Introduction of Dynamic Job Matching Optimization for Grid Middleware using Site Sonar Infrastructure Monitoring

Kalana Wijethunga[1,2], Costin Grigoras[1], Latchezar Betev[1] and Indika Perera[2]

[1]CERN, Geneva, Switzerland
[2]Department of Computer Science and Engineering, University of Moratuwa, Sri Lanka

E-mail: kalana.16@cse.mrt.ac.lk

Abstract. In the realm of Grid middleware, efficient job matching is paramount, ensuring that tasks are seamlessly assigned to the most compatible worker nodes. This process hinges on meticulously evaluating a worker node's suitability for the given task, necessitating a thorough assessment of its infrastructure characteristics. However, adjusting job matching parameters poses a significant challenge due to the involvement of both central and site services within the Grid middleware. This necessitates deploying a new middleware version across the entire Grid, introducing potential bugs and raising the risk of a single point of failure.

Furthermore, the inherent limitations in the number of available job matching parameters, stemming from insufficient infrastructure monitoring in pilot jobs, further complicate the task for Grid middleware developers.

This paper introduces an entirely new approach for dynamically adding and modifying job matching parameters in Grid middleware, leveraging the Site Sonar Grid Infrastructure monitoring framework. This solution empowers Grid administrators to seamlessly add or modify job matching parameters without altering the core middleware code. This flexibility enables dynamic job matching based on diverse infrastructure properties of worker nodes. By decoupling job matching parameters from the Grid middleware, the proposed approach enhances flexibility, mitigates complexities, and reduces risks associated with introducing and changing job matching parameters.

This transformative approach bolsters the adaptability of Grid middleware for heterogeneous systems, fostering optimized resource allocation.

Keywords: Grid Monitoring, Grid Infrastructure, Dynamic Job Matching, Site Sonar

1 Introduction

Computing Grids are a popular computing paradigm that is used to execute large scale computation in a cost effective way by aggregating separate computing resources in different organizations to a single entity. The ALICE experiment(1) at CERN heavily relies on the computing sites in Worldwide Large Hadron Collider Grid (WLCG)(2) along with more non-WLCG computing sites, aggregated to form The ALICE Grid to process the massive amount of data generated from the ALICE experiment.

The ALICE Grid currently consists of 54 computing sites, comprising 9500+ shared worker nodes that contains a sum of 600,000+ logical CPU cores. JAliEn(3) is used as the Grid middleware for the ALICE Grid. When a user submits a job to the Grid, JAliEn takes on the responsibility of executing it in the most suitable node in the Grid. Currently, this is done by matching a limited set of infrastructure parameters like CPU cores, memory, disk space requested by the job to the amount of resources available in the worker node. While this provides proper Grid functionality to the ALICE Grid, it can be seen that the efficiency of resource usage can be greatly improved by introducing more fine tuned job matching. Introducing a flexible, fine tuned job matching process with the current architecture of JAliEn is not possible because it requires upgrading both central and site services and deploying a new release across the Grid to introduce a new job matching parameter.

This paper introduces a novel approach to enhance Grid job matching by introducing the ability to dynamically change job matching parameters depending on the requirement of the Grid using the Site Sonar Infrastructure monitoring framework.

2 Background

MonALISA(4) is the main monitoring tool used to monitor the ALICE Computing Grid. It collects an extensive set of information like job efficiency, CPU cores used, disk space available, job status, job walltime etc. However, MonALISA collects information from the point of view of the jobs submitted to the Grid. Each job executing on the Grid monitors the current state of the host it is executed on and report that data to the upstream monitoring server. Although this provides insights about the health and efficiency of the whole Grid setup, it is not enough to provide information about the infrastructure of the worker nodes in the Grid and their resources.

Recently, "Site Sonar"(5) was introduced to address this problem. Site Sonar is a flexible and extensible infrastructure monitoring framework that collects infrastructure properties of the worker nodes in the ALICE computing Grid. It runs before the job payload execution and reports a wide range of information about node infrastructure like CPU family, CPU cores, memory information, operating system, system architecture etc. Currently, Site Sonar runs 36 test probes for each node and reports more than 220 parameters per node. This information has been crucial to understand the current status of the Grid, readiness of different computing sites for new JAliEn features, trends in changes in infrastructure across the Grid, etc.

