Expandindo uma Arquitetura para HPC em Nuvens Computacionais Utilizando Conceitos de Computação Autonômica

Emanuel F. Coutinho[1], **Gabriel A. L. Paillard**[1]
Leonardo O. Moreira[1], **Ernesto Trajano de Lima**[1]

[1] Instituto Universidade Virtual (UFC Virtual)
Universidade Federal do Ceará (UFC) – Fortaleza – CE – Brasil

{emanuel,gabriel,leoomoreira,ernesto}@virtual.ufc.br

Abstract. *High Performance Computing (HPC) has high application in solving complex problems in several areas, usually requiring a lot of processing and bandwidth. Cloud Computing has received much attention from both industry and academia in recent years, due to its dynamic and flexible character. Autonomic Computing mechanisms, such as control loops and rules, can be employed to improve the characteristics of both environments, reducing human intervention and improving quality. The aim of this work is to propose features expansions of an HPC architecture for a Cloud Computing environment using concepts of Autonomic Computing.*

Resumo. *Computação de Alto Desempenho (HPC) tem elevada aplicação na resolução de problemas complexos nas mais diversas áreas, normalmente exigindo muito processamento e largura de banda. Computação em Nuvem tem recebido muita atenção tanto da indústria quanto da academia nos últimos anos, devido ao seu caráter dinâmico e flexível. Mecanismos de Computação Autonômica, como loops de controle e regras, podem ser empregados para melhorar características dos dois ambientes citados, reduzindo a intervenção humana e melhorando a qualidade. O objetivo desse trabalho é propor expansões das funcionalidades de uma arquitetura de HPC proposta sobre um ambiente de Computação em Nuvem utilizando conceitos de Computação Autonômica.*

1. Introdução

Aplicações de Computação de Alto Desempenho (*High Performance Computing* - HPC) focam em problemas complexos em ciências, engenharias e negócios [Rodrigo Álvarez et al. 2015]. Tais aplicações são caracterizadas por requererem alta largura de banda, redes complexas e alto poder computacional. Em relação à complexidade da plataforma subjacente, uma infraestrutura de HPC é mais simples que uma nuvem computacional, devido a Computação em Nuvem ser mais genérica e executar uma diversidade de aplicações, diferente do HPC. Além disso, naturalmente aplicações executando em nuvens computacionais possuem diferentes cargas de trabalho se comparadas ao HPC. Em nuvens computacionais, as aplicações possuem características dinâmicas atendidas por características específicas do ambiente, como virtualização e elasticidade. Em

contraste, HPC não oferece elasticidade nem adição de recursos virtuais, dificultando o desempenho de aplicações diante de mudanças nas cargas de trabalho.

O monitoramento de recursos computacionais, como CPU e memória, se torna essencial tanto para os provedores quanto para usuários de nuvens computacionais. Uma forma de monitorar aplicações em nuvem de modo mais efetivo é por meio da utilização de mecanismos de Computação Autonômica. Um sistema autônomo ou autonômico é composto por um conjunto de elementos autônomos, sendo este o componente responsável pela gestão do seu próprio comportamento em conformidade com políticas e por interagir com outros elementos autônomos, que fornecem ou consomem serviços computacionais [Kephart and Chess 2003]. Mecanismos de Computação Autonômica, como *loops* de controle e regras, podem ser empregados no monitoramento de uma nuvem computacional, possibilitando a adição ou remoção de recursos do ambiente conforme limiares de uso pré-estabelecidos.

O modelo de Computação em Nuvem tem como objetivo dois benefícios: redução de custos e flexibilidade. O primeiro consiste na redução de custos pela aquisição e composição da infraestrutura necessária para o atendimento das necessidades de negócio. Entretanto, empresas não necessitam adquirir uma infraestrutura complexa para hospedar suas aplicações, utilizando infraestruturas terceirizadas com garantia de serviço e pagando apenas pelos recursos utilizados. Além disso, a infraestrutura da nuvem pode ser composta sob demanda e por recursos heterogêneos e de baixo custo, que atenda o segundo benefício do modelo (flexibilidade). Neste sentido, a infraestrutura pode escalar tanto no nível de *hardware* quanto de *software*.

