UNIVERSIDADE FEDERAL DO OESTE DO PARÁ
IEG - INSTITUTO DE ENGENHARIA E GEOCIÊNCIAS
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

VITOR TORRES EMERIQUE

MONITORAMENTO DE RECURSOS COMPUTACIONAIS EM *CLUSTER* DE ALTO DESEMPENHO: IMPLEMENTAÇÃO E AVALIAÇÃO COM FOCO NA SUSTENTABILIDADE

Santarém - Pará
2024

VITOR TORRES EMERIQUE

MONITORAMENTO DE RECURSOS COMPUTACIONAIS EM *CLUSTER* DE ALTO DESEMPENHO: IMPLEMENTAÇÃO E AVALIAÇÃO COM FOCO NA SUSTENTABILIDADE

Trabalho de Conclusão de Curso apresentado como requisito para obtenção do grau de Bacharel em Ciência da Computação, pela Universidade Federal do Oeste do Pará, no Instituto de Engenharia e Geociências.
Orientador(a): Prof°.Dr. Marcelino da Silva Silva

Santarém - Pará
2024

FORMULÁRIO DE AVALIAÇÃO DE TCC

Identificação:

Título do Trabalho:	Monitoramento de Recursos Computacionais em *Cluster* de alto desempenho: Implementação e Avaliação com Foco na Sustentabilidade.
Aluno (a): Vitor Torres Emerique	
Orientador (a): Marcelino Silva da Silva	

Avaliação:

Examinador (a) 1: Marcelino Silva da Silva	**Nota:** 9,8
Assinatura:	

Examinador (a) 2: Fábio Manoel França Lobato	**Nota:** 9,7
Assinatura:	

Examinador (a) 3: Josecley Fialho Goes	**Nota:** 10,0
Assinatura:	

Parecer:

Após apresentação e arguição da banca, tendo o discente respondido todas as questões de forma satisfatória, a banca deliberou pela aprovação.

Resumo da Avaliação:

- [x] Aceitação incondicional
- [ ] Aceitação condicionada a modificações (especificar no verso)
- [ ] Recusado

Nota Final: 9,8

Santarém-PA, 25 de outubro de 2024.

Presidente da Banca Examinadora

Dados Internacionais de Catalogação-na-Publicação (CIP)
Sistema Integrado de Bibliotecas – SIBI/UFOPA

E53m Emerique, Vitor Torres

Monitoramento de recursos computacionais em *Cluster* de alto desempenho: implementação e avaliação com foco na sustentabilidade. / Vitor Torres Emerique. - Santarém, 2024.

53 p. : il.

Inclui bibliografias.

Orientador: Marcelino Silva da Silva.

Trabalho de Conclusão de Curso (Graduação) – Universidade Federal do Oeste do Pará, Instituto de Engenharia e Geociências, Bacharelado em Ciência da Computação.

1. Computação de alto desempenho. 2. Sustentabilidade - Amazônia Legal. 3. Monitoramento OpenPBS. 4. Computação verde. I. Silva, Marcelino Silva da, *orient*. II. Título.

CDD: 23 ed. 005.4

Bibliotecária - Documentalista: Cátia Alvarez – CRB/2 843

Dedico aos meus pais e meu irmão, Izaías, Rosângela e Vinicius Emerique pelo constante apoio nessa jornada acadêmica para suportar limitações que acreditava carregar.

AGRADECIMENTOS

Agradeço primeiramente ao Deus altíssimo pela vida e saúde para avançar em meus estudos. Aos meus pais pelo forte apoio na caminhada acadêmica e familiares que se fizeram presentes durante esse processo, aos caros colegas do laboratório de computação aplicada que estavam no percurso nessa fase de minha vida.

"Porque a sabedoria é melhor do que as joias, e tudo o que se possa desejar não se compara com ela"

BÍBLIA, Provérbios 8:11

RESUMO

A implantação do sistema de monitoramento computacional em *cluster* de alto desempenho é uma ferramenta benéfica para gestão estratégica dos recursos computacionais, em vista que o monitoramento provê o entendimento do comportamento dos recursos, permitindo análises diversas. Nesse sentido, considerando que sistemas computacionais de alto desempenho são equipamentos escassos, com poucos centros disponíveis devido ao alto custo, manutenção, energia e refrigeração; a gestão estratégica para otimizar e aumentar o tempo de vida se fazem necessários. Além disto, observando o contexto da Amazônia Legal onde esse trabalho foi desenvolvido, no qual temas como sustentabilidade e redução da emissão de dióxido de carbono são cruciais, o monitoramento de sistemas computacionais se torna uma ferramenta de extrema importância para a adoção de estratégias que visam a redução da pegada de carbono na atmosfera. A literatura apresenta diversas ferramentas para monitoramento, porém, existe uma lacuna em relação a sistemas que utilizam o gerenciador de recursos OpenPBS em relação à coleta de métricas por meio do Prometheus, a qual foi necessário desenvolver um *exporter* personalizado em consonância com comandos de monitoramento do gerenciador de recursos OpenPBS apresentando gráficos em *dashboard* personalizável por meio da ferramenta Grafana de visualização de dados, os quais foram coletados 20 métricas diferentes relacionadas a disponibilidade dos nós, recursos consumidos pelos usuários e estado de *jobs*. Este trabalho visa contribuir para o desenvolvimento sustentável da Amazônia e implementar um sistema de monitoramento de recursos computacionais como ferramenta precursora para introdução da computação verde no *cluster* alinhado ao Plano de Desenvolvimento Institucional. Nessa percepção, foram identificados os dados de interesse provenientes do gerenciador de recursos OpenPBS, coletados, armazenados e disponibilizados para visualização em gráficos. Logo, este trabalho é relevante como um produto para administradores interessados em entender o comportamento do *cluster*.

Palavras-chave: Computação de alto desempenho; Sustentabilidade na Amazônia Legal; Monitoramento OpenPBS; Computação verde.

ABSTRACT

Implementing a high-performance cluster computing monitoring system is a beneficial tool for the strategic management of computing resources, given that monitoring provides an understanding of the behavior of resources, enabling various analyses. In this sense, considering that high-performance computing systems are scarce equipment, with few centers available due to the high cost, maintenance, energy and cooling; strategic management to improve and increase their lifespan is necessary. Furthermore, observing the context of the Legal Amazon where this work was developed, where issues such as sustainability and reduction of carbon dioxide emissions are crucial, the monitoring of computing systems becomes an extremely important tool for adopting strategies that aim to reduce the carbon footprint in the atmosphere. The literature presents several monitoring tools, however, there is a gap between systems that use OpenPBS resource management about the collection of metrics through Prometheus, which made it necessary to develop a customized exporter in line with OpenPBS resource manager monitoring commands, displaying customizable dashboard graphs through the Grafana data visualization tool, which found 20 different surveys related to node availability, resources consumed by users and employment status. This work aims to contribute to the sustainable development of Amazon and implement a computational resource monitoring system as a precursor tool for the introduction of green computing in the cluster aligned with the Institutional Development Plan. In this perception, the data of interest from the OpenPBS resource manager were identified, obtained, stored and made available for visualization in graphs. Therefore, this work is relevant as a product for specific administrators who want to understand the behavior of the cluster.

Keywords: High-performance computing; Sustainability in the Legal Amazon; OpenPBS monitoring, Green computing.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ALGORITMOS

LISTA DE ABREVIATURAS E SIGLAS

API REST	*Application Programming Interface Representation State Transfer*
CPU	*Central Process Unit*
DATA SOURCE	Origem de onde o Grafana obtém os dados que serão exibidos nos painéis e dashboards
DRAM	*Dynamic Random Access Memory*
DSR	*Design Science Research*
GB	*GigaByte*
GPU	*Graphical Process Unit*
HPC	*High Process Compute*
IP	*Internet Protocol*
JSON	*JavaScript Object Notation*
PDI	Plano de Desenvolvimento Institucional
PDTIC	Plano Diretor de Tecnologia da Informação e Comunicação
RAM	*Random Access Memory*
SBC	Sociedade Brasileira de Computação
SSCAD-WIC	Simpósio em Sistemas Computacionais de Alto Desempenho - *Workshop* de Iniciação Científica em Arquitetura de Computadores e Computação de Alto Desempenho
SSD	*Solid State Drive*
TB	*TeraByte*
TSDB	*Time Series Database*

SUMÁRIO

1 INTRODUÇÃO

Nesse capítulo são apresentadas os argumentos iniciais do presente trabalho, contextualizando os conceitos por trás da pesquisa, de modo a compreender seu valor. A frente são apresentados os objetivos que serão alcançados ao decorrer do estudo, os potenciais impactos identificados. Logo, será apresentado a organização do trabalho, expondo um visão global da estrutura do trabalho.

1.1 CONTEXTUALIZAÇÃO E PROPOSTA

Cluster High performance Computing (HPC) impacta as mais diversas áreas do conhecimento, sendo utilizada por cientistas e engenheiros em diferentes setores (ALONSO et al., 2017). Um *cluster* utiliza dois ou mais computadores chamados de nós, trabalhando com visão de realizar tarefas conjuntamente, é um sistema computacional de processamento paralelo e distribuído que reúne um conjunto de máquina conectadas entre si concentrando seus poderes computacionais (SILVA et al., 2016; BUYYA et al., 1999). São diversas as aplicações que utilizam a computação de alto desempenho (IGNÁCIO; DIAS, 2023), como o processamento massivo de dados ou aplicaçôes de modelagem climática e inteligência artificial. Os nós computacionais do *cluster* são gerenciados por um sistema de agendamento de *job* e gerenciamento das cargas de trabalho (HAGER; WELLEIN, 2010; ROBEY; ZAMORA, 2021). Dessa maneira, os nós trabalham em paralelo em prol dos *jobs*. O poder do *cluster* de alto desempenho não se relaciona pela Lei de Moore, que explica que a capacidade de poder de processamento dobra a cada dois anos, mas o poder computacional está fortemente ligada ao paralelismo (HAGER; WELLEIN, 2010).

A pegada de carbono dos equipamentos de *clusters* ocorrem desde a coleta da matéria-prima, construção do equipamento, transporte, ciclo de vida o qual os principais emissores na sua produção são os componentes de armazenamento de disco rígido e placas gráficas e *Dynamic Random Acess Memory* (DRAM); por fim, na reciclagem (LI et al., 2023). Dessa forma, o monitoramento de recursos na fase da vida útil possibilita gerar percepções norteadoras relevantes para os administradores desses sistemas de alto poder de processamento.

Sendo assim, sistemas de monitoramento de recursos computacionais são uma importante ferramenta para tomada de decisões estratégicas e para melhor entender o comportamento do ativo digital (KASHIN; VOEVODIN, 2023). Além disso, permitir a análise do uso de recursos possibilita entender o desempenho do sistema e identificar suas anomalias (SORKUNLU et al., 2017).

A propagação do uso das tecnologias da informação trouxe, em paralelo, o aumento da produção de lixo eletrônico. Por meio da popularização da tecnologia, produtos competitivos implicaram na redução do ciclo de vida dos produtos, que em poucos dias podem se torna ultrapassado. A forte adoção tecnológica, ocasionou o aumento do consume de energia, logo superando a aviação na pegada de carbono (MISHRA et al., 2014; NETO et al., 2024). Nesse sentido, com foco na sustentabilidade, a adoção da computação verde é uma estratégia sustentável que lida com o *hardware*, visando reduzir influências negativas no ambiente, proporcionando maior desempenho dos equipamentos de modo que reduza a pegada de carbono e aumente a vida útil do equipamento (ROSA, 2020).

Examinando que computadores de alto desempenho são recursos escassos, com pouco centros disponíveis, devido ao seu custo de aquisição, bem como alto custo de manutenção, energia e refrigeração; a gestão estratégica para otimizar o seu uso e aumentar o seu tempo de vida se fazem prementes (NETO et al., 2022). A adoção da computação verde é necessária, objetivando a utilização eficiente dos recursos computacionais, minimizando os impactos ambientais sem comprometer as necessidades tecnológicas (MOHAPATRA et al., 2019). A quantidade de gás carbônico emitido em uma rede elétrica varia conforme o horário do dia e a localização devido ao tipo de geração (RADOVANOVIĆ et al., 2022); a energia renovável fotovoltaica pode reduzir em 20 vezes a emissão na pegada de carbono (LI et al., 2023); sendo que as máquinas de alto desempenho são responsáveis pela emissão de mais de cem milhões de toneladas de dióxido de carbono por ano (COCAÑA-FERNÁNDEZ et al., 2019), considerando todo seu ciclo de vida, desde a produção do equipamento, utilização e descarte eletrônico segundo a ABNT NBR ISO 14040 (Associação Brasileira de Normas Técnicas, 2009). Nesse sentido, considerando que o *cluster*, objeto de trabalho do presente estudo, é localizado na Amazônia Legal, sendo necessárias medidas que tendem a reduzir transtorno ambiental da pegada de carbono deste ativo digital.

Frente ao dado contexto, este estudo planeja apresentar o processo de implementação do sistema de monitoramento de recursos com base nos dados fornecidos pelo OpenPBS[1] , monitorando métricas relacionadas a informações gerais dos recursos computacionais, disponibilidade dos nós, memória RAM e CPU apresentando uma visão geral, recursos por usuário e estados dos jobs submetidos. Desse modo, foi empregada a ferramenta Prometheus[2], um *kit* de ferramentas de alerta e monitoramento de sistemas e sua biblioteca disponibilizada oficialmente para algumas linguagens de programação.