Matching jobs to the most suitable node is done by JAliEn in the ALICE Grid. A job can request different amounts of resources and it is assigned to a worker node that contains resources more than the requested amount. It uses a few crucial factors like number of CPU cores, available memory, and available disk space for this job matching process. Currently, JAliEn has hardcoded functions to read each of these resource limits and to report them for the job matching process. This comes with a considerable burden of these parameters being tightly bound with the job matching process. Adding, removing, or changing of one of these parameters require upgrading central services to read new parameters, upgrading site services to report new parameters and releasing a new JAliEn deployment across the Grid that includes these code changes. Owing to this factor, Grid administrators would prefer to constrain the job matching parameters to a small set to avoid frequent changes to them. However, this means that fine-grained job matching and dynamic changes to these parameters are not possible. Given that Site Sonar has the ability to collect these parameters and change them on-demand without any deployment rollouts, it can be used as a way to address this problem.

3 Methodology

Historically, Grid workflow management and Grid site infrastructure monitoring has been treated as two separate facets as they can be managed independently to have a functional Grid. Grid workflow management was done by central Grid middleware by the Grid administrators, and the Grid site monitoring was either done by using separate software or it was done only by the site itself for internal maintenance purposes. However, it can be seen that the newly emerging user requirements need an integration and sharing of information between these two facets to run jobs on the most suitable node. For example, Grid jobs historically required a certain amount of CPU, memory and disk space to execute, which is examined by the Grid middleware and reported to the Job Matcher. With the emergence of new CPU architectures like ARM, computation techniques like parallel programming with

GPUs, users need more specific targeting of worker nodes since accessing the information like CPU model, GPU model from the Grid middleware is not straightforward.

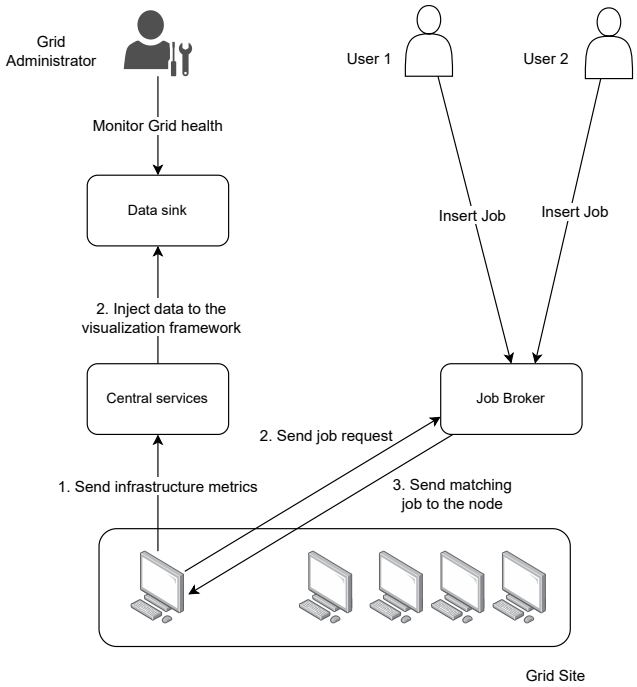

Figure 1: Proposed methodology for integration

To address this issue, we suggest treating both Grid workflow management and Grid site monitoring as a single facet. A Grid Workflow Management System (WMS)/middleware builds its global queue by submitting pilot jobs to Grid sites. The Grid site monitoring framework can be integrated to run as part of this pilot job, since it run on the bare host before the Grid middleware initiates its containerization process and is guaranteed to run in each worker node in the Grid, assuming that enough pending jobs exist in the Grid queue. Even if the pilot job is started inside a virtual environment by the Grid site, the host would not report all of its attributes, as some resources such as memory are partitioned. However, this would not be a problem as the important information is the current resource allocation to the Grid, which can be seen even if the pilot job is run inside a virtual environment

Previously, potential improvements that can be made to the Grid middleware were limited because Grid worker nodes are vastly different in their infrastructure by design of a Computing Grid and the middleware had only limited access to infrastructure of the worker nodes. This novel approach of executing site monitoring tests from the Grid middleware substantially increases the potential features that can be introduced to the Grid middleware as it introduces complete observability to the host infrastructure from within the middleware itself. For example, different OS- or GPU-specific features can be easily implemented with the introduced approach.

