Kumo: um serviço para portabilidade em multi-nuvens heterogêneas

Marcus Rafael Xavier
Universidade Federal de Pernambuco
Jornalista Aníbal Fernandes, s/n,
Cidade Universitária
Recife, Pernambuco 50740-560
mrxl@cin.ufpe.br

Carlos Andre G. Ferraz
Universidade Federal de Pernambuco
Jornalista Aníbal Fernandes, s/n,
Cidade Universitária
Recife, Pernambuco 50740-560
cagf@cin.ufpe.br

Ioram Schechtman Sette
Centro de Estudos e Sistemas
Avançados do Recife (CESAR)
Rua Bione, 220, Bairro do Recife
Recife, Pernambuco 50030-390
iss@cesar.org.br

ABSTRACT

Cloud computing is maturing and becoming ubiquitous in people's daily lives. As a result, cloud environments are providing more and more services with better quality of service. Cloud customers, however, have suffered from the vendor lock-in problem, in such a way that those who wish to migrate to another cloud provider require partial or total reimplementation of applications and virtual infrastructure. Moreover, the problem of heterogeneity among distinct cloud environments makes it difficult for the portability of resources between them. Therefore, this work focuses on the development of an ontology to handle multi-cloud heterogeneity, and thus, bring interoperability in the form of a service to perform the portability of virtual machines between different providers.

KEYWORDS

Multi-nuvem, Aprisionamento, Heterogeneidade, Portabilidade.

1 INTRODUÇÃO

Com o passar dos anos, várias tecnologias como virtualização, computação em grade, e arquiteturas orientadas a serviço (SOA) têm amadurecido e contribuído significativamente para tornar a computação na nuvem viável [9].

A computação na nuvem permitiu que os serviços web (do inglês, *web services*) fossem ainda mais disseminados, possibilitando a hospedagem de aplicações web em larga escala, e fazendo com que empresas dependam cada vez mais de recursos em nuvem para implantação de seus sistemas.

Juntamente com o atual amadurecimento e migração de infraestruturas para as nuvens, novos problemas surgem. No cenário com uma única nuvem destaca-se o aprisionamento tecnológico ou *vendor lock-in*, enquanto que no cenário multi-nuvem, cenário cujo o aprisionamento tecnológico não é presente, destaca-se o problema da heterogeneidade.

A solução pra esses e outros problemas da computação na nuvem deve-se começar por estratégias que tragam interoperabilidade para esses ambientes heterogêneos, permitindo assim que usuários se libertem do aprisionamento causado pelas implementações proprietárias e não padronizadas promovidas pelos provedores de serviços de nuvem.

In: XVIII Workshop de Teses de Dissertações (WTD 2018), Salvador, Brasil. Anais do XXIV Simpósio Brasileiro de Sistemas Multimídia e Web: Workshops e Pôsteres. Porto Alegre: Sociedade Brasileira de Computação, 2018.

ISBN 978-85-7669-435-9.

Inicialmente, é necessário o uso de alguma estratégia para que se possa conhecer as ações comuns a todas as nuvens, as comuns apenas a algumas nuvens, e ações que são específicas de cada nuvem (ver Figura 1). Logo, por ser indicada para representação formal de conhecimento, através de classes (*server*), relacionamento (*tem um ou vários*), e atributos (*volume*), optou-se pelo uso de uma ontologia.

Neste trabalho é realizado o desenvolvimento de uma ontologia com foco em portabilidade de máquinas virtuais. Com isso, busca-se conhecer as operações sobre essas VMs de modo que se possa a partir dessa ontologia, desenvolver um serviço que possa realizar o gerenciamento de recursos, como máquinas virtuais, na estratégia de multi-nuvem heterogêneas.

Entretanto, a principal motivação é a especificação de um sistema chamado Kumo, que foca em realizar migração de máquinas virtuais entre multi-nuvens heterogêneas. As migrações de máquinas virtuais são atualmente um problema em multi-nuvem em ambientes homogêneos e heterogêneos, e esse é o desafio que esse trabalho trata.

2 CONTEXTUALIZAÇÃO

Essa seção é dedicada a contextualização dos problemas principais em ambientes de múltiplas nuvens, o de aprisionamento tecnológico, também conhecido na literatura como *vendor lock-in*, e a heterogeneidade das tecnologias e interfaces implementadas pelos provedores de serviços de nuvem (CSPs).