[1]Agendador de *jobs* e gerenciador de carga de trabalho em ambiente de computação de alto desempenho, <https://www.openpbs.org/>;

[2]<https://prometheus.io/docs/introduction/overview/>;

A linguagem Go[1] foi utilizada para desenvolver um *exporter*[2] customizado para a coleta e concentração dos dados de séries temporais, permitindo a visualização e análise dos gráficos em um *dashboard* na plataforma Grafana[3], uma solução também *open-source*.

O trabalho visou cumprir com diretrizes institucionais de sustentabilidade, no Plano de Desenvolvimento Institucional da (Universidade Federal do Oeste do Pará, 2024) da Ufopa por meio do sistema de monitoramento de recursos implementado no *cluster* Tapajós, como objeto de estudo, possibilitando a gestão estratégica deste ativo, já influenciando no planejamento de obtenções como o sistema de refrigeração por precisão e na necessidade de se expandir o armazenamento considerando o uso atual do *cluster*, uma importante ferramenta para a tomada de decisão, ferramenta útil para auxiliar em novos planejamentos de Plano Diretor de Tecnologia da Informação e Comunicação (Universidade Federal do Oeste do Pará, 2019) da instituição em relação ao âmbito da computação de alto desempenho.

Os resultados obtidos são úteis para o gerenciamento do sistema computacional com o intuito de se estabelecer como uma ferramenta para tomadas de decisão de ações referentes ao *cluster* de alto desempenho e para entender o comportamento da utilização dos recursos, garantindo um sistema mais tolerante a falhas ao identificar antecipadamente o problema por meio do monitoramento.

1.2 OBJETIVOS

Diante do contexto apresentado, direcionam-se os seguintes objetivos:

1.2.1 Objetivo geral

O objetivo deste trabalho é implementar uma ferramenta de monitoramento de recursos computacionais no cluster de alto desempenho da Universidade Federal do Oeste do Pará, no *cluster* Tapajós.

1.2.2 Objetivo específico

À luz do objetivo geral, defini-se os objetivos específicos:

---

[1]Linguagem de Programação desenvolvida pela Google. <https://go.dev/>;

[2]Componente do Prometheus que realiza a coleta dos dados de um sistema específico. <https://prometheus.io/docs/operating/security/#exporters>;

[3]Ferramenta responsável pela visualização permitindo análises temporais. <https://grafana.com/oss/grafana/>.

- Identificar as métricas do sistema computacional de alto desempenho por meio do OpenPBS;
- Desenvolver uma solução eficaz para a coleta automatizada e precisa das métricas providas pelo OpenPBS;
- Criar dashboards dinâmicos para o monitoramento contínuo e a análise detalhada das métricas, visando aprimorar o desempenho e prevenir falhas.

1.3 POTENCIAIS IMPACTOS

Este trabalho tem como função implementar um sistema monitoramento de recursos computacionais no *cluster* como uma ferramenta percursora para a adoção da computação verde, focando na sustentabilidade e alinhando com o PDI da Ufopa em relação às diretrizes de sustentabilidade. Nesse sentido, o monitoramento dos recursos dará uma ampla visão sobre o comportamento do *cluster*, além de auxiliar na tomada de decisões perante a análise temporal comportamental dos recursos.

Para este trabalho espera-se que o sistema de monitoramento dos recursos seja uma ferramenta de alto valor para a tomada de decisões e relação a melhora do sistema, aquisição de hardwares, tomada de estratégias para o melhor desempenho do *cluster* adotando a computação verde como princípio fundamental, para assim contribuir para a redução da emissão de carbono na atmosfera, redução do consumo de energia com estratégia de otimização do sistema computacional e o aumento da vida útil do *hardware*.

Este trabalho contribui para preconização da ciência aberta disponibilizando em repositórios o *exporter* desenvolvido e a estrutura Docker para coletar as métricas oriundas do gerenciador de recursos computacionais de alto desempenho OpenPBS, utilizando o *client library* Prometheus para efetuar as coletas e a estrutura para realizar o armazenamento e visualização das métricas por meio de contêiner virtual.

Espera-se também que com a disponibilidade do repositório, estruturado com métricas escritas em inglês, possa ser alcançado ao nível mundial este *exporter*, considerando a importância do monitoramento em sistema de alto desempenho computacional.

1.4 ORGANIZAÇÃO DO TRABALHO

O restante do trabalho está organizado da seguinte forma:

No Capítulo 2, são apresentados trabalhos pertinentes que abordam sobre o processo de implementação de sistemas de monitoramento em computação de alto desempenho,

que adotam diferentes estratégias e ferramentas no processo de coleta das métricas, análise e visualização.

No Capítulo 3 é apresentado o método de pesquisa adotado incluindo as etapas de implementação do sistema. Também é dada uma visão geral do problema, a motivação para a implementação, os objetivos da solução em relação ao sistema, como se configura o *design* da aplicação, a implementação das ferramentas utilizadas, o processo de implantação partindo da identificação dos dados, coleta das métricas, armazenamento das métricas, exibição e análise em *dashboard*.

No Capítulo 4 é apresentado o produto final, expondo os gráficos presentes de informações referentes a características gerais do *cluster*, recursos consumidos, disponíveis e por usuário.

Por fim, no Capítulo 5, as considerações finais são apresentadas, concluindo a implantação do sistema e destacando as principais contribuições deste trabalho para a comunidade acadêmica. Além disso, são discutidos possíveis direcionamentos para pesquisas futuras, identificando lacunas e oportunidades para o aprimoramento do sistema.

2 TRABALHOS RELACIONADOS

Neste capítulo, serão apresentados trabalhos que exploram diversas ferramentas e abordagens para coleta e visualização de métricas do ambiente de alto desempenho. Apresentando o diferencial deste trabalho para os demais expostos.

Em BAUMANN et al. (2017) os autores desenvolveram e implementaram um sistema de monitoramento de temperatura de um *cluster* utilizando sensores de temperatura digitais conectados a um *Raspberry Pi*, buscando otimizar o desempenho das máquinas em relação ao calor propagado. Por outro lado, (CHI; ZHOU, 2019) utilizaram um enfoque fundamentado em *software*, coletando dados dos sensores das próprias máquinas computacionais, gerenciados pelo *kernel* do sistema operacional, utilizando uma abordagem arquitetural de cliente-servidor distribuído com objetivo de melhorar ferramentas existentes para garantir um monitoramento em tempo real via protocolos de rede, consumindo pouco recursos computacionais para realizar o monitoramento. Este trabalho salienta focar no gerenciador de recursos do sistema HPC por meio da coleta, armazenamento e visualização de métricas mediante softwares.

SAPUTRA et al. (2024) segue o monitoramento por meio de software, utilizando o sistema Prometheus, diante o sistema diretamente ligado ao *Kernel* do sistema operacional semelhantemente a (CHI; ZHOU, 2019). O sistema de monitoramento apresentado por (SAPUTRA et al., 2024) permite uma melhor administração dos dados ao poder ser ligado a diversos nós, sendo necessário a presença do *exporter* a cada máquinas. A exibição dos dados via *dashboard* pelo Grafana no sistema proposto por (SAPUTRA et al., 2024) também pode gerar alerta por meio do Telegram. O presente trabalho destaca-se por implantar um *exporter* customizado juntamente ao gerenciador de recursos OpenPBS na linguagem Go, utilizando a biblioteca oficial do Prometheus abordando apenas um *target exporter* para coleta de dados.

É possível perceber que existem diversos meios de realizar a coleta de métricas e visualizá-las. A abordagem feita pelo trabalho de (EITZINGER et al., 2019) objetivou-se na interface gráfica de monitoramento de cluster de pequeno e médio porte, se direcionando para a visualização de *jobs* específicos e usuários, motivado pela disponibilidade de apenas ferramentas genéricas de visualização e análise de métricas, voltando-se a utilizar uma API Rest para realizar as coletas de métricas, armazenando os dados no InfluxDB do tipo TSDB e visualizando os dados na interface web programada em PHP. Nesse sentido, este trabalho se difere por focar no ambiente sustentável por meio do monitoramento de recursos computacionais do *cluster*.

Equivalentemente, KUNZ (2022), implementou e desenvolveu um *exporter* personalizado para o *workload manager* e *job scheduling system* Slurm[1] motivado pela limitação do monitoramento do *cluster* visando implementar um sistema de monitoramento e estudar e analisar o comportamento de comportamentos anômalos do cluster. Nesse sentido, este trabalho diferencia por oferecer toda a estrutura de software nas dependências do sistema HPC, Prometheus, *exporter* e Grafana. (STANISIC; REUTER, 2020) anuncia um *design* e implantação de um sistema de monitoramento HPC com uma solução arquitetural apresentando os passos de coleta, análise e exposição dos dados, um interposto executa em todos os nós do *cluster*, concentrando dados de monitoramento do Slurm e outras informações de métricas, persistência de dados e a visualização ocorrem por meio da ferramenta Splunk[2]. Nesse viés, o presente trabalho se diferencia por monitorar os nós de processamento computacional de todos os estados de disponibilidade e os usuários que usam o sistema HPC.

É possível visualizar na Tabela 2.1 para melhor compreensão sobre o conjunto de ferramentas de coleta e visualização dos trabalhos relacionados.

Trabalho	**Ferramenta de Coleta**	**Ferramenta de Visualização**	**Gerenciador de recursos**
(BAUMANN et al., 2017)	Sensores Digitais e Raspberry Pi	Python Threading Library	-
(CHI; ZHOU, 2019)	pseudo-file system	Terminal	-
(EITZINGER et al., 2019)	API Rest + InfluxDB	Interface Web em PHP	Slurm
(STANISIC; REUTER, 2020)	Hpcmd	Splunk	Slurm
(KUNZ, 2022)	Prometheus	Grafana	Slurm
(SAPUTRA et al., 2024)	Prometheus	Grafana	-

Tabela 2.1 – Ferramentas de coleta e visualização utilizadas pelos trabalhos relacionados.

É importante destacar que, pela nossa melhor compreensão, não existe na literatura trabalhos que explorem o monitoramento de recursos computacionais utilizando Prometheus e o gerenciador de recursos OpenPBS como alvo para a coleta de métricas. Este trabalho se destaca dos correlatos por implementar o sistema monitoramento de recursos no *Cluster* localizado na Amazônia Legal na Universidade Federal do Oeste do Pará, seguindo os princípios da computação verde e alinhando-se com o PDI da instituição de ensino e por desenvolver um exporter personalizado Prometheus para o gerenciador de recursos OpenPBS. A proposta opera métricas oriundas do OpenPBS, desenvolvendo um *exporter* customizado Prometheus para coleta e armazenamento das métricas e o Grafana para visualização e análise em *dashboard*.

[1]Um sistema de gerenciamento de cluster e agendamento de tarefas de código aberto, tolerante a falhas e altamente escalável para clusters Linux grandes e pequenos

[2]Ferramenta que realiza a coleta dos dados, armazenamento, visualização e permite análise das métricas.

3 MATERIAIS E MÉTODOS

Com a pretensão de alcançar os objetivos relacionados ao trabalho, neste capítulo é apresentado a metodologia de pequisa adotada para o desenvolvimento deste trabalho. A princípio foi realizado um levantamento das principais ferramentas utilizadas no sistema de monitoramento implantado.

A metodologia de pesquisa selecionada para este trabalho foi o *Design Science Research* (DSR), as etapas apresentado por (PEFFERS et al., 2007; PEFFERS et al., 2020). A Abordagem do DSR se objetiva por progredir um artefato para solucionar um problema em um cenário específico e formar conhecimento técnico-científico (PIMENTEL et al., 2020). Esta metodologia foi escolhida por sua adequação, apresentando as etapas relacionadas ao objeto de estudo deste trabalho, o processo consiste em seis fases como apresentado na Figura 3.1, sendo essas: (I) Identificação do problema e motivação, (II) Objetivos para solução, (III) Design, (IV) Implementação, (V) Avaliação e (VI) Comunicação.

