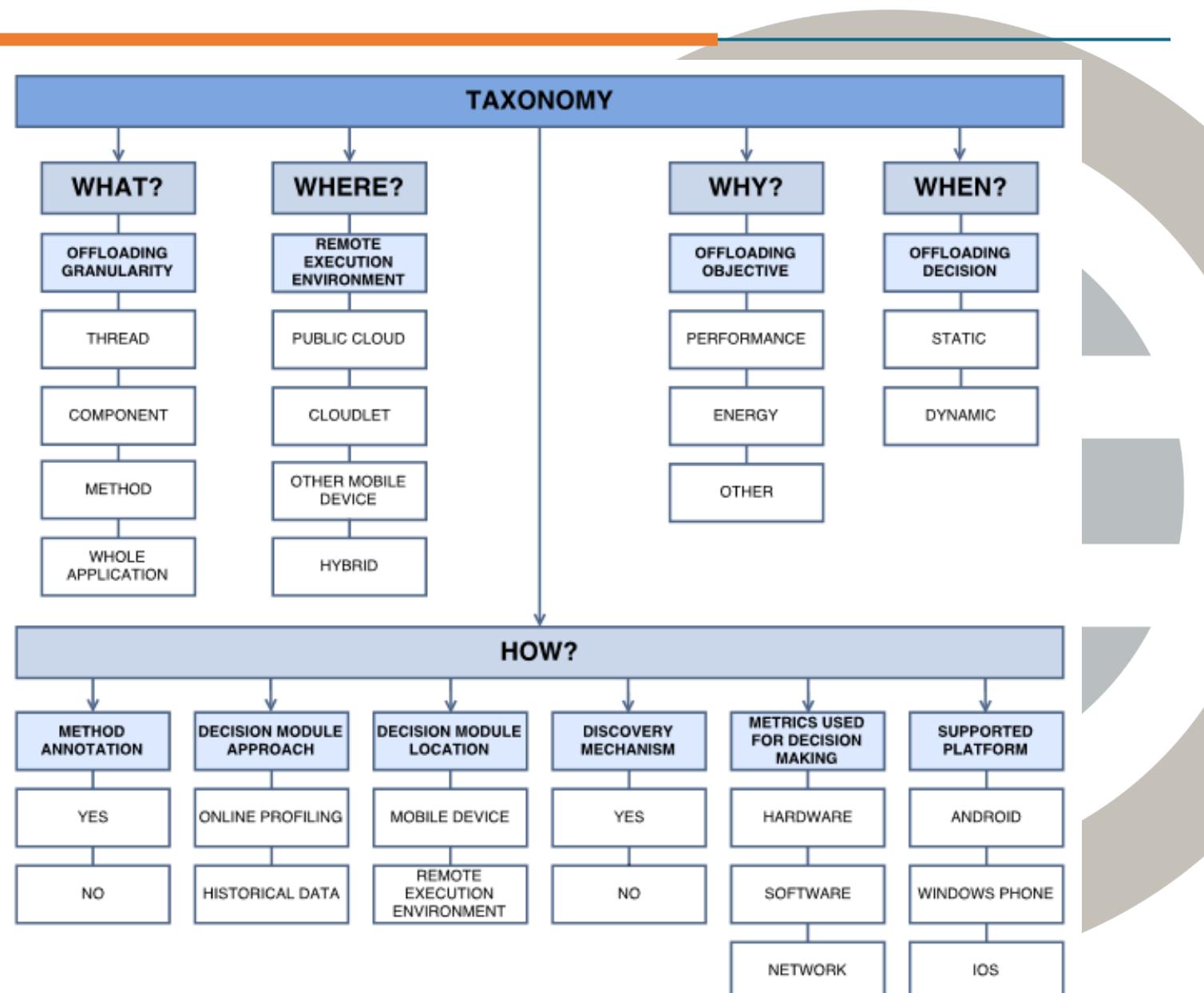
Mobile Cloud Computing

- Fazer offloading:
 - Quando?
 - Como?
 - O que?
 - Onde?
 - Por que?



Mobile Cloud Computing

- Revisão de literatura
- Taxonomia inspirada nos trabalhos de:
 - Sharifi et al. (2012)
 - Liu et al. (2015)



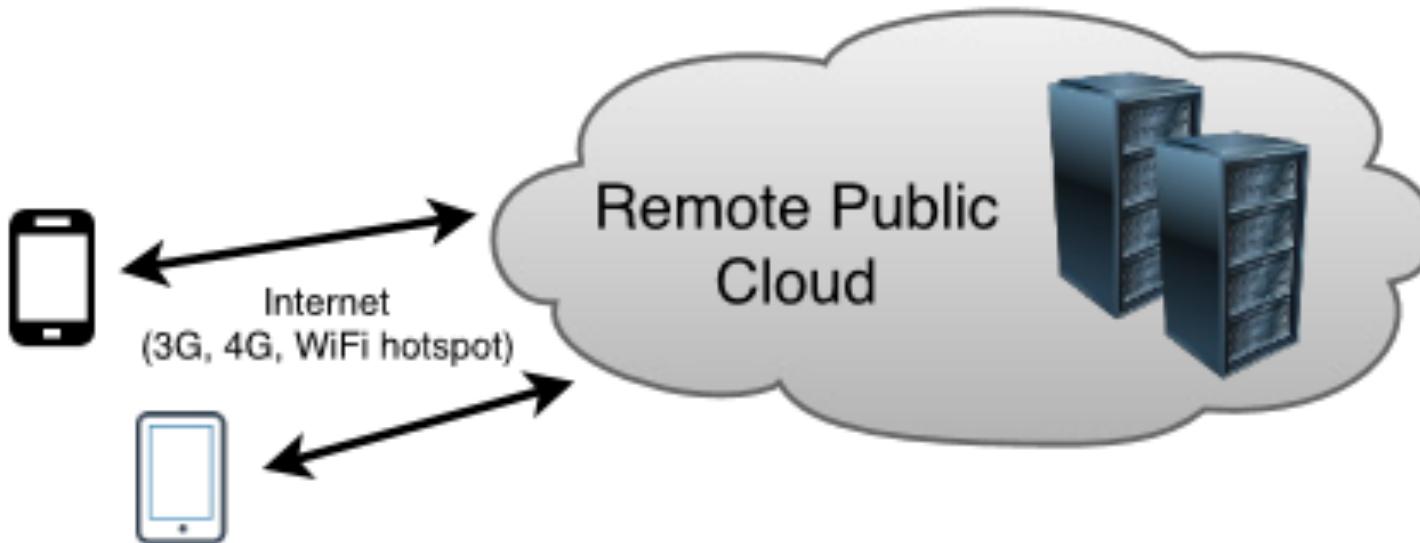
Mobile Cloud Computing

- Por que fazer offloading?
 - Melhorar o desempenho das aplicações
 - Reduzir o consumo de energia do dispositivo móvel
 - Outros motivos. Por exemplo:
 - Estender recursos de armazenamento do dispositivo móvel
 - Compartilhar recursos



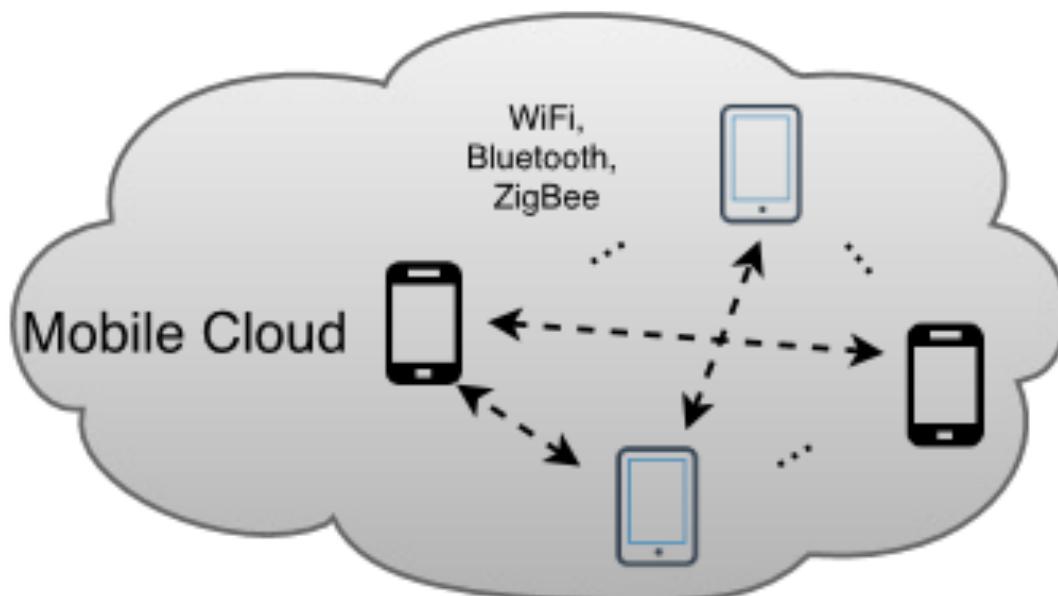
Mobile Cloud Computing

- Onde fazer offloading?
 - Nuvem Pública



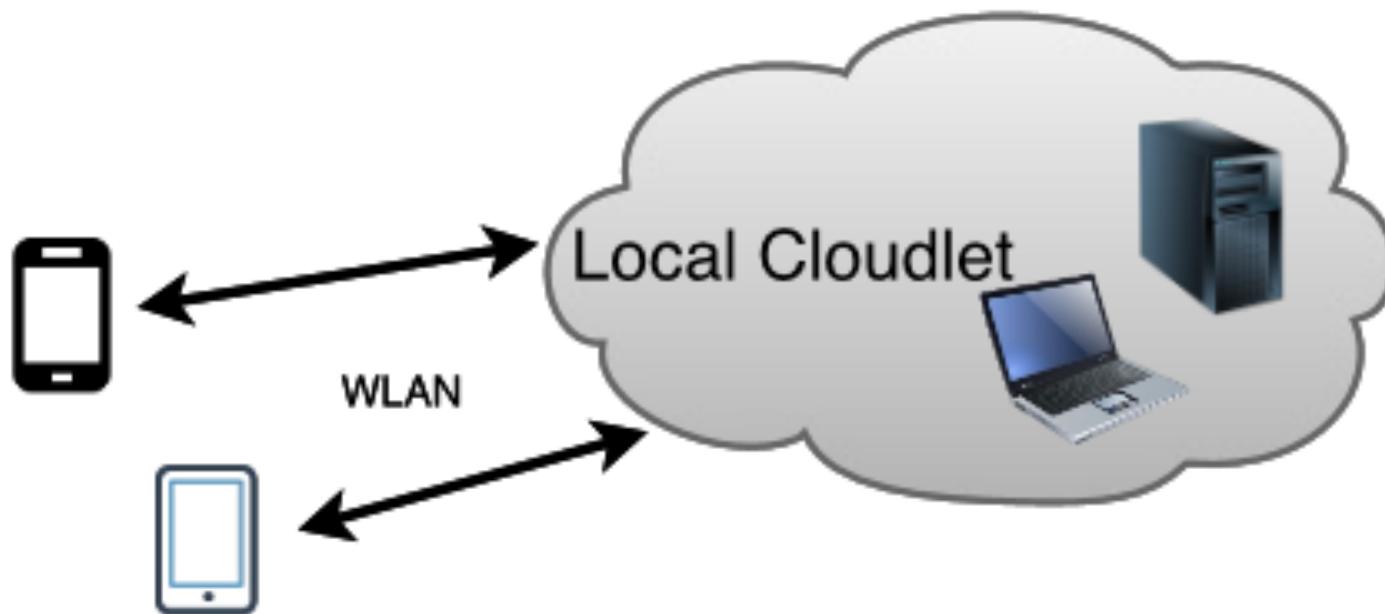
Mobile Cloud Computing

- Onde fazer offloading?
 - Outro dispositivo móvel



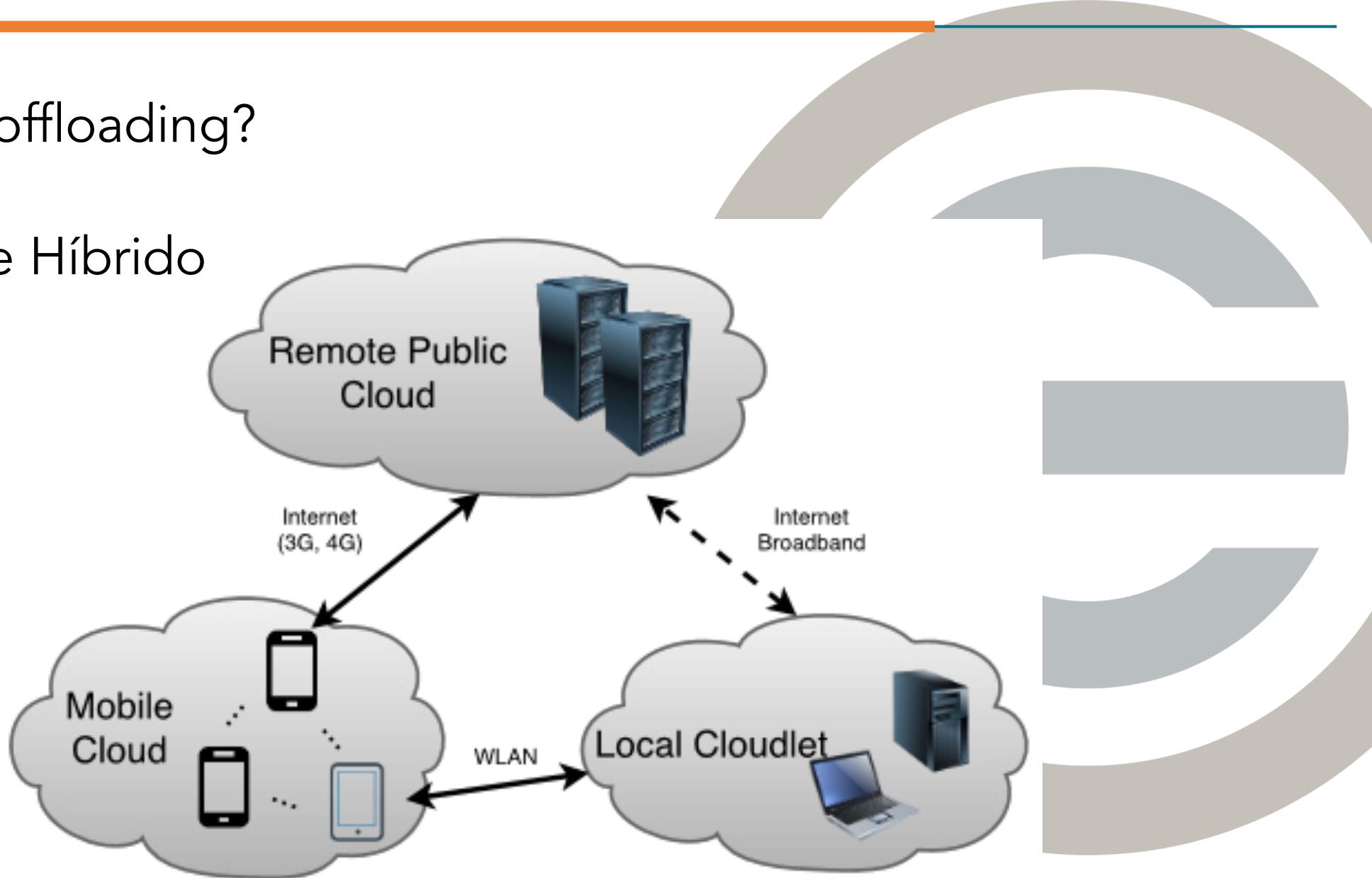
Mobile Cloud Computing

- Onde fazer offloading?
 - Cloudlet [Satyanarayanan et al. 2009]



Mobile Cloud Computing

- Onde fazer offloading?
 - Ambiente Híbrido



Mobile Cloud Computing

- Fazer offloading de quê?
 - Métodos
 - Componentes
 - Threads
 - Aplicação Inteira



Mobile Cloud Computing

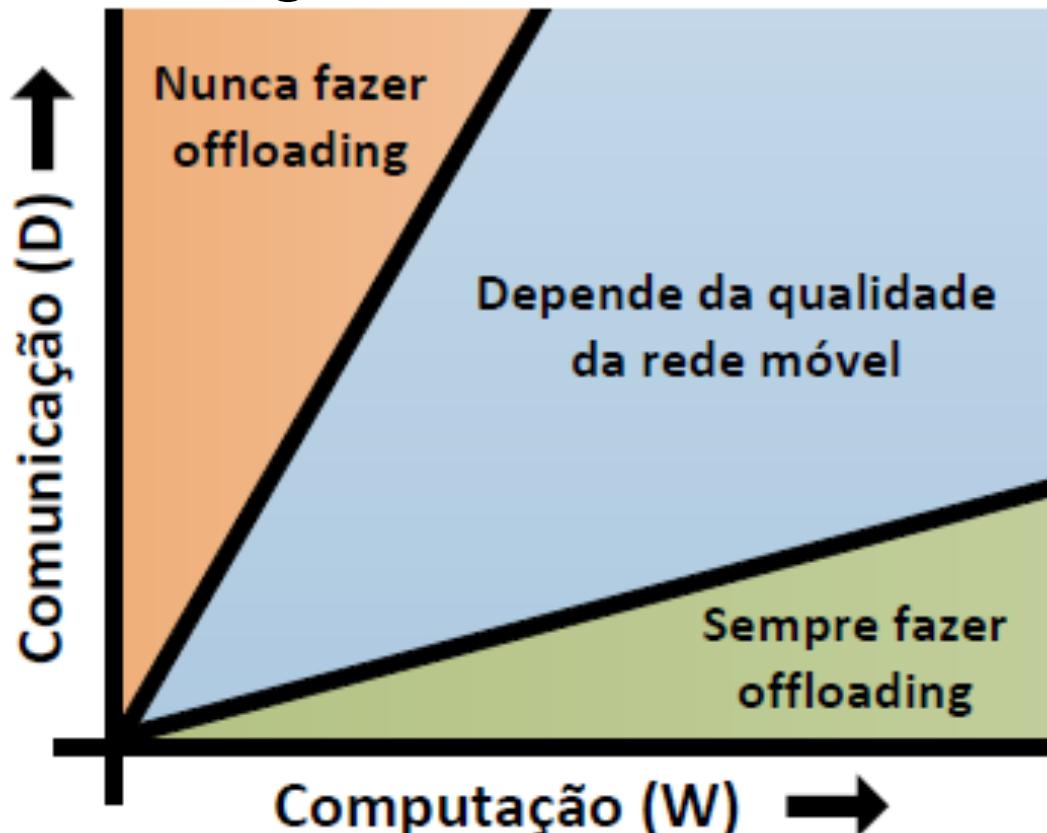
- Quando fazer offloading?
 - “Quando a técnica de offloading aumenta o desempenho do dispositivo móvel?”. [Kumar et al., 2013]

$$\frac{W}{P_m} > \frac{D_u}{T_u} + \frac{W}{P_c} + \frac{D_d}{T_d}$$

Kumar, K., Liu, J., Lu, Y.-H., and Bhargava, B. A survey of computation offloading for mobile systems. *Mobile Networks and Applications*, 2013.

Mobile Cloud Computing

- Quando fazer offloading?



Trade-off de quando fazer offloading para melhorar o desempenho.

Kumar, K. and Lu, Y.-H. Cloud computing for mobile users: Can offloading computation save energy? Computer, 2010.