Nesse contexto, o objetivo desse trabalho é propor expansões para serem aplicadas como melhorias sobre uma arquitetura para HPC em nuvens computacionais, proposta em [Paillard et al. 2015], utilizando conceitos do auto gerenciamento (*self-**).

2. Descrição da Arquitetura

A arquitetura proposta em [Paillard et al. 2015] apresenta em alto nível componentes para a execução de aplicações HPC em nuvens computacionais. A Figura 1 exibe seus componentes, seus relacionamentos internos e relacionamentos com demais dispositivos.

O componente principal da arquitetura é o **HPCaaS** (*HPC-as-a-Service*), responsável pelo provimento de uma interface para a construção e execução de aplicações ou serviços que requerem a criação de uma infraestrutura de *cluster* para HPC, submissão de *jobs*, serviços de preempção e alocação em nuvens computacionais. Sua ideia é habilitar uma plataforma como serviço (PaaS) para HPC em nuvens de diferentes tipos. O usuário deve utilizar o componente **Interface** para a construção e execução de suas aplicações por meio de chamadas aos demais componentes do HPCaaS. O componente **Cria *Cluster*** é responsável pela criação e configuração do *cluster* na nuvem, utilizando como dados de entrada a localização da nuvem (ou nuvens) na qual as máquinas virtuais serão instanciadas. Ele também configura o ambiente e possibilita a implementação de aplicações HPC. O componente **Submissão de *Jobs*** é responsável pelo acondicionamento de aplicações (e.g. programas MPI ou qualquer aplicação que utilize um serviço HPC na nuvem) e inicializa a execução da aplicação a ser executada no *cluster* configurado previamente na nuvem. O componente **Preempção** executa políticas de escalonamento responsáveis pelas trocas de processos HPC executando na plataforma. Caso a plataforma não atenda às