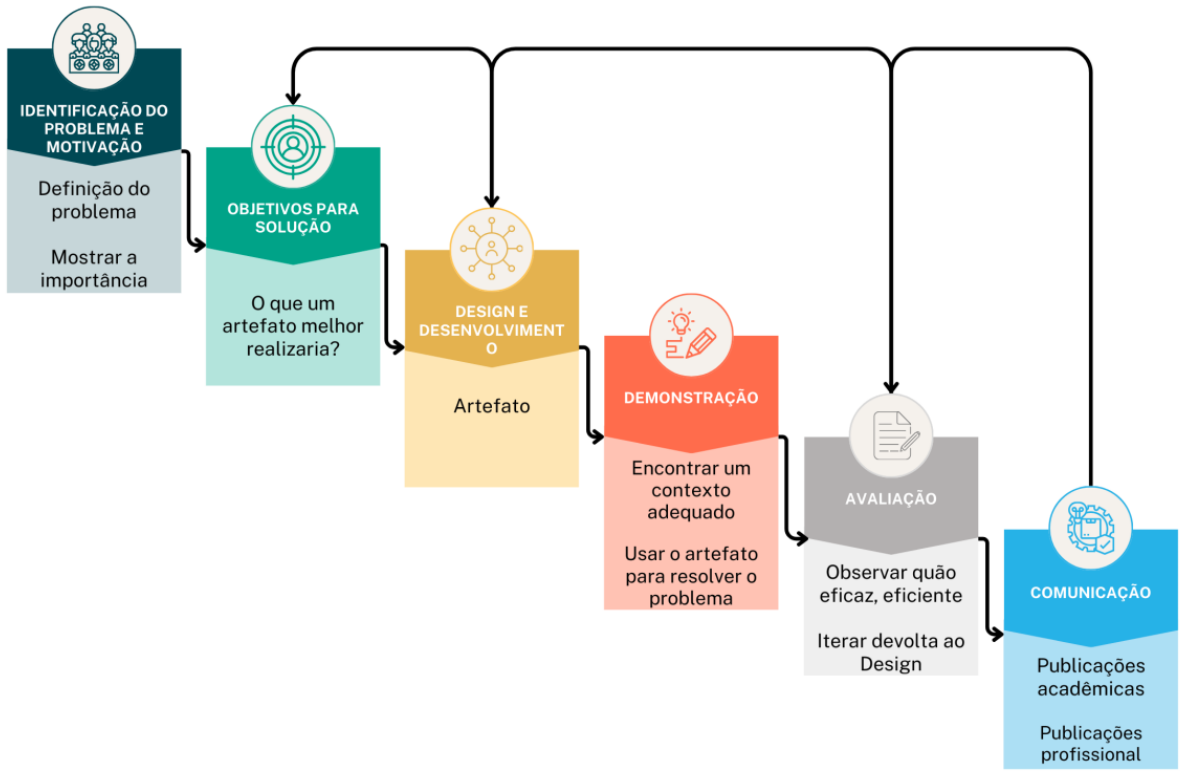


Figura 3.1 – Modelo do *Design Science Research*. Adaptado de (PEFFERS et al., 2020).

3.1 IDENTIFICAÇÃO DO PROBLEMA E MOTIVAÇÃO

Na fase (I) a primária para este trabalho foi realizado a identificação do problema e motivação para nortear as próximas etapas.

3.1.1 Identificação do Problema

Com base no contexto apresentado neste trabalho, foi identificado a ausência de monitoramento de computacional dinâmico, não permitindo a análise das métricas em relação ao consumo de recursos no *Cluster*, como objeto de estudo. Logo, o problema encontrado é apresentado pela falta de observabilidade computacional em relação aos recursos computacionais.

3.1.2 Motivação

Em relação à subseção 3.1.1 a motivação para solucionar esse problema se dá pela utilização dos recursos computacionais precisamente para reduzir as influências negativas no meio ambiente, levando em considerando que o monitoramento do *cluster* é estático e não permite a análise do comportamento através das métricas no *Cluster* Tapajós localizado no meio da Amazônia Legal. Adotar a computação verde por meio do monitoramento de recursos computacionais para auxiliar na tomada de decisão provendo *insighs* por meio da análise temporal dos recursos para minimizar os impactos ambientais sem afetar o poder computacional, aumentar à longevidade do equipamento computacional e reduz a emissão de CO_2 na atmosfera. Com isso, seguindo a metodologia, delineou-se como **Objetivo da Solução** na seção 3.2.

3.2 OBJETIVO DA SOLUÇÃO

O objetivo do trabalho foi apresentado no Capítulo 1 na Seção 1.1 no sexto parágrafo.

3.3 DESIGN E DESENVOLVIMENTO

Essa seção irá expor o design, como as ferramentas trabalham em conjunto para coletar, armazenar e visualizar as métricas e o desenvolvimento do *exporter*.

3.3.1 Design

O *Cluster* Tapajós é atualmente o maior Sistema de Computação de Alto Desempenho do Oeste do Pará, com capacidade de cerca de 100 TFLOPS, realizando aproximadamente 100 trilhões de operações por segundo. Sua infraestrutura conta com:

- 08 nós de cálculo com CPUs AMD 7532 (128 núcleos físicos), 256GB RAM e 2TB SSD;
- 01 nó de cálculo acelerado com GPU NVIDIA V100 (32GB) e FPGA Xilinx U200 (64GB), além de CPUs AMD 7532, 256GB RAM e 2TB SSD;

- 01 nó de administração com CPUs AMD 7532, 256GB RAM e 2TB SSD;
- 01 nó de armazenamento com 160TB brutos, CPU Intel 4214R, 192GB RAM e 160TB SSD;
- Switch de 100-Gigabit Ethernet QSFP28 para processamento de dados;
- Switch Gigabit Ethernet para gerenciamento.

Para os aspectos de software, o *cluster* Tapajós conta com o *workload manager and job scheduling* **OpenPBS** na versão 20.0.1, responsável por realizar o agendamento dos jobs e o gerenciamento das cargas de trabalho em ambiente de alto desempenho, tendo como principal característica ser *open-source*, baseado no núcleo do *PBS professional*, um software comercial.

Partindo para o **Design**, após o levantamento das ferramentas voltadas para a telemetria e uma estratégia de coleta das métricas por meio da utilização de apenas um *endpoint*[1], percebeu-se a necessidade da ferramenta Prometheus para coleta de métrica de séries temporais e o armazenamento. Posteriormente, para exibição de gráficos, selecionou-se a Grafana, uma ferramenta de visualização e análise desses dados. Dessa forma, foi necessário desenvolver um *exporter* personalizado para coletar as métricas do *cluster*. Um exemplo de *exporter*[2] e a estrutura do *Dashboard* se encontram disponíveis publicamente em repositório, visando atender aos princípios da ciência aberta conforme preconizado pela Sociedade Brasileira de Computação. Na Figura 3.2 é exposto o fluxo de execução em uma visão global do trabalho.

Conforme pode ser visto na Figura 3.2, o fluxo se inicia a partir dos comandos de monitoramento de recursos e nós do **OpenPBS**, coletando informações sobre o status dos *jobs* submetidos, informações sobre o status dos nós e utilização de recursos. Após o entendimento do ambiente computacional, a estratégia adotada se deu por utilizar apenas um *endpoint* do *exporter* Customizado , pois não haveria necessidade de configurar um *exporter* para cada nó. Assim, foi utilizado um dos nós computacionais para esta aplicação, mitigando o uso de mais recurso computacional, configurado no *Login node*. Os dados de séries temporais são concentrados no **Prometheus *Server*** que realiza a requisição das métricas e as armazena no banco de dados *Time Series Data Base*[3] (TSDB). Assim sendo possível visualizar os dados em dashboard personalizado utilizando o recurso de consulta do

[1]Uma fonte de métricas que pode ser extraída, geralmente correspondendo a um único processo. <https://prometheus.io/docs/introduction/glossary/#endpoint>;

[2]<https://github.com/vitoremerique/Openpbs_Exporter.git>;

[3]Fluxos de valores que possuem carimbos de data/hora que pertencem a mesma métrica.<https://prometheus.io/docs/introduction/glossary/#time-series>.

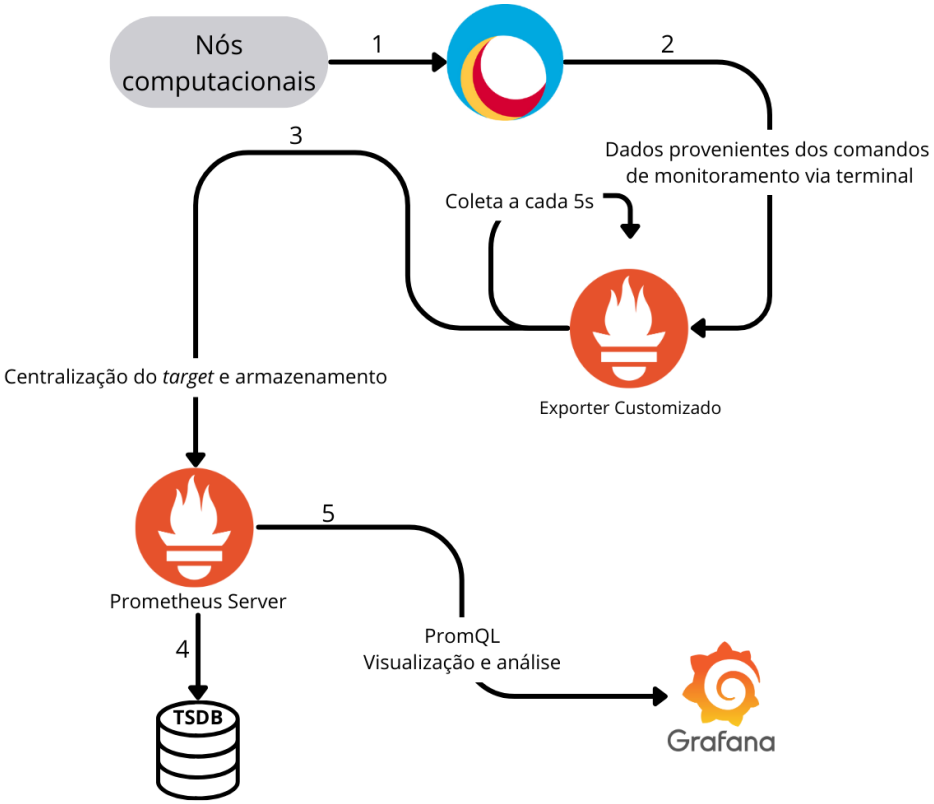


Figura 3.2 – Fluxo de execução geral do sistema.

Prometheus, o *PromQL*[1]. A Tabela 3.1 apresenta alguns exemplos de comandos utilizados do OpenPBS para desenvolver o *exporter*.

Tabela 3.1 – Comandos de monitoramento utilizados do OpenPBS.

Comando	Descrição
pbsnodes	Exibe informações detalhadas sobre todos os nós do cluster.
qstat	Mostra o status dos jobs na fila.

A Tabela 3.1 mostra comandos de monitoramento proveniente do OpenPBS. Por meio deles, foi possível alcançar os dados de saída alvo. Em seguida, após obter os dados via código terminal, utilizou-se a biblioteca do Prometheus para criar um servidor *exporter* obtendo as métricas pro meio dos *parsers* implementados. Os dados de saída foram convertidos em métricas do tipo *Gauge*[2]. Nesse sentido, foi possível disponibilizar para o Prometheus *server* o *endpoint* do *exporter* e requisitar e armazenatr as métricas.

Após o processo de coleta e armazenamento das métricas coletadas, utilizou-se o Prometheus *server* como um *data source* na plataforma Grafana, permitindo montar

[1]Uma linguagem de consulta prometheus, permite uma grande quantidade de operações como agregação, divisão, previsão e junção <https://prometheus.io/docs/prometheus/latest/migration/#promql>;

[2]Métrica que representa um valor numérico único que pode diminuir ou aumentar arbitrariamente. <https://prometheus.io/docs/tutorials/understanding_metric_types/#gauge>.

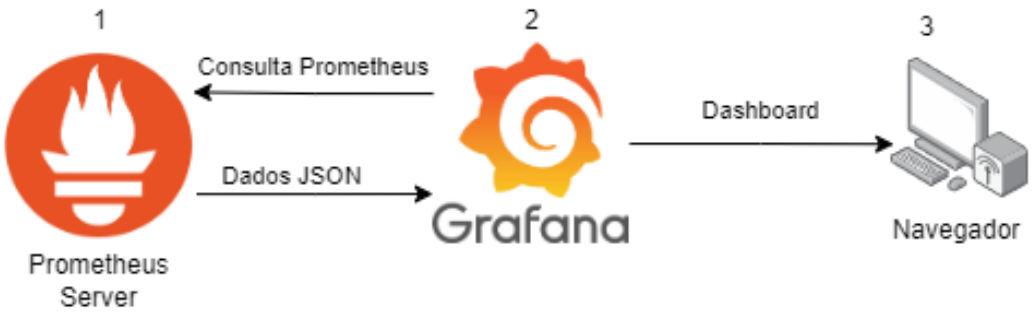


Figura 3.3 – Fluxo do Grafana com Prometheus.

gráficos específicos para cada métrica, além de possibilitar a correlação de métricas por meio de gráficos, visualizando e analisando o que ocorreu em um passado já coletado ou monitorando os recursos com um *delay* de 5 segundos a cada nova coleta. Isso foi possível devido o Prometheus possuir sua própria *query language* chamada *PromQL*, utilizada para fazer a chamada de requisição para visualização em tempo real com atualização periódica, na Figura 3.3 é possível ter uma visão geral.

3.3.2 Implementação

Na fase de **Implementação** serão expostos os recursos utilizados para coletar das métricas no exporter customizado desenvolvido conforme a sua presença na Figura 3.2.

Nesta fase foram identificados os dados que seriam coletados por meio dos comandos de monitoramento do OpenPBS; foi realizado o processamento desses dados para se tornarem métricas no *endpoint exporter*; o serviço do*endpoint* foi configurado no *Systemd* para garantir a disponibilidade do serviço; após isso, foi definida a integração do serviço de métricas ao serviço de visualização, priorizando inicialmente o armazenamento das métricas e a visualização em dashboard.