A computação em nuvem tornou-se uma tendência para empresas construírem suas infraestruturas virtuais, em vez de usar recursos locais (do inglês, *on-premises*). Com isso, essas empresas que utilizavam recursos locais iniciaram a migração para a nuvem, e aquelas novas empresas criadas a partir de então, já utilizam a nuvem como a plataforma para receber toda sua infraestrutura.

Com isso, um problema relevante na computação na nuvem surge. Uma vez que uma empresa estabelece sua infraestrutura em um provedor de nuvem, será difícil ou até mesmo impossível a migração para outro provedor. Sendo necessária a reimplementação parcial ou total da infraestrutura, bem como de aplicações desenvolvidas utilizando bibliotecas fornecidas pelo provedor, que são compatíveis apenas com seus serviços.

Quando essa situação acontece, é dito que uma empresa está aprisionada tecnologicamente ao seu provedor de nuvem, esse que não fornece opções na forma de ferramentas para facilitarem a migração para um outro provedor [3]. Assim, uma possibilidade não otimizada é a de baixar os dados e configurações feitas, para que possam ser importados e replicados em outro provedor. Porém

mesmo nesse caso mais manual de migração, um segundo problema pode acontecer, esse problema é a heterogeneidade.

No contexto da portabilidade de recursos entre nuvens, heterogeneidade é compreendida como a situação em que um usuário que é detentor de um recurso em um dado formato de arquivo proprietário, e que esse formato só pode ser compreendido pelo provedor que o usuário ou empresa está aprisionado. Com isso, mesmo de posse do arquivo que representa o recurso que é necessário migrar, não há garantia que o provedor de destino possa entender e executar esse recurso.

3 INTEROPERABILIDADE

Para propor qualquer solução para o problema de aprisionamento tecnológico, um bom ponto de partida é entender o que significa a ideia de trazer interoperabilidade para as nuvens. De acordo com o documento da organização Object Management Group [4], interoperabilidade *"deve ser vista como a capacidade de serviços de nuvem pública, serviços de nuvem privada e outros sistemas diversos dentro da empresa de entender as interfaces de aplicativos e serviços de cada um, a configuração, as formas de autenticação e autorização, formatos de dados, etc. a fim de trabalhar uns com os outros."*.

Na computação na nuvem, interoperabilidade está diretamente relacionada a compatibilidade entre os tipos e formatos de dados usados para comunicação entre interfaces de programação (APIs) como o JSON[1], entre linguagens de programação e aplicações que possam ser executadas por vários provedores. Também se pode considerar interoperabilidade entre componentes de infraestrutura, como arquivos referentes a máquinas virtuais (arquivos de *appliances*) executáveis por vários hipervisores, ou imagens de contêineres.

Sendo a interoperabilidade uma característica fundamental não apenas em ambientes de nuvem, mas também nos ambientes multi-nuvem, sejam eles homogêneos ou heterogêneos. Ainda existem muitos desafios para instalação, manutenção e operação desses ambientes pela falta dessa interoperabilidade. Em [9], oito desses desafios estão listados e aqui descritos quanto a sua motivação para ambientes de multi-nuvem.

1. **Rede** - Conectividade (gerência da infraestrutura física ou virtual de rede); Endereçamento (mobilidade do endereço IP); *Naming* (nomes para identificar recursos nas várias nuvens); *Multicasting* (envio seletivo de pacotes para dispositivos em nuvens diferentes).
2. **Economia** - Mercado (estratégias de negociação referente ao uso recursos de outras nuvens); Preço (definições de como comprar/vender/alugar/emprestar recursos com outras nuvens); Contabilidade e Faturamento (quantificação de uso de recursos de/por terceiros e estratégias de pagamento, pré e pós-pago).
3. **Provisionamento** - Descoberta (descoberta de recursos em outras nuvens); Seleção (escolha por provedores baseado nas garantias e qualidade de serviço); Alocação (estratégias de reservar recursos previamente ou sob demanda em outras nuvens).
4. **Acordo de Nível de Serviço (SLA)** - Gerência (gerenciamento de contratos SLA pelas nuvens); Acordo de Nível de Serviço da Federação (contrato que determina quais condições um nuvem precisa para interoperar com outras nuvens); Monitoramento e Dependência (verificação das nuvens quanto o cumprimento dos contratos de SLA).
5. **Segurança** - Confiança (garantia de confiança baseado em ranking de provadores mais confiáveis); Autorização e Autenticação (acesso e ações que um nuvem poderá usar de outra quando previamente autenticada); Políticas e Interoperabilidade Semântica (garantia de equivalência semântica quanto às permissões das nuvens).
6. **Autonomia** - gerência autônoma dos recursos nas múltiplas nuvens.
7. **Monitoramento** - garantia de coleta de métricas de utilização em múltiplas nuvens.
8. **Portabilidade** - Portabilidade de Dados (padronização de dados e de seus formatos); Portabilidade de VMs (migração de VMs entre nuvens homogêneas e heterogêneas).