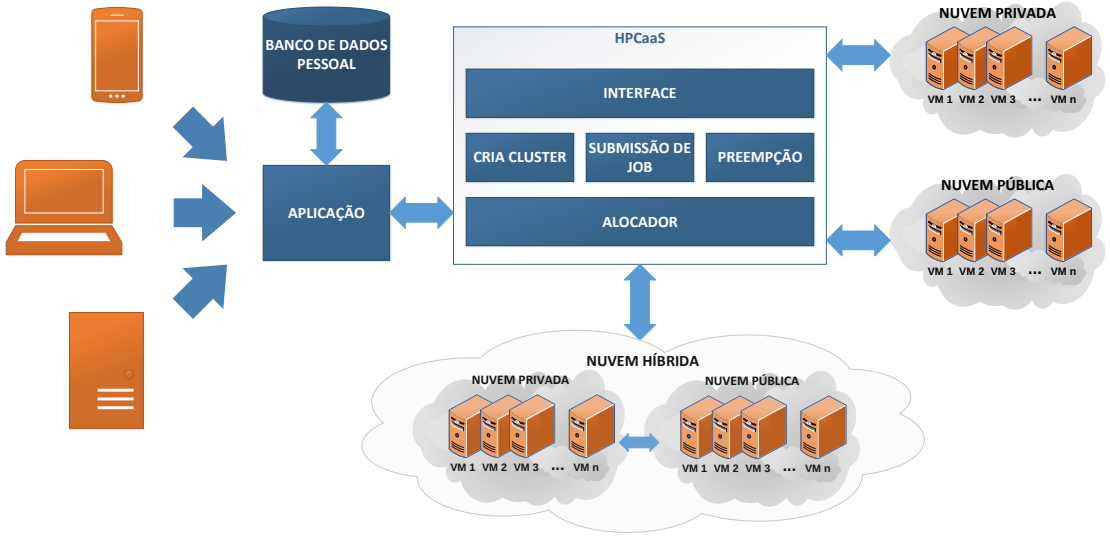


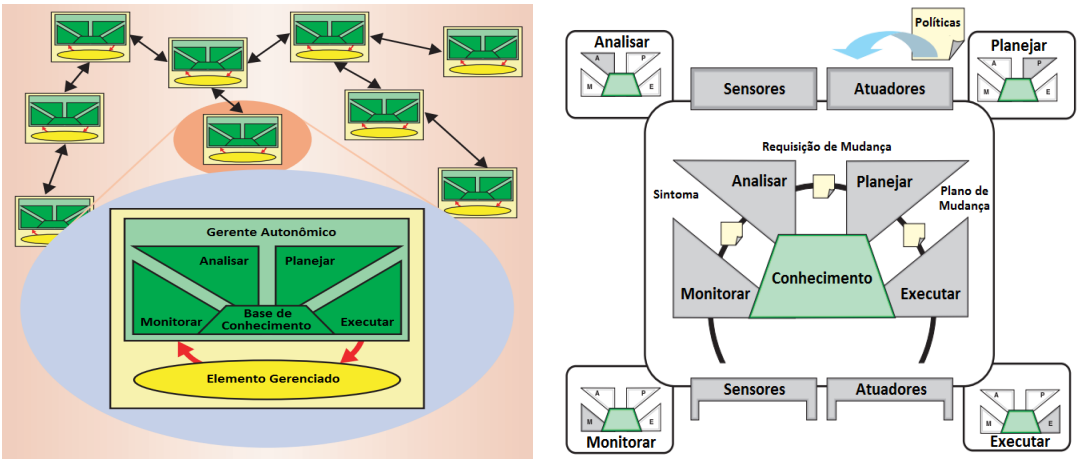
Figura 1. Arquitetura genérica para HPC em nuvem [Paillard et al. 2015]

demandas dos vários *jobs*, o componente de preempção habilita o acesso compartilhado dos recursos de maneira justa. O componente **Alocador** é responsável pela comunicação da plataforma com e entre as nuvens. Também é responsável pelas credenciais, criação e utilização dos usuários, criando máquinas virtuais e implantando os componentes e serviços na nuvem, além da alocação propriamente dita. O **Banco de Dados Pessoal** é utilizado para armazenar dados das aplicações dos usuários e configurações para posterior ajuste, e dados históricos de utilização do ambiente e da API. Desse modo, também é possível monitorar a utilização dos recursos, auditar e fazer recomendações baseadas no uso. Por fim, a **Aplicação** pode ser desenvolvida em qualquer plataforma (e.g. *mobile*, *desktop* ou *web*). A única restrição é que a aplicação utilize o componente HPCaaS. Por meio desse componente é possível a construção de aplicações para o usuário (SaaS) para o acesso a aplicações HPC em nuvens computacionais.

3. Elementos do Auto Gerenciamento

A essência da Computação Autonômica é o auto gerenciamento, cuja intenção é livrar os administradores de sistemas de detalhes da operação e manutenção, e prover aos usuários recursos (*hardware* ou *software*) com desempenho adequado [Kephart and Chess 2003]. Os elementos do auto gerenciamento podem ser utilizados para promover auto configuração, auto reparação, auto otimização e auto proteção em sistemas computacionais, conforme as definições de [Kephart and Chess 2003].

Sistemas autonômicos são grupos interativos de elementos autonômicos (sistemas individuais constituintes que contém recursos e distribuem serviços) [Kephart and Chess 2003]. Elementos autonômicos gerenciam seu comportamento interno e relacionamentos com outros elementos autonômicos conforme políticas estabelecidas. O gerenciamento do próprio sistema resulta em diversas interações através de elementos autonômicos do auto gerenciamento dos elementos autonômicos individuais. A Figura 2 (a) exibe um elemento autonômico tipicamente composto por um ou mais elementos gerenciados acoplados a um único gerenciador autonômico, que os controla.



(a) (b)

Figura 2. (a) Estrutura de um elemento autonômico [Kephart and Chess 2003] e (b) detalhes funcionais de um gerente autonômico [IBM 2005]

Cada elemento autonômico é responsável pelo gerenciamento de seus estados internos, comportamento e interações com o ambiente, consistindo na troca de sinais e mensagens entre elementos e o mundo externo. O comportamento de um elemento interno e seus relacionamentos é orientado pelos objetivos de seus projetistas, por outros elementos que detém autoridade sobre ele ou contratos com elementos pares de consentimento explícito. O elemento pode requerer assistência de outros elementos para executar metas.