3.3.3 Identificação dos dados

Primeiramente foram identificadas e definidas as métricas a serem coletadas através dos comandos de monitoramento do *cluster* conforme a Tabela 3.1 de comandos provenientes do OpenPBS, por meios desta, foi possível realizar a coleta de 20 métricas diferentes relacionadas a uso de recursos e estado do *cluster*, bem como dos *jobs* do sistema. A partir da Tabela 3.2 é possível visualizar as métricas nomeadas para o *exporter*

Prometheus.

3.3.4 Processamento dos dados

Após a identificação das métricas conforme a Tabela 3.2 foi realizado no exporter a coleta das informações oriundas dos comandos de monitoramento por meio do openPBS, que ainda necessitavam de limpeza dos dados, o qual foi utilizado filtros de fluxo de dados, recursos este, padrão nas distribuições Linux que juntamente com os comandos da Tabela 3.1 realizaram as respectivas filtragens, posterior ao comando, foi realizado a extração dos dados alvo através do uso de *parsers* programado juntamento com o uso de Regex, uma sequência de expressões regulares que define um padrão de pesquisa, especificamente para cada caso.

Tabela 3.2 – Métricas coletadas.

Métrica	Descrição
openpbs_cpu_assigned_unit	Unidades de CPU em uso.
openpbs_cpu_available_unit	Unidades disponíveis de CPU.
openpbs_cpu_total	Total de unidades de CPU.
openpbs_job_count	Quantidade de *jobs*.
openpbs_job_exiting	Quantidade de *jobs* em estado de *exiting*.
openpbs_job_held	Quantidade de *jobs* em estado de *held*.
openpbs_job_queued	Quantidade de *jobs* em estado de *queued*.
openpbs_job_running	Quantidade de *jobs* em estado de *running*.
openpbs_memory_available_gb	Quantidade em Gigabyte de memória RAM disponível.
openpbs_memory_usage_gb	Quantidade em Gigabyte de memória RAM em uso.
openpbs_node_available	Quantidade de nós disponível.
openpbs_node_busy	Quantidade de nós em estado ocupado.
openpbs_node_count	Quantidade total de nós do *cluster*.
openpbs_node_down	Quantidade de nós no estado *down*.
openpbs_node_drained	Quantidade de nós no estado *drained*.
openpbs_node_offline	Quantidade de nós no estado *offline*.
openpbs_node_reserved	Quantidade de nós no estado *reserverd*.
openpbs_node_unknown	Quantidade de nós no estado desconhecido.
openpbs_user_memory_usage_unknown	Quantidade de memoria utilizada por usuário.
openpbs_user_cpu_usage	Quantidade de CPU utilizada por usuário.

O exporter implementado possui um algoritmo para melhor entendimento do

desenvolvimento, conforme é apresentado no Algoritmo 1 como ocorreu a fase de coleta dos dados.

Algoritmo 1: Coleta de métricas do OpenPBS com Prometheus

Inicialização:

Registrar variáveis Prometheus para métricas do cluster e dos usuários

Configurar servidor HTTP para expor as métricas na porta 8080

Execução contínua a cada 5 segundos:

while *Execução contínua* **do**

- Capturar saída de comandos do sistema (`qstat` e `pbsnodes`)
- Filtrar a saída com expressões regulares
- Extrair contagem de *jobs* (`qstat | wc -l`)
- Extrair estado dos nós (`pbsnodes -a`)
- Extrair estados dos *jobs* (`qstat -a`)
- Extrair uso de memória e CPU por usuário (`qstat -f`)
- Atualizar as variáveis Prometheus com os valores extraídos

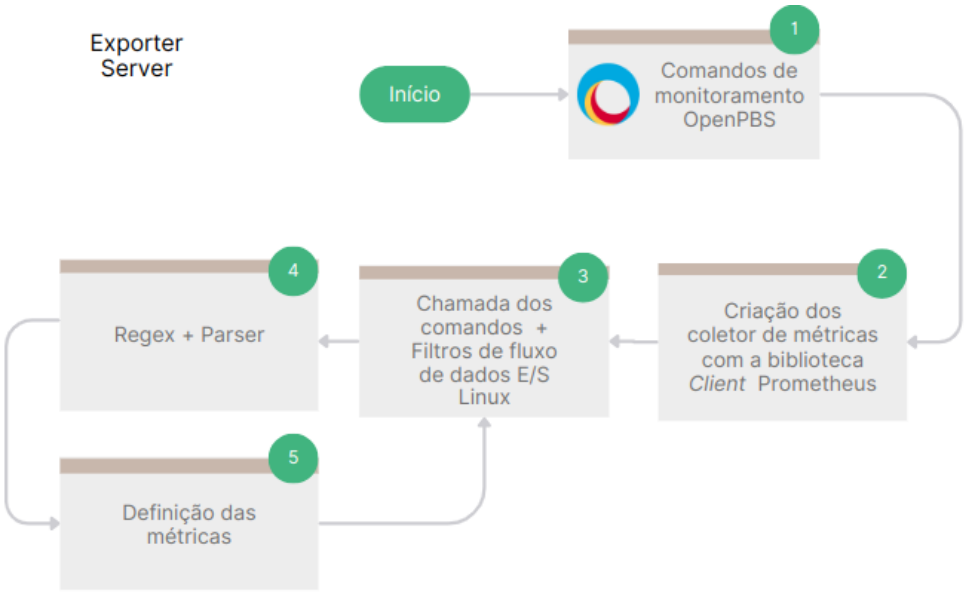
end

Serviço:

Servir métricas na rota `/metrics` via HTTP

Logo após o processo de filtragem, definições de pesquisa com regex e definições das métricas com a biblioteca *client* Prometheus com métricas do tipo Gauge, foi possível transformar simples informações de monitoramento em métricas acessíveis, esse fluxo pode ser observador por meio da Figura 3.4 esse fluxo se repete a cada 5 segundos.

Figura 3.4 – Fluxo interno do Exporter.



3.3.5 Gerenciamento do serviço exporter no cluster

Caso por ventura houver o desligamento da máquina que comportar o *exporter* é necessário ser configurado um serviço no sistema operacional para garantir o funcionamento do *exporter* após ligado novamente.

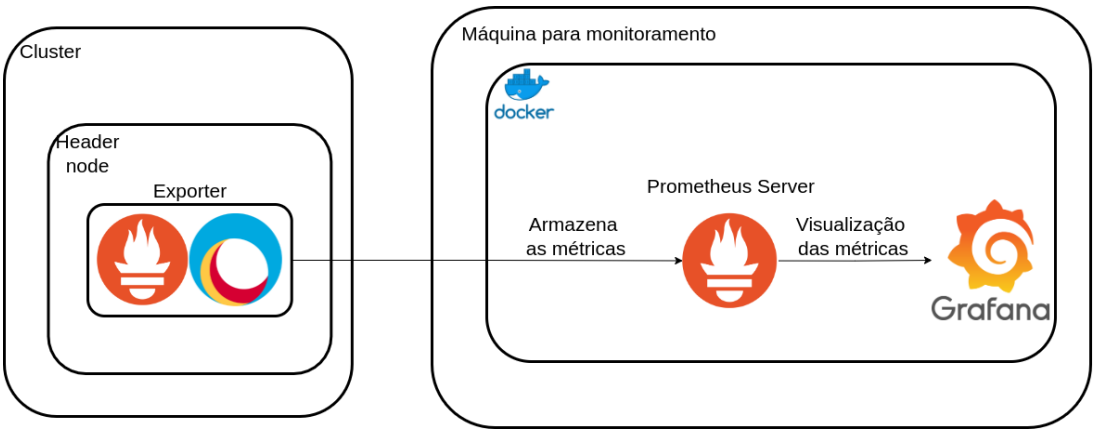
Nesse sentido, foi necessário configurar no *systemd* do sistema operacional, responsável por inicializar e gerenciar processos e serviços linux durante o boot da máquina, para garantir o funcionamento do exporter mesmo que a máquina seja reiniciada (CLINTON; NEGUS, 2020).

3.3.6 Integração de Métricas: Soluções para Armazenamento e Visualização de Dados

Nessa etapa da implementação foi realizada na máquina computacional dedicada ao monitoramento dos recursos, ou seja, essa configuração é organizada fora dos nós do cluster, porém é possível que seja configurada no recinto do *cluster*. Nesse sentido, a utilização da ferramenta Docker[1] por meio de contêineres virtuais facilitando as configurações iniciais. A configuração das ferramentas por meio conteinerização permite definir as configurações iniciais do Prometheus permitindo reproduzir em qualquer máquina que possua o Docker. A configuração de virtualização dessas ferramentas[2] está disponível publicamente em um repositório.

A Figura 3.5 visa oferecendo uma visão geral da conteinerização do armazenamento das métricas e da visualização dos conteúdos.

Figura 3.5 – Comunicação do exporter com a visualização das métricas.



[1] <https://www.docker.com/>

[2] <https://github.com/vitoremerique/Prometheus-Grafana-Docker.git>;

3.3.6.1 Armazenamento das métricas

Conforme pode ser visualizado na Figura 3.5 por meio do *exporter*, se comunica com o Prometheus *server* através do protocolo HTTP o qual são estabelecidos no arquivo prometheus.yml necessitando configurar os endereços *Internet Protocol* (IP) dos endpoints exporter, foi configurado o endereço local da rede da instituição e o externo para garantir o monitoramento de qualquer lugar.

Conferindo o Quadro 3.1 para mais detalhes é possível visualizar como foi configurado o *endpoint* do *exporter*, por motivos de segurança não foram expostos os respectivos endereços digitais.

Quadro 3.1 – Configuração do Prometheus Server.

Configuração prometheus.yml

```
- job_name: 'pbs-internal'
  static_configs:
    - targets: ['SEU-IP INTERNO']
- job_name: 'pbs-external'
  static_configs:
    - targets: ['SEU-IP EXTERNO']
```

3.3.6.2 Visualização dos dados

Consoante a Figura 3.5 após realizar a coleta e armazenamento das métricas, é necessário adicionar um *data source* Prometheus na plataforma Grafana conforme a Figura 3.6. É obrigatório endereçar o (*localhost*) com sua respectiva porta do Prometheus Server para possibilitar o acesso às métricas computacionais do *cluster*. Após preencher o enderenço, é necessário clicar no *Save & Test* para garantir que esteja correto.

Com o *data source* configurado, é importante criar um *dashboard* onde será possível configurar e correlacionar métricas através dos visualizadores. Foi disponibilizado um arquivo JSON nos repositórios para importar as estruturas dos visualizadores no Grafana. A Figura 3.7 expõem a área de criação do *dashboard* na plataforma Grafana. Pós-criação do dashboard, é possível realizar a configuração dos visualizadores, é possível por meio da linguagem PromQL receber dados e especificá-los mediante filtros, expressões e queries, permitindo correlacionar métricas.

Figura 3.6 – Configuração do *data source* no Grafana.