3.1 Estratégias

Em [9] são mencionadas quatro alternativas para trazer interoperabilidade para nuvens, divididas em duas categorias. Primeiramente são apresentadas duas categorias onde a mudança necessária para alcançar esse requisito é implementada e centrada nos provedores de nuvem (*provider-centric*), e em seguida são apresentadas outras duas abordagens, centradas e implementadas nos clientes de nuvem (*client-centric*).

3.1.1 ***Nuvem híbrida.*** A interoperabilidade de nuvem híbrida é uma abordagem *provider-centric* que permite que um provedor de serviço tenha uma parceria com uma nuvem pública. Assim que a parceria for formada, se o uso de recursos da nuvem privada aumentar e não houverem mais recursos disponíveis, a nuvem privada utiliza recursos do provedor de serviços de nuvem pública.

3.1.2 ***Nuvem federada.*** A federação de nuvem é outra abordagem *provider-centric* para fornecer interoperabilidade na computação em nuvem. No cenário de federação, os provedores de nuvem compartilham seus recursos ociosos dos *data centers*, com os regulamentos e o acordo de nível de serviço (SLA) definido pelos membros da federação.

3.1.3 ***Aplicações multi-nuvem.*** Aplicativos com várias nuvens são uma abordagem de interoperabilidade *client-centric* que permite aos desenvolvedores escreverem suas aplicações independente da tecnologia de nuvem. Se necessário migrar para outro provedor, o aplicativo deve estar pronto para ser reimplantado sem muito esforço no novo ambiente. Os *frameworks* e bibliotecas como o projeto Apache Libcloud[2] podem ser usados para auxiliar esse desenvolvimento.

3.1.4 ***Broker multi-nuvem.*** O broker multi-nuvem é outra abordagem de interoperabilidade *client-centric*. Nela, as requisições feitas pelos usuários são recebidas pelo broker, que cria uma nova requisição com a mesma semântica para a nuvem do provedor de serviço escolhida pelo usuário.

[1] www.json.org

[2] libcloud.apache.org

4 TRABALHOS RELACIONADOS

Na presente literatura são documentados trabalhos relacionados a identificação de formas para conquistar a interoperabilidade em ambientes multi-nuvens. No sentido de migração de aplicações entre provedores, uma proposta encontrada em [7] foca na combinação da arquitetura de micro-serviços para a separação de responsabilidades, e na utilização de contêineres para empacotamento das aplicações e de suas dependências.

Na estratégia de utilização de contêineres, é possível portar as aplicações devido vários provedores utilizarem as mesmas tecnologias para a execução e orquestração desses contêineres. O Docker[3] é atualmente a plataforma para contêineres mais difundida entre os provedores, enquanto que a ferramenta de orquestração Kubernetes[4] é a mais adotada para realizar interoperabilidade de contêineres Docker, permitindo migração dos contêineres entre infraestruturas homogêneas (que executem Docker).

Diferentemente, em [5] é descrita uma outra alternativa para interoperabilidade multi-nuvem. Nessa abordagem, usuários administradores da plataforma *LambdaLink* podem cadastrar imagens de discos virtuais que podem ser utilizadas para o lançamento de instâncias em múltiplas nuvens heterogêneas. Entretanto, o problema da portabilidade de máquinas virtuais entre as nuvens não é tratado pela plataforma. Em resumo, tal plataforma permite apenas o compartilhamento de imagens de disco.