Fig. 1 shows an overview of the proposed approach. Once the pilot job starts its execution, it runs a set of tests on the worker node to collect its infrastructure attributes. These attributes are then fed into a central service that injects the data into a data sink. The data is indexed at the data sink and presented in the form of visualizations to the Grid administrators. These visualizations provide a complete overview of the Grid infrastructure. An interesting set of parameters from the collected attributes hand-picked by the Grid administrators are sent to the Job Broker (also known as Job Matcher) in parallel. These parameters are directly used in the job matching process, allowing more fine-tuned job matching capabilities than before. The set of hand-picked attributes can be changed on-demand providing dynamic job matching capabilities to the Grid middleware.

4 Implementation

The Grid workflow management in the ALICE Grid is done using JAliEn. JAliEn is a lightweight open source grid framework built around other open source components using the combination of a web service and distributed agent model. Similar to other Grid WMS, JAliEn builds a global queue across its Grid sites by submitting pilot jobs.

Site Sonar is a flexible and extensible infrastructure monitoring tool for Grid sites. It was deployed few years ago in ALICE Grid and provides many useful insights about the infrastructure of the worker nodes available in ALICE Grid. It has been used in multiple studies(6)(7) to identify the features that can be introduced to JAliEn based on the available infrastructure of the Grid.

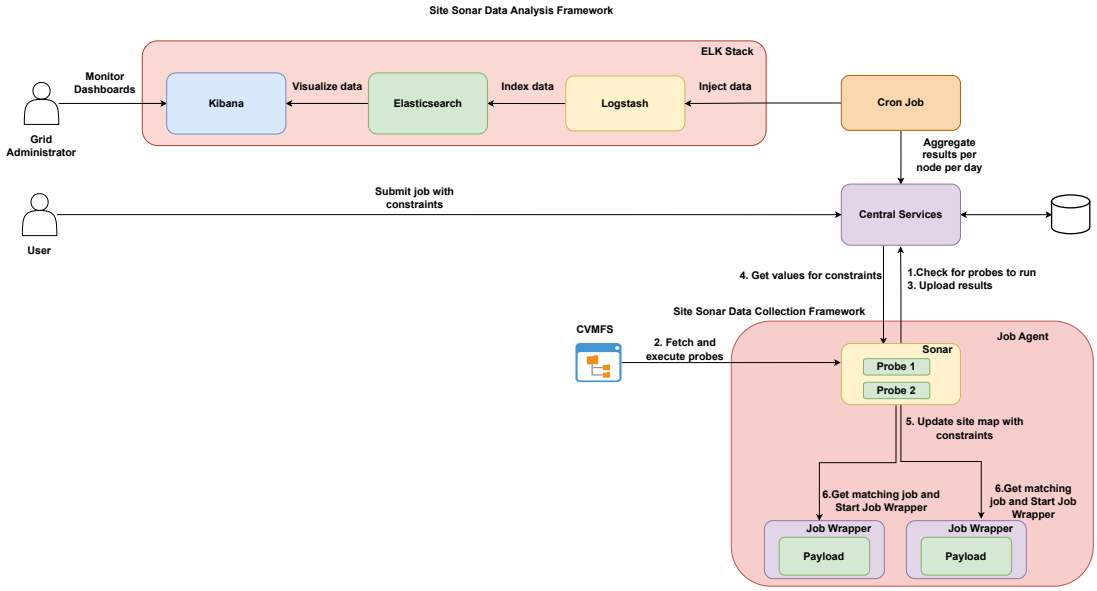

Figure 2: New job submission flow after integrating Site Sonar with JAliEn

Initially, Site Sonar was an independent tool that ran separately to collect the infrastructure data from the ALICE Grid. To achieve the outcome proposed in the methodology, we integrated Site Sonar to run as a part of JAliEn's job submission. As soon as a pilot job (known as Job Agent in JAliEn) is started on a worker node, it starts the "Sonar" component that handles the integration. Fig. 2 shows the implemented flow of the integration.

1. The Sonar contacts the central services to check which test probes should run. This allows the admins to enable/disable tests on-demand and define custom TTLs for the test result collection. The data is collected from a worker node only if the TTL of the last test result has expired.