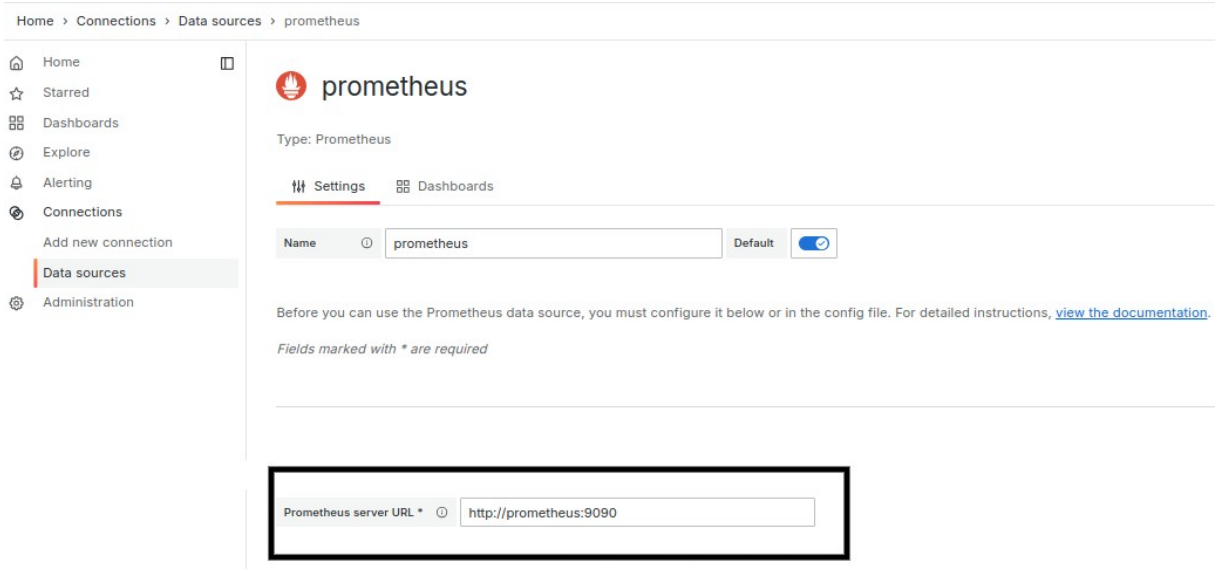
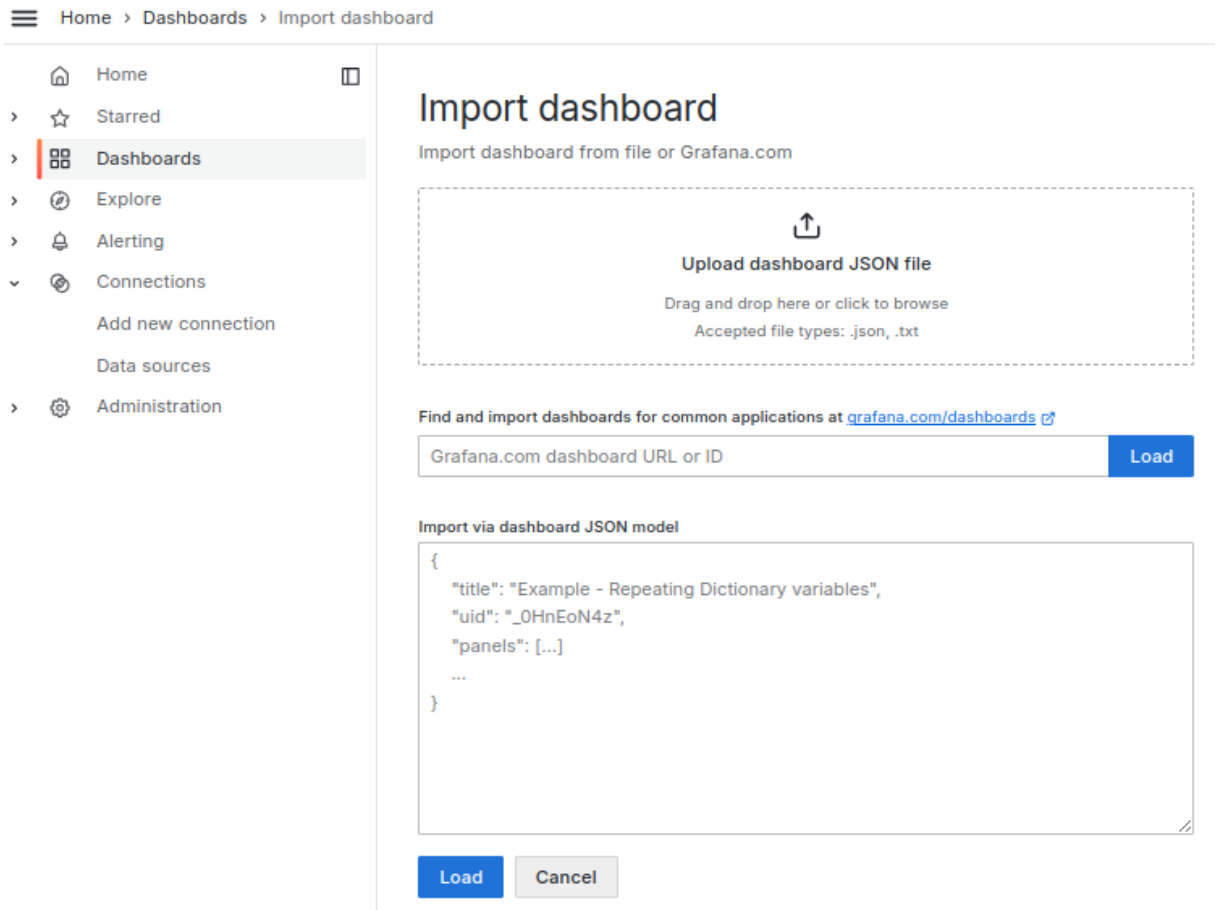


Figura 3.7 – Criação do grafana dashboard.



3.4 DEMONSTRAÇÃO

Na etapa **Demonstração**, foram realizados testes de inclusão de *jobs* na fila e analisada a confiabilidade dos dados no *dashboard*. Foi efetuada uma investigação utilizando

os comandos da Tabela 3.1 para garantir que os dados eram exibidos corretamente. Após a realização dos testes manuais, foi possível afirmar que o sistema implementado exibe nos gráficos as métricas corretamente.

Abaixo é possível visualizar na Figura 3.8 o *script* utilizado para realizar os testes de confiabilidade dos dados, comparando os comando de monitoramento do gerenciador de recursos com os dados exibidos no *dashboard*.

```
vitor.emerique.discente@tapajos:~> cat test.sh
#!/bin/bash
#PBS -N TestJob
#PBS -l nodes=2:ppn=1
#PBS -l walltime=00:10:00

echo "Starting job..."
echo "Hostname: $(hostname)"
echo "Job ID: $PBS_JOBID"
echo "Job Name: $PBS_JOBNAME"
echo "Allocated Nodes: $PBS_NUM_NODES"
echo "Processors Per Node: $PBS_NUM_PPN"
echo "Working Directory: $PBS_O_WORKDIR"

# Your actual job commands go here
# Example:
sleep 30
```

Figura 3.8 – Script para submissão de teste.

É possível visualizar o job submetido por nome “TestJob” na Figura 3.9 como exemplo, sendo utilizando o comando de monitoramento **qstat test.sh** para submeter um job na fila. O nome dos demais *jobs* juntamente com usuário foram omitidos por questões de privacidade.

```
vitor.emerique.discente@tapajos:~> qstat
Job id            Name             User              Time Use S Queue
----------------  ---------------- ----------------  -------- - -----
13608.tapajos                                                0 Q accel
14193.tapajos                                         02:38:32 R cpu
14207.tapajos     TestJob          vitor.emerique.*  00:00:00 R workq
```

Figura 3.9 – Verificação do Job.

Após a submissão do job “TestJob” foram comparados os dados visualizados pelo terminal com os dados das tabelas exibida no *dashboard*. Para melhor explicação, serão apresentados uso de memória RAM e CPU por usuário.

Conforme a Figura 3.10 é possível visualizar o consume de memória RAM do job submetido juntamente ao nome do usuário que a submeteu.

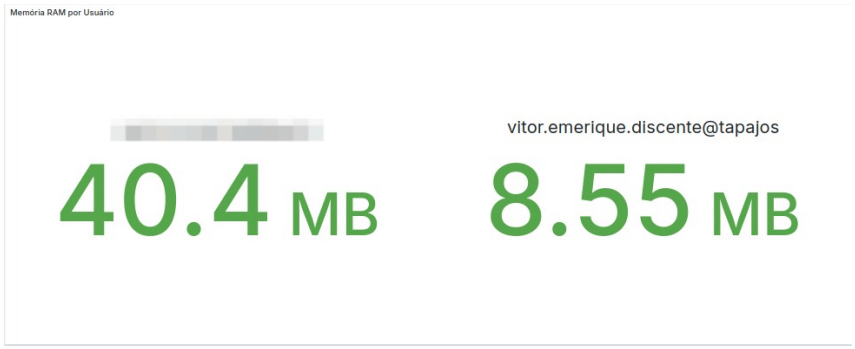


Figura 3.10 – Consumo de RAM por usuário.

Através da Figura 3.11 é possível visualizar a quantidade CPU solicitada pelo script da Figura 3.8, sendo solicitado dois nós com cada um disponibilizando uma CPU que juntos chegam a soma de 2 unidades de CPU exibindo juntamente o nome do usuário.

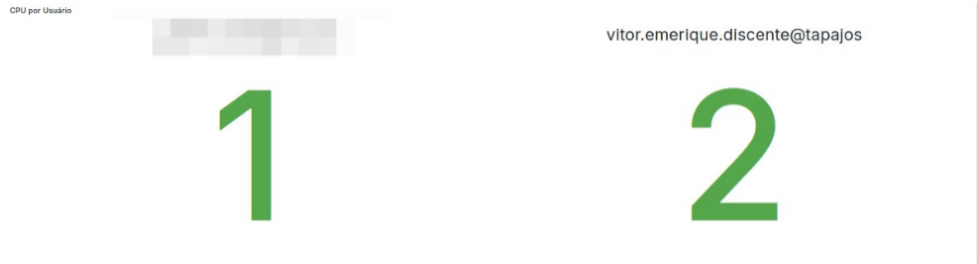


Figura 3.11 – Consumo de CPU por usuário.

Para melhor compreensão da quantidade de RAM sendo utilizada, foi usado o comando "qstat -f id_do_job"para verificar informações específicas de recursos usados, logo, na Figura 3.12 é possível visualizar em Kbytes o uso de memória RAM e numero de CPUs utilizadas. É importante ressaltar que esses teste foram em pleno funcionamento do *cluster*, não houve um ambiente controlado para realização dos testes.

```
vitor.emerique.discente@tapajos:~> qstat -f 14213
Job Id: 14213.tapajos
    Job_Name = TestJob
    Job_Owner = vitor.emerique.discente@tapajos
    resources_used.cpupercent = 0
    resources_used.cput = 00:00:00
    resources_used.mem = 8996kb
    resources_used.ncpus = 2
    resources_used.vmem = 45500kb
    resources_used.walltime = 00:00:10
```

Figura 3.12 – Verificando recursos usados pelo Job.

3.5 AVALIAÇÃO

Seguindo a metodologia DSR, passou-se para a etapa de **Avaliação**, onde foi possível atestar que a implantação do sistema de monitoramento de recursos sobre o uso e *status* atual do *Cluster*, usuários e monitoramento de filas de *jobs*. Além de garantir a compreensão do sistema, o sistema garante que ele seja mais tolerante a falhas, tenha alta disponibilidade, eficiência e contribua para a redução na pegada de carbono. Na etapa de avaliação também é possível utilizar o sistema para alcançar as diretrizes de sustentabilidade do PDI, possibilitando também o planejamento estratégico em vista ao aprimoramento da infraestrutura tecnológica, na busca da eficácia e eficiência nas aplicações deste importante ativo.

3.6 COMUNICAÇÃO

Por fim, a última etapa do DSR prevê a comunicação, que está sendo feita por meio de relatórios técnicos, repositórios públicos e submissão de artigo científico.

Como fruto desse trabalho, foi submetido e aprovado o artigo científico intitulado "Implantação e Avaliação de um Sistema de Monitoramento de Recursos Computacionais de *Cluster*: um enfoque em desenvolvimento sustentável" no XXV Simpósio em Sistemas Computacionais de Alto Desempenho (SSCAD - WIC) realizado pela Sociedade Brasileira de Computação (SBC), realizado na cidade de São Carlos - SP, com classificação Qualis B4. A versão final do artigo pode ser encontrado no Apêndice A.

Os resultados obtidos são melhor descritos na próxima Capítulo.

4 RESULTADOS

Nessa seção, serão expostos os resultados obtidos por essa pesquisa. As ferramentas utilizadas tem como principal objetivo implementar um sistema de monitoramento um *cluster* de alto desempenho se direcionando ao uso sustentável de forma que seu impacto ambiental seja reduzido, como um primeiro passo para a introdução da computação verde por meio da ferramenta de monitoramento de recursos, monitorando os principais recursos por meio de informações disponibilizadas pelo gerenciador de recursos computacionais do *cluster*.

O sistema é estruturado através do Prometheus, responsável pela coleta e armazenamento das métricas que em conjunto com o Grafana possibilita visualizar em *dashboard* de forma personalizada gráficos sobre a uso atual dos recursos do *cluster* bem como permite a análise temporal do comportamento dos recursos em uma faixa de tempo. Os gráficos apresentam informações relevante dos recursos, os gráficos estão divididos em quatro níveis, informações gerais, nós do *cluster*, usuário e OpenPBS *jobs*.

Embora as métricas tenham apresentado informações relevantes ao usuário, pode ser melhor ampliado as diversidades de informações com novos gráficos que apresentem métricas fora do escopo do gerenciador de recursos do *cluster*, como informações de rede e armazenamento.

A seção de informações gerais apresenta um panorama sobre o *cluster*, fornecendo informações sobre a quantidade total de *Central Processing Unit* (CPU) do *cluster* e o total de nós de processamento. Assim, expondo ao administrador uma visão sobre o porte do sistema. É possível visualizar na Figura 4.1 como é apresentado o gráfico.

Figura 4.1 – Visão geral do *cluster*.

A seção de nós do *cluster* é responsável por viabilizar o entendimento sobre a situação dos nós de processamento e fornecer informações pertinentes relacionadas a disponibilidade dos recursos de *Random-Acess memory* (RAM) e CPU. Nesse sentido, as informações relatadas nessa seção, os gráficos exibem ao usuário informações sobre a disponibilidade e ocupação dos recursos de RAM e CPU de forma geral, não obstante, exibe o *status* dos nós. A seção de usuário detalha as métricas de uso por usuário exibindo a quantidade de memória RAM e CPU que cada usuário está utilizando. Por fim, a seção destinada aos *jobs*, responsável por apresentar informações sobre os trabalhos somando o total de todos os estados e especificando também a quantidade de *jobs* em cada tipo de estado no *cluster*. Portanto, o dashboard permite o monitoramento ativo apresentado pela seções e a análise das métricas.