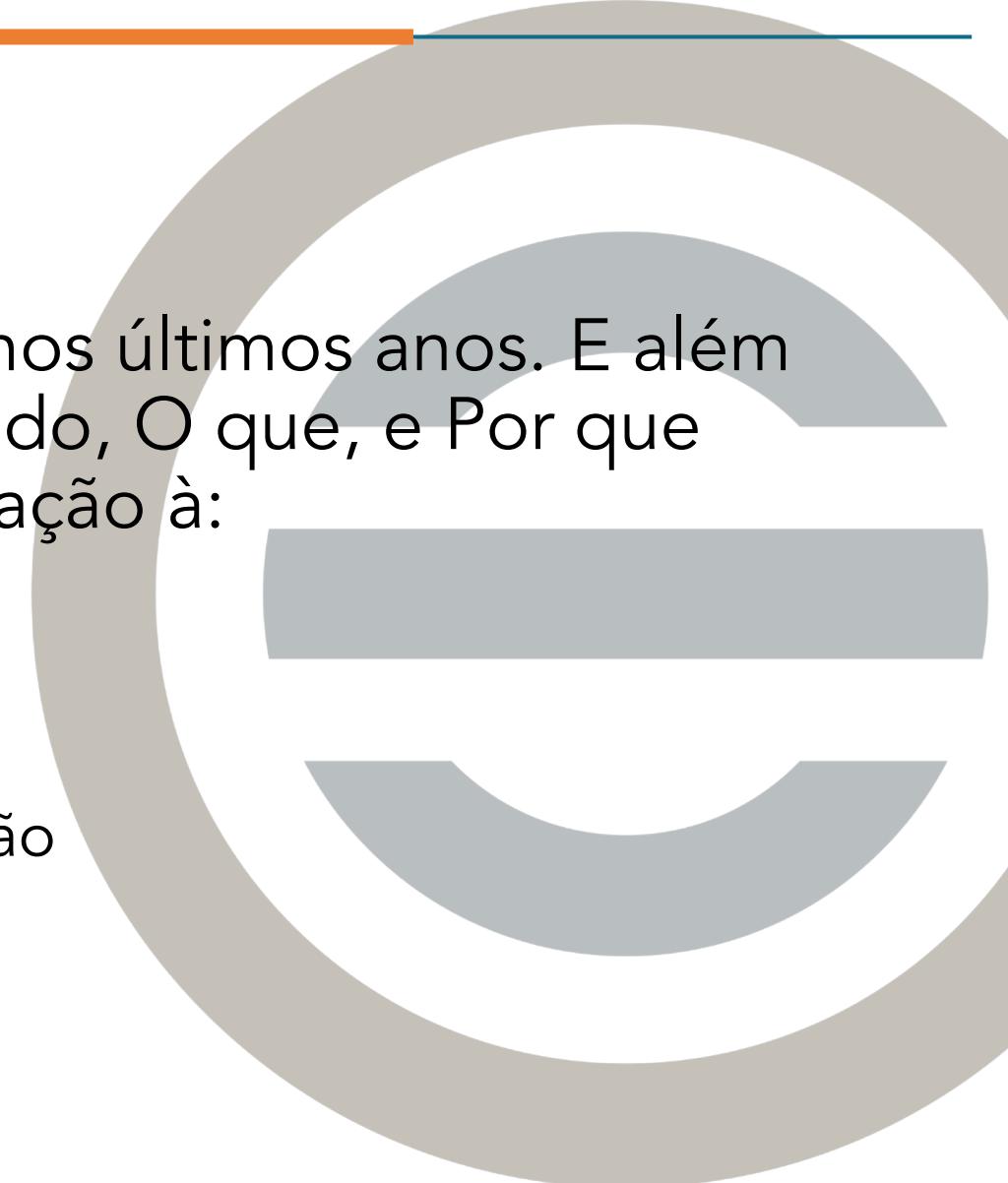
Mobile Cloud Computing

- Quando fazer offloading?
 - Offloading Estático
 - SEMPRE tenta fazer offloading
 - Offloading Dinâmico
 - Faz offloading QUANDO VALE A PENA



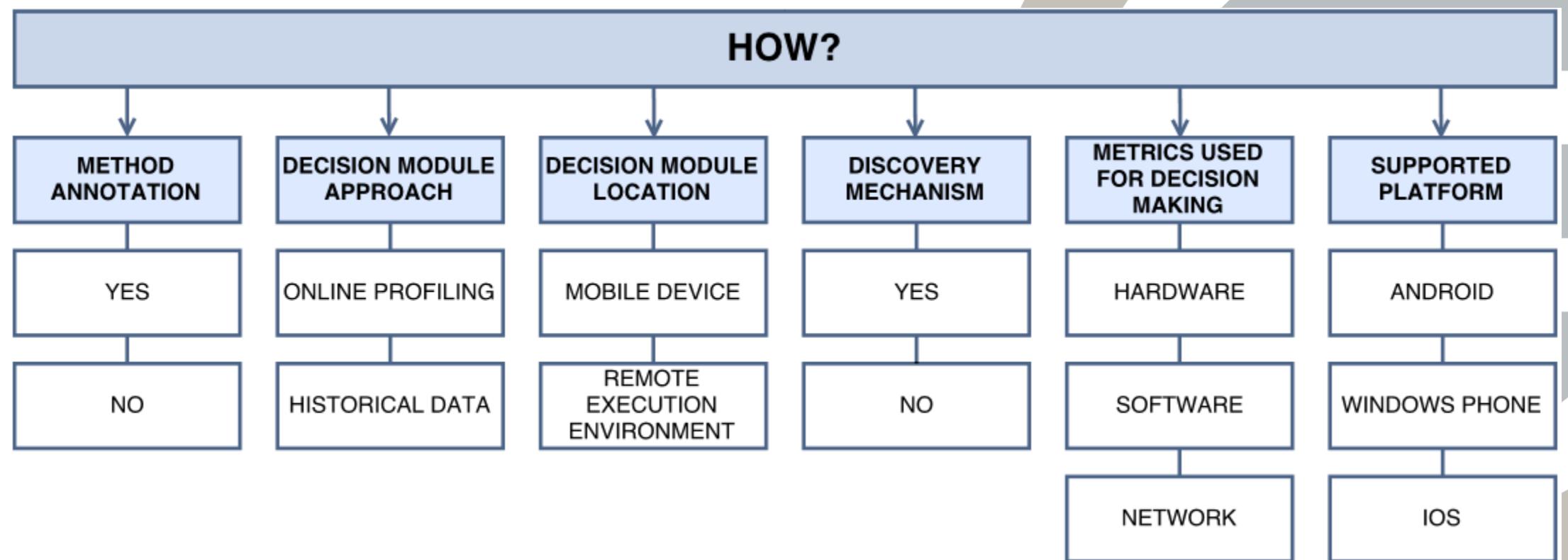
Mobile Cloud Computing

- Como fazer offloading?
 - Várias soluções têm sido propostas nos últimos anos. E além de variar com relação a Onde, Quando, O que, e Por que fazer offloading, elas variam com relação à:
 - Utilização de Anotações
 - Localização do módulo de decisão
 - Características do módulo de decisão
 - Métricas usadas para tomada de decisão
 - Plataformas suportas
 - Mecanismo de descoberta



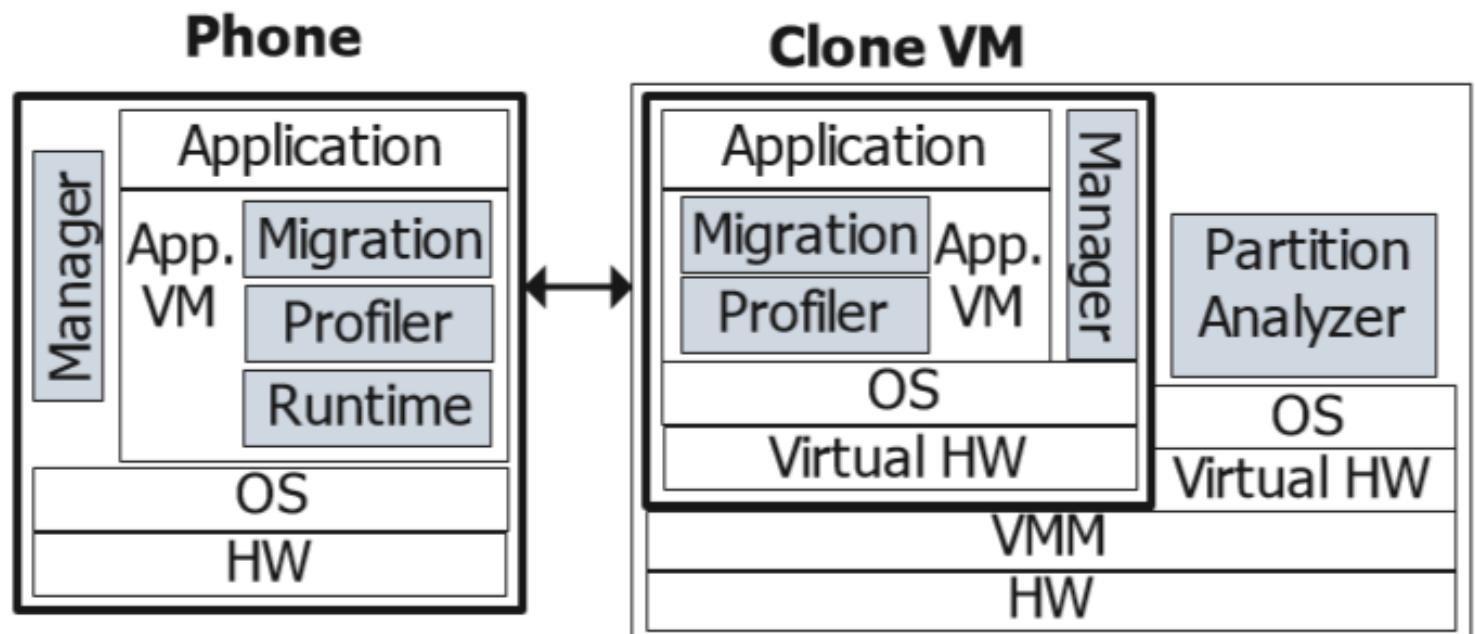
Mobile Cloud Computing

- Como fazer offloading?



Mobile Cloud Computing (Soluções)

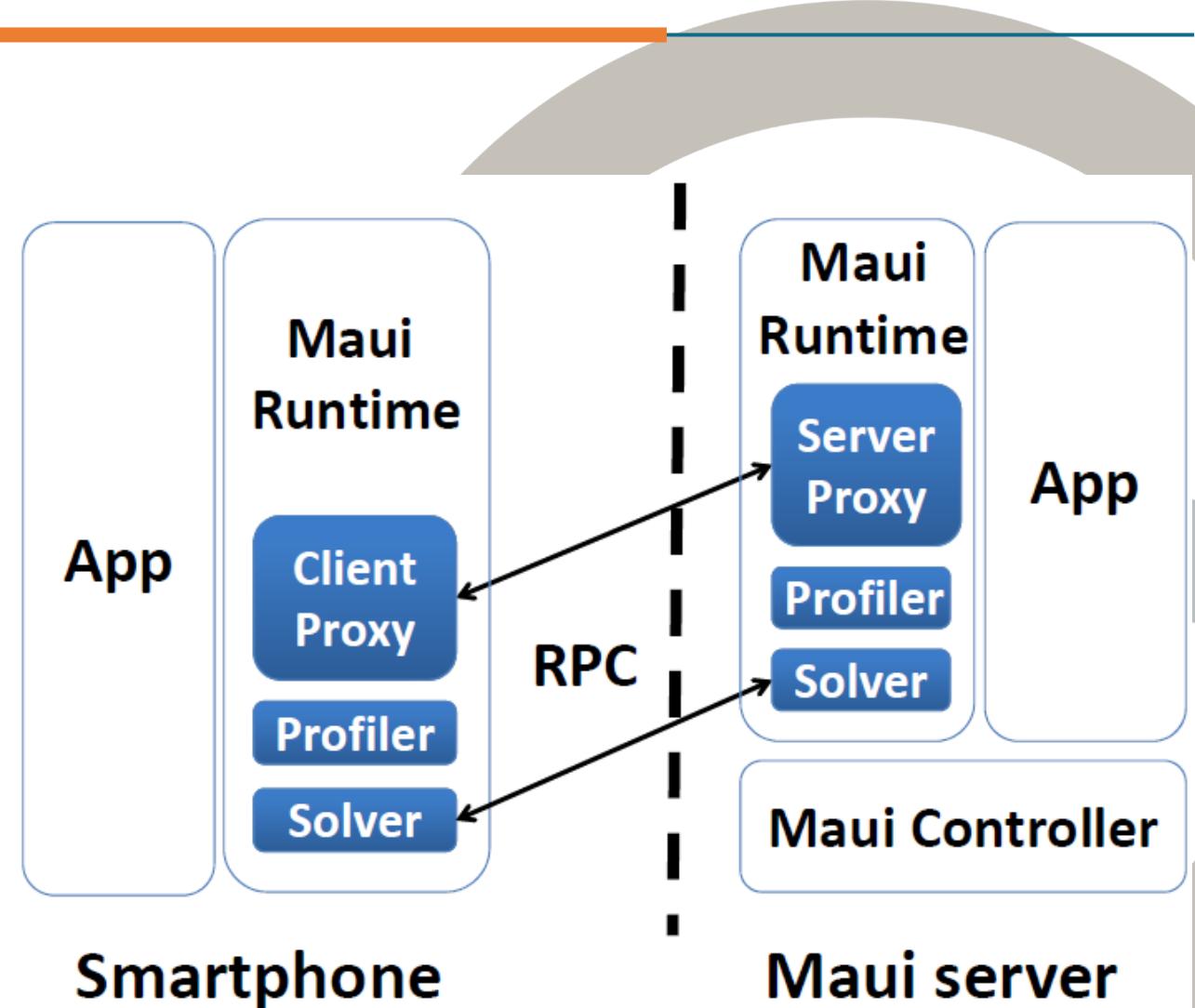
- CloneCloud
 - Máquina virtual do Android modificada para fazer offloading de threads
 - Servidor remoto (clone do smartphone) executa na Nuvem ou Cloudlet
 - Offloading dinâmico com o objetivo de melhorar o desempenho e economizar energia



Chun, Byung-Gon, et al. "Clonecloud: elastic execution between mobile device and cloud." Proceedings of the sixth conference on Computer systems. ACM, 2011.

Mobile Cloud Computing (Soluções)

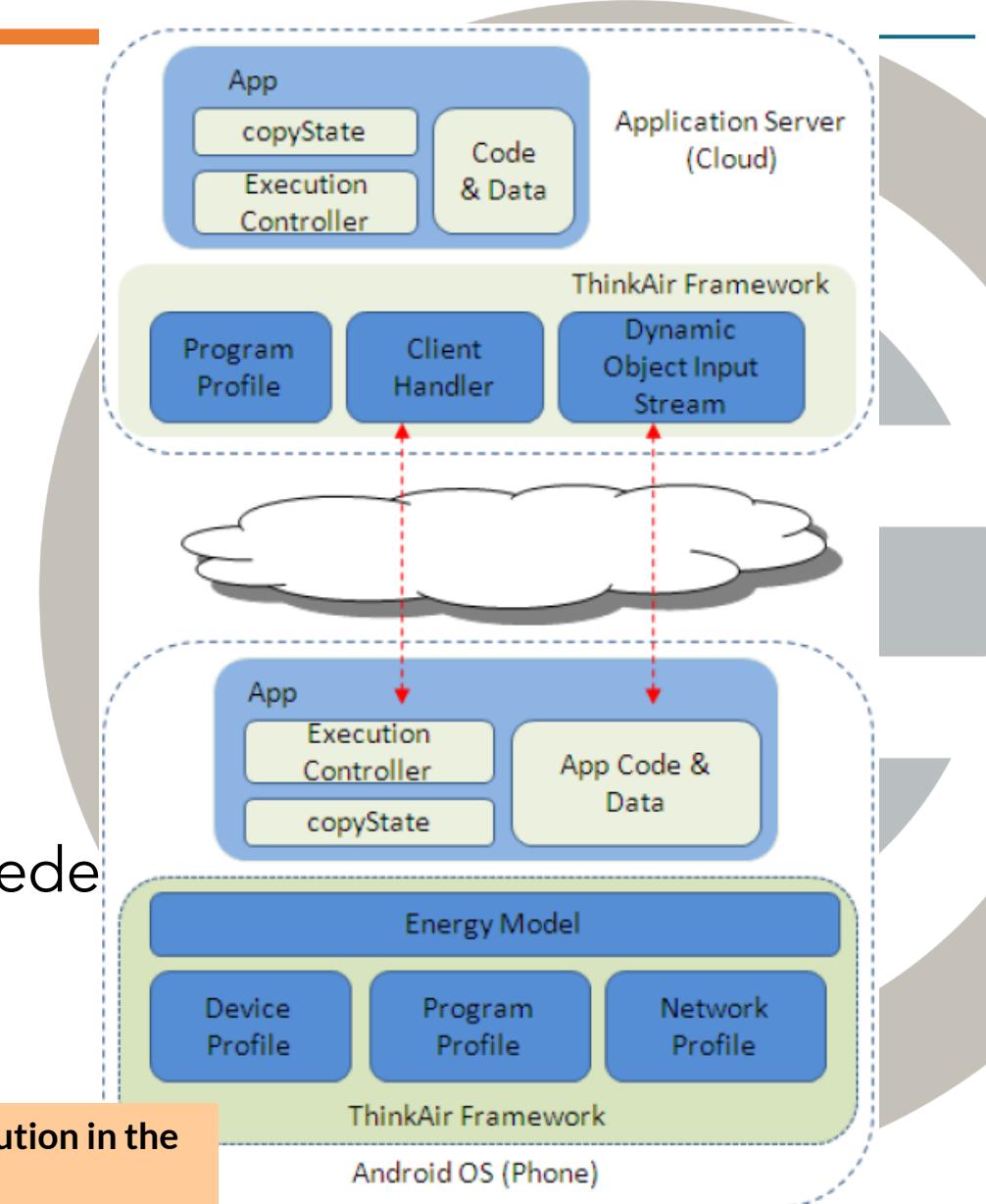
- MAUI
 - Offloading dinâmico
 - Foco em economia de energia
 - Utiliza anotações
 - Exclusivo para Windows Phone



Cuervo, Eduardo, et al. "MAUI: making smartphones last longer with code offload." Proceedings of the 8th international conference on Mobile systems, applications, and services. ACM, 2010.

Mobile Cloud Computing (Soluções)

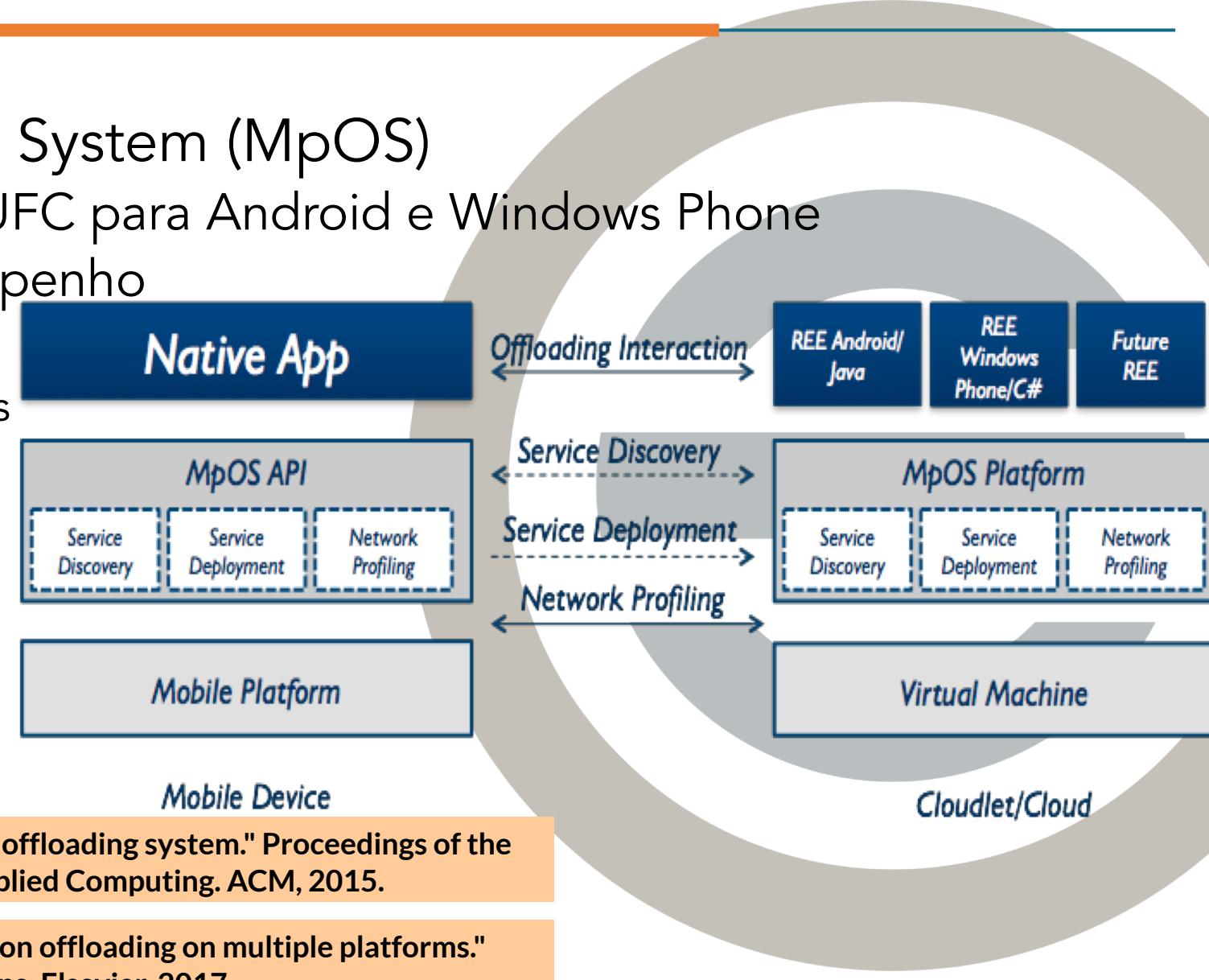
- ThinkAir
 - Offloading dinâmico
 - Utiliza anotações
 - Objetivos:
 - Economizar energia
 - Melhorar desempenho
 - Reduzir custo de utilização da nuvem
 - Exclusivo para Android
 - Tem componentes para fazer profiling de recursos do dispositivo, do aplicativo e da rede



Kosta, Sokol, et al. "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading." Infocom, IEEE, 2012.

