**Disciplina**: Sistemas Distribuídos. **Professor**: Alcides Calsavara

**Alunas:** Ana Carolina Moises de Souza / Nabila Coutinho

**Resumo do Artigo:** A systematic literature review on energy efficiency in cloud software architectures

Giuseppe Procacciantia, Patricia Lagoa, Stefano Bevinic

VU University Amsterdam, De Boelelaan 1081a, Amsterdam, The Netherlands

Politecnico di Torino, Corso Duca degli Abruzzi, 24, Torino, Italy

IBM Rome Software Lab, Via Sciangai 53, Roma, Italy

Accepted 16 November 2014

Sustainable Computing: Informatics and Systems

**Problema**: Falta de evidencia cientifica quanto a eficiência energética de arquitetura de software baseado em *Cloud*.  
**Objetivo**: Investigar diversos artigos relacionados à eficiência energética de software.  
Introdução: A demanda por energia elétrica utilizado por ICT irá crescer de 60 TWh/y em 2005 para 250 TWh/y em 2017. Por tanto é necessário repensar as tecnologias tornando as mais eficiente energeticamente e sustentáveis.  
Um dos princípios de *Cloud Computing* é a redução de hardware físico, uma vez que as máquinas são virtuais, sendo assim *Cloud* está intrinsicamente ligado a economia de energia.  
Em termos de prestação de serviços, a redução de custos de energia é visto como um objetivo (SLO) para níveis de serviços de contratuais (SLA). Um exemplo que autor cita é "A conta de energia elétrica do cliente deve ser reduzida 20% em um ano", isso pode ser uma vantagem para quem fornece o serviço de cloud e para quem recebe o serviço.  
No entanto, o problema está na oferta desse serviço, pois a arquitetura de software para cloud é muito complexa e não é conhecido o impacto dessa arquitetura no uso eficiente de energia. Existe vários software que medem o consumo de energia, no entanto este avança no intuito de responder se é possível quantificar os efeitos do consumo de energia utilizando certa abordagem de arquitetura de software e quais arquiteturas aumentariam a eficiência energética de cloud.  
**Método de Pesquisa Revisão Sistemática de Literatura:** A técnica de revisão sistemática de literatura permite a formulação de padrão de busca de artigos em base cientificais, exclusão e inclusão dos resultados selecionados, analise dos resultados selecionados onde pode ser empregado uma técnica para leitura desses artigos e rastreabilidade (organização das referências). Neste estudo foi utilizada as seguintes palavras chaves para busca: *“software architecture” AND cloud AND service AND “(energy OR power) efficiency” AND (SLA OR SLO OR “service level“)* na base de dados Google Scholar no período entre os anos 2000 a 2013. A técnica de *Coding* para análise qualitativa dos dados foi utilizada para descobrir padrões e tendências. Os códigos foram estruturados em *Strategy* (quais soluções de software fornece eficiência energética), *Technique* (qual é técnica utilizada na estratégia) e *Component* (qual é a arquitetura conceitual utilizada quando a técnica é aplicada). Por fim foram encontrados inicialmente, 149 estudos com base nas palavras chave. Após a revisão do título, abstract e leitura completa do texto, seguindo as técnicas de revisão sistemática citadas, 26 artigos foram selecionados. Desse estudos a maior parte foi produzida em 2011, quando houve muitos estudos sobre cloud, ou seja, são estudos recentes. Os tipos estudos foram classificados por publicação: 11 journals, 12 conferências e 3 teses de doutorado. Também foi considerado os estudos quanto a sua validação: 14 apresentou validação acadêmica, 2 validação na indústria e 10 não validaram a solução proposta. Por tanto, o autor informa que não existe uma maturidade sobre o tópico de arquitetura de software e eficiência energética em cloud considerando a evidencia na indústria.   
Eficiência energética em arquitetura de software: Os resultados alcançados com a revisão sistemática são explicados de acordo com o que foi identificado na aplicação da técnica *Coding*, isto é *Strategies*, *Techniques* e *Components*. Sobre as Estratégias (*Strategies)* o autor identificou *Energy Monitoring* que diz respeito a softwares que monitoram o consumo de energia. *Self-Adaptation*, que é relacionado a adaptação do comportamento do software com o objetivo de aumentar a eficiência energética. *Cloud Federation* que abrange a possibilidade de alugar ou negociar serviços de cloud baseado em na especificação de consumo de energia. Baseado nestas estratégias foi identificado nos estudos que *Self-Adaptation* é o mais adotado e *Energy Monitoring* é na maioria dos casos, adotado em combinação com Self-Adaptation. No entanto, *Cloud Federation* não é tão citado nos estudo. De acordo com o autor isto ocorre por que ambientes *multi-cloud* não é muito comum provavelmente por causa de padronização e segurança. Referente as Técnicas, as de maiores destaque foram *Consolidation* e *Workload