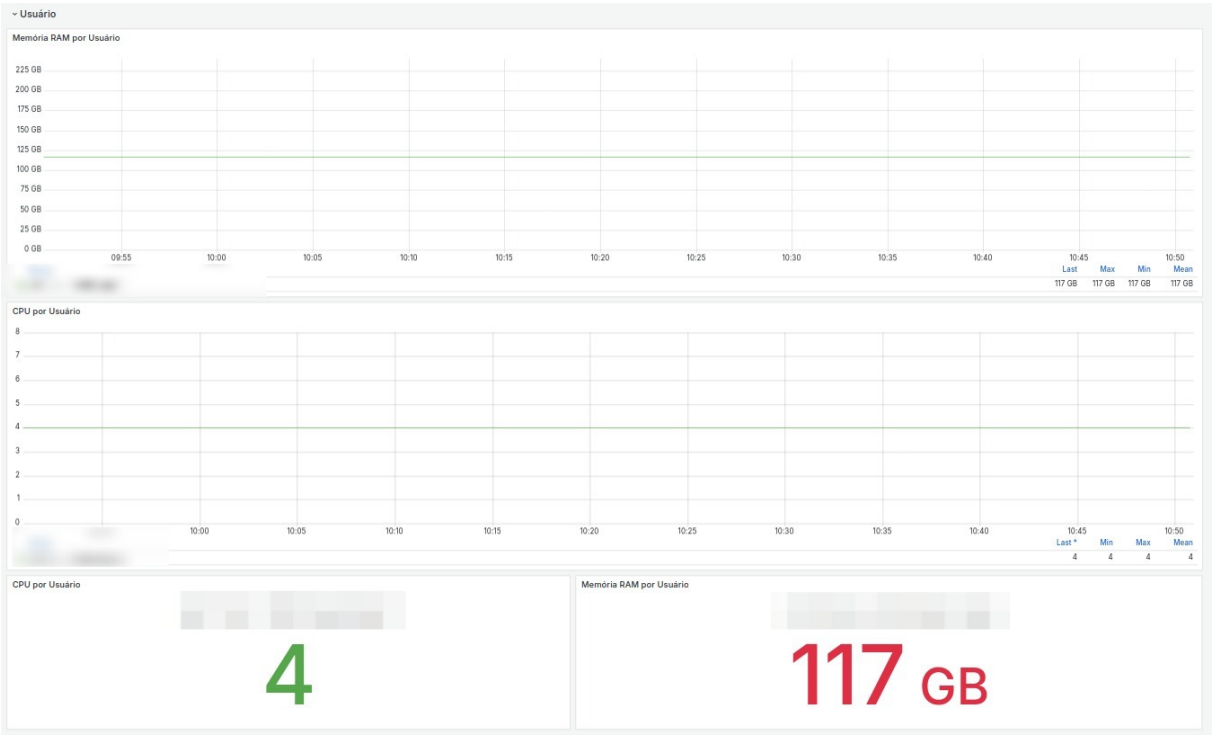
Em relação aos nós do *cluster*, foi possível visualizar os *status* dos nós e o consumo e disponibilidade de CPU, bem como o uso de *Random-Acess memory* (RAM), conforme apresentado na Figura 4.2. Os valores totais de CPU e RAM foram produto da soma dos recursos de todos os nós de processamento.

Figura 4.2 – Monitoramento do uso dos nós de CPU e de memória.

A solução desenvolvida permitiu também monitorar a utilização dos recursos com relação aos usuários. Na Figura 4.3 são apresentados o uso de recursos por usuários, como CPU e RAM, visualizadores do tipo *time series e Stat* foram utilizadas para exibir qual usuário e quanto recurso esta alocado a ele. Os dados dos usuários do *cluster* foram

omitidos devido a questões de privacidade.

Figura 4.3 – Recursos utilizados por usuário.



Por fim é exibido uma visão sobre os status dos *jobs* conforme a apresentado Figura 4.4, destacando-se a quantidade de *jobs* e o status da sua execução. Foram utilizados visualizadores de *time series e stat* para exibir as oscilações cronológicas dos status e contribuir para análises.

Figura 4.4 – Status de *jobs*.

Antes de ser implementado era possível visualizar todas as métricas apresentadas por meio dos comandos da Tabela 3.1 via terminal, porém não era possível analisar o comportamento desses nós, *jobs* e usuários no decorrer do tempo, devido essas informações não serem salvas e nem possuia um interface gráfica para monitorar. Outra via, a partir

desse sistema é possível entender o comportamento do sistema para auxiliar na tomada de decisão em relação a estratégias referentes ao *cluster*.

Portanto, o sistema de monitoramento implementado, garante a coleta e tratamento dos dados através do módulo *exporter* Prometheus e armazenamento das métricas no Prometheus *Server*. Além disso, através do PromQL no Grafana, permite a visualização das métricas mediante uma interface gráfica utilizando tabelas e a análise das métricas em uma faixa de tempo determinada pelo usuário.

5 CONSIDERAÇÕES FINAIS

Esse trabalho teve como objetivo principal implementar uma ferramenta de monitoramento de recursos computacionais no *cluster* de alto desempenho da Universidade Federal do Oeste do Pará campus Tapajós. Os resultados mostraram que a ferramenta pôde monitorar os nós computacionais, recursos de memória RAM e CPU e recursos por usuário, atingindo os objetivos propostos.

A principal contribuição deste trabalho foi a implementação de uma solução prática para o monitoramento do *cluster* de qualquer lugar que forneça acesso à rede de internet. A ferramenta preenche uma lacuna referente a soluções de código aberto, em vista da contribuição a ideologia *open-source*, disponibilizando em repositórios o *exporter* desenvolvido para o software agendamento de *jobs* e gerenciamento de recursos computacionais OpenPBS responsável pela coleta dos dados e a configuração preliminar do contêiner Docker contendo o Prometheus Server e o Grafana para possibilitar o armazenamento e visualização dos dados. Nesse sentido, é interessante ressaltar que não foram encontrados trabalhos que apresentaram solução de módulo *exporter* Prometheus para o gerenciador de recursos OpenPBS.

Neste Trabalho foi apresentado o processo de implementação e avaliação de um sistema de monitoramento de *cluster* de alta performance utilizando as ferramentas Prometheus e Grafana. Destaca-se o desenvolvimento de um *exporter* que lida com a biblioteca mantida oficialmente pelo Prometheus e disponibilizá-lo em repositório público juntamente a estrutura dashboard modelada no Grafana. Foi abordado o fluxo de trabalhado na coleta dos dados até a fase de *dashboard*, expondo os principais componentes responsáveis e as métricas monitoradas. A seção de resultados demonstrou que o sistema implementado oferece um visão do estado dos *jobs* e recursos, permitindo a gestão eficiente, auxiliando na tomada de decisão.

O monitoramento não se limita a recursos computacionais de *hardware*, mas possui um gama de níveis de monitoramento, esta é apenas uma das camadas de monitoramento que se pode utilizar para buscar entender o comportamento do sistema. Além do monitoramento de *hardware*, é possível monitora aplicações, sistemas, *cluster*. Nesse sentido, esse trabalho navega por duas camadas, no nível de *cluster* e ao nível de *hardware*, essa relação entre as duas camadas está presente na seguinte forma: monitorar os status dos *jobs* e os recursos consumidos pelos *jobs* nos nós computacionais. Dessa forma, o sistema de monitoramento personalizado é de grande valor para os administradores, garantindo pesquisas mais produtivas e a redução do custo de operação.

Diante do exposto, o presente trabalho alcançou o proposito de implementar um sistema de monitoramento baseado nas tecnologias apresentadas, alinhando-se com o PDI da Universidade Federal do Oeste do Pará, visando promover a computação verde. Através dessa ferramenta, é possível entender o comportamento do sistema, permitindo a tomada de decisões para otimizar o uso e aumentando o tempo de vida do *cluster*, minimizando impactos ambientais na região Amazônica sem comprometer o poder computacional, permitindo o desenvolvimento sustentável por meio deste ativo.

O trabalho apresenta limitações quando relacionada a variação de métricas, o escopo de métricas se limitou a apenas dados providos dos comandos de monitoramento do gerenciador de recursos OpenPBS, se limitando a apenas métricas de recursos de CPU,RAM, recursos por usuário, *status* na fila.

Nesse sentido para trabalhos futuros, visa-se ampliar o conjunto de métricas de armazenamento, rede e processos, expandir o dashboard e definir alertas Prometheus, alcançar outros hardwares, assim como explorar a diversidade de métrica que possam ser coletadas. Portanto, visando garantir informações mais detalhadas e precisas para a tomada de decisão da gestão estratégica e o entendimento do comportamento mais amplo do *cluster*.

REFERÊNCIAS

ALONSO, P.; RANILLA, J.; VIGO-AGUIAR, J. High-performance computing: the essential tool and the essential challenge. **The Journal of Supercomputing**, Springer, v. 73, p. 1–3, 2017.

Associação Brasileira de Normas Técnicas. **ABNT NBR ISO 14040: gestão ambiental – avaliação do ciclo de vida – princípios e estrutura**. Rio de Janeiro, 2009.

BAUMANN, M.; GEBHART, F.; MATTES, O.; NIKAS, S.; HEUVELINE, V. Development and implementation of a temperature monitoring system for hpc systems. **Preprint Series of the Engineering Mathematics and Computing Lab**, n. 07, 2017.

BUYYA, R. et al. High performance cluster computing: Architectures and systems (volume 1). **Prentice Hall, Upper SaddleRiver, NJ, USA**, v. 1, n. 999, p. 29, 1999.

CHI, W.; ZHOU, W. A realtime monitoring method for cluster system running state based on network. In: IOP PUBLISHING. **Journal of Physics: Conference Series**. [S.l.], 2019.

CLINTON, D.; NEGUS, C. **Ubuntu Linux Bible**. [S.l.]: John Wiley & Sons, 2020.

COCAÑA-FERNÁNDEZ, A.; GUIOTE, E. S. J.; RANILLA, J.; SÁNCHEZ, L. Improving eecluster to optimize the carbon footprint and operating costs of hpc clusters. In: IEEE. **2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)**. [S.l.], 2019. p. 1–6.

EITZINGER, J.; GRUBER, T.; AFZAL, A.; ZEISER, T.; WELLEIN, G. Clustercockpit—a web application for job-specific performance monitoring. In: IEEE. **2019 IEEE International Conference on Cluster Computing (CLUSTER)**. [S.l.], 2019. p. 1–7.

HAGER, G.; WELLEIN, G. **Introduction to high performance computing for scientists and engineers**. [S.l.]: CRC Press, 2010.

IGNÁCIO, A. L. J.; DIAS, W. R. A. Análise do desempenho computacional de algoritmos paralelizados com openmp e mpi executados em raspberry pi. In: SBC. **Simpósio em Sistemas Computacionais de Alto Desempenho (SSCAD)**. [S.l.], 2023. p. 41–48.

KASHIN, D.; VOEVODIN, V. Verifying the correctness of hpc performance monitoring data. In: SPRINGER. **International Conference on Parallel Computing Technologies**. [S.l.], 2023. p. 197–208.

KUNZ, P. Hpc job-monitoring with slurm, prometheus and grafana. 2022.

LI, B.; ROY, R. B.; WANG, D.; SAMSI, S.; GADEPALLY, V.; TIWARI, D. Toward sustainable hpc: Carbon footprint estimation and environmental implications of hpc systems. In: **Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis**. [S.l.: s.n.], 2023. p. 1–15.

MISHRA, D.; AKMAN, I.; MISHRA, A. Theory of reasoned action application for green information technology acceptance. **Computers in human behavior**, Elsevier, v. 36, p. 29–40, 2014.

MOHAPATRA, S. K.; NAYAK, P.; MISHRA, S.; BISOY, S. K. Green computing: a step towards eco-friendly computing. In: **Emerging trends and applications in cognitive computing**. [S.l.]: IGI global, 2019. p. 124–149.

NETO, A. J. A.; NETO, J. A. C.; MORENO, E. D. Low-cost clusters on big data - a systematic study. **Proceedings of the 11th Euro American Conference on Telematics and Information Systems**, 2022.

NETO, A. J. de M.; MÉXAS, M. P.; NETO, J. V. Práticas de tecnologia da informação verde: uma revisão sistemática da literatura. **Revista de Gestão e Secretariado**, v. 15, n. 7, p. e3943–e3943, 2024.

PEFFERS, K.; TUUNANEN, T.; GENGLER, C. E.; ROSSI, M.; HUI, W.; VIRTANEN, V.; BRAGGE, J. **Design Science Research Process: A Model for Producing and Presenting Information Systems Research**. 2020. Disponível em: <https://arxiv.org/abs/2006.02763>.

PEFFERS, K.; TUUNANEN, T.; ROTHENBERGER, M. A.; CHATTERJEE, S. A design science research methodology for information systems research. **Journal of management information systems**, Taylor & Francis, v. 24, n. 3, p. 45–77, 2007.

PIMENTEL, M.; FILIPPO, D.; SANTOS, T. M. dos. Design science research: pesquisa científica atrelada ao design de artefatos. **RE@ D-Revista de Educação a Distância e eLearning**, v. 3, n. 1, p. 37–61, 2020.

RADOVANOVIĆ, A.; KONINGSTEIN, R.; SCHNEIDER, I.; CHEN, B.; DUARTE, A.; ROY, B.; XIAO, D.; HARIDASAN, M.; HUNG, P.; CARE, N. et al. Carbon-aware computing for datacenters. **IEEE Transactions on Power Systems**, IEEE, v. 38, n. 2, p. 1270–1280, 2022.