Mobile Cloud Computing (Soluções)

- Multiplatform Offloading System (MpOS)
 - Desenvolvido no GREat/UFC para Android e Windows Phone
 - Foca no ganho de desempenho
 - Offloading dinâmico
 - Baseado em regras simples
 - É possível fazer Estático
 - Faz profiling de rede
 - Utiliza anotações
 - C# e Java



Mobile Cloud Computing (Soluções)

Solução	Offloading	Granularidade	Decisão (Objetivo)	Ambiente de Execução
μ Cloud	Estático	Componente	Energia	Nuvem
MACS	Dinâmico	Componente*	Desempenho	Nuvem ou Cloudlet
MapCloud	Estático	Componente	Desempenho	Híbrido (Nuvem e Cloudlet)
eXCloud	Dinâmico	Aplicação	Desempenho	Nuvem Cloudlet
AIOLOS	Dinâmico	Método	Desempenho Energia	Nuvem
MpOS	Dinâmico Estático	Método	Desempenho	Híbrido (Nuvem e Cloudlet)
Scavenger	Dinâmico	Método	Desempenho Energia	Nuvem ou Cloudlet
MAUI	Dinâmico	Método	Desempenho Energia	Nuvem ou Cloudlet
ThinkAir	Dinâmico	Método	Desempenho Energia, Custo Energia Energia, Custo	Nuvem ou Cloudlet Cloudlet
CloneCloud	Dinâmico*	Thread	Desempenho Energia	Nuvem ou Cloudlet
COMET	Dinâmico	Thread	Desempenho	Nuvem

Integração MCC e CAM

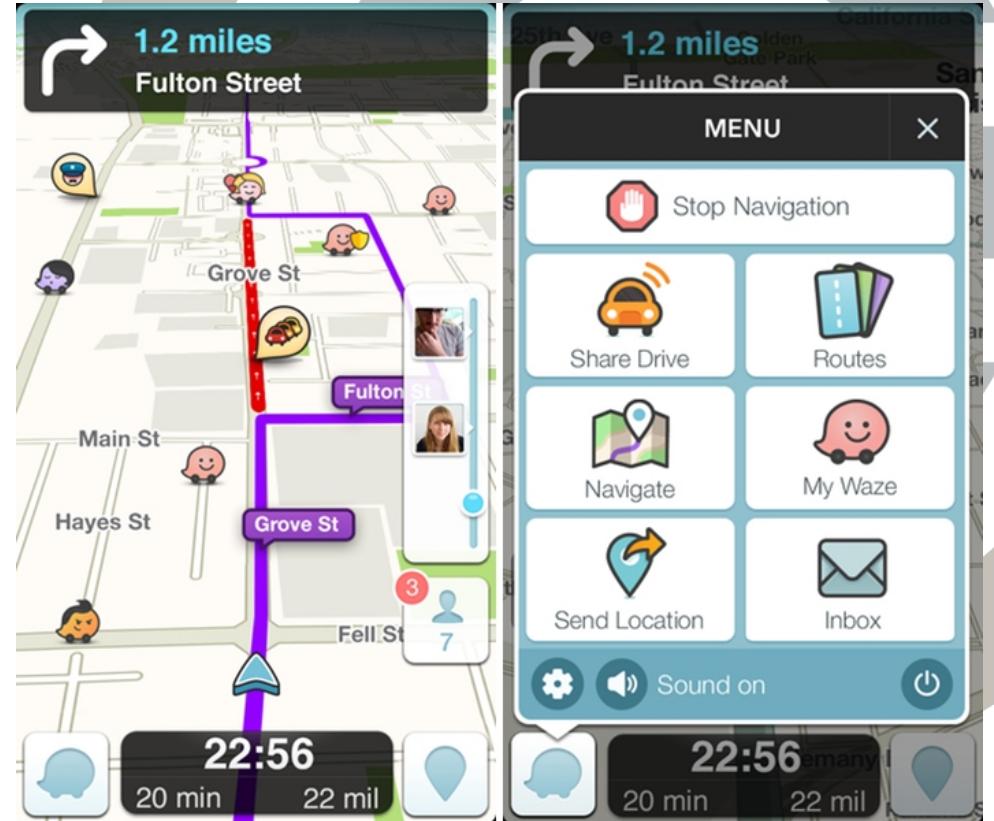
- Fazer inferência de situações envolvendo o contexto do usuário requer recursos de processamento e dados que podem não estar disponíveis localmente, no smartphone dos usuários.
- O uso de recursos complexos (e.g., vídeo, áudio e imagens) para determinar o contexto atual do usuário é uma forte tendência para aplicações móveis. Como processar tais informações no dispositivo?
- Diversos trabalhos na área de MCC usam o termo “context” ou “context-aware”, mas a maioria se refere apenas ao contexto em que o dispositivo móvel está executando (e.g., tipo de conexão, quantidade de bateria disponível).
- “O uso de serviços de nuvem por aplicações móveis sensíveis ao contexto é um caminho sem volta.” [Naqvi et al 2013]

Integração MCC e CAM (Cenários)

- Crowdsensing
 - Cenário em que muitos dispositivos móveis podem capturar e computar dados contextuais. Com o compartilhamento colaborativo de tais dados, pode-se extrair informações e inferir processos de interesse comum.



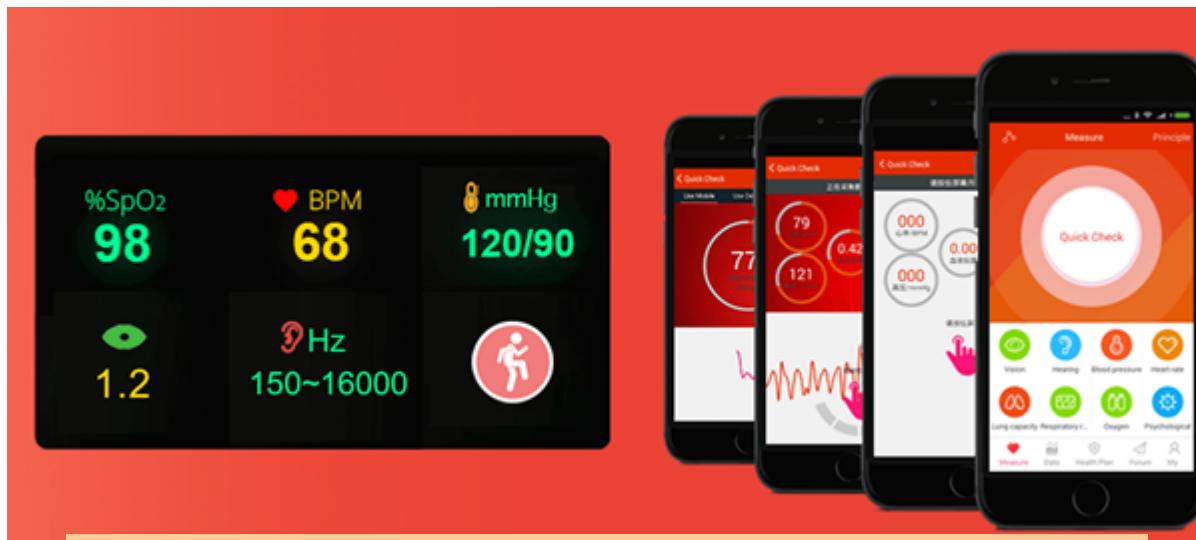
Ra, Moo-Ryong, et al. "Medusa: A programming framework for crowd-sensing applications." 10th MobiSys. ACM, 2012.



Integração MCC e CAM (Cenários)

- Healthcare and Well-being

- Com o aumento da presença de microsensores, inclusive em smartphones, é possível fazer inferências sobre a posição e atividades executadas pelos usuários (e.g., se a pessoa está em pé, deitada, correndo, andando).
- O reconhecimento de tais atividades depende do monitoramento e análise de dados contextuais (e.g., batimento cardíaco, nível de respiração).



iCare health Monitor: <http://icarefit.com>



Nike+ app: https://www.nike.com/us/en_us/c/nike-plus/nike-app

Integração MCC e CAM (Cenários)

- Augmented Reality
 - Uma aplicação típica requer suporte a detecção e tracking de objetos. Dados contextuais são importantes conhecer o ambiente em que o usuário está e poder prover conteúdo personalizado.



Pokemon Go: <http://www.pokemongo.com/>



Dinosaur 4D+ Cards app:
<https://www.octagonstudio.com/4d>

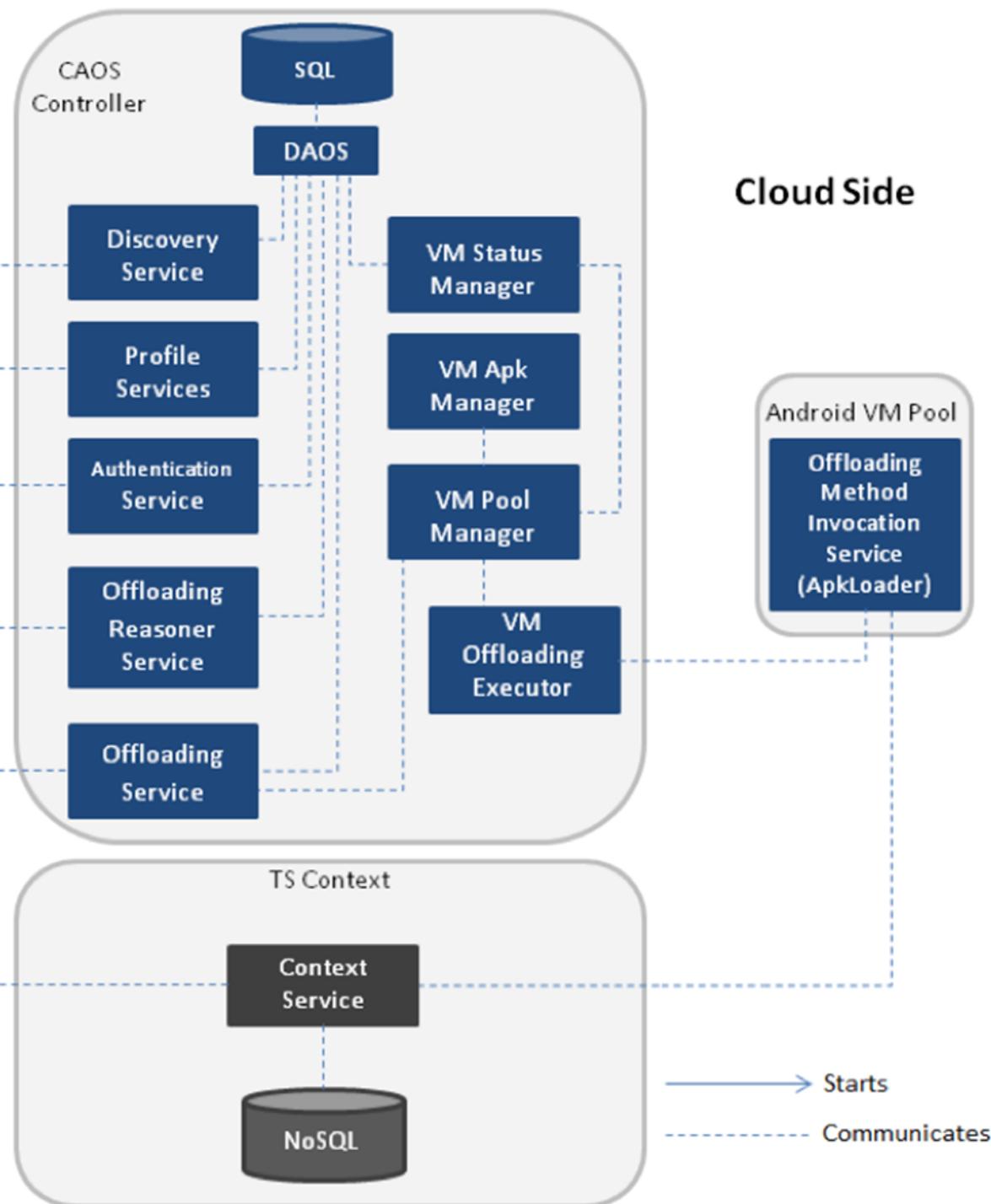
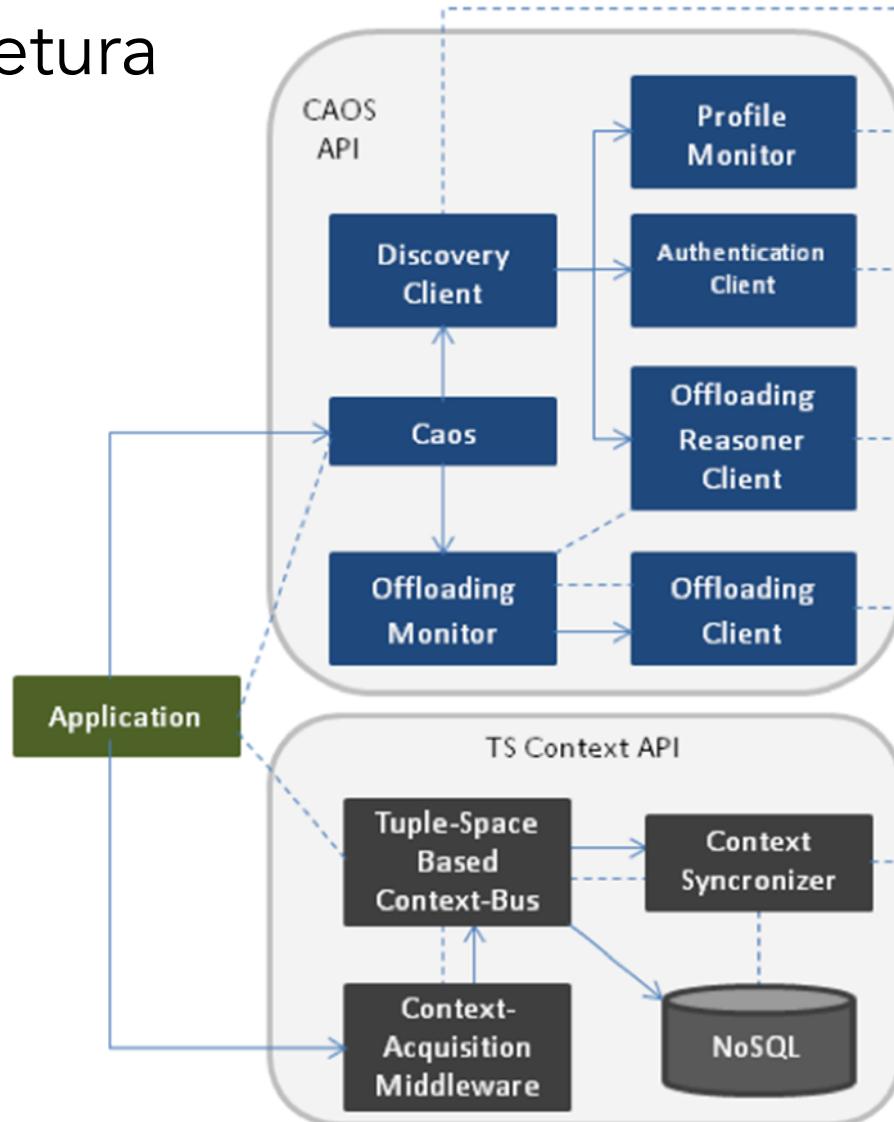
CAOS (Context Acquisition and Offloading System)

- Desenvolvido no GREat/UFC
- Integra MCC e CAM
- Baseado nos trabalhos LoCCAM e MpOS
- Offloading de dados e métodos
- Monitoramento de métricas de rede, hardware e software
- Paradigma Pub/Sub
- Offloading para a Nuvem ou Cloudlet
- Tomada de decisão dinâmica
- Descoberta automática de Cloudlets
- Diferentes políticas de sincronização e privacidade de dados contextuais
- Modularização da aquisição de contexto