Sensores provêm mecanismos para coletar dados sobre o estado e transições de um elemento, consistindo um conjunto de propriedades que expõem informação sobre o estado atual de um recurso gerenciado, e um conjunto de eventos de gerenciamento (não solicitados, mensagens ou notificações) que ocorrem quando o recurso gerenciado sofre alterações de estado que mereça relato [IBM 2005]. Atuadores são mecanismos que modificam o estado do elemento conforme decisões tomadas pelo sistema autônomo, consistindo uma coleção de operações de escrita que permitem alterar o estado do elemento gerenciado, e uma coleção de operações implementadas pelos gerentes autonômicos que permite aos recursos gerenciados realizar requisições ao seu gerente [IBM 2005].

Um elemento ou recurso gerenciado é um componente do sistema controlado, podendo ser um recurso único ou uma coleção de recursos, controlado por sensores e atuadores [IBM 2004]. O gerente autonômico é um componente que implementa um *loop* de controle, e realiza ações de: gerenciar a coleta (filtragem e relato de dados coletados pelos sensores); analisar e aprender sobre o elemento gerenciado; e acumular conhecimento e predizer ações futuras [IBM 2004]. A Figura 2 (b) exibe uma arquitetura semelhante à proposta por [Kephart and Chess 2003], porém com mais detalhes e inclusão dos sensores e atuadores. Em essência, as duas arquiteturas são idênticas, pois possuem as mesmas funcionalidades e objetivos, além das funções de monitorar, analisar, planejar e executar. As ações de um ciclo de vida de um gerente autonômico são: auto configuração, auto reparação, auto otimização e auto proteção [Kephart and Chess 2003].

4. Propostas para a Expansão

Baseado na aplicação dos quatro elementos do auto gerenciamento sobre a arquitetura para HPC proposta, destacamos um conjunto de expansões para os componentes descritos. De maneira geral, aplicando recursos para monitorar, analisar, planejar e executar, apoiado por sensores e atuadores, possibilitaria ainda mais a integração entre o ambiente particular do usuário, o ambiente da plataforma de HPC e as demais nuvens computacionais utilizadas para recursos. A utilização do componente **Banco de Dados Pessoal** teria um papel maior como base de conhecimento. Entretanto, para a arquitetura, este componente atualmente está disposto em um ambiente diferente da plataforma. O ideal é que a plataforma possuísse seu próprio banco de dados, e assim os dados seriam globais, podendo ser utilizados para diversas operações de caráter proativo, e potencial predição.

4.1. Auto Configuração

Em um ambiente de nuvem computacional é comum a utilização de diferentes tipos de recursos (e.g. CPU, memória e disco), providos por diferentes tipos de equipamentos e máquinas virtuais. Sendo assim, a alocação dos recursos conforme a demanda é uma atividade crucial, pois caso ela não seja adequada, pode implicar em desperdício de recursos e consumo energético ineficiente. Com a auto configuração, o ambiente pode ajustar a quantidade de recursos necessária para suprir a demanda, que pode ser dinâmica, demorada e diversificada. Além disso, a elasticidade é uma capacidade da nuvem altamente relacionada com a auto configuração, pois ela ajusta os recursos conforme a necessidade, evitando ociosidade e desperdício. Para a arquitetura proposta, o componente **Cria *Cluster*** pode se beneficiar de recursos autonômicos para a composição de *clusters* mais eficientes do ponto de vista da utilização de recursos, aproveitando melhor os recursos nas nuvens disponíveis para a plataforma.

4.2. Auto Reparação

Muitas operações, apoiadas pelo monitoramento e pelo mecanismos de *loops* de controle, podem ser claramente aplicadas em nuvens computacionais, principalmente relacionadas à disponibilidade de serviços e de máquinas virtuais em diferentes nuvens. O monitoramento, que pode apoiar eventos reativos ou proativos, pode constantemente verificar a disponibilidade dos serviços de computação, providos pelos fornecedores, assim como a disponibilidade das nuvens onde o *cluster* instancia as máquinas virtuais. Nesse contexto, novas máquinas podem ser incluídas, balanceadores de carga podem ser implementados para prover a disponibilidade dos serviços e recursos, e o monitoramento, ou algum mecanismo de verificação pode reintegrar os novos componentes ao ambiente da plataforma, ou reintegrar os componentes recuperados comprometidos em caso de reparação.