ROBEY, R.; ZAMORA, Y. **Parallel and high performance computing**. [S.l.]: Simon and Schuster, 2021.

ROSA, M. R. da. Adoption of green it in the university environment: systematic review of sustainability practices in educational institutions. **AtoZ: novas práticas em informação e conhecimento**, v. 9, n. 2, p. 79–87, 2020.

SAPUTRA, M. Y. E.; ARIEF, S. N.; WIJAYANINGRUM, V. N.; SYAIFUDIN, Y. W. et al. Real-time server monitoring and notification system with prometheus, grafana, and telegram integration. In: IEEE. **2024 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS)**. [S.l.], 2024. p. 1808–1813.

SILVA, H. B. da; GARCIA, V. C.; SOUSA, T. C. de; RABÊLO, R. A. L.; SOARES, J. H. C.; LOPES, L. A. Extração de conhecimento em um survey sobre o uso de computação em nuvem no brasil. 2016.

SORKUNLU, N.; CHANDOLA, V.; PATRA, A. Tracking system behavior from resource usage data. In: IEEE. **2017 IEEE International Conference on Cluster Computing (CLUSTER)**. [S.l.], 2017. p. 410–418.

STANISIC, L.; REUTER, K. Mpcdf hpc performance monitoring system: Enabling insight via job-specific analysis. In: SPRINGER. **Euro-Par 2019: Parallel Processing Workshops: Euro-Par 2019 International Workshops, Göttingen, Germany, August 26–30, 2019, Revised Selected Papers 25**. [S.l.], 2020. p. 613–625.

Universidade Federal do Oeste do Pará. **Plano Diretor de Tecnologia da Informação e Comunicação**. 2019. Disponível em: <https://www.ufopa.edu.br/media/file/site/ctic/documentos/2021/9890c9a6c71dca4c8e537642653d8922.pdf>.

Universidade Federal do Oeste do Pará. **Plano de Desenvolvimento Institucional - PDI**. 2024. Disponível em: <https://pdi.ufopa.edu.br/media/file/site/pdi/documentos/2024/2bbc3be3fa445f93422176bada4aaebe.pdf>.

APÊNDICE A – ARTIGO SUBMETIDO PARA SSCAD-WIC

Implantação e Avaliação de um Sistema de Monitoramento de Recursos Computacionais de *Cluster*: um enfoque em desenvolvimento sustentável

Vitor Torres Emerique[1], **Fábio Manoel França Lobato**[1], **Marcelino da Silva Silva**[1]

[1]Instituto de Engenharia e Geociências

Universidade Federal do Oeste do Pará – Santarém – PA – Brasil

vitor.emerique@discente.ufopa.edu.br, marcelino.ss@ufopa.edu.br

Abstract. *Monitoring systems improve the strategic management of computer resources. The literature presents various monitoring tools, but there is a gap in relation to systems that use the OpenPBS resource manager. This work aims to implement a computer resource monitoring system as a precursor tool for introducing green computing into the cluster aligned with the Institutional Development Plan of a Federal Higher Education Institution located in the heart of the Amazon region. The system aims to reduce the carbon footprint, contributing to the sustainable development of the Legal Amazon. The successful implementation of the monitoring system contributes to a more comprehensive view of the high-performance computer, guiding the strategic management of this critical asset.*

Resumo. *A implantação do sistema de monitoramento computacional aprimora a gestão estratégica de recursos computacionais. A literatura apresenta diversas ferramentas para monitoramento, porém, há uma lacuna em relação a sistemas que utilizam o gerenciador de recursos OpenPBS. Este trabalho visa contribuir para o desenvolvimento sustentável da Amazônia legal implementando um sistema de monitoramento de recursos computacionais como ferramenta precursora para introdução do green computing no cluster alinhada ao Plano de Desenvolvimento Institucional de uma Instituição Federal de Ensino Superior localizada no coração da região Amazônica. A implantação bem-sucedida do monitoramento contribui para uma visão mais abrangente do sistema de alto desempenho, guiando a gestão estratégica deste importante ativo.*

1. Introdução

Considerando que computadores de alto desempenho são recursos escassos, com pouco centros disponíveis, devido ao seu custo de aquisição, bem como alto custo de manutenção, energia e refrigeração; a gestão estratégica para otimizar o seu uso e aumentar o seu tempo de vida se fazem prementes. Ademais, a necessidade do desenvolvimento sustentável, a adoção da *green computing*[1] é necessária, objetivando a utilização eficiente dos recursos computacionais, minimizando os impactos ambientais sem comprometer as necessidades tecnológicas [Mohapatra et al. 2019]. Convém pontuar que as máquinas de

[1]Visa a utilização eficiente dos recursos computacionais minimizando os impactos ambientais

alto desempenho são responsáveis pela emissão de mais de cem milhões de toneladas de dióxido de carbono por ano [Cocaña-Fernández et al. 2019]. Nesse sentido, levando em conta que o *cluster*, objeto de trabalho do presente estudo, é localizado na Amazônia Legal, são necessárias medidas que tendem a reduzir transtorno ambiental da pegada de carbono deste ativo digital.

Sendo assim, sistemas de monitoramento de recursos computacionais são uma importante ferramenta precursora para tomada de decisões e para o melhor entendimento do comportamento do ativo digital [Kashin and Voevodin 2023]. Além disso, permitir a análise do uso de recursos possibilita entender o desempenho do sistema e identificar suas anomalias [Sorkunlu et al. 2017]. A emissão de gás carbônico ocorre também no transporte, ciclo de vida o qual os principais emissores na sua produção são os componentes de armazenamento de disco rígido e placas gráficas e *Dynamic Random Acess Memory* (DRAM), por fim, na reciclagem [Li et al. 2023]. Dessa forma, o monitoramento de recursos gera percepções que possibilitam o direcionamento na tomada de decisões, sendo uma ferramenta relevante como solução para a introdução do *green computing* no *cluster*.

Frente ao exposto, este estudo visa apresentar o processo de implementação do sistema de monitoramento de recursos com base nos dados fornecidos pelo OpenPBS[1]. Desse modo, foi utilizada a ferramenta Prometheus[2] para realizar a coleta dos dados e armazenamento das métricas. A linguagem Go[3] foi utilizada para desenvolver um *exporter*[4] customizado para a coleta e concentração dos dados de séries temporais, permitindo a visualização e análise dos gráficos em um *dashboard* na plataforma Grafana [5], uma solução também *open-source*. O trabalho visou cumprir com diretrizes institucionais de sustentabilidade, previstos no Plano de Desenvolvimento Institucional (PDI) por meio da abordagem de *green computing* e monitoramento de recursos computacionais, possibilitando também a gestão estratégica deste ativo, já influenciando no planejamento de aquisições como o sistema de refrigeração por precisão e na necessidade de se expandir o armazenamento considerando o uso atual do *cluster*.

O restante deste artigo está organizado como segue. Na Seção 2 são apresentados os trabalhos relacionados. Os materiais e métodos são descritos na Seção 3. Na Seção 4 são expostos e discutidos os resultados. Na Seção 5 são apresentadas as considerações finais.

2. Trabalhos Relacionados

[Baumann et al. 2017] desenvolveu e implementou um sistema de monitoramento de temperatura de um *cluster* utilizando sensores de temperatura digitais conectados a um *Raspberry Pi* buscando otimizar o desempenho do hardware em relação ao calor dissipado. Os

[1]Agendador de *jobs* e gerenciador de carga de trabalho em ambiente de computação de alto desempenho. https://www.openpbs.org/

[2]Um *kit* de ferramentas de alerta e monitoramento de sistemas e sua biblioteca disponibilizada oficialmente para algumas linguagens de programação. https://prometheus.io/docs/introduction/overview/

[3]Linguagem de Programação desenvolvida pela Google. https://go.dev/

[4]Componente do Prometheus que realiza a coleta dos dados de um sistema específico. https://prometheus.io/docs/operating/security/#exporters

[5]Ferramenta responsável pela visualização permitindo análises temporais. https://grafana.com/oss/grafana/

dados foram lidos por meio da linguagem Python utilizando a *Python Threading Library*. Por outro lado, [Chi and Zhou 2019] utiliza uma abordagem baseada em *software*, coletando dados dos sensores dos próprios computadores, gerenciados pelo *kernel* do sistema operacional, com uma abordagem arquitetural cliente-servidor distribuída com objetivo de melhorar ferramentas já existentes e garantindo o monitoramento em tempo real via protocolo de rede. Este corrente trabalho, salienta o foco na coleta de métricas mediante softwares, correspondentes ao gerenciador de recursos do sistema HPC.

[Saputra et al. 2024] também monitora via software, utilizando o sistema Prometheus, em detrimento ao sistema diretamente ligado ao *Kernel* do sistema operacional tal como [Chi and Zhou 2019]. Este sistema de monitoramento proposto por [Saputra et al. 2024] permite uma melhor gestão dos dados, ao poder ser ligado a múltiplos nós, sendo necessário a presença do *exporter* em cada uma das máquinas. Além de exibir os dados via *dashboard* pelo Grafana, o sistema proposto por [Saputra et al. 2024] também tem a capacidade de gerar alerta por meio do Telegram. O presente trabalho destaca-se por implementar um *exporter* personalizado para o gerenciador de recursos OpenPBS na linguagem Go, utilizando a biblioteca oficial do Prometheus com a abordagem de apenas um *target exporter* para coleta de dados.

Semelhantemente, [Kunz 2022], além da implementação, desenvolveu um *exporter* personalizado para o *workload manager* e *job scheduling system* Slurm[1]. Nesse sentido, este trabalho destaca-se por prover toda a estrutura de software nas dependências do sistema HPC, Prometheus, *exporter* e Grafana. [Stanisic and Reuter 2020] apresenta um *design* e implantação de um sistema de monitoramento HPC com uma solução arquitetural apresentando os passos de coleta, análise e exposição dos dados, um intermediário executa em todos os nós do *cluster*, concentrando informações de monitoramento do Slurm e outras informações de métricas, persistência de dados e a visualização utilizadas pela ferramenta Splunk[2]. Nesse viés, o presente trabalho se destaca por monitorar os nós de processamento computacional de todos os estados de disponibilidade e os usuários que utilizam o sistema HPC.

É importante ressaltar que, pelo nosso melhor conhecimento, não há na literatura trabalhos que explorem o monitoramento de desempenho utilizando o gerenciador de recursos OpenPBS como alvo para a coleta de métricas. Este trabalho se diferencia dos correlatos por implementar o monitoramento de recurso computacional em um *Cluster* localizado na Universidade Federal do Oeste do Pará, seguindo os princípios de *green computing* e alinhando-se com o PDI. A proposta utiliza métricas provenientes do OpenPBS, desenvolvendo um *exporter* personalizado Prometheus para coleta das métricas e o Grafana para visualização e análise em *dashboard*.

3. Materiais e Métodos

A metodologia de pesquisa selecionada para este trabalho foi o *Design Science Research* (DSR), processo estabelecido por [Peffers et al. 2020], o processo consiste em seis fases como apresentado na Figura 1, sendo essas: (I) Identificação do problema e motivação,

[1]Um sistema de gerenciamento de cluster e agendamento de tarefas de código aberto, tolerante a falhas e altamente escalável para clusters Linux grandes e pequenos

[2]Ferramenta que realiza a coleta dos dados, armazenamento, visualização e permite análise das métricas.

(II) Objetivos para solução, (III) Design, (IV) Implementação, (V) Avaliação e (VI) Comunicação.

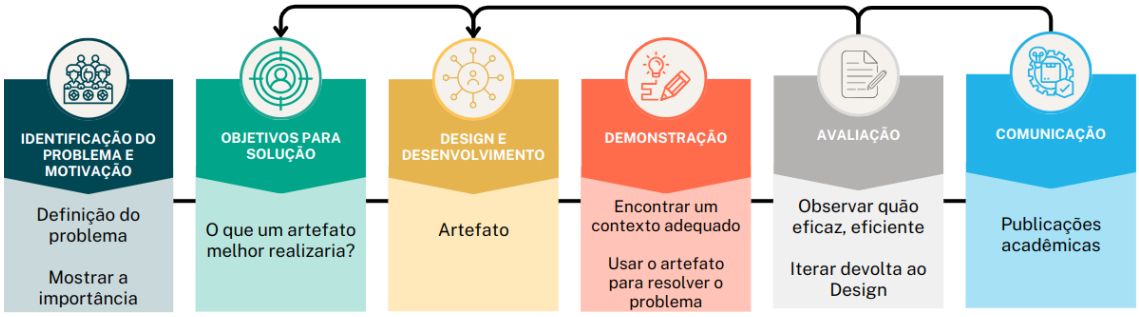


Figura 1. Modelo do *Design Science Research*. Adaptado de [Peffers et al. 2020].

Na *Identificação do Problema e Motivação* foi identificada a ausência de monitoramento computacional em relação ao consumo de recursos no *Cluster* que serve como objeto de estudo. Logo, o problema encontrado, representa-se pela falta de observabilidade computacional em relação aos recursos computacionais. Com isso, seguindo a metodologia, delineou-se como **Objetivo da Solução**: Desenvolver um sistema de monitoramento de recursos computacionais e garantir um impacto positivo no funcionamento do *cluster*, seguindo as diretrizes de sustentabilidade apontadas no Plano de Desenvolvimento Institucional.