Diferentemente de [7] e [5], este trabalho trata o problema de máquinas virtuais criadas em ambientes de nuvem, possibilitando a portabilidade das VMs entre multi-nuvens de tecnologias heterogêneas. Mesmo com propósito diferente, a proposta [7] se mostra promissora para a implantação do serviço Kumo, que pode ser implementado utilizando micro-serviços (permite a evolução do Kumo para agregar novas funcionalidades, como o gerenciamento multi-nuvem) quanto o uso contêineres para implantação do Kumo, trazendo a possibilidade de migrar o serviço e ter múltiplas instâncias do mesmo. Já como no trabalho [5], Kumo poderia ter um repositório de imagens de disco para facilitar além da migração, a criação de novas máquinas virtuais.

5 O PROBLEMA DA PORTABILIDADE

Como mencionado, a portabilidade é um desafio na computação em nuvem. O domínio de alguns provedores de nuvem em relação a outros aumenta o aprisionamento de clientes a uma única plataforma de nuvem, impedindo a portabilidade de dados ou software criado por eles [1]. Nesta seção, algumas motivações e desafios relevantes em relação à portabilidade são discutidos.

Em [1], alguns pontos interessantes são discutidos sobre a comunidade de computação em nuvem. Em primeiro lugar, a comunidade deve promover padrões comuns e interoperabilidade entre os serviços de nuvem pública, o que maximizaria os ganhos econômicos para os provedores e também seria um ponto de partida para transformar a portabilidade multi-nuvem em realidade. Em seguida, a comunidade precisaria desenvolver a mentalidade para divulgar amplamente o princípio de facilitação do acesso a dados para conquistar a portabilidade, evitando assim o aprisionamento.

[3] www.docker.com
[4] kubernetes.io

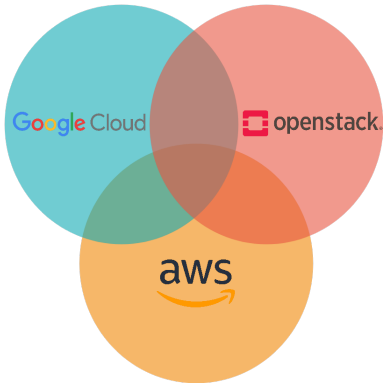

Figura 1: Nuvens selecionadas para a ontologia

Um exemplo disso é a comunidade OpenStack [5], que é um projeto de código aberto e gratuito para lançar nuvens privadas e públicas. Embora o OpenStack seja desenvolvido por qualquer pessoa que queira contribuir e que muitos colaboradores do projeto sejam funcionários dos provedores de nuvem, a comunidade garante que projetos de código aberto e até mesmo fechados criados por fornecedores, interoperem com o OpenStack.

As motivações para a portabilidade são, além da eliminação do aprisionamento, a possibilidade de importar e exportar dados (por exemplo, motivado por uma outra nuvem que tenha uma estratégia de recuperação de desastres melhor); SLA aprimorado (por exemplo, um recurso pode ser executado melhor no ambiente de nuvem de outro provedor com um hipervisor de melhor desempenho para execução de uma máquina virtual) e também a opção de escolha baseada no preço (outra nuvem pode ter um data center em uma região com eletricidade mais barata e assim permitir menor preço [3]).

Tão importante quanto migrar os dados de uma nuvem para outra, é migrar componentes de infraestrutura para outra nuvem, por exemplo, para satisfazer aos requisitos tecnológicos de uma aplicação [1]. A migração de uma máquina virtual de uma nuvem heterogênea envolve muitas decisões, por exemplo, decidir se a migração será executada de forma *live* (neste caso, o usuário continua a usar a máquina virtual original até que uma cópia dela seja lançada na nuvem de destino); se o hipervisor da máquina virtual na nuvem de destino pode executá-la; decidir como configurar a rede; o que fazer se a máquina virtual tiver um volume anexado; e assim por diante.

Nesse sentido, considerando uma migração feita de forma manual por um operador de nuvem, para cada passo necessário durante a migração, esse operador deverá conhecer todos os comandos necessários para a execução da migração, isso para cada uma das tecnologias das nuvens heterogêneas, iniciando em como autenticar em todas as nuvens envolvidas na operação. Para lidar com tais preocupações, a literatura mostra que o conceito de ontologia pode ser usado para mapear recursos às ações possíveis ao recurso "máquina virtual". Na seção 6 essa abordagem é discutida e desenvolvida.

6 ONTOLOGIA

Conforme mencionado em [1], no contexto da computação em nuvem, uma ontologia pode ser usada para obter interoperabilidade entre diferentes provedores de nuvem e seus serviços.