4.3. Auto Otimização

Uma vez que recursos estão sendo utilizados de diferentes fontes, e muitas vezes possuem um custo financeiro associado, faz sentido sua melhor distribuição/alocação nas diferentes infraestruturas. Além disso, a seleção dos recursos (e.g. CPU, memória e disco) também pode ser melhor executada, trabalhando inclusive o conceito de consolidação de recursos, promovendo economia de energia e menos emissão de CO_2. O monitoramento da utilização dos recursos auxilia na otimização desses recursos. Outra operação da auto otimização é na criação dos *clusters*, promovida pelo componente **Cria *Cluster***, onde se

define o melhor arranjo para a construção da base para o processamento das operações sobre as nuvens computacionais disponíveis. O componente **Submissão de *Jobs*** também pode ser aprimorado por recursos de otimização. Por fim, a melhor definição da estrutura, apoiada por um monitoramento contínuo das diversas nuvens utilizadas pela plataforma, pode sugerir novos arranjos, novas conexões e implicar em serviços mais estáveis e consolidados, sem a necessidade de intervenção humana, melhorando o desempenho do ambiente.

4.4. Auto Proteção

Ataque a nuvens computacionais ocorrem como a qualquer ambiente computacional, sendo importante sua preservação. Como as nuvens muitas vezes são públicas ou híbridas, seu acesso, tanto a seus serviços quanto a sua infraestrutura, é público. Mesmo com as devidas credenciais, as nuvens sofrem com diversos tipos de ataques. Um monitoramento nesse contexto procuraria antecipar potenciais problemas de disponibilidade na infraestrutura, assim como de desempenho, detectando intrusos e relatando tentativas de acessos indevidos e reais. O acesso à plataforma de HPC também deve ser verificado constantemente, pois ataques internos também podem ocorrer. O componente **Alocador** é responsável pela comunicação da plataforma com e entre as nuvens, e no contexto da auto proteção ele seria um forte candidato e se beneficiar de capacidades de proteção.

5. Conclusão

A arquitetura proposta visa a disponibilização de recursos de HPC, de forma mais transparente ao usuário, tanto para desenvolvedores de aplicações quanto para serviços, utilizando nuvens computacionais. A ideia é a utilização de recursos de HPC de uma maneira mais fácil, mais barata e mais disponível. A utilização de conceitos de Computação Autonômica podem aprimorar a plataforma em diversos aspectos, tais como: desempenho, segurança, monitoramento, diminuição da intervenção humana e melhor utilização dos recursos. Esse trabalho é motivacional, portanto todas as ideias descritas para o auto gerenciamento são potenciais trabalhos futuros. Sendo assim, acreditamos que a aplicação de conceitos de Computação Autonômica sobre a plataforma HPCaaS só viria a beneficiar a disponibilização e qualidade dos serviços para seus usuários.

Referências

IBM (2004). A pratical guide to the ibm autonomic computing toolkit. Technical report, IBM.

IBM (2005). An architectural blueprint for autonomic computing. Technical report, IBM.

Kephart, J. O. and Chess, D. M. (2003). The vision of autonomic computing. *Computer*, 36(1):41–50.

Paillard, G. A. L., Coutinho, E. F., de Lima, E. T., and Moreira, L. O. (2015). An architecture proposal for high performance computing in cloud computing environments. In *4th International Workshop on ADVANCEs in ICT INfrastructures and Services (ADVANCE 2015)*, Recife.

Rodrigo Álvarez, G. P., Östberg, P.-O., Elmroth, E., and Ramakrishnan, L. (2015). A2l2: An application aware flexible hpc scheduling model for low-latency allocation. In *Proceedings of the 8th International Workshop on Virtualization Technologies in Distributed Computing*, VTDC '15, pages 11–19, New York, NY, USA. ACM.