CAOS

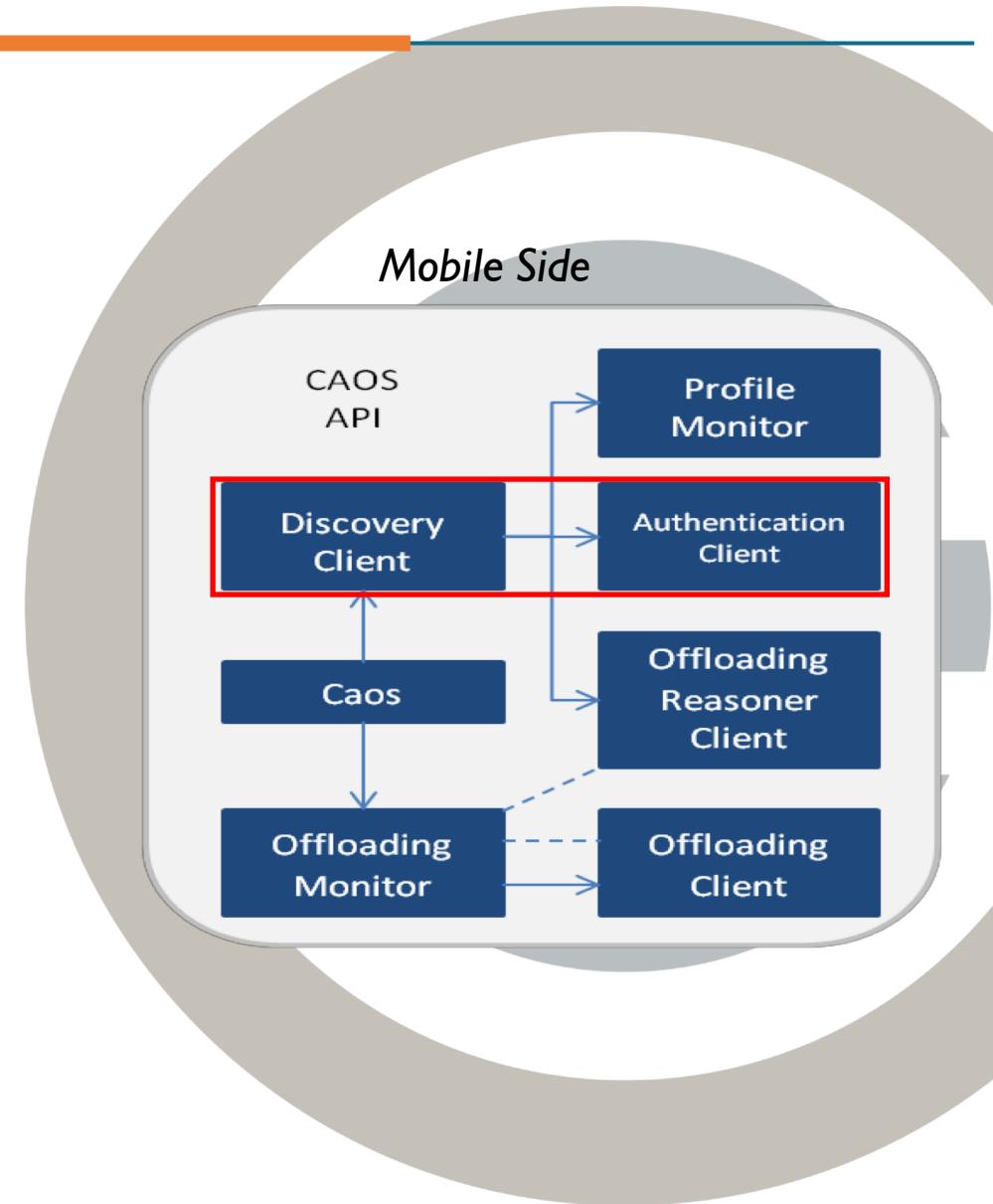
Mobile Side

- Arquitetura



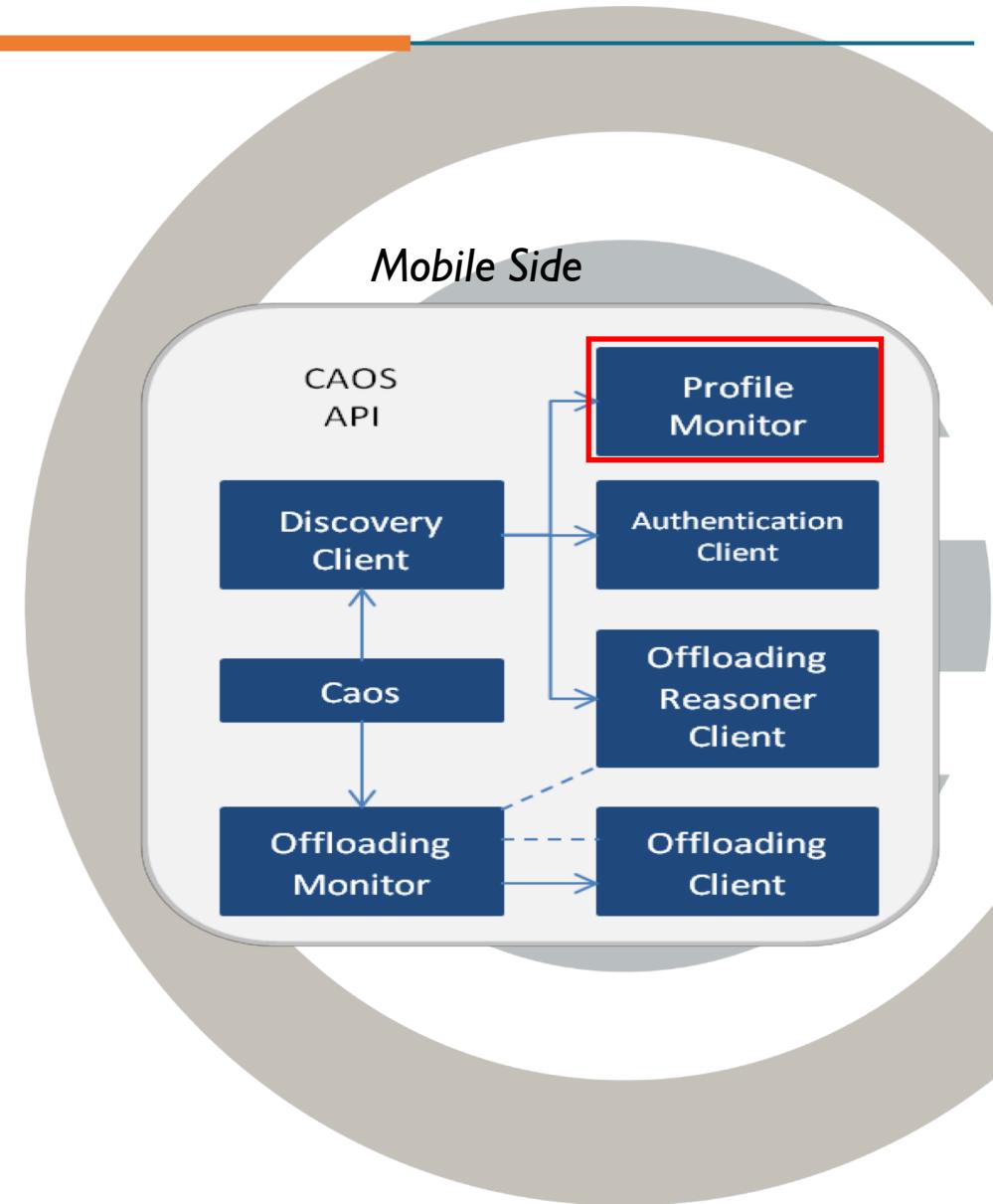
CAOS

- Discovery Client
 - Usa um mecanismo baseado em UDP/Multicast para descobrir CAOS Controllers em execução na rede local.
- Authentication Client
 - Mecanismo simples que autentica o dispositivo móvel no CAOS Controller, e autoriza o acesso aos demais serviços.



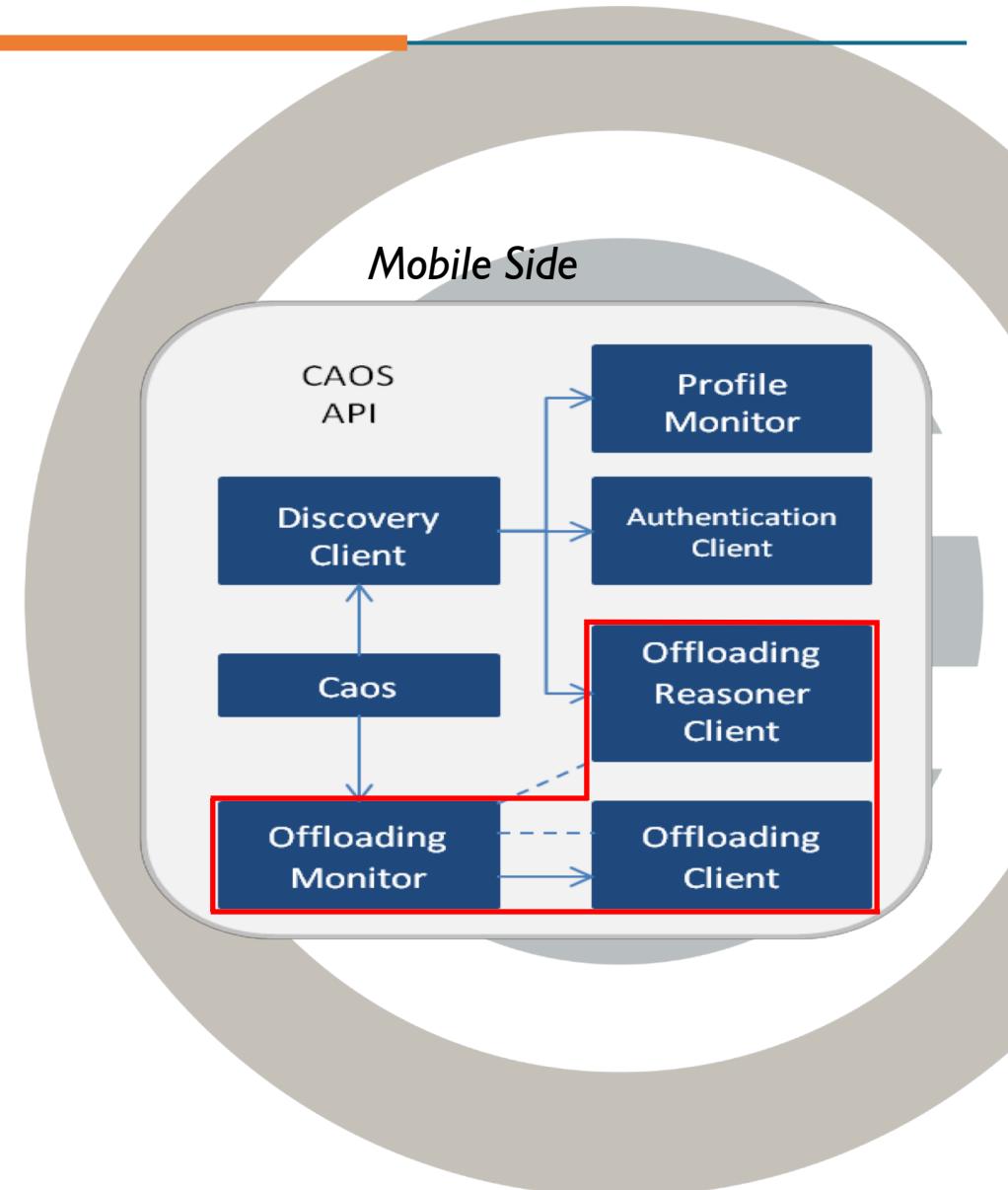
CAOS

- Profile Monitor
 - Responsável por monitorar diversas métricas do ambiente e do dispositivo móvel, e enviá-las periodicamente aos Profile Services.
 - Métricas de rede, software e hardware
 - Exemplo: taxa de upload e download, latência, carga da bateria, memória e tipo do processador, etc.



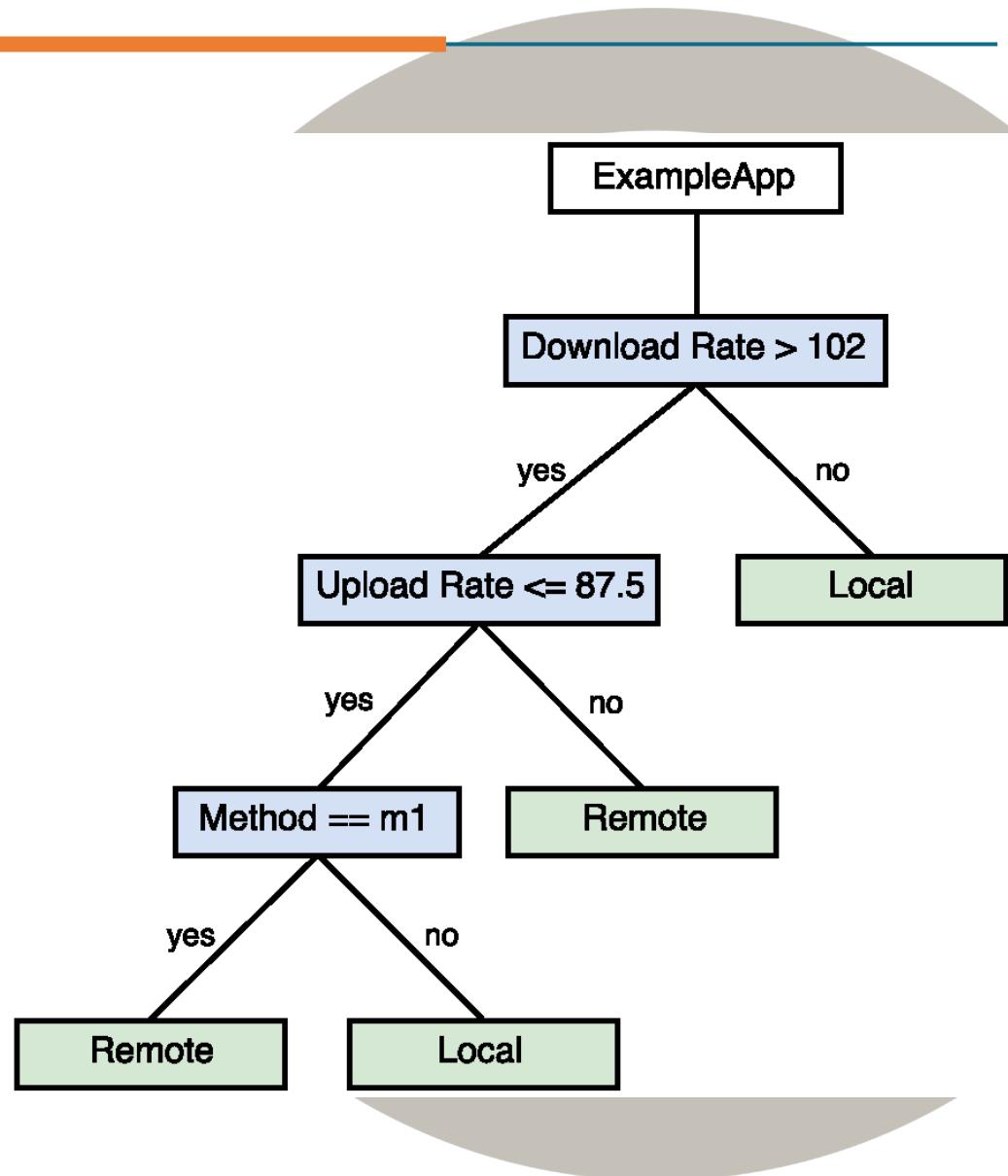
CAOS

- Offloading Monitor
 - Monitora a execução da aplicação e intercepta o fluxo de execução sempre que um método anotado é chamado.
- Offloading Reasoner Client
 - Auxilia a decisão de offloading usando uma árvore de decisão, que é enviada pelo Offloading Reasoner Service.
- Offloading Client
 - Inicia o processo de offloading



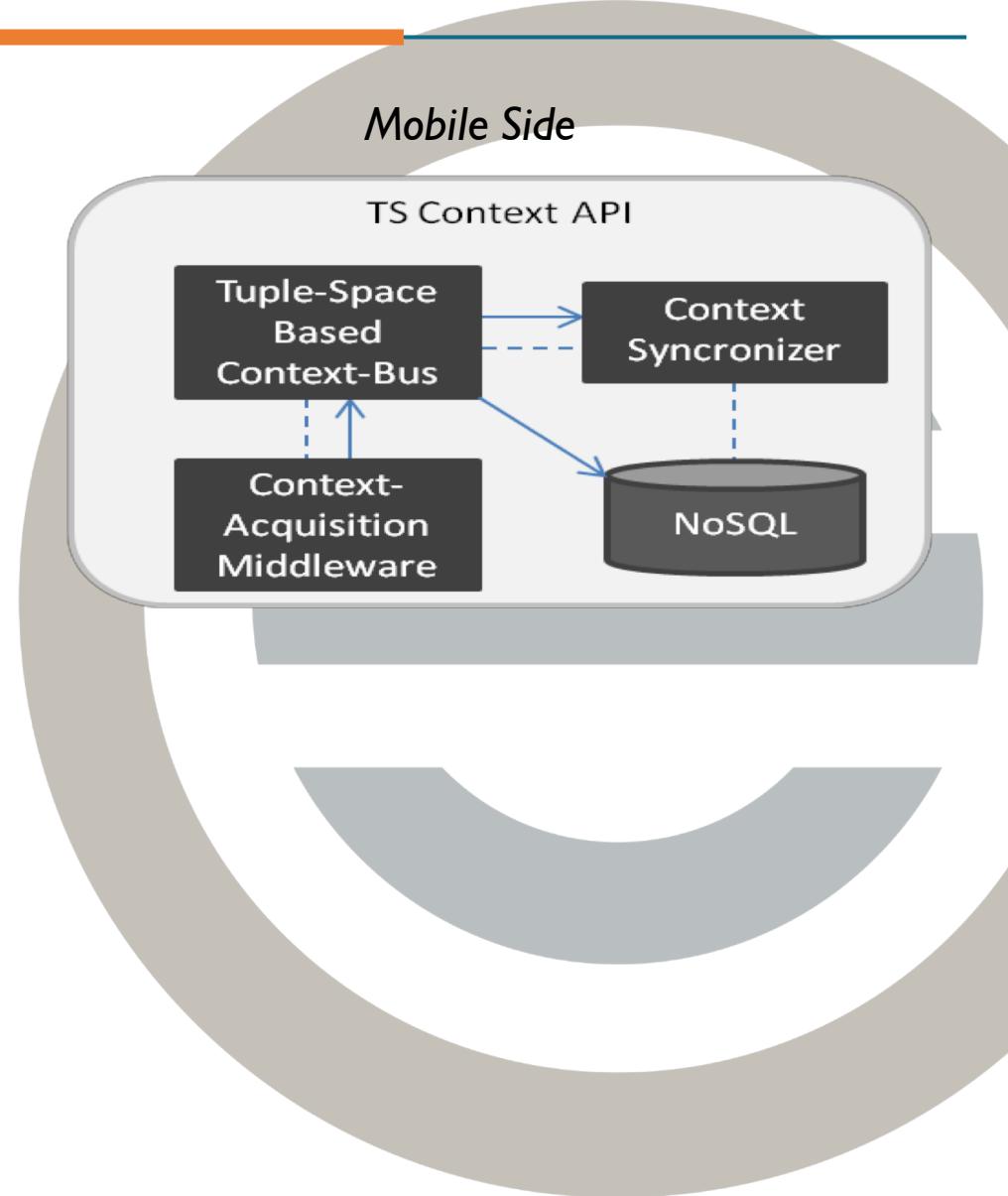
CAOS

- Offloading Monitor
 - Monitora a execução da aplicação e intercepta o fluxo de execução sempre que um método anotado é chamado.
- Offloading Reasoner Client
 - Auxilia a decisão de offloading usando uma árvore de decisão, que é enviada pelo Offloading Reasoner Service.
- Offloading Client
 - Inicia o processo de offloading



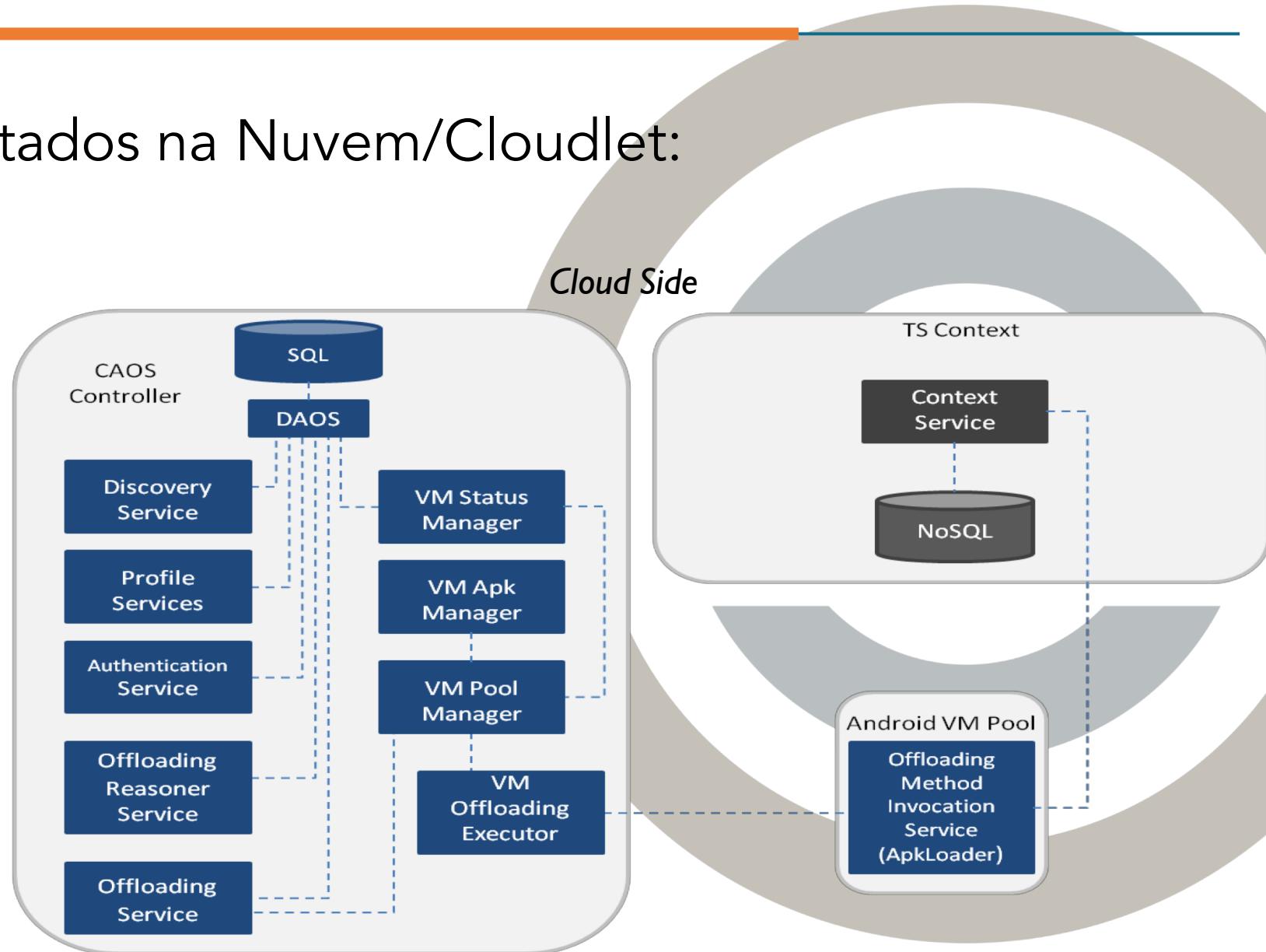
CAOS

- Context-Acquisition Middleware
 - Responsável por gerenciar o ciclo de vida (i.e., search, deploy, start e stop) dos sensores virtuais que encapsulam os mecanismos de aquisição de contexto.
- Tuple-Space Based Context-Bus
 - Armazena as informações contextuais em um formato de espaço de tuplas e entregam tais informações às aplicações usando notificações através de um barramento de eventos contextuais.
- Context Synchronizer
 - Sincroniza dados contextuais entre os dispositivos móveis e a Nuvem/Cloudlet.