O conhecido trabalho "*O que são ontologias, e por que precisamos delas?*"[2], define o que é uma ontologia:

> *Ontologia é um vocabulário de representação, geralmente especializado em algum domínio ou assunto. Mais precisamente, não é o vocabulário como tal que se qualifica como uma ontologia, mas os conceitos que os termos do vocabulário pretendem captar. Assim, traduzir os termos de uma ontologia de uma língua para outra, por exemplo, do inglês para o francês, não altera conceitualmente a ontologia.*

Devido a heterogeneidade das nuvens, se houver necessidade de portabilidade multi-nuvem, uma ontologia pode ser um recurso para entender o que é comum a todas as nuvens, o que é comum em algumas nuvens, e o que é específico de cada nuvem. Com isso, é possível conhecer onde a heterogeneidade nas nuvens acontece e também definir um *"vocabulário especializado"* comum que, parafraseando [2], permite traduzir os termos da ontologia de uma nuvem para outra, por exemplo, da Amazon Web Services para OpenStack, sem conceitualmente alterar a ontologia.

Na criação de uma ontologia para interoperabilidade, primeiramente é fundamental definir quais componentes dentre os vários disponíveis na computação em nuvem, são relevantes para a portabilidade das máquinas virtuais. Usando a experiência da utilização da nuvem, entende-se que a máquina virtual é um recurso gerenciado pelo serviço de computação. No entanto, uma observação que vem à mente é que a máquina virtual tem relação não só com o serviço de computação (*computing*), mas está relacionada ao serviço de rede (*networking*), imagem de disco (*image*), e discos virtuais (*volume*).

Para a ontologia desenvolvida neste trabalho, foi utilizada uma estratégia *bottom-up* onde as nuvens foram analisadas a partir de suas definições de políticas de autorização, para que em seguida fosse possível encontrar aqueles recursos que são comuns as políticas de cada uma das nuvens, e assim mapear as ações comuns a cada um desses recursos. Esse trabalho foca exclusivamente em recursos e em ações comuns às três nuvens, que é representado pela interseção entre os conjuntos na Figura 1.

Dessa forma, a primeira decisão tomada para iniciar o desenvolvimento da ontologia, foi estabelecer que o recurso foco da ontologia seria a máquina virtual (*server*, na ontologia), não apenas considerando como parte do serviço de computação, mas como parte de todos os serviços que são relacionados a ela, como rede, imagem de disco e volume. Por exemplo, do componente de armazenamento em bloco foi considerado o recurso volume, isso porque a máquina virtual pode ter um volume associado. Outro exemplo ocorre no serviço de rede, onde foi considerada a interface de rede, pois este recurso associado a máquina virtual para trazer comunicação a ela via rede. Em resumo, não foram adicionados à investigação recursos de serviços que não têm relação nenhuma com o recurso principal da ontologia, a máquina virtual.

[5] www.openstack.org/community

Dando continuidade ao desenvolvimento da primeira versão da ontologia, foram escolhidas as nuvens que seriam usadas nesse trabalho, que foram:

Amazon Web Services (AWS) é uma plataforma de nuvem pública que oferece computação, armazenamento de banco de dados, entrega de conteúdo e etc. Atualmente, é o serviço de nuvem pública mais usado [8].

Google Cloud Platform (GCP) é a opção de serviço de nuvem pública da Google. Ela fornece menos serviços do que a AWS, mas por outro lado, tem sido a primeira opção dos usuários de nuvem quando pensam em migrar de seu provedor atual para um novo provedor [8].

OpenStack (OS) é um software de código aberto para implantação de nuvens públicas e privadas, capaz de gerenciar recursos de computação, armazenamento e rede [6].

Após a escolha pelas três nuvens, a etapa seguinte foi definir quais informações seriam usadas para extrair os recursos e suas ações. Para isso, chegou-se as políticas de autorização do Amazon Web Services [6] (AWS), do Google Cloud Platform [7] (GCP) e do OpenStack [8] (OS). A escolha por essas políticas veio por serem recursos estruturados e que contemplam todas as possibilidades em ações que um usuário, com o papel de administrador pode realizar em cada uma dessas nuvens.