Scheduling*, relacionados à Estratégia *Self-Adptation*, isso se deve ao fator dessas técnicas já serem utilizadas em cloud. Foram descobertas algumas dependências entre as técnicas, por exemplo algoritmos de escalonamento e maquinas virtuais são relacionadas ao componente de monitoramento de energia. Outra dependência de técnicas é entre *Service Adaptation* e *Energy Broker*, por meio de informações sobre energia existe a troca de serviços de *cloud*, isto está ligado ao componente *Cloud Federation*. Sobre Componentes (*Components)*, é importante lembrar que o relacionamento é muitos para muitos, isto quer dizer que as técnicas usam um número de componentes e cada componente podem ser usados pode mais de uma técnica. Neste código foi encontrado mais frequentemente o componente *VM Allocator* e *Workload Scheduler*, dentro da estratégia *Self-Adaptation*, porém o que chamou atenção foi as ocorrências sobre *SLA Violation Checker*, pois teve sua validação na indústria, o que contribui para a resposta da pergunta de pesquisa, no sentido de sugerir que a eficiência energética e qualidade de software é importante para arquitetura de software. O autor ainda menciona que existe a dois tipos de *SLA Violation*: um interno - que está ligado ao componente *Self-Adaptation*, isto é, monitoração interna do servidor; e externo, onde considera o SLA negociado de serviços entre provedores e consumidores.  
Stakeholder overview: Outra intenção desta pesquisa era identificar os envolvidos afetados ou interessados numa solução arquitetural de software para eficiência energética. Foram identificados quatro principais *stakeholders*: *End-User* (6 estudos), *Service Provider* (10 estudos), *System Architect* (13 estudos) e *Infrastructure Manager* (12 estudos). Como esperado *Infrastructure Manager* e *System Architects* são quem mais participa nas soluções apresentadas, no entanto os *End-users* são quem menos participa do processo, isso implica que os usuários finais desconhecem os benefícios das solução que promovem eficiência energética.  
Threats to validity: O autor deixa claro que sua pesquisa não está imune de ameaças de validade e a divide em dois tipos interna e externa. A respeito da ameaça interna, ele considera a base de dados Google Scholar como uma ameaça, no entanto justifica a escolha baseado em argumentos de dois pesquisadores no campo de SLR onde os dois falam que o Google Scholar é base apropriada e que tem melhorado muito sua estrutura nos últimos anos. Além dessa ameaça interna, o processo de seleção dos estudos foi feito por um pesquisador o que introduz um viés subjetivo a pesquisa. Para mitigar este risco foi introduzido o critério de inclusão e exclusão de estudos. Dentro deste processo de seleção, os estudos foram selecionados verificando o título, absatract e o texto completo. Sendo assim, a seleção desses estudos tendem a ter mais erros como falso positivo do que falso negativo (quando a exclusão de estudos relevantes). De acordo com o autor as ameaças externas pode ser encontradas na fase de data analises, pois foi adotada a técnica de Coding, onde a análise é qualitativa o que pode ter viés subjetivos, para mitigar esse risco dois pesquisadores analisaram e discutiram os resultados. Outra ameaça externa é sobre a quantidade de soluções não validadas nos estudos encontrados. Embora a validação dessas soluções estejam fora do escopo da revisão sistemática, o autor acha importante considerar a validação desses estudos como ameaça externa. Para identificar cada solução sem validação foi feita uma tabela no anexo que informa qual estudo tem a validação e qual não tem.  
**Conclusão:** Este artigo contribuiu para identificar aspectos de arquitetura de software em ambientes cloud que estão relacionados com eficiência energética. Foram encontrados 26 artigos que cobriam estes aspectos e a partir deles foram divididos em três códigos: *Strategies*, *Techniques* e *Components*. As principais estratégias identificadas foram: *Energy Monitoring*, *Self-Adaptation* e *Cloud Federation*. No entanto, *Self-Adaptation* é a mais adotada para atingir eficiência energética. Esses códigos auxiliam arquitetos de software a descrever e analisar solução de software que sejam energeticamente eficientes. Além disso, o estudo identifica que usuário finais não tem conhecimento de software que são eficientes energicamente. Isto poderia contribuir para o meio ambiente e benefícios econômicos, por tanto mais pesquisas são necessárias para identificar o que e como comunicar sobre softwares energicamente eficientes ao usuário. Ao final, como trabalhos futuros o autor pretende verificar software reusáveis quais as táticas, design e padrões para modelar software energeticamente eficientes. Além disso, é necessário planejar um estudo de caso no mundo real de como arquiteturas de software se relacionam com eficiência energética.