2. Sonar fetches the enabled set of tests from CVMFS (CernVM File System)(8) and executes them on the worker node. The tests are kept in the CVMFS distributed file system as it allows us to add/edit/remove the content of the tests dynamically.

3. The executed test results are uploaded to the central services for long term persistence.

4. Central Services are contacted by the Sonar to obtain the values for a set of predefined infrastructure parameters out of the collected results. This is handled by a single endpoint in the form of a JSP (Java Server Page) servlet deployed in the central services. A servlet is used here as its code can be changed on-demand without redeploying. The servlet will return the values for the considered parameters for that node.

5. These parameters and their values are then injected to the "Site Map" which is sent to the Job Broker for obtaining the most suitable job.

6. Lastly, the pilot job contacts the Job Broker with these parameters. The job broker can now use the newly added parameters and their values in the job matching process. The important factor is that these matching parameters can now be changed dynamically without any code changes and Grid middleware deployment rollouts.

5 Outcome

As the final outcome of the work, Site Sonar which is the Grid infrastructure monitoring tool of ALICE Grid is now integrated with JAliEn which is the Grid WMS of the the ALICE Grid. This integration has introduced a brand new set of possibilities that could not be easily explored before.

As the initial phase, we introduced the CPU model as a job matching parameter. This allows targeting some jobs to specific CPU models that yield better job efficiency. Further, we introduced the Operating System, the presence of AVX support, and the hostname as additional parameters. A highly anticipated feature will be the ability to match jobs based on the GPU model which is to be introduced next.

Ultimately, the work presented in this paper has allowed the Grid administrators to provide much more fine-grained job matching that results in more efficient use of resources. It has also granted the users the privilege of easily requesting new matching parameters that are short lived or changed frequently. We can also see fewer job failures in the Grid that occur as a result of matching jobs to worker nodes with incompatible infrastructures.

6 Conclusion

In this paper we have introduced a new approach to job matching in Computing Grids by integrating Grid infrastructure monitoring frameworks with Grid middleware. The new approach provides flexible, dynamic job matching capabilities to existing Grid middleware allowing more fine-tuned job matching in Grids leading to better resource usage and fewer incompatibility issue. The proposed solution is already integrated and running in production in JAliEn at the ALICE Computing Grid, leveraging the infrastructure monitoring capabilities of Site Sonar. The solution has proven to be effective in the Grid domain and has become an important addition to the Grid middleware domain.

References

[1] Aamodt K, Quintana AA, Achenbach R, Acounis S, Adamová D, Adler C, et al. The ALICE experiment at the CERN LHC. Journal of Instrumentation. 2008;3(08):S08002.

[2] Shiers J. The worldwide LHC computing grid (worldwide LCG). Computer Physics Communications. 2007;177(1-2):219-23.

[3] Pedreira MM, Grigoras C, Yurchenko V. JAliEn: the new ALICE high-performance and high-scalability Grid framework. In: EPJ Web of Conferences. vol. 214. EDP Sciences; 2019. p. 03037.

[4] Legrand I, Newman H, Voicu R, Cirstoiu C, Grigoras C, Dobre C, et al. MonALISA: An agent based, dynamic service system to monitor, control and optimize distributed systems. Computer Physics Communications. 2009;180(12):2472-98.

[5] Wijethunga K, Storetvedt M, Grigoras C, Betev L, Litmaath M, Amarasinghe G, et al. Site Sonar-A Flexible and Extensible Infrastructure Monitoring Tool for ALICE Computing Grid. In: EPJ Web of Conferences. vol. 295. EDP Sciences; 2024. p. 04037.

[6] Storetvedt M, Betev L, Helstrup H, Hetland KF, Kileng B. The ALICE Grid Workflow for LHC Run 3. In: EPJ Web of Conferences. vol. 295. EDP Sciences; 2024. p. 04042.

[7] Ferrer MB, Grigoras C, Badia RM. Dynamic scheduling using CPU oversubscription in the ALICE Grid. In: EPJ Web of Conferences. vol. 295. EDP Sciences; 2024. p. 04020.

[8] Blomer J, Aguado-Sánchez C, Buncic P, Harutyunyan A. Distributing LHC application software and conditions databases using the CernVM file system. In: Journal of Physics: Conference Series. vol. 331; 2011. p. 042003.