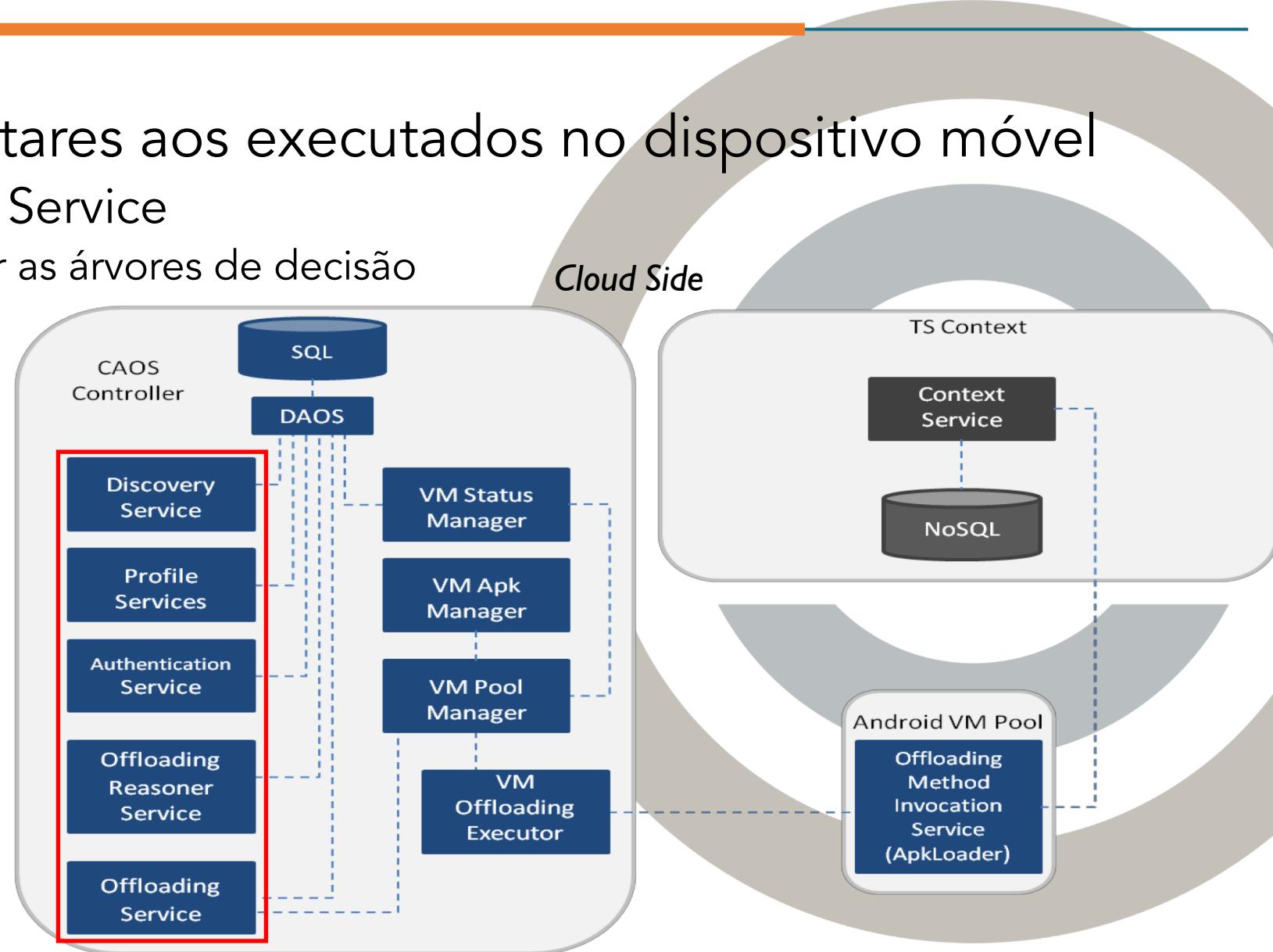
CAOS

- Componentes executados na Nuvem/Cloudlet:
 - CAOS Controller
 - Android VM Pool
 - TS Context



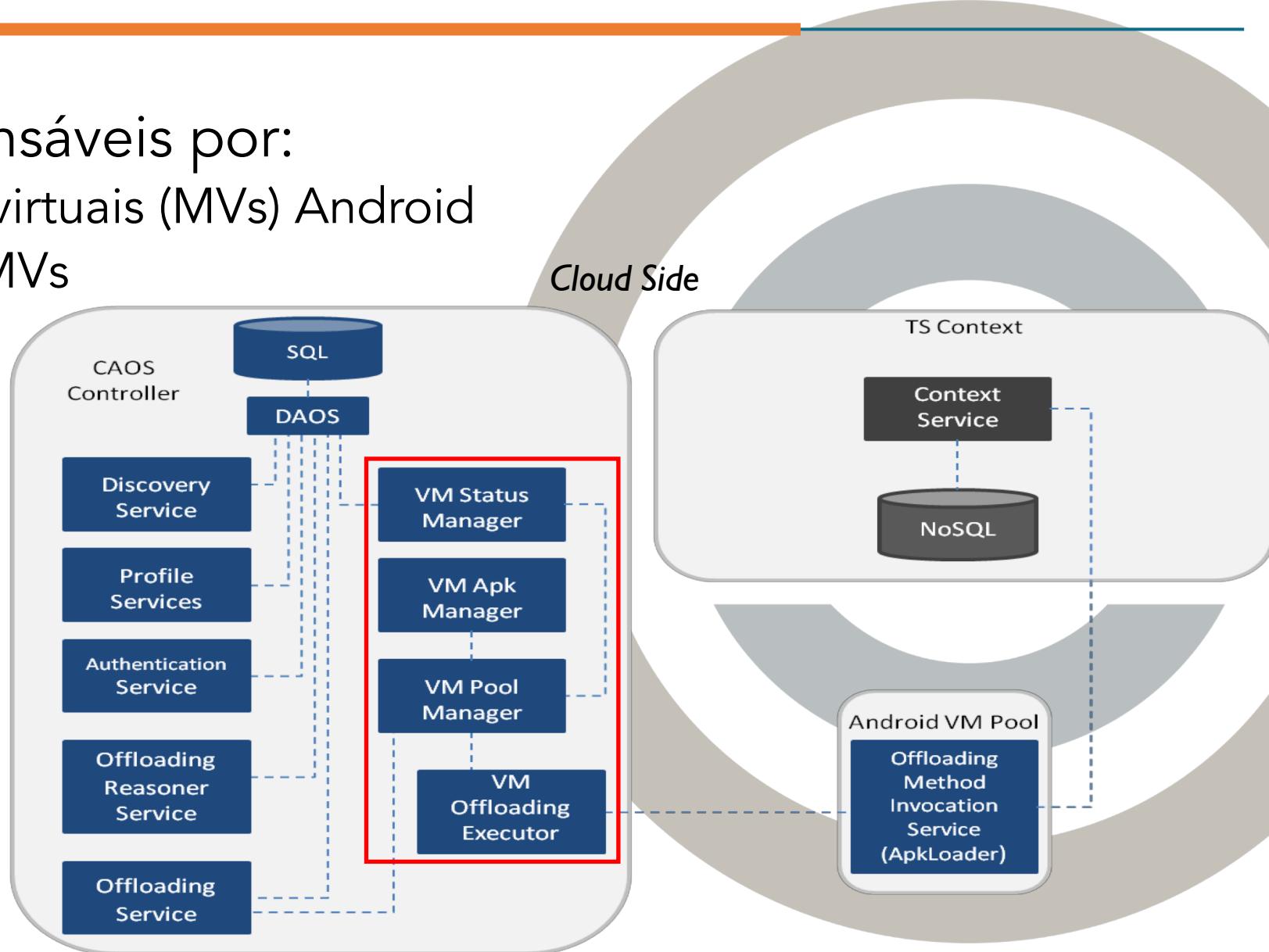
CAOS

- Serviços complementares aos executados no dispositivo móvel
 - Offloading Reasoner Service
 - Responsável por criar as árvores de decisão
 - Online profiling
 - Dados históricos
 - Authentication Service
 - Offloading Service
 - Discovery Service
 - Profile Services



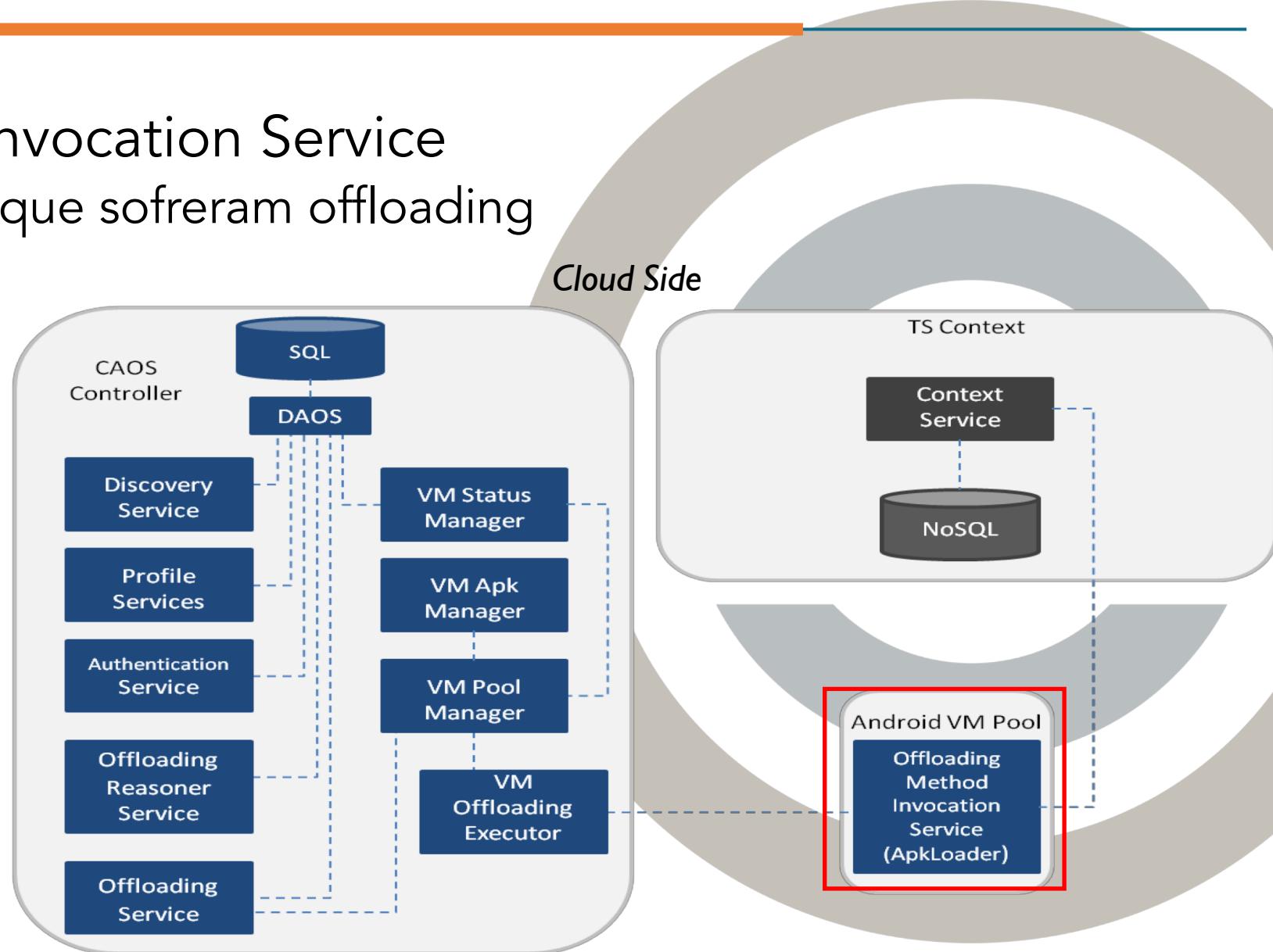
CAOS

- Componentes responsáveis por:
 - Gerenciar máquinas virtuais (MVs) Android
 - Implantar APKs nas MVs
 - Monitorar as MVs
 - Requisitar offloading à MV apropriada



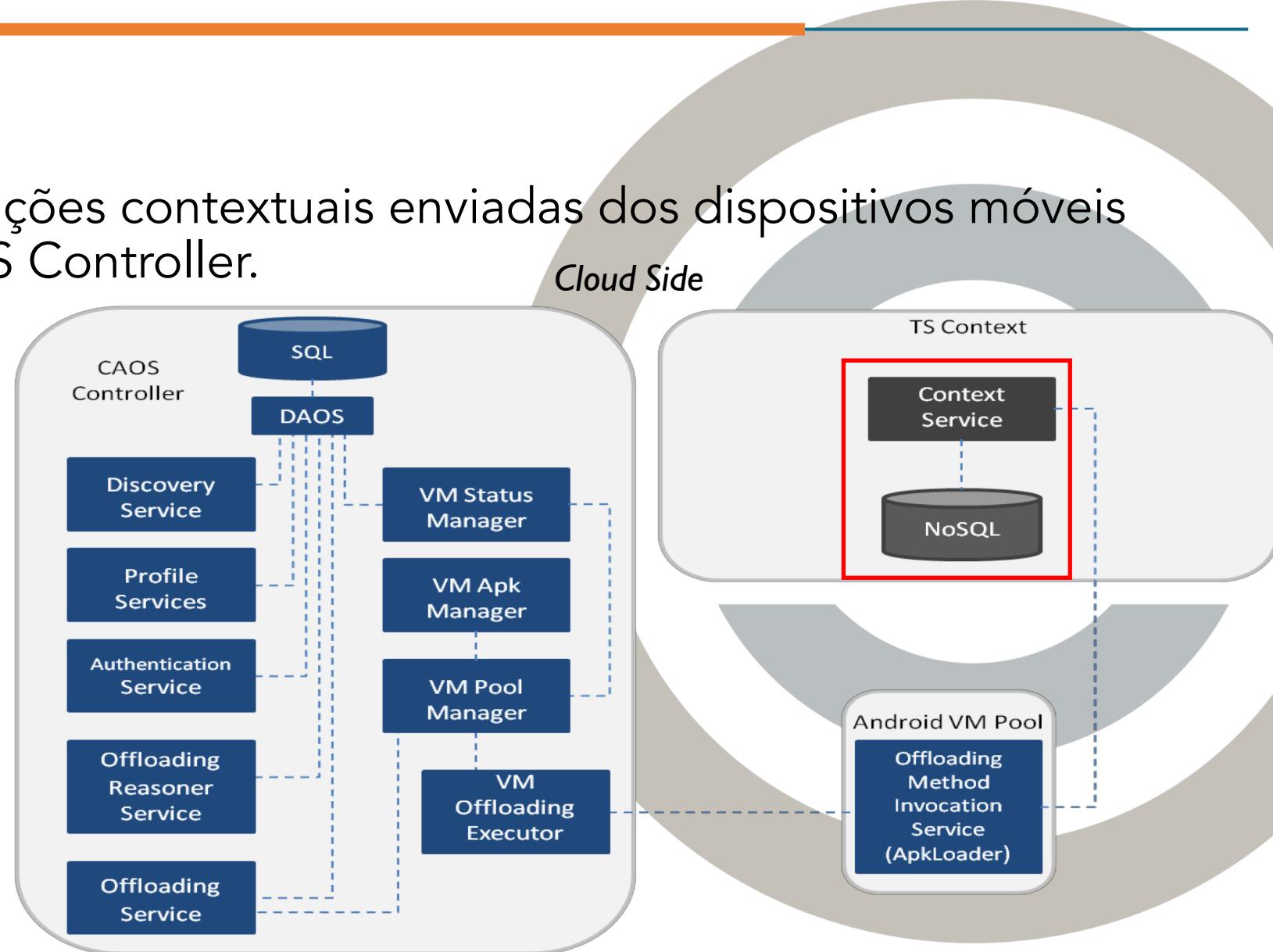
CAOS

- Offloading Method Invocation Service
 - Executa os métodos que sofreram offloading



CAOS

- Context Service
 - Armazena as informações contextuais enviadas dos dispositivos móveis conectados ao CAOS Controller.
 - Um banco de dados NoSQL é utilizando para manter as informações contextuais



CAOS

- Modelo de Contexto (5 campos do LoCCAM + 4 novos)
 - *IdDevice*: Representa a identificação do dispositivo móvel
 - *IdApp*: Representa a identificação da aplicação
 - *Visible*: Identifica a visibilidade do dado contextual
 - *GlobalTimestamp*: O instante em que o dado é inserido na nuvem

CAOS

- Exemplo de contexto
 - *IdDevice*: 34569213456
 - *IdApp*: br.ufc.great.myphotos
 - *Visible*: 0
 - *GlobalTimestamp*: 2345671231566



