ALMA MATER STUDIORUM · UNIVERSITÀ DI BOLOGNA

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

Analisi e sviluppo di middleware DDS per la gestione dei consumi in sistemi HPC

Relatore
Prof. Andrea Bartolini

Candidato
Giacomo Madella

Ottobre 2023

Abstract

Indice

Introduzione

In questo ultimo decennio nel contesto dei sistemi informatici si sono manifestate delle sempre crescente richieste di capacità e potenza computazionale mentre le tecnologie ad esse associate si sono avvicinate sempre più ai propri limiti fisici. L'efficienza energetica nei data center è diventata una delle principali preoccupazioni, non solo a causa dei costi monetari, ma anche per la sostenibilità ambientale. Il consumo di elettricità dei data center è in costante aumento, ed è urgente applicare ottimizzazioni hardware e software per ottenere le migliori prestazioni per watt.

1.0.1 HPC

1.0.2 Power Management

1.0.3 DDS

Power management

2.1 Definizione

2.2 Componenti PowerStack

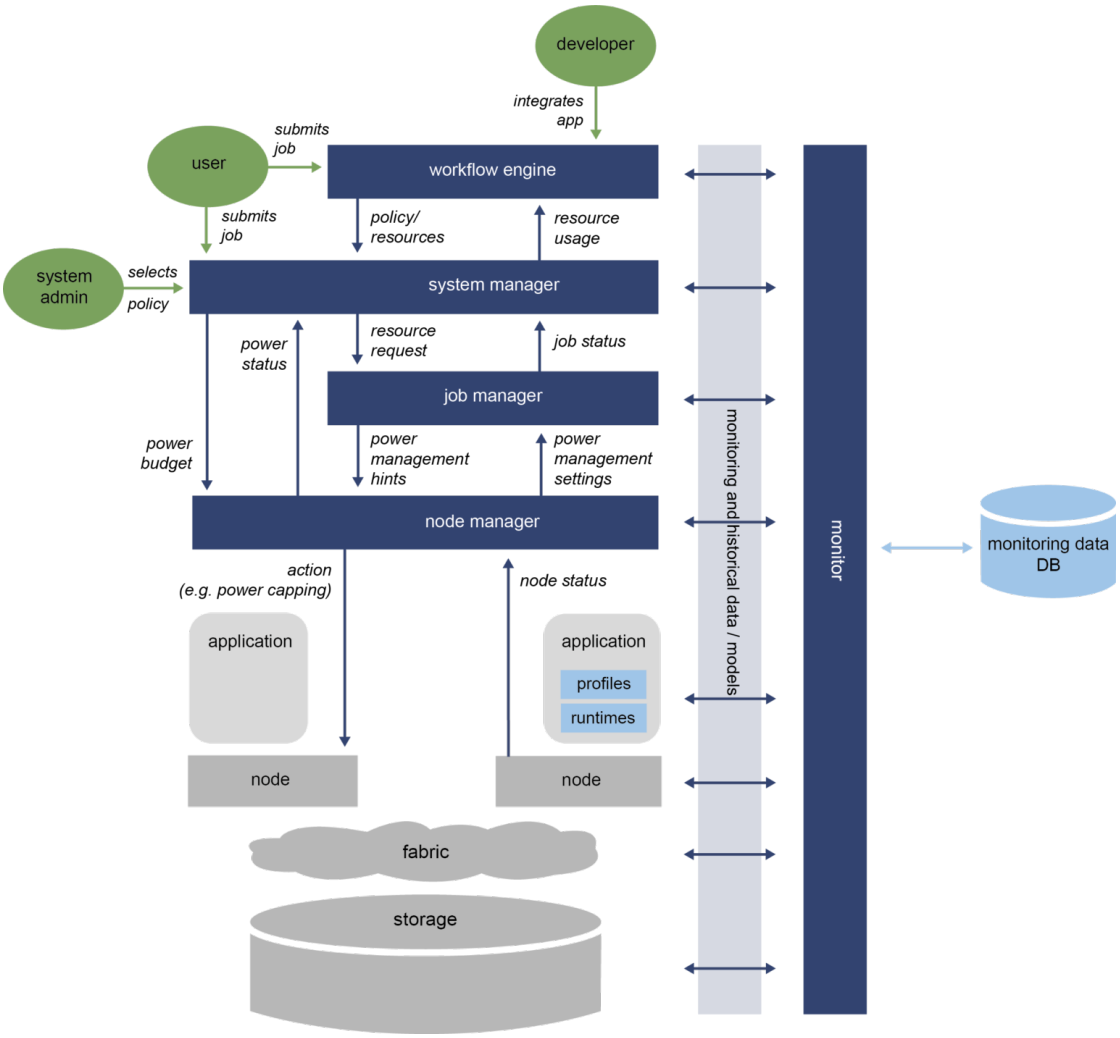
2.2.1 Workflow engine

The workflow engine analyses the dependencies and resource requirements of each workflow and decides on how to break the workflow into specific jobs that will be fed to the system manager.