Para encontrar as políticas na AWS foi necessário fazer login com uma conta, navegar para o Identity and Access Management (IAM) e acessar as políticas para o serviço de computação EC2, para o papel denominado *AmazonEC2FullAccess*. As políticas do GCP, análogas à AWS, foram encontradas no seu serviço de IAM, onde as políticas de autorização para o papel *Compute Admin* puderam ser encontradas. No OpenStack, foi necessário acessar a página do GitHub[9] para todos os serviços necessários: o Nova (computação), o Cinder (volume), o Neutron (rede) e o Glance (imagem).

Após a criação de uma tabela para receber todas as políticas de autorização para as três nuvens, as documentações dessas nuvens também foram analisadas, uma a uma. O objetivo dessa etapa foi descobrir possíveis ações que tivessem sintaxe diferentes, mas que semanticamente representassem a mesma ação. Conceitualmente, o resultado do trabalho até este ponto é representado pela interseção entre os três conjuntos na Figura 1.

Em seguida foi criado um mapeamento entre os recursos e suas ações para as três nuvens. Um fragmento desse mapeamento pode ser visto na Tabela 1 que representa o recurso *Image* (imagem) nas três nuvens. Assim, é possível ver como foram mapeadas cada uma das ações de *Image*, de maneira que todas as ações da ontologia (última coluna da Tabela 1) sejam nomes em alto nível e representem semanticamente o que cada uma das ações significa.

O resultado conquistado na ontologia pode ser conferido na Figura 2, que representa a ontologia na linguagem *Ontology Web Language* (OWL) e visualmente representada na ferramenta Protegé[10]. Na Figura 2, são mostrados todos os recursos e as ações em alto nível com nomes padronizados. Isso busca facilitar o tratamento de futuras aplicações desenvolvidas tendo como base a

[6] aws.amazon.com
[7] cloud.google.com
[8] www.openstack.org
[9] github.com/openstack
[10] protege.stanford.edu

Tabela 1: Exemplo de mapeamento na ontologia para o recurso Image

AWS	GCP	OS	Ontologia
Copy, Create, Import, Register	Create	Add, CopyFrom, Upload	Create
—	Get, GetFromFamily, GetIamPolicy	Download, Get	Read
Deregister, Register	Deprecate, SetIamPolicy, Update, UserReadOnly	Communitize, Modify, Publicize	Update
Deregister	Delete	Delete	Delete
—	List	—	List
—	SetLabels	—	Add
—	SetLabels	—	Remove

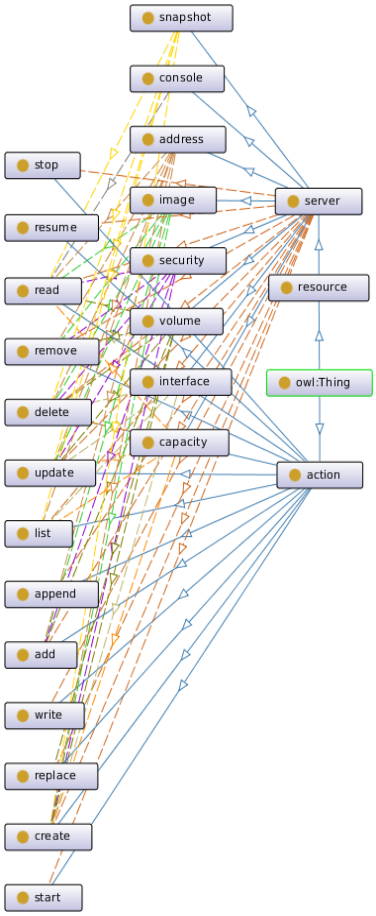

Figura 2: Primeira versão da ontologia: Conceitos, atributos e relacionamentos

ontologia, como Kumo que é um serviço especializado na realização de migrações de máquinas virtuais em ambientes multi-nuvem heterogêneos.

7 KUMO

Nesta seção, o serviço Kumo, sua API e drivers são detalhados. Kumo é a palavra na língua japonesa para nuvem e é representado

