# 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.

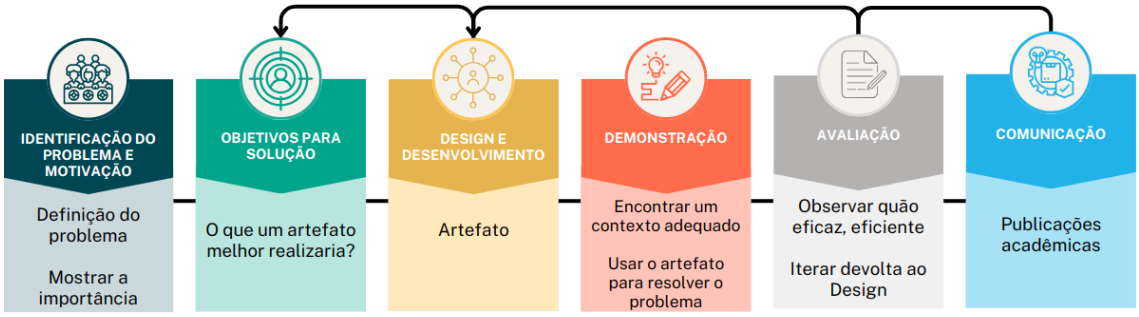


**Figura 1. Modelo do *Design Science Research*. Adaptado de [Peffers et al. 2020].**

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

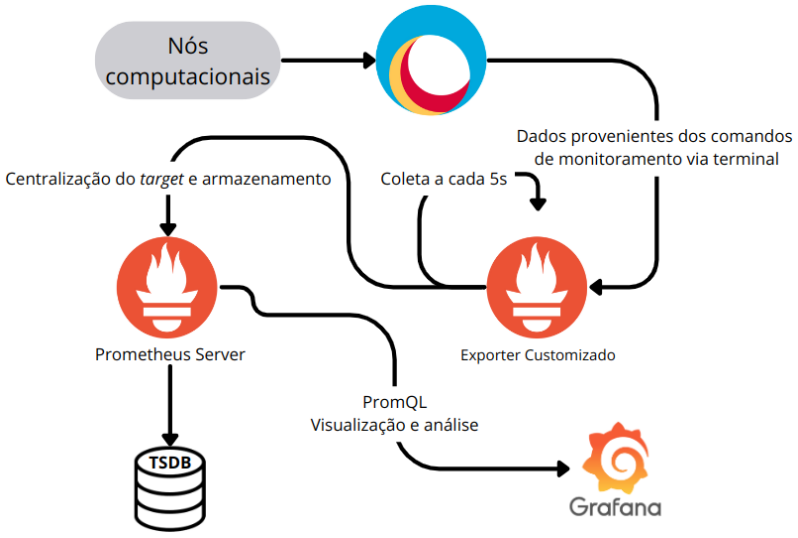


**Figura 2. Fluxo de execução geral**

| **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.

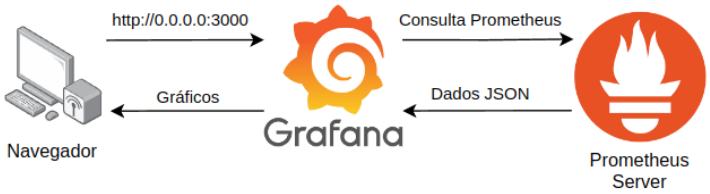


**Figura 3. Fluxo do Grafana com Prometheus**

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.

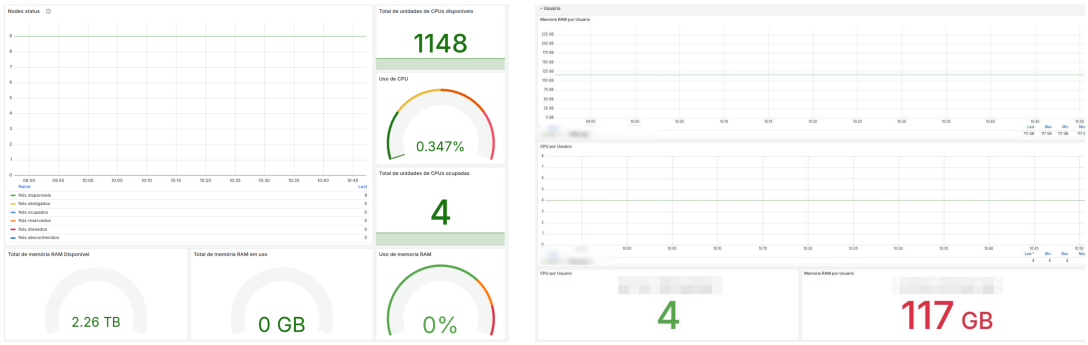


**Figura 5. Monitoramento do uso de GPU e memória dos nós e Recursos utilizados por usuário**

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.