Partindo para o **Design e Implementação**, após o levantamento de ferramentas voltadas para a telemetria e uma estratégia de coleta das métricas por meio da utilização de apenas um *target*, percebeu-se a necessidade da ferramenta Prometheus para coleta de métrica de séries temporais e o armazenamento. Posteriormente, para exibição de gráficos, selecionou-se a Grafana, uma ferramenta de visualização e análise desses dados foi utilizada. Dessa forma, foi necessário desenvolver um *exporter* personalizado para coletar as métricas do *cluster*. Um exemplo de *exporter* e a estrutura do *Dashboard* se encontra disponível publicamente em um repositório[1], visando atender aos princípios da ciência aberta conforme preconizado pela Sociedade Brasileira de Computação. Na Figura 2 é exposto o fluxo de execução em uma visão global do trabalho.

Conforme pode ser visto na Figura 2, o fluxo se inicia a partir dos comandos de monitoramento de *jobs* e nós provenientes do OpenPBS, coletando informações sobre o *status* dos *jobs* submetidos, informações sobre o *status* dos nós e utilização de recursos. Após o entendimento de ambiente computacional, a estratégia adotada foi utilizar apenas um *target* e um exporter, pois não haveria necessidade de configurar um *exporter* para cada nó. Assim, foi necessário que um dos nós utilize esta aplicação, mitigando o uso de mais recurso computacional, configurado no *Header node*. Os dados de séries temporais são concentrados no Prometheus *Server* sendo armazenados em um banco de dados, o *Time Series Data Base* (TSDB). A Tabela 1 apresenta alguns exemplos de comandos utilizados pelo OpenPBS.

A Tabela 1 mostra comandos de monitoramento proveniente do OpenPBS. Por meio deles foi possível alcançar os dados de saída alvo. Em seguida, após obter os da-

[1]https://github.com/vitoremerique/Openpbs_Exporter.git

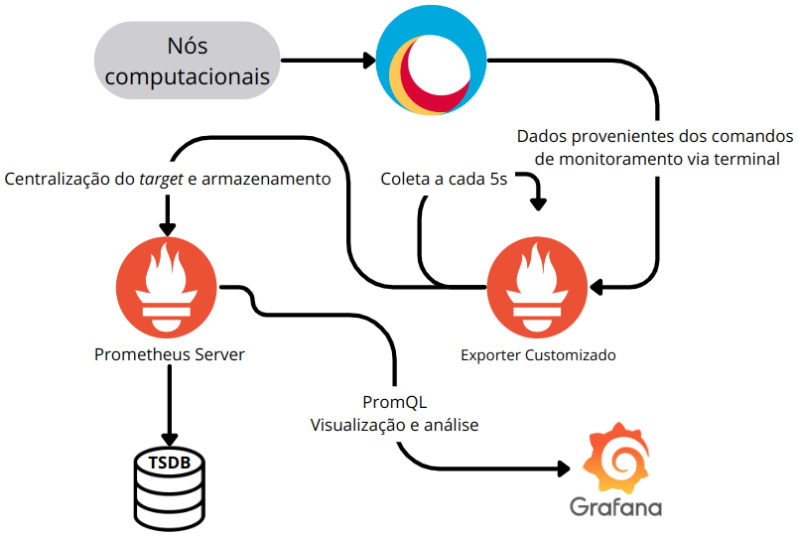


Figura 2. Fluxo de execução geral

Comando	Descrição
pbsnodes	Exibe informações detalhadas sobre todos os nós do *cluster*
qstat	Mostra o *status* dos *jobs* na fila

Tabela 1. Comandos de monitoramento utilizados do OpenPBS

dos via código terminal, utilizou-se a biblioteca do Prometheus para criar um servidor *exporter* pro meio dos *parsers* implementados. Os dados de saída foram convertidos em métricas do tipo *Gauge*[1]. Nesse sentido, foi possível disponibilizar para o Prometheus *server* o *target* oriundo do *exporter* e concentrar e armazenas as métricas.

Após o processo de coleta e armazenamento das métricas coletadas, utilizou-se o Prometheus *server* como um *data source* na plataforma Grafana, permitindo montar gráficos específicos para cada métrica, além de possibilitar a correlação de métricas por meio de gráficos, visualizando e analisando o que ocorreu em um passado já coletado ou monitorando os recursos com um *delay* de 5 segundos, senda esse o tempo de atualização mínimo no Grafana, com isso as coletas Prometheus se mantém nesse mesmo tempo de atualização. Isso foi possível devido o Prometheus possuir sua própria *query language* chamada *PromQL*, utilizada para fazer a chamada de requisição para visualização em tempo real com atualização periódica, na Figura 3 é possível ter uma visão geral.

Seguindo a metodologia DSR, passou-se para a etapa de **Demonstração**, foram realizados testes de inclusão de *jobs* na fila e analisado a confiabilidade dos dados no *dashboard*. Foi efetuada uma investigação utilizando os comandos da Tabela 1 para garantir que os dados eram exibidos corretamente.

Na etapa **Avaliação**, a implantação do sistema de monitoramento de recursos possibilitou entender o comportamento do sistema computacional por meio do uso de recursos, cobrindo uma lacuna do sistema computacional de alto desempenho do *cluster*,

[1]Métrica que representa um valor numérico único que pode diminuir ou aumentar arbitrariamente.

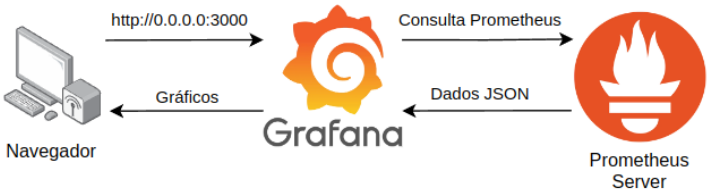


Figura 3. Fluxo do Grafana com Prometheus

sendo uma importante ferramenta para a tomada de decisões como uma solução para a introdução do *green computing* relacionado a estratégia que possibilitem por meio das análises das métricas alcançar um ambiente mais sustentável do cluster. Além disso, foi possível sanar uma lacuna do PDI da universidade em relação ao desenvolvimento sustentável.

Por fim, a última etapa do DSR prevê a comunicação, que está sendo feita por meio de relatórios técnicos, repositórios públicos e submissão de artigos científicos como o presente estudo. Os resultados obtidos são melhor descritos na próxima Seção.

4. Resultados

Nessa seção, serão expostos os resultados obtidos por essa pesquisa. As principais métricas utilizadas estão relacionadas a disponibilidade dos nós do *cluster*, informações relacionadas ao uso de memória RAM e CPU, Uso de recursos por usuário e informes sobre os *jobs*. O primeiro ponto foi visualizar as Informações gerais do *cluster*. Esses dados procuram dar uma visão mais ampla da grandeza do *cluster* exibindo a quantidade total de *Central Processing Unit* (CPU) do *cluster* e o total de nós, conforme a Figura 4, as tabelas são do tipo *stat*.

Figura 4. Visão geral do *cluster*

Em relação aos Nós do *cluster*, foi possível analisar os *status* dos nós e o consumo e disponibilidade de CPU, bem como o uso de *Random-Acess memory* (RAM), conforme apresentado no lado A da Figura 5.

A solução desenvolvida permitiu também monitorar a utilização dos recursos com relação aos usuários. No lado B da Figura 5 são apresentados o uso de recursos por usuários, como CPU e RAM, tabelas do tipo *time series e Stat* foram utilizadas para exibir qual usuário e quanto recurso esta alocado a ele. Os dados dos usuários do *cluster* foram omitidos devido a questões de privacidade.

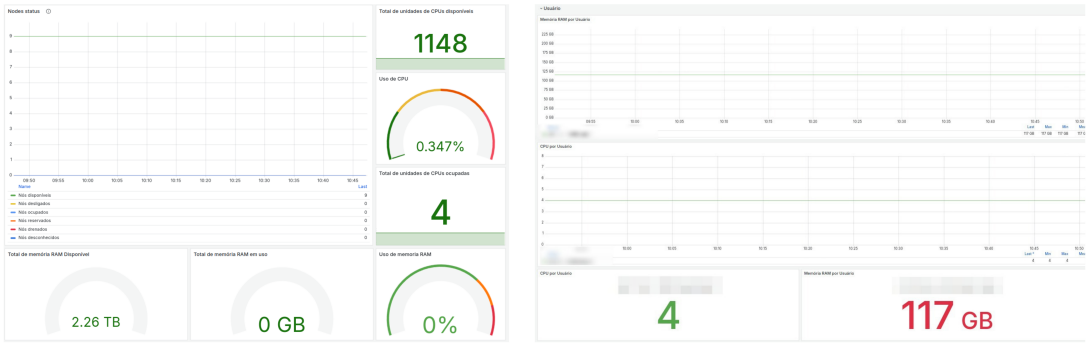


Figura 5. Monitoramento do uso de GPU e memória dos nós e Recursos utilizados por usuário

Por fim é exibido uma visão sobre os *status* dos *jobs* conforme é apresentado na Figura 6, destacando-se a quantidade de *jobs* e o *status* da sua execução. Foi utilizado uma tabela de *time series e stat* para exibir as oscilações cronológicas dos *status* e contribuir para análises.

Figura 6. *Status* de *jobs*

5. Considerações Finais

Neste artigo apresentamos o processo de implementação e avaliação de um sistema de monitoramento de *cluster* de alta performance utilizando Prometheus e Grafana. Destaca-se o desenvolvimento de um *exporter* que lida com a biblioteca mantida oficialmente pelo Prometheus. Abordamos o fluxo de trabalhado na coleta dos dados até a fase de *dashboard*, conforme a Figura 2, expondo os principais componentes responsáveis e as métricas monitoradas.

Diante do exposto, o presente trabalho alcançou objetivo de implementar um sistema de monitoramento baseado nas tecnologias apresentadas, alinhando-se com o PDI da Instituição Federal de Ensino Superior, aderindo o *green computing* por meio do monitoramento de recursos computacionais como ferramenta precursora para estratégias de redução de consumo de energia consequentemente na pegada de carbono por meio da solução proposta, com benefícios de otimização do uso dos recursos e o aumento do tempo de vida do *cluster*, minimizando impactos ambientais na região Amazônica sem comprometer o poder computacional, permitindo o desenvolvimento sustentável por meio deste ativo.

Agradecimentos

Este trabalho foi apoiado pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq)-DT-303031/2023-9; e pela Fundação Amazônia de Amparo a Estudos e Pesquisas (FAPESPA) PRONEM-FAPESPA/CNPq nº 045/2021.

Referências

[Baumann et al. 2017] Baumann, M., Gebhart, F., Mattes, O., Nikas, S., and Heuveline, V. (2017). Development and implementation of a temperature monitoring system for hpc systems. *Preprint Series of the Engineering Mathematics and Computing Lab*, (07).

[Chi and Zhou 2019] Chi, W. and Zhou, W. (2019). A realtime monitoring method for cluster system running state based on network. In *Journal of Physics: Conference Series*, number 2. IOP Publishing.

[Cocaña-Fernández et al. 2019] Cocaña-Fernández, A., Guiote, E. S. J., Ranilla, J., and Sánchez, L. (2019). Improving eecluster to optimize the carbon footprint and operating costs of hpc clusters. In *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–6. IEEE.

[Kashin and Voevodin 2023] Kashin, D. and Voevodin, V. (2023). Verifying the correctness of hpc performance monitoring data. In *International Conference on Parallel Computing Technologies*, pages 197–208. Springer.

[Kunz 2022] Kunz, P. (2022). Hpc job-monitoring with slurm, prometheus and grafana.

[Li et al. 2023] Li, B., Basu Roy, R., Wang, D., Samsi, S., Gadepally, V., and Tiwari, D. (2023). Toward sustainable hpc: Carbon footprint estimation and environmental implications of hpc systems. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15.

[Mohapatra et al. 2019] Mohapatra, S. K., Nayak, P., Mishra, S., and Bisoy, S. K. (2019). Green computing: a step towards eco-friendly computing. In *Emerging trends and applications in cognitive computing*, pages 124–149. IGI global.

[Peffers et al. 2020] Peffers, K., Tuunanen, T., Gengler, C. E., Rossi, M., Hui, W., Virtanen, V., and Bragge, J. (2020). Design science research process: A model for producing and presenting information systems research.

[Saputra et al. 2024] Saputra, M. Y. E., Arief, S. N., Wijayaningrum, V. N., Syaifudin, Y. W., et al. (2024). Real-time server monitoring and notification system with prometheus, grafana, and telegram integration. In *2024 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS)*, pages 1808–1813. IEEE.

[Sorkunlu et al. 2017] Sorkunlu, N., Chandola, V., and Patra, A. (2017). Tracking system behavior from resource usage data. In *2017 IEEE International Conference on Cluster Computing (CLUSTER)*, pages 410–418. IEEE.

[Stanisic and Reuter 2020] Stanisic, L. and Reuter, K. (2020). Mpcdf hpc performance monitoring system: Enabling insight via job-specific analysis. In *Euro-Par 2019: Parallel Processing Workshops: Euro-Par 2019 International Workshops, Göttingen, Germany, August 26–30, 2019, Revised Selected Papers 25*, pages 613–625. Springer.