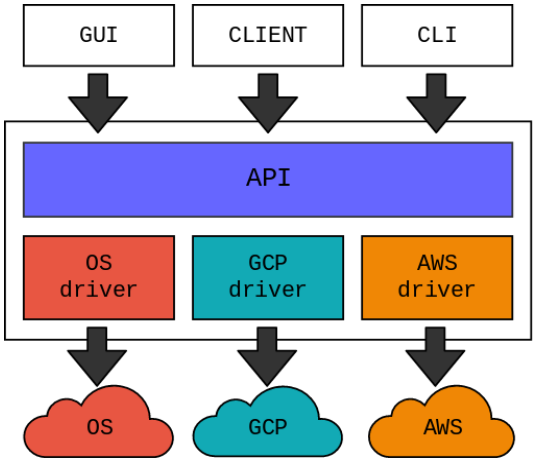

Figura 3: Arquitetura do Kumo conductor

pelo caractere do alfabeto Kanji 雲. Kumo é responsável por implementar e expor uma API web que permitirá aos usuários registrar suas VMs e também acionar outras ações, como migrar. Para isso, Kumo possui o conceito de drivers, que são os especialistas em realizar operações necessárias à migração.

API destina-se a expor a execução remota de código e é necessário para resolver o problema de heterogeneidade através de sua interface unificada, em que os usuários solicitarão migrações para as nuvens integradas ao Kumo, de acordo com as definições da ontologia;

Drivers são as implementações de código que executam as ações necessárias nas nuvens de origem e destino (por exemplo, autenticação). É a maneira de tornar o Kumo capaz de interoperar com as múltiplas nuvens;

Conductor é a aplicação responsável por iniciar a API e executar as ações definidas e implementadas pelos drivers, bem como realizar o controle de ciclo de vida das ações definidas pelo serviço.

Tendo apresentado os principais componentes que compõem o Kumo, é interessante entender o fluxo onde um usuário solicita uma migração, até o estágio em que a VM é provisionada e entra em execução na nuvem destino. Na Figura 4, é mostrado um processo de alto nível de migração de uma máquina virtual.

Na primeira ação da Figura 4, (1) solicita a migração de uma VM da nuvem de origem para uma nuvem de destino por meio da API

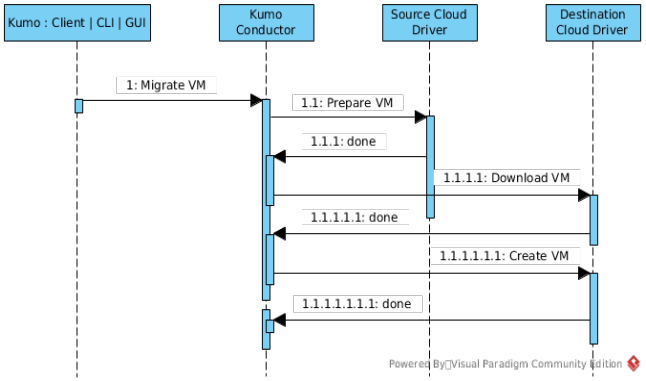

Figura 4: Diagrama de sequência da migração usando Kumo

do Kumo executada pelo conductor. Em (1.1), o processo de preparação de VM na nuvem de origem é iniciado pelo driver da nuvem de origem. A fase de preparação pode incluir o desligamento da VM; criação de um *snapshot*; buscar configurações de rede e armazenamento; obter metadados. Após essas etapas, o Kumo conductor recebe todas as informações necessárias sobre essa VM da nuvem de origem (1.1.1). Caso nenhum erro ocorra, o próximo passo é acionado (1.1.1.1). Nessa fase, o driver Kumo da nuvem de destino executará o registro do arquivo da VM e também replicará todas as configurações que foram coletadas da nuvem de origem para a de destino. Como o conductor sabe que todas as configurações foram feitas com sucesso (1.1.1.1.1), ele solicita a criação da VM na nuvem de destino (1.1.1.1.1.1), garantindo também todas as configurações que realizamos na VM e com isso a migração termina (1.1.1.1.1.1.1).

7.1 Ferramentas Kumo

Nesta seção, as ferramentas que permitem a operação do Kumo são detalhadas. Essas ferramentas possibilitam a conexão com o Kumo, e o desenvolvimento de uma aplicação para execução no serviço, permitindo uma melhor interação dos operadores com o serviço.

7.1.1 Kumo Client. Os usuários do Kumo podem precisar criar seus próprios aplicativos e *scripts* para automatizar suas migrações e cargas de trabalho. Por causa disso, o cliente Kumo será a maneira mais fácil de acessar, gerenciar e migrar os recursos por meio de código.

7.1.2 Kumo CLI. Para ter uma melhor experiência de usuário, os operadores que precisam gerenciar manualmente grandes quantidades de recursos, seja através de *scripts* de automação ou por meio do terminal do sistema operacional.