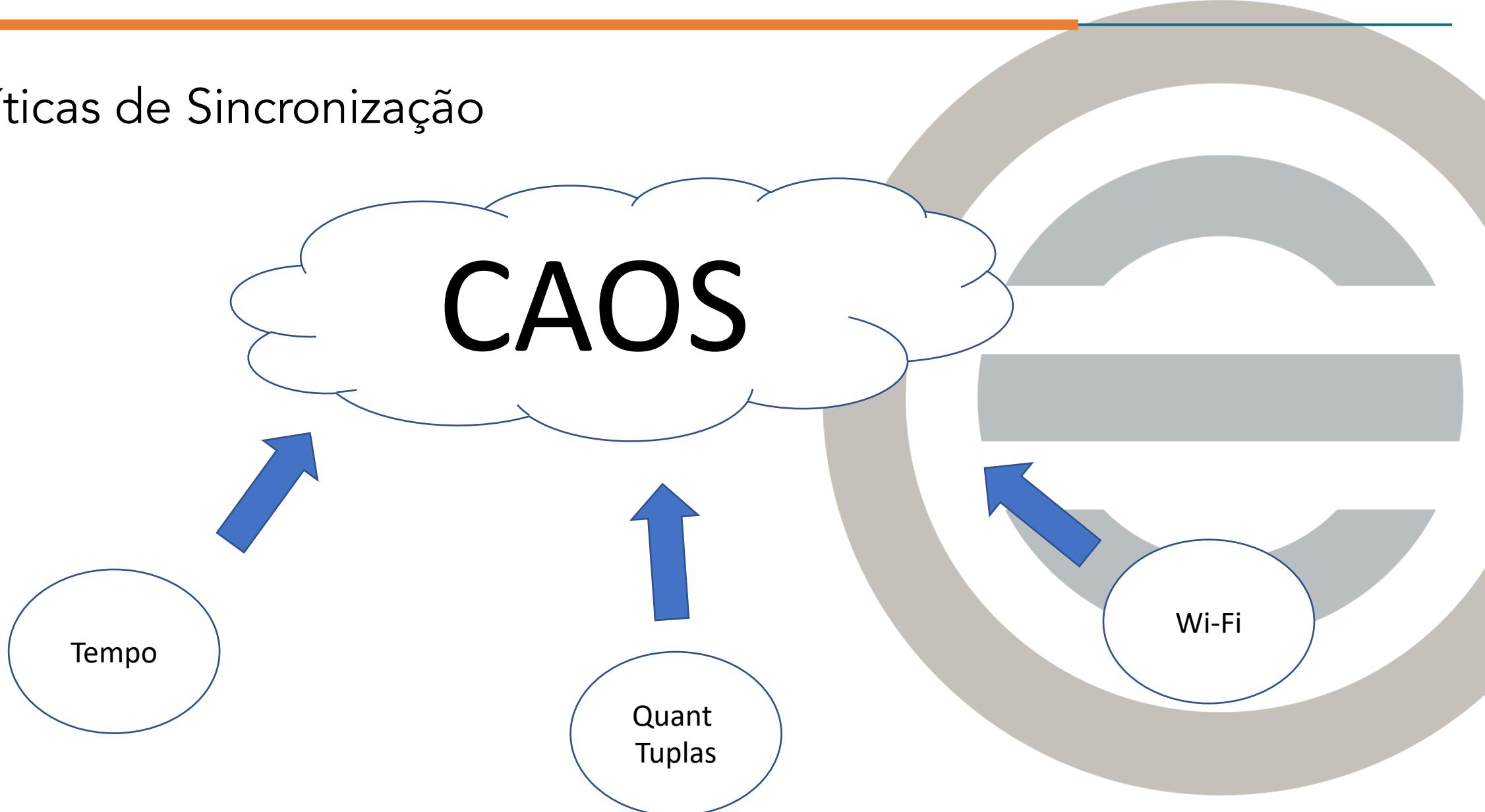
CAOS

- Políticas de Privacidade
 - Visibilidade dos dados
 - Pública (visible = 1)
 - Privada (visible = 0)
 - *Gateway: cloudlet confiável*

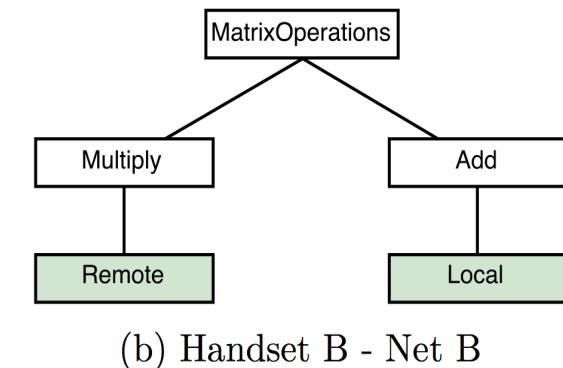
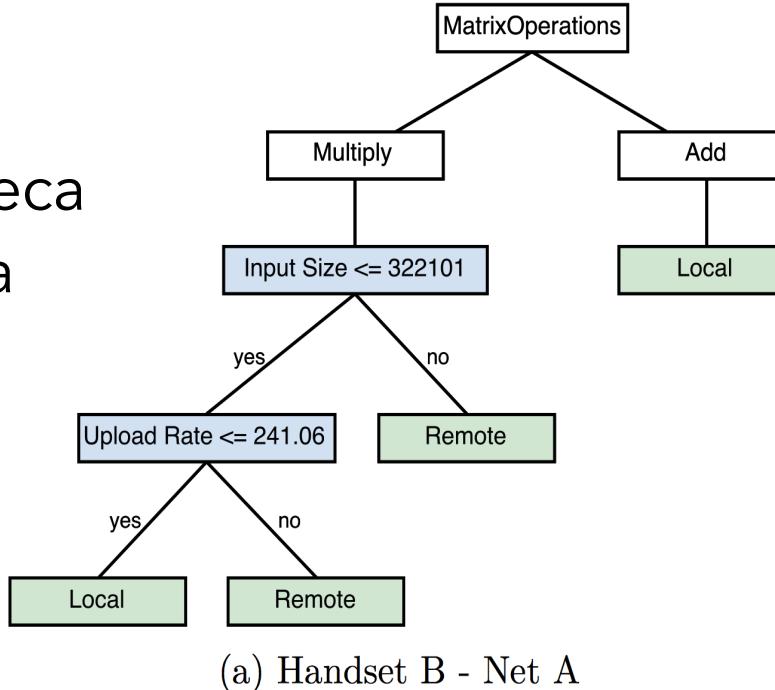


CAOS

- Políticas de Sincronização



- Mais detalhes sobre a decisão de quando fazer offloading
 - O Offloading Reasoner Service cria uma árvore de decisão para cada Usuário/Aplicação utilizando o histórico de execuções de offloading e dados de monitoramento
 - O algoritmo J48 da biblioteca Weka é utilizado para gerar a árvore de decisão



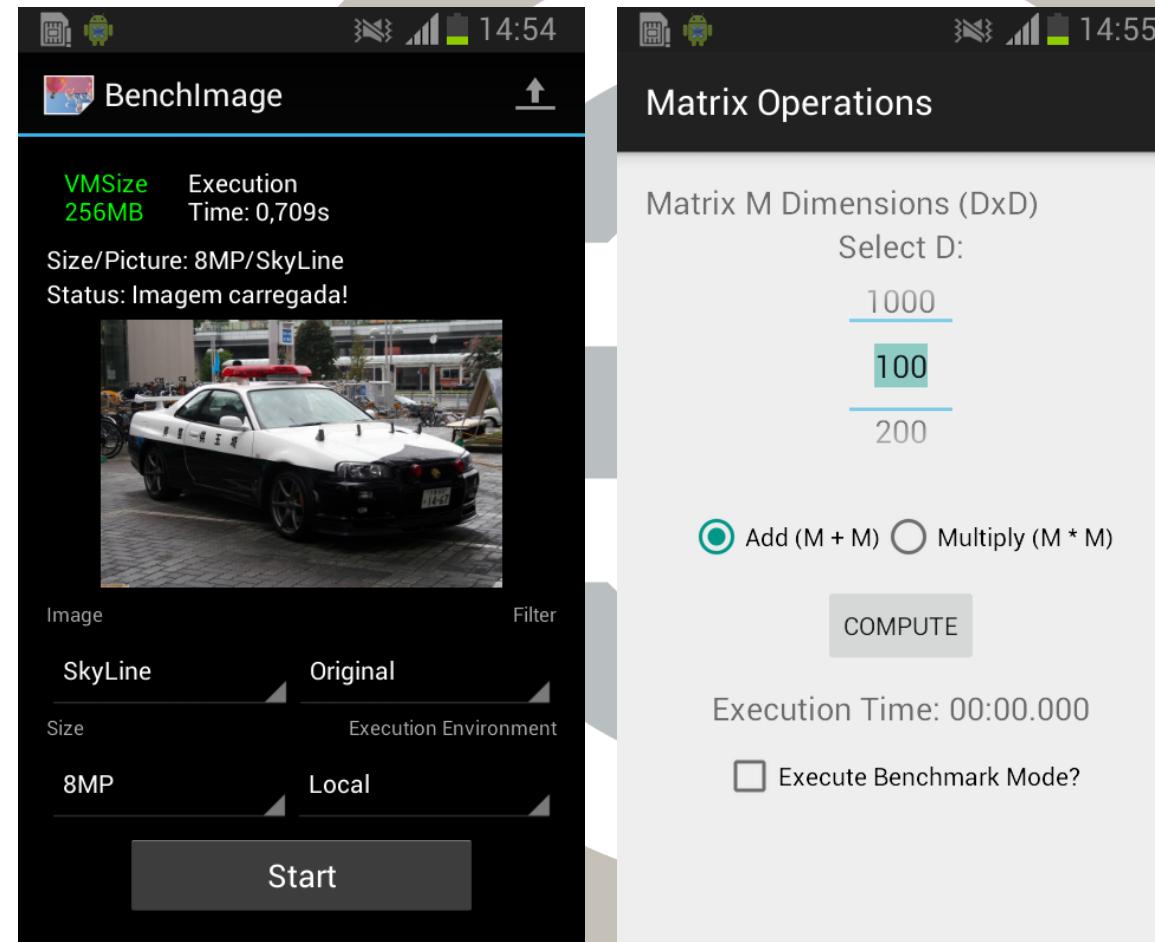
- Detalhes de implementação
 - IDE Android Studio
 - Couchbase (a NoSQL database)
 - Bouncy Castle encryption API
 - JCoAP (Java Constrained Application Protocol)
 - CAOS Controller modules were implemented using Java SE/EE
 - Apache TomCat
 - RESTful Web - Jersey framework
 - PostgreSQL database - Hibernate framework
 - MongoDB (a NoSQL database)
 - VirtualBox hypervisor
 - Android x86 version (4.4.2) Kitkat.



CAOS

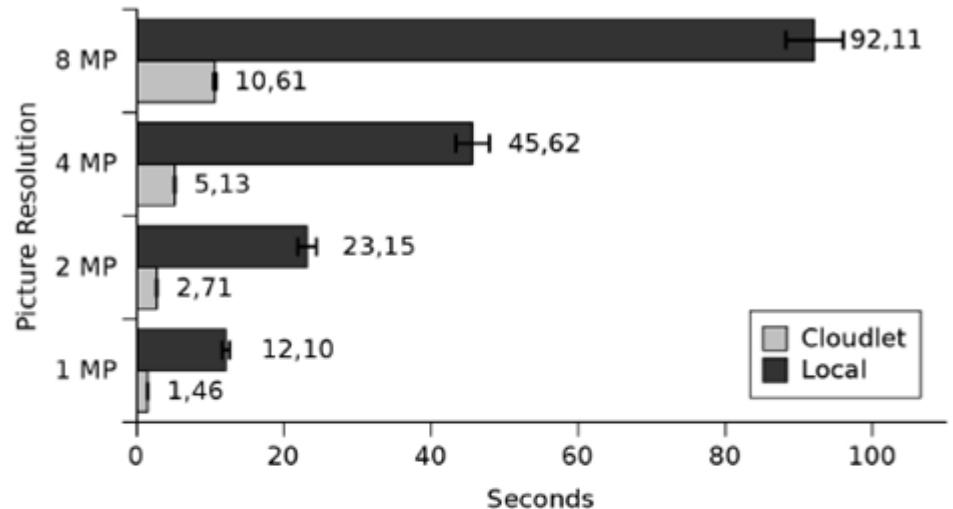
Alguns resultados:

- Aplicações
 - BenchImage
 - MatrixOperations
- Dispositivo móvel
 - Android 4.1.0, 1 GB de RAM e processador ARM Cortex A9
- Cloudlet (rede 802.11n)
 - Laptop, SO Linux Mint 17.2, 8 GB de RAM e processador Core i5-4200U
 - MV Android: Android x86 4.4.2, 4 GB de RAM e 2 CPUs virtuais.

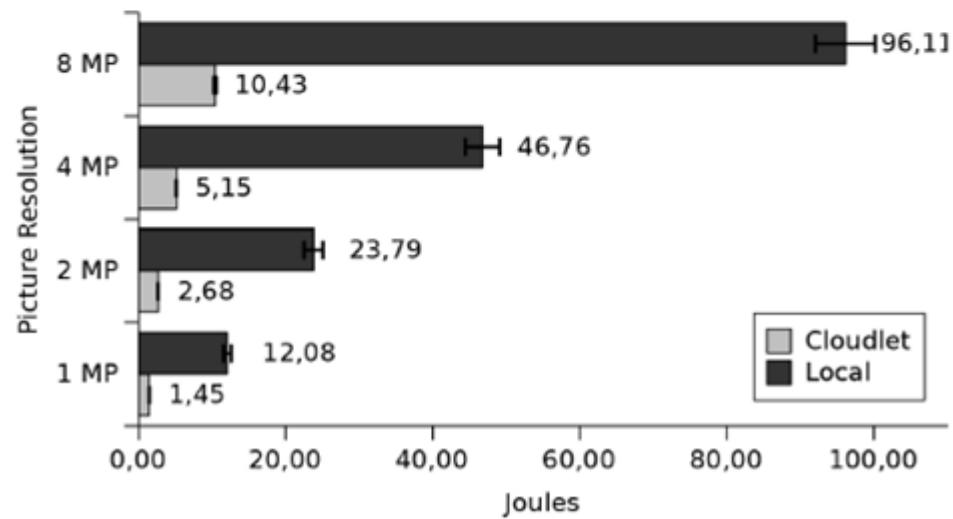


BenchImage

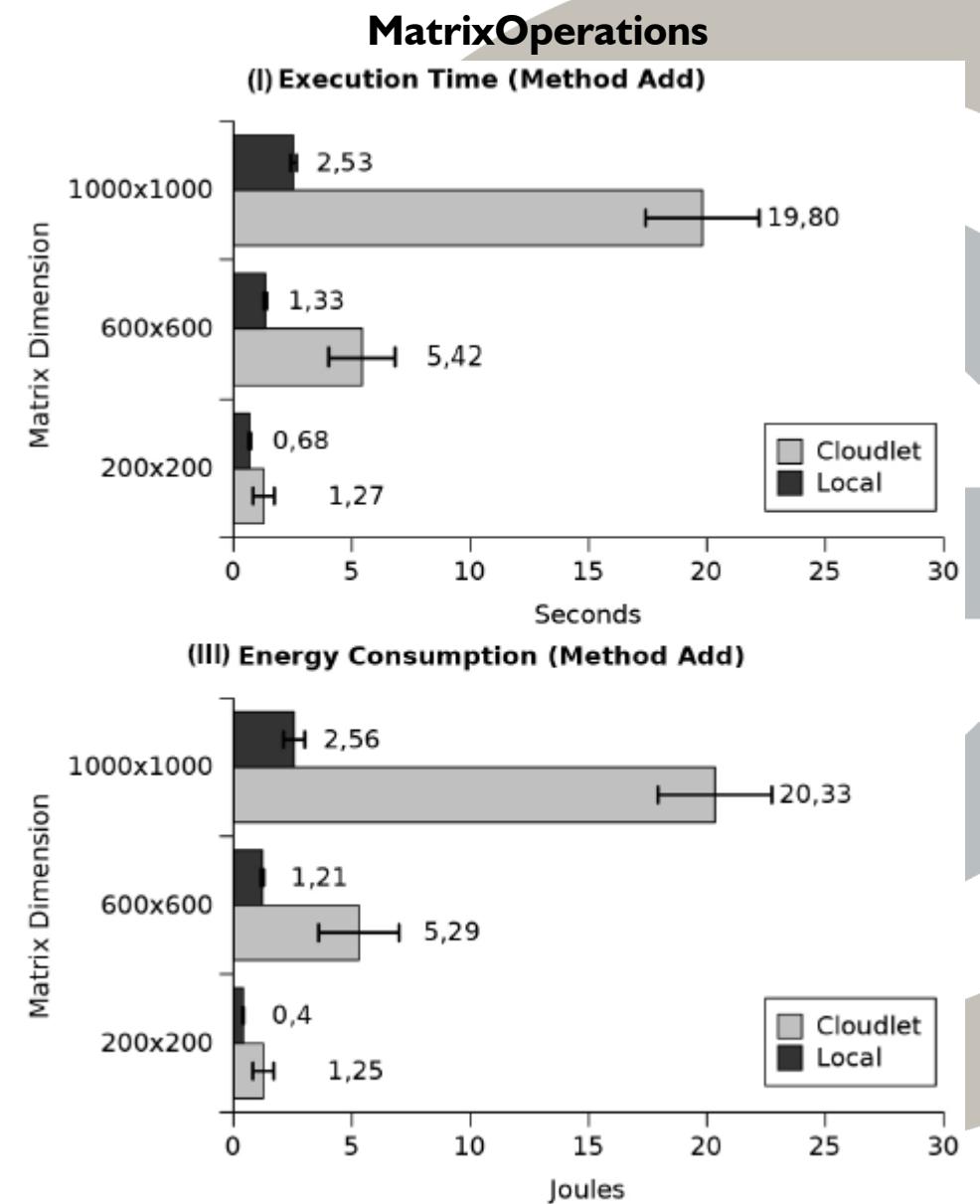
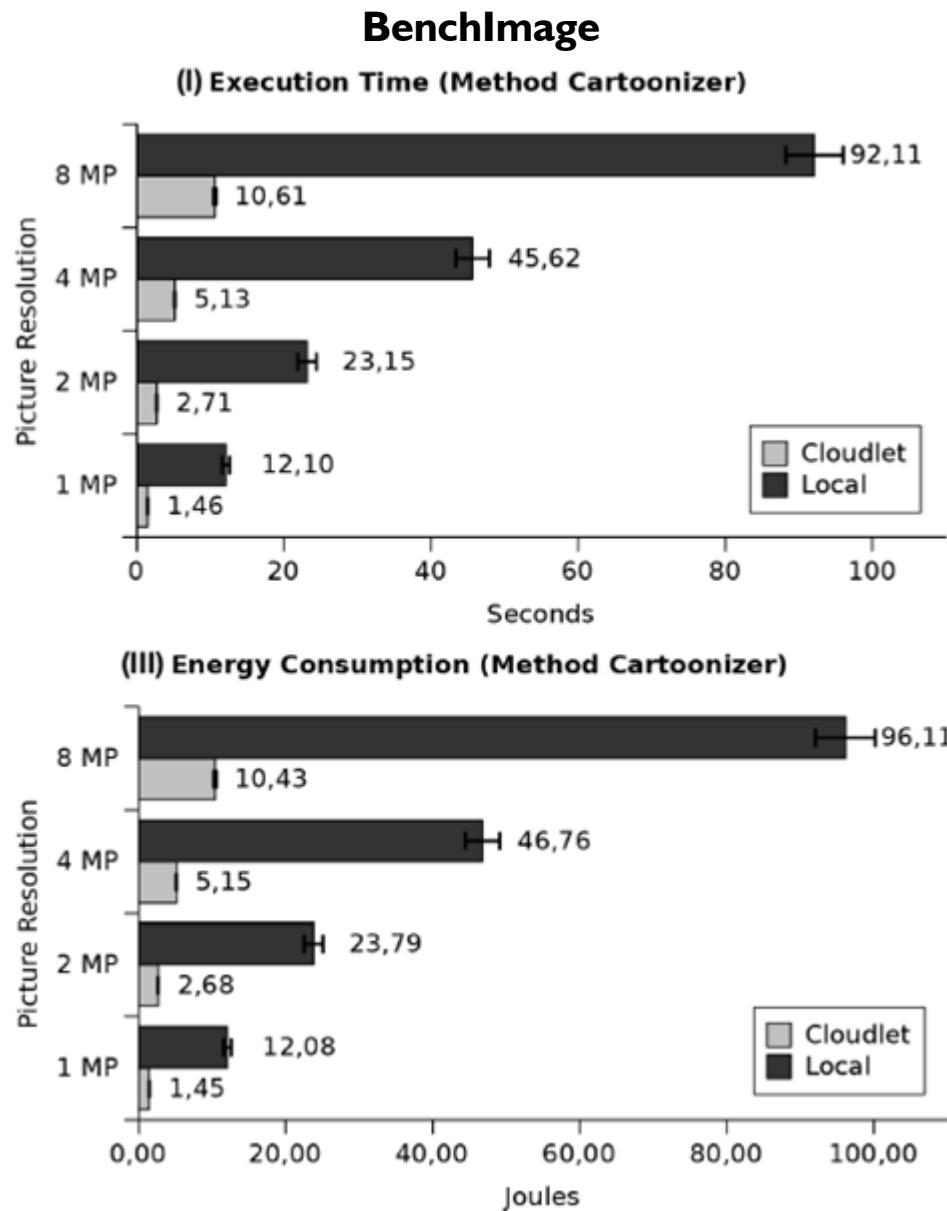
(I) Execution Time (Method Cartoonizer)



(III) Energy Consumption (Method Cartoonizer)