Job schedulers allow high-performance computing users to efficiently share the computing resources that comprise an HPC system. Users submit batch jobs into one or more batch queues that are defined within the job scheduler. The job scheduler examines the overall set of pending work waiting to run on the computer and makes decisions about which jobs to place next onto the computational nodes within the computer. Generally speaking, the job scheduler attempts to optimize some characteristic such as overall system utilization or fast access to resources for some subset of batch jobs within the computing center's overall workload. The various queues that are defined within the job scheduler may be designated as having higher or lower priorities and may be restricted to some subset of the center's users, thus allowing the job scheduler to understand distinctions of importance of certain jobs within the overall workflow.

2.2.2 System Manager

Receives as input a set of jobs to be scheduled within the system and indicatively decides upon when to schedule each job, to which specific compute nodes to map it, and under which power budget or setting. For this, it constantly monitors and records power and energy telemetry data, and controls power budgets/settings and/or user fairness


developer
integrates app
user
submits job
workflow engine
submits job
policy/ resources
resource usage
system admin
selects policy
system manager
power status
resource request
job status
job manager
power budget
power management hints
power management settings
node manager
action (e.g. power capping)
node status
application
application
profiles
runtimes
node
node
fabric
storage
monitoring and historical data / models
monitor
monitoring data DB

2.2.3 Job Manager

By analysing the profiles, Job Manager decides the target power management knob (CPU power cap, CPU clock frequency scaling or any others) as well as performs code tuning as an option.

2.2.4 Node Manager

The node manager provides access to node-level hardware controls and monitors. Moreover, the node manager implements processor level and node level power management policies, as well as preserving the power integrity, security and safety of the node.

2.2.5 Monitor

The monitor is responsible for collecting in-band and out-of-band data for performance, resource utilization, status, power, energy, with minimal footprint, collecting, aggregating, and analysing various metrics, and pushing necessary real-time data to the other entities.

REGALE

3.1 Power Stack

3.2 Problematica

DDS & RTPS

DDS (Data Distribution Service)[1] e RTPS (Real-Time Publish-Subscribe)[2] costituiscono due soluzioni fondamentali nel campo delle comunicazioni distribuite e real-time. Queste tecnologie svolgono un ruolo importante nella la trasmissione di dati tra dispositivi e applicazioni interconnesse, rivestendo particolare importanza in scenari complessi come i sistemi embedded, in IoT e applicazioni ad alte prestazioni come l'HPC (High-Performance Computing).

4.1 DDS

Data Distribution Service è un protocollo di comunicazione incentrato sullo scambio di dati per sistemi distribuiti. Questo si basa su modello chiamato Data-Centric Publish Subscribe (DCPS) I principali attori che vengono coinvolti sono:

- Publisher: responsabile della creazione e configurazione dei DataWriter. Il DataWriter è l'entità responsabile della pubblicazione effettiva dei messaggi. Ciascuno avrà un Topic assegnato sotto il quale vengono pubblicati i messaggi;
- Subscriber: responsabile di ricevere i dati pubblicati sotto i topic ai quali si iscrive. Serve uno o più oggetti DataReader, che sono responsabili di comunicare la disponibilità di nuovi dati all'applicazione;
- Topic: collega i DataWriter con i DataReader. È univoco all'interno di un dominio DDS;
- Dominio: utilizzato per collegare tutti i publisher e subscriber appartenenti a una o più domini di appartenenza, che scambiano dati sotto diversi topic. Il DomainParticipant funge da contenitore per altre entità DCPS, e svolge anche la funzione di costruttore di entità Publisher, Subscriber e Topic fornendo anche servizi di QoS;
- Partizione: costituisce un isolamento logico di entità all'interno dell'isolamento fisico offerto dal dominio;

Inoltre DDS definisce le cosidette Qualità di Servizio (QoS policy) che servono configurare il comportamento di ognuno di questi attori.

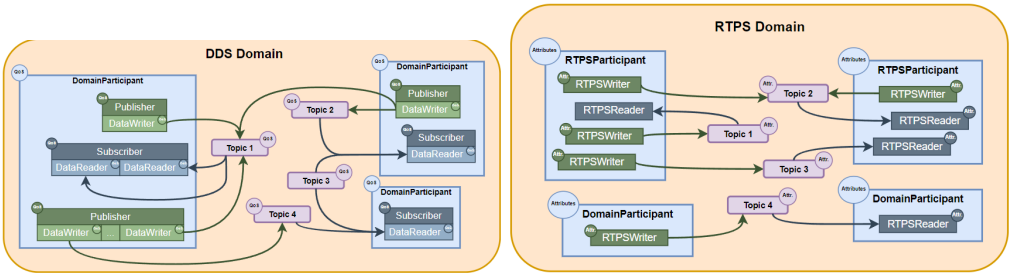
4.2 RTPS

Real-Time Publisher Subscribe protocol è un protocollo-middleware utilizzato da DDS per gestire la comunicazione su diversi protocolli di rete come UDP/TCP e Shared Memory. Il suo principale scopo è quello di inviare messaggi real-time, con un approccio best-effort e cercando di massimizzare l'efficienza. E' inoltre progettato per fornire strumenti per la comunicazione unicast e multicast. Le principali entità descritte da RTPS sono:

- RTPSWriter: endpoint capace di inviare dati;
- RTPSReader: endpoint abilitato alla ricezione dei dati;

Ereditato da DDS anche RTPS ha la concezione di Dominio di comunicazione e come questo, le communcazioni a livello di RTPS girano attorno al concetto di Topic prima definito. L'unità di comunicazione è chiamata **Change** che rappresenta appunto un cambiamento sui dati scritti sotto un certo topic. Ognuno degli attori registra questi *Change* in una struttura dati che funge da cache. In particolare la sequenza di scambio è:

1. il *change* viene aggiunto nella cache del RTPSWriter;
2. RTPSWriter manda questa *change* a tutti gli RTPSReader che conosce;
3. quando RTPSReader riceve il messaggio, aggiorna la sua cache con il nuovo *change*.



(a) Schema DDS **(b)** Schema RTPS

Figura 4.1. Confronto tra architettura DDS e RTPS

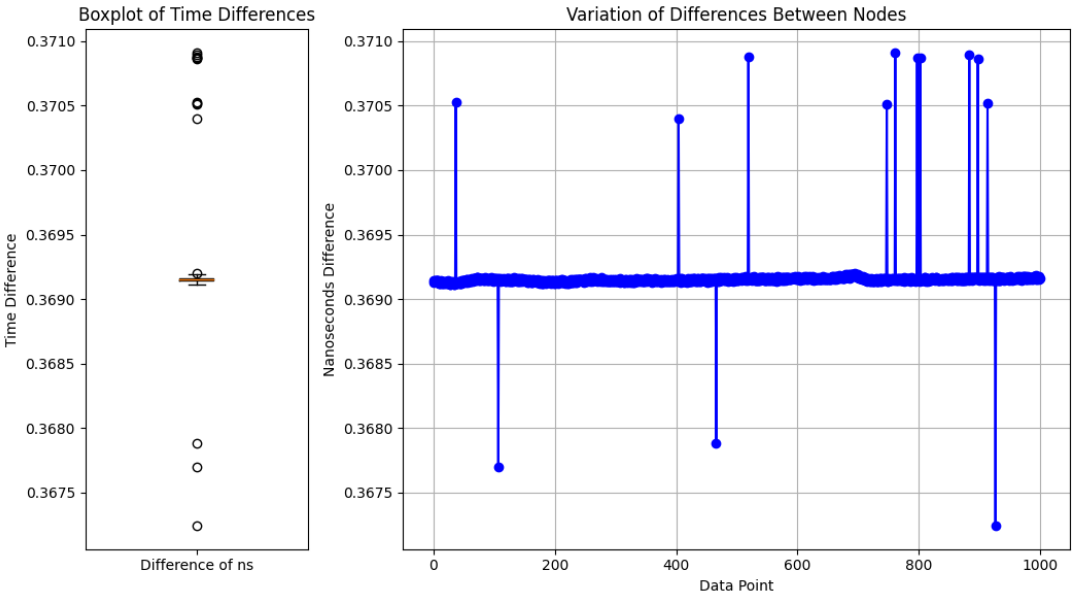
Casi di studio e valutazioni

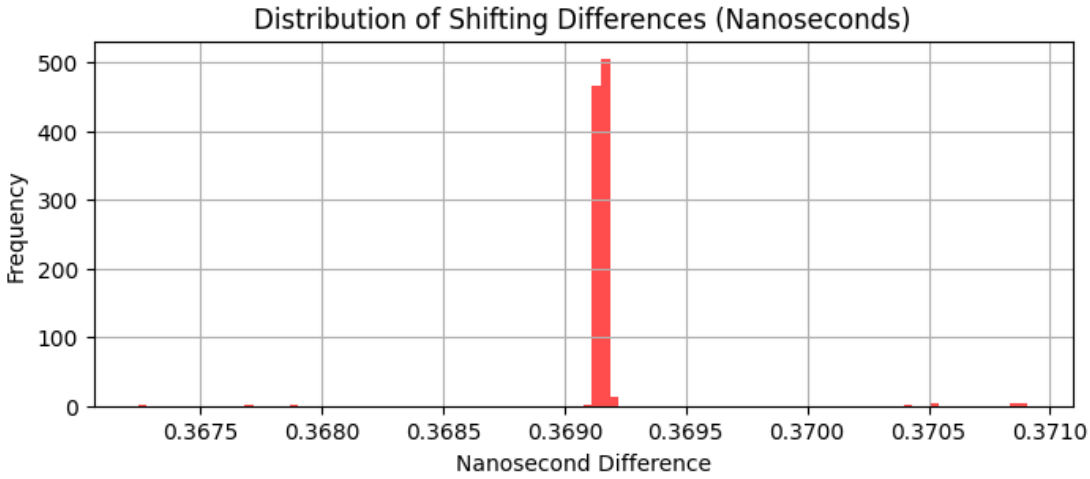
In