7.1.3 Kumo GUI. Opcionalmente, será possível operar o serviço Kumo para executar migrações usando um navegador web através da sua interface gráfica (GUI). Após a integração desse componente ao ecossistema Kumo, os usuários poderão gerenciar com mais rapidez e facilidade seus recursos em várias nuvens, mesmo usuários sem muita experiência em operações.

8 CONCLUSÕES

Atualmente, a primeira versão de uma ontologia para ambientes multi-nuvens heterogêneos com foco na portabilidade de máquinas virtuais foi desenvolvida. A abordagem utilizada para o desenvolvimento de tal ontologia foi *bottom-up*, uma vez que foram analisadas políticas de autorização de três nuvens heterogêneas para que fossem extraídos os recursos e ações comuns a elas, chegando assim ao que se chama de *core* da ontologia.

Também foi detalhada uma proposta de arquitetura para o serviço que implementará os conceitos definidos pela ontologia. Este serviço chamado Kumo, será uma outra contribuição da pesquisa e também será responsável pela portabilidade de máquinas virtuais entre nuvens heterogêneas. Neste trabalho, é definida a arquitetura do serviço Kumo e seus componentes que serão implementados durante a execução da pesquisa.

Devido a sua arquitetura flexível, outros conceitos poderão posteriormente ser integrados ao serviço Kumo, como a possibilidade de gerenciamento de infraestrutura de maneira unificada ou ainda a adição de capacidades de migração de outros componentes de infraestrutura, como contêineres. Para isso, é necessária previamente a evolução da ontologia para mapear os novos recursos, bem como suas ações.

9 AGRADECIMENTOS

Este trabalho foi possível pela bolsa de mestrado fornecida pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES). Programa PROEX, Edital 0487 e Processo 1741270.

REFERÊNCIAS

[1] Nick Bassiliades, Moisis Symeonidis, Panagiotis Gouvas, Efstratios Kontopoulos, Georgios Meditskos, and Ioannis Vlahavas. 2018. PaaSport semantic model: An ontology for a platform-as-a-service semantically interoperable marketplace. *Data & Knowledge Engineering* 113 (2018), 81–115. https://doi.org/10.1016/j.datak.2017.11.001

[2] B. Chandrasekaran, J. R. Josephson, and V. R. Benjamins. 1999. What are ontologies, and why do we need them? *IEEE Intelligent Systems and their Applications* 14, 1 (Jan 1999), 20–26. https://doi.org/10.1109/5254.747902

[3] Divyaa Manimaran Elango, Frank Fowley, and Claus Pahl. 2018. An Ontology-Based Architecture for an Adaptable Cloud Storage Broker. In *Advances in Service-Oriented and Cloud Computing*, Zoltán Ádám Mann and Volker Stolz (Eds.). Springer International Publishing, Cham, 86–101.

[4] Object Management Group. 2017. *Interoperability and Portability for Cloud Computing: A Guide* (2.0 ed.). Object Management Group, https://www.omg.org/cloud/deliverables/CSCC-Interoperability-and-Portability-for-Cloud-Computing-A-Guide.pdf.

[5] Kate Keahey, Pierre Riteau, and Nicholas P. Timkovich. 2017. LambdaLink: An Operation Management Platform for Multi-Cloud Environments. In *Proceedings of the10th International Conference on Utility and Cloud Computing (UCC '17)*. ACM, New York, NY, USA, 39–46. https://doi.org/10.1145/3147213.3147224

[6] OpenStack. 2018. Open source software for create private and public clouds. (2018). https://www.openstack.org

[7] R. Pellegrini, P. Rottmann, and G. Strieder. 2017. Preventing vendor lock-ins via an interoperable multi-cloud deployment approach. In *2017 12th International Conference for Internet Technology and Secured Transactions (ICITST)*. 382–387. https://doi.org/10.23919/ICITST.2017.8356428

[8] Statista. 2018. Public cloud service usage worldwide 2018. (2018). https://www.statista.com/statistics/511467/worldwide-survey-public-coud-services-running-application

[9] Adel Nadjaran Toosi, Rodrigo N. Calheiros, and Rajkumar Buyya. 2014. Interconnected Cloud Computing Environments: Challenges, Taxonomy, and Survey. *ACM Comput. Surv.* 47, 1, Article 7 (May 2014), 47 pages. https://doi.org/10.1145/2593512