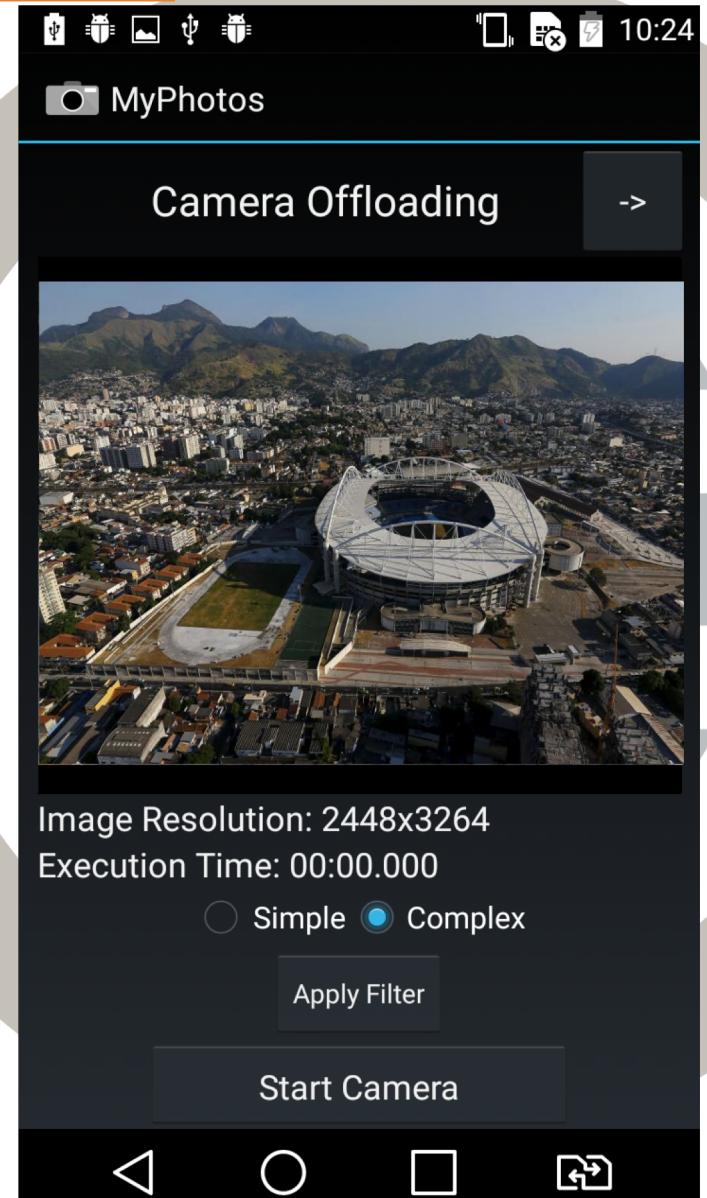
CAOS



Hands on

MyPhotos

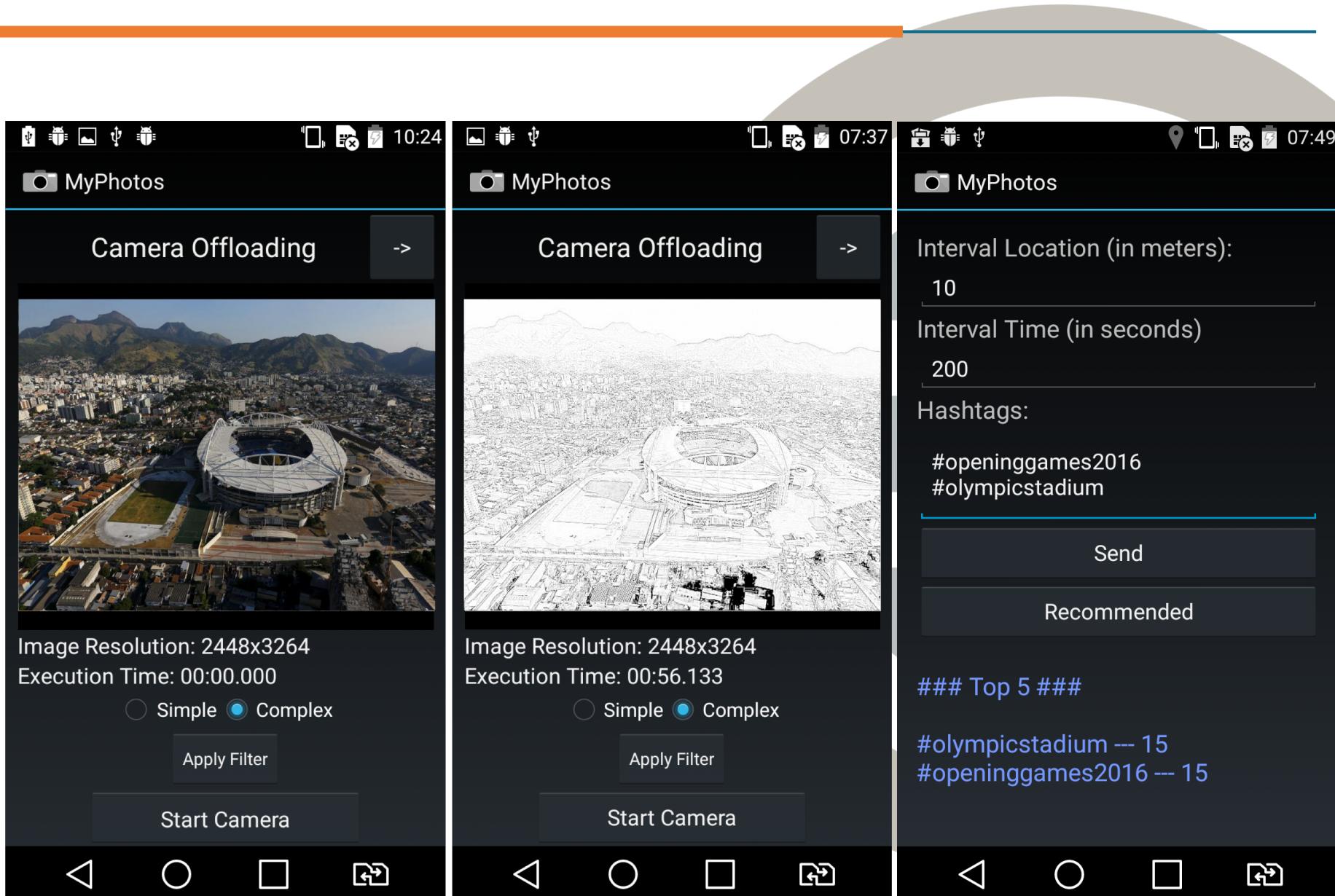
- Aplicação multimídia sensível ao contexto
- Parecida com o Instagram
 - Aplicar filtros em fotos
 - Utilizar hashtags para marcar as fotos
- Além das hashtags, pode utilizar dados contextuais (e.g., localização e data/hora em que foram capturadas) para marcar as fotos



Hands on

MyPhotos

- Filtros:
 - Simple
(Redtone)
 - Complex
(Cartoonizer)



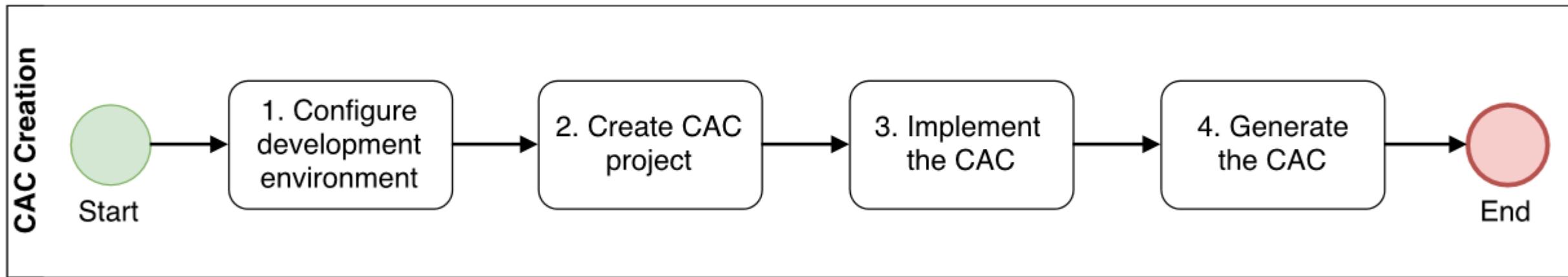
Hands on

Desenvolvendo a aplicação MyPhotos com o framework CAOS

1. Criação dos CACs
2. Implementação da aplicação MyPhotos
3. Instalação e configuração do CAOS Services

Hands on

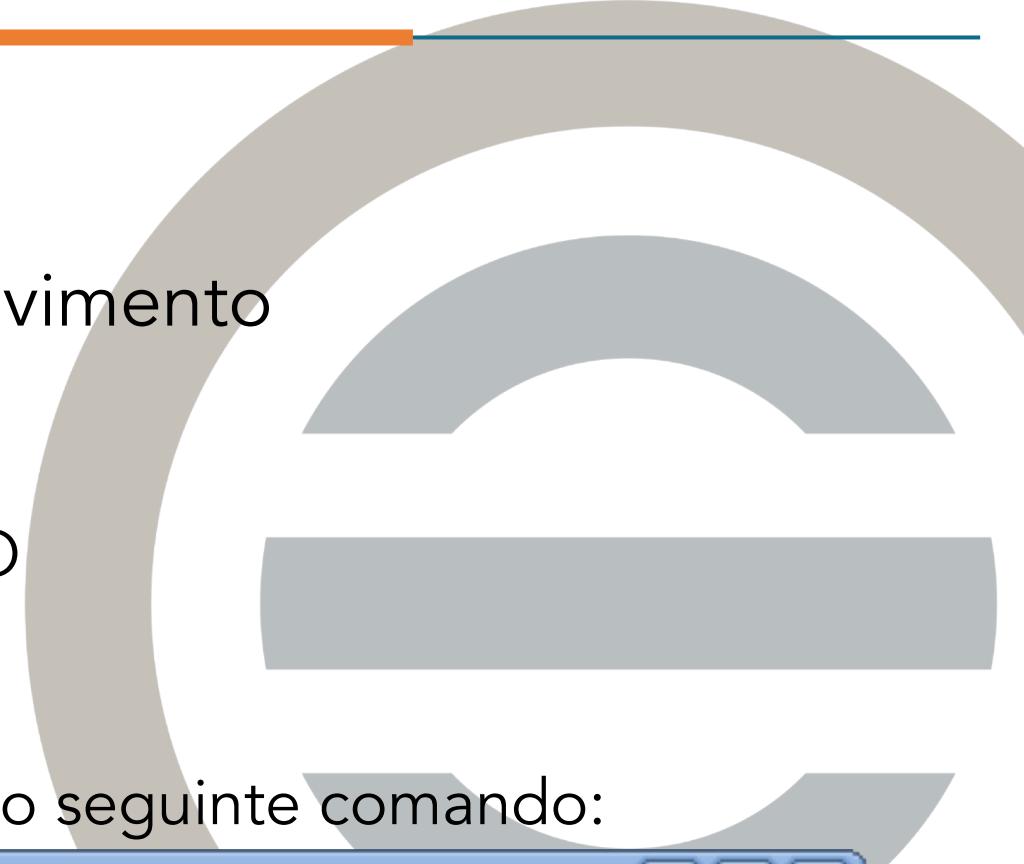
1. Criação dos CACs



Hands on

1. Criação dos CACs

- Configurar o ambiente de desenvolvimento
- Fazer download dos scripts
- Colocar a pasta de scripts no *PATH* do SO
- Reiniciar a máquina
- Instalar as dependências Maven
 - Entrar na pasta dos scripts e executar o seguinte comando:

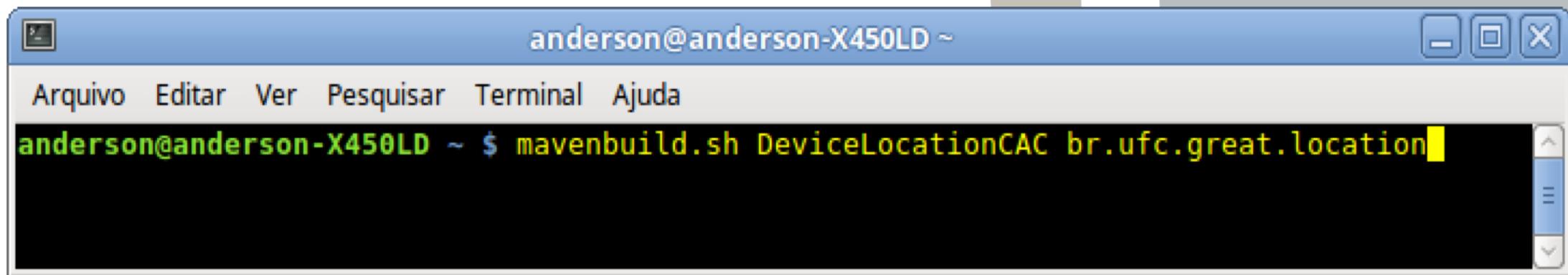


```
anderson@anderson-X450LD ~/git2/sac-build-scripts
Arquivo Editar Ver Pesquisar Terminal Ajuda
anderson@anderson-X450LD ~/git2/sac-build-scripts $ ls
cacbuilder.sh libs mavenbuild.sh mavenbuild.sh~ mavenlibs.sh
anderson@anderson-X450LD ~/git2/sac-build-scripts $ ./mavenlibs.sh
```

Hands on

1. Criação dos CACs

- Criar um projeto de CAC
- Escolha qualquer pasta e execute o seguinte comando:
mavenbuild.sh <NomeDoCAC> <NomeDoPacote>



A screenshot of a Linux terminal window titled "anderson@anderson-X450LD ~". The window has a blue header bar with standard window controls. Below the title bar is a menu bar with options: Arquivo, Editar, Ver, Pesquisar, Terminal, and Ajuda. The main terminal area is black with white text. A command is being typed in: "anderson@anderson-X450LD ~ \$ mavenbuild.sh DeviceLocationCAC br.ufc.great.location". The command "mavenbuild.sh" is highlighted in green, while the arguments "DeviceLocationCAC" and "br.ufc.great.location" are highlighted in yellow.

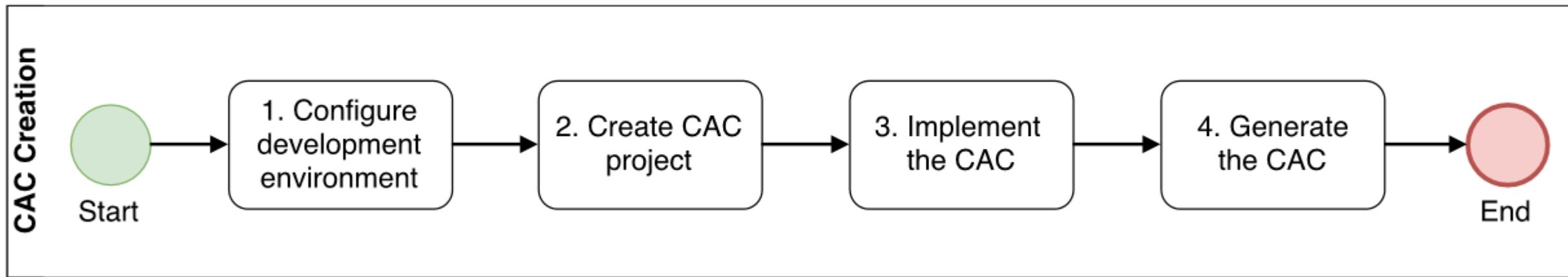
Hands on

1. Criação dos CACs

- Implementar o CAC
- Importar o projeto para o Eclipse
- Na pasta src, procurar pela classe principal (mesmo nome do projeto)
- Definir a CK do CAC
- Implementar o método onStart (lógica do CAC)
- Gerar o CAC
- Uma vez implementado, executar o projeto como uma aplicação Java

Hands on

2. Implementação da aplicação MyPhotos



Hands on

2. Implementação da aplicação MyPhotos
 - Adicionar a biblioteca do CAOS ao projeto
 - Antes, importe o projeto base do MyPhotos no Android Studio.

Hands on

2. Implementação da aplicação MyPhotos

- Implementar as interfaces e identificar os métodos que podem sofrer offloading

```
1 public interface Effect {  
2     @Offloadable  
3     public byte[] simple(byte source[]);  
4  
5     @Offloadable  
6     public byte[] complex(byte source[]);  
7     ...  
8 }  
9 public class ImageEffect implements Effect {  
10    public byte[] simple(byte source[]){ ... }  
11    public byte[] complex(byte source[]){ ... }  
12 }
```

Hands on

2. Implementação da aplicação MyPhotos

- Implementar as interfaces e identificar os métodos que podem sofrer offloading

```
1 public interface Read {  
2     @Offloadable  
3     public String readHashTags(double latitude, double longitude,  
4                                 double intervalTime, double intervalLocation);  
5 }  
6  
6 public class ReadHashtags implements Read {  
7     public String readHashTags(double latitude, double longitude,  
8                                 double intervalTime, double intervalLocation){ ... }  
9 }
```

Hands on

2. Implementação da aplicação MyPhotos

- Configurar a Activity principal

```
1 @Caos
2 public class MyPhotosMainActivity extends Activity implements CaosContextListener {
3
4     @Inject(ImageEffect.class)
5     private Effect effect;
6     @Inject(ReadHashtags.class)
7     private Read read;
8     ...
9     String CK_TAGS = "context.ambient.noise";
10    String CK_LOCATION = "context.device.location";
11    ...
12    @Override
13    protected void onCreate(Bundle bundle) {
14        ...
15        Caos caos = Caos.getConfig().getInstance();
16        caos.start(this, this);
17        ...
```

Hands on

2. Implementação da aplicação MyPhotos

- Declarar os interesses da aplicação

```
27     ...
28     @Override
29     public void onServiceConnected(ISysSUService service) {
30         try {
31             caos.putInterest(CK_LOCATION);
32             caos.putInterest(CK_TAGS);
33         } catch (RemoteException e) {
34             e.printStackTrace();
35         }
36     }
37 }
```

Hands on

2. Implementação da aplicação MyPhotos
 - Compilar e gerar o APK

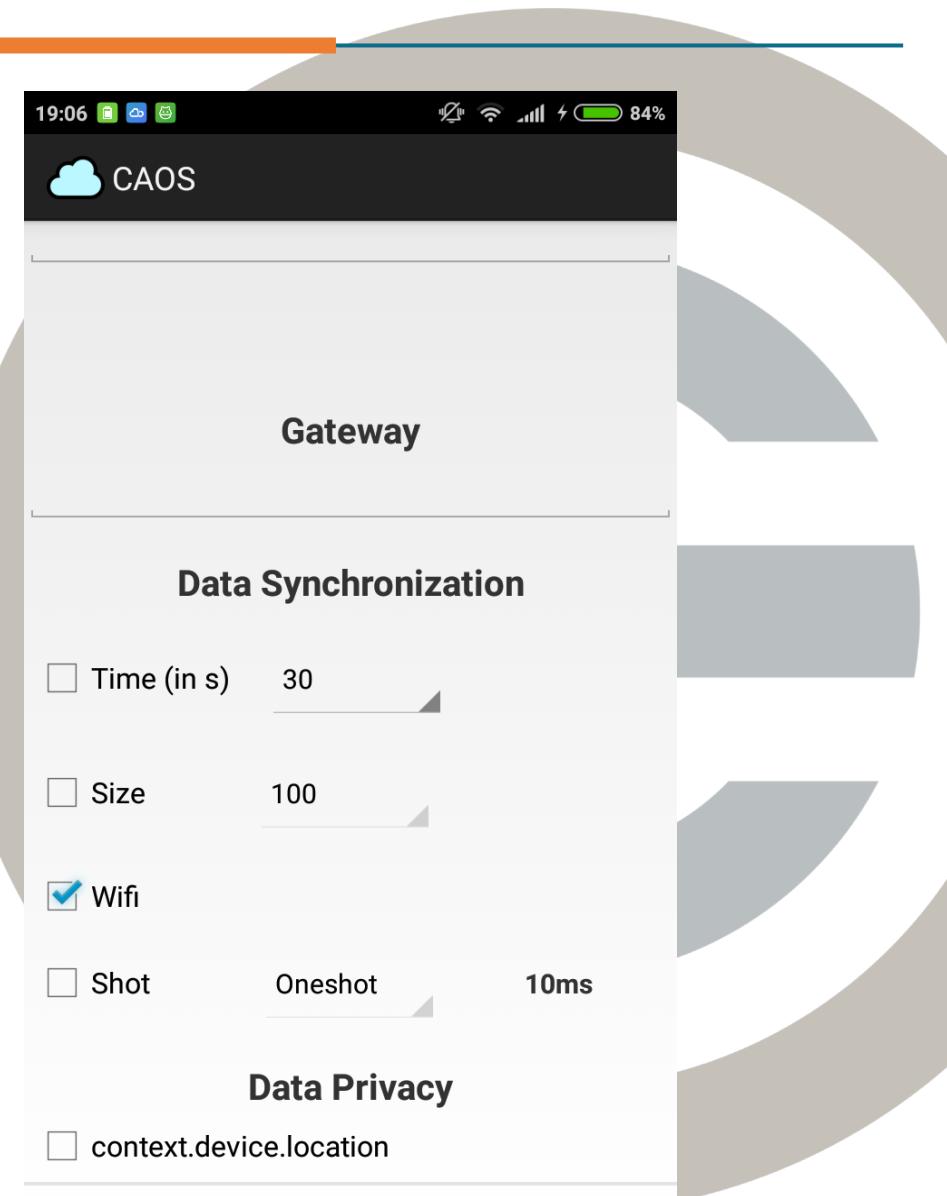
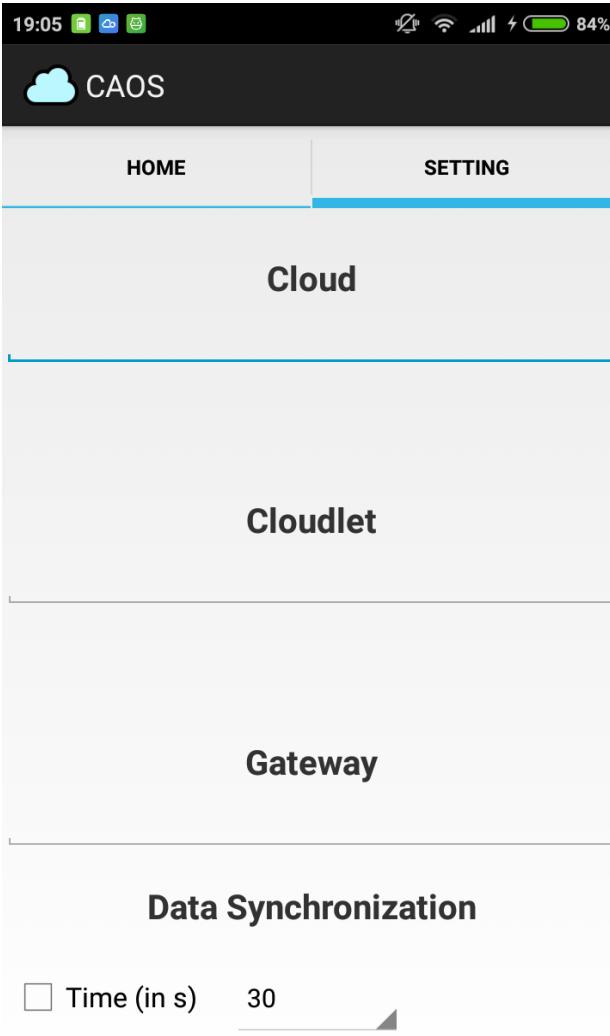
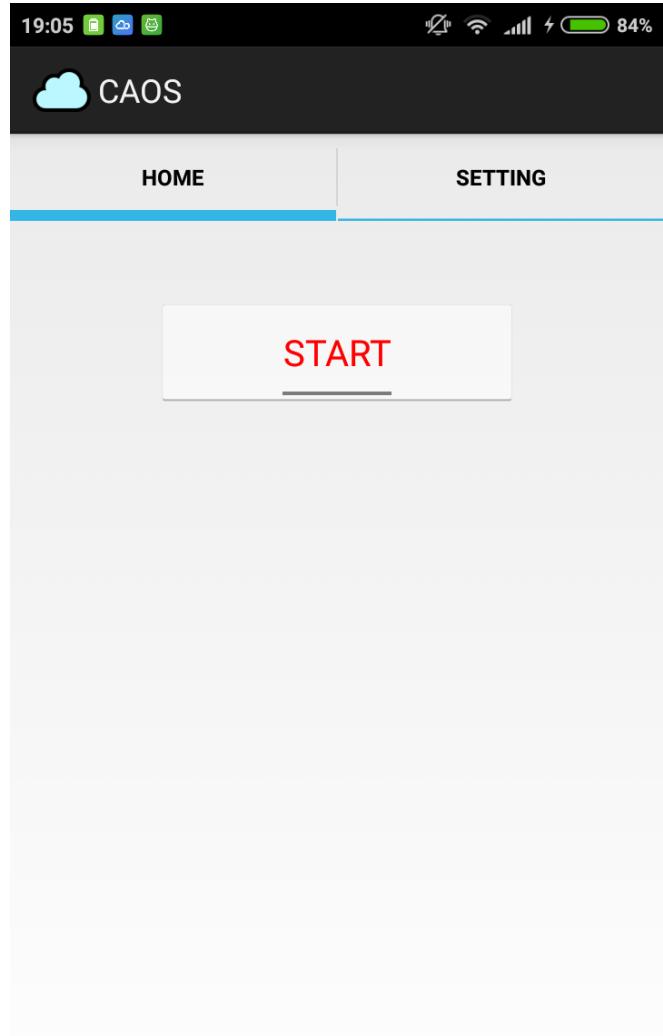


Hands on

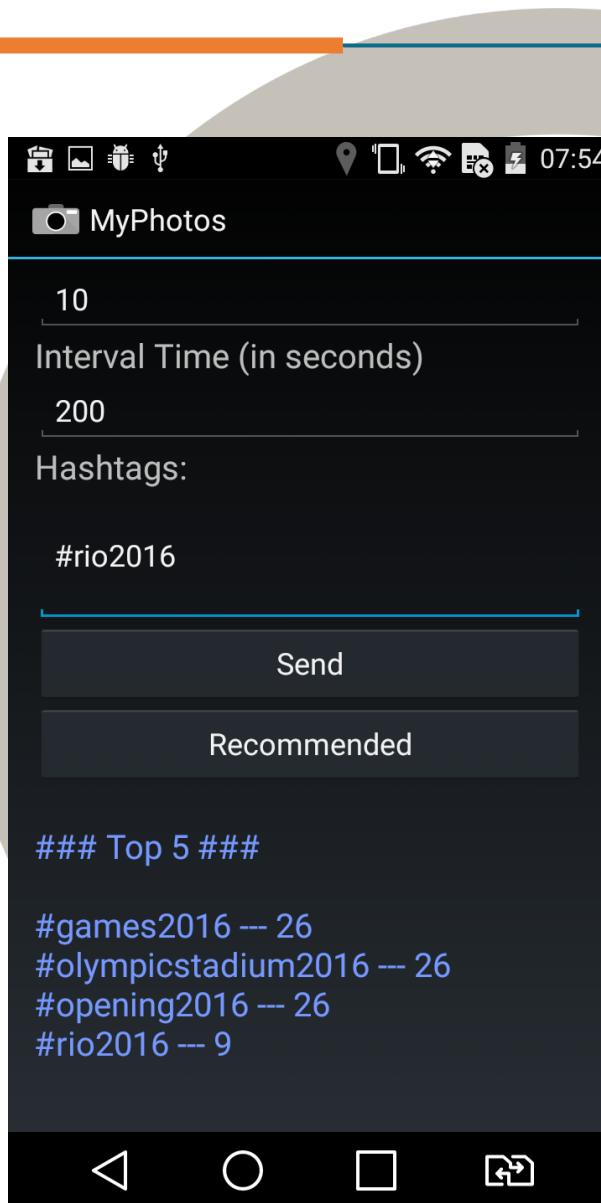
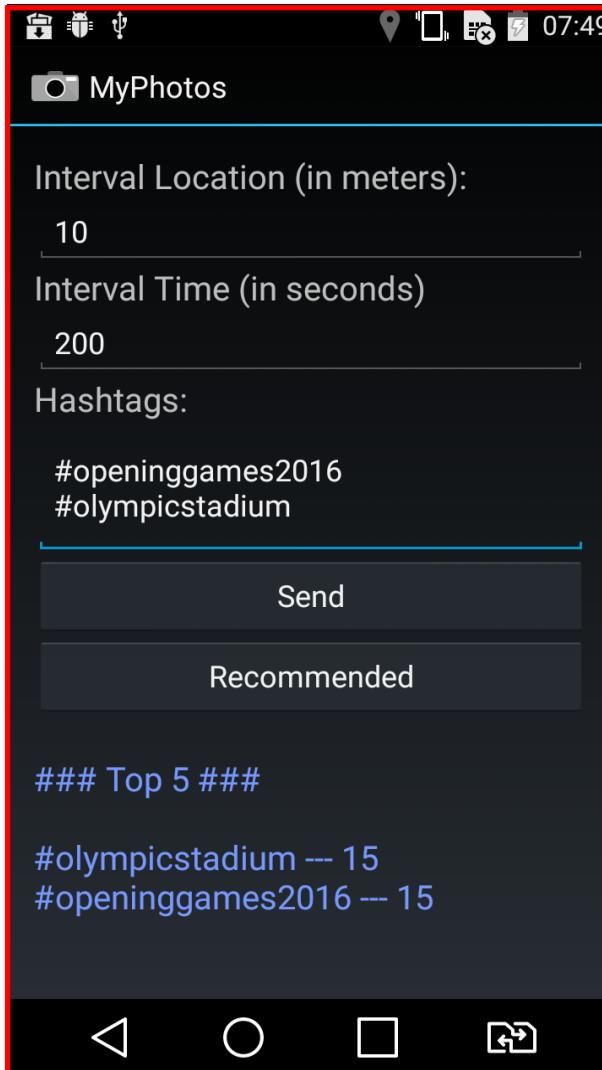
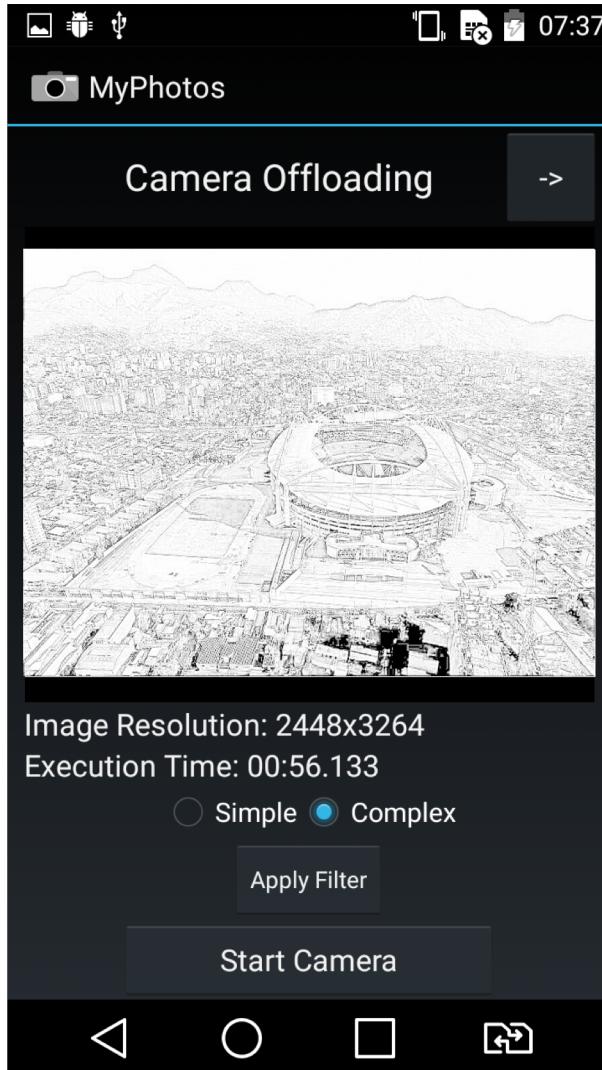
3. Instalação e configuração do CAOS Services

- Instalar o CAOS Service (APK) no dispositivo móvel
- Configurar o CAOS Service
 - Iniciar o aplicativo
 - Apertar em Settings
 - Se um Cloudlet estiver sendo executado na mesma rede local do dispositivo móvel irá aparecer, no campo Cloudlet, o endereço IP e Porta do serviço remoto
 - Configurar como os dados contextuais serão sincronizados e como eles serão sensíveis. Basta marca as opções.
 - Depois de configurado, basta apertar em Start

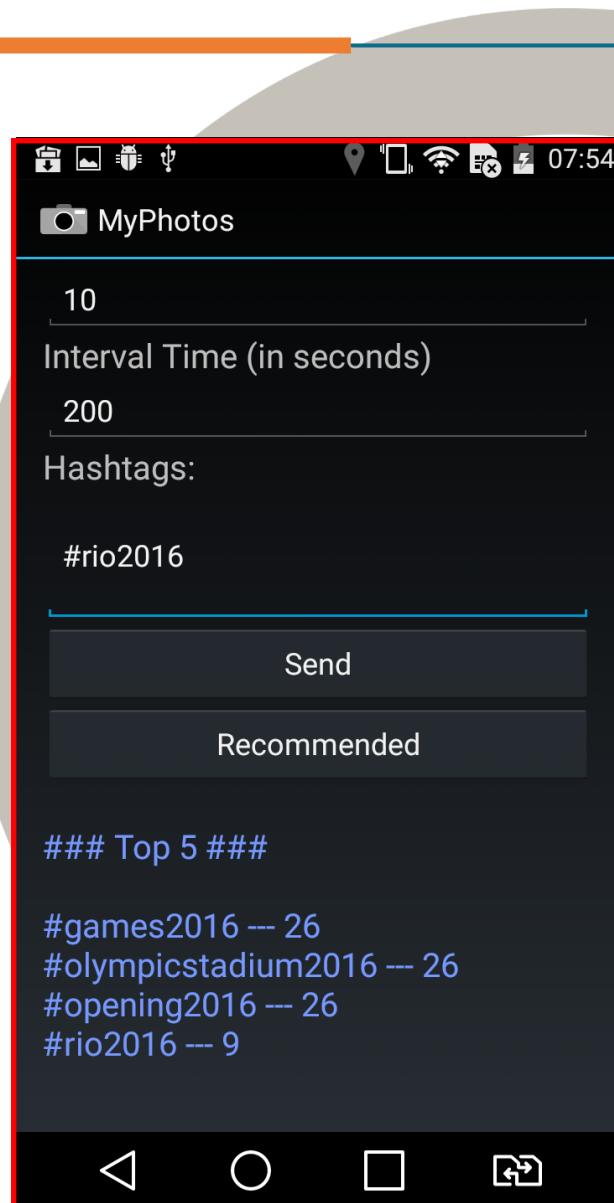
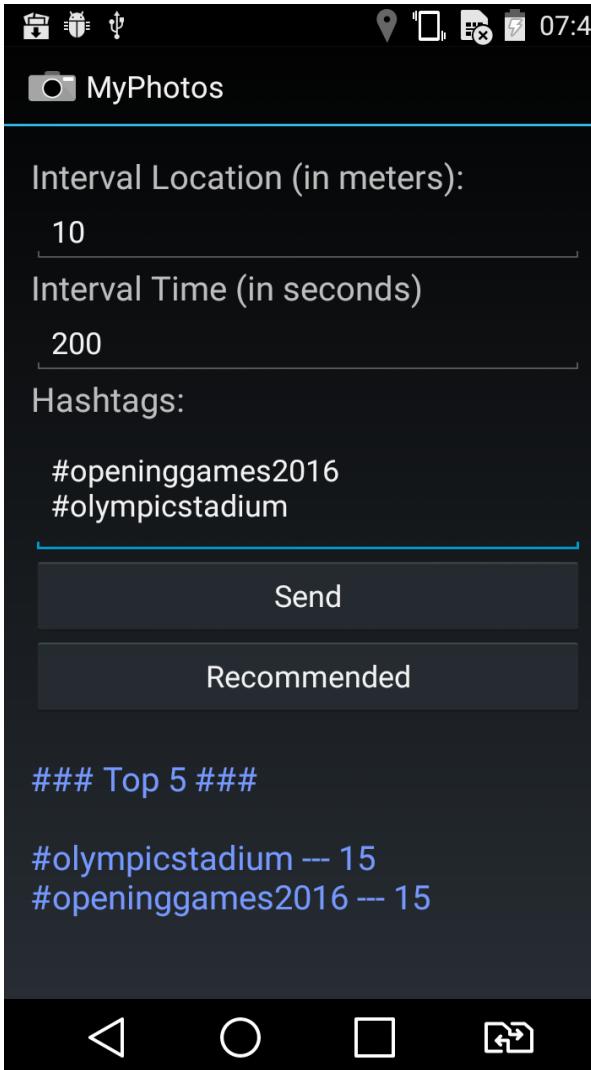
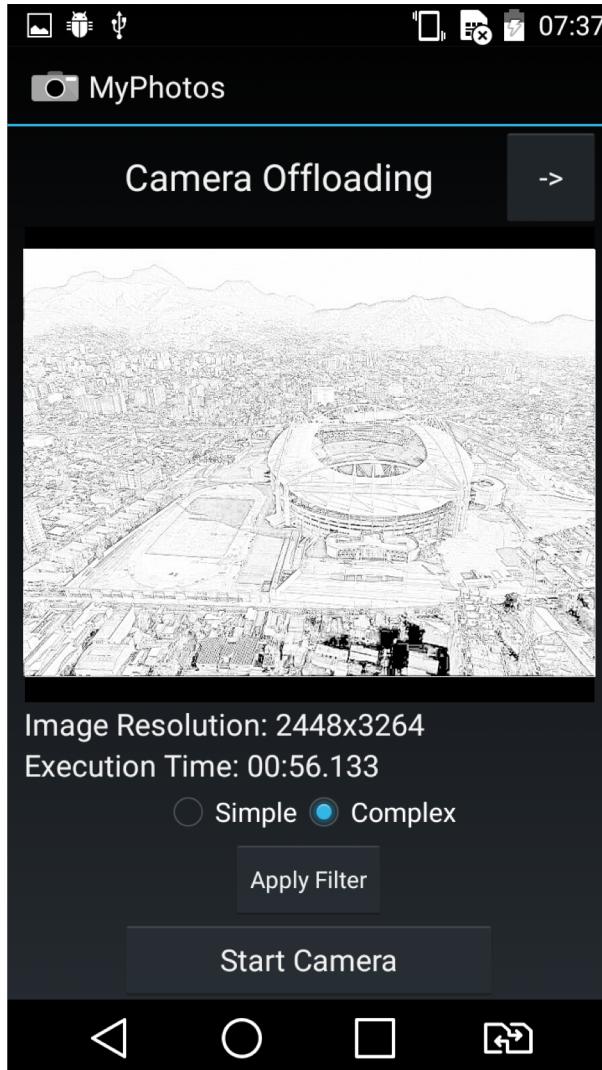
Hands on



Hands on (Executando o MyPhotos)



Hands on (Executando o MyPhotos)



Tendências e Desafios de Pesquisa

- Custos para transferência de dados brutos para o processamento de inferências em serviços de nuvem.
- Alta latência na comunicação entre dispositivos móveis e a nuvem.
- Segurança e privacidade.
- Consumo de energia.
- Disponibilidade de infraestruturas de MCC em larga escala.
- Falta de “killer applications”
 - Deep learning e processamento de vídeo são fortes candidatos

Conclusão e Trabalhos Futuros

- Aplicações móveis cada vez mais complexas
- Integração entre MCC e CAM é uma tendência (realidade?!)
- Trabalhos Futuros
 - Fazer uma refatoração completa dos serviços do CAOS
 - Microserviços para lidar com elasticidade
 - Adicionar novas features ao framework
 - Deploy automático de CACs no smartphone e APKs no CAOS Controller
 - Fazer offloading entre dispositivos móveis (device-to-device)

Obrigado

Perguntas?

Fernando A. M. Trinta <fernando.trinta@dc.ufc.br>

Paulo A. L. Rego <pauloalr@ufc.br>

Francisco A. A. Gomes <franciscoanderson@great.ufc.br>

Em breve: <http://caos.great.ufc.br>



UNIVERSIDADE
FEDERAL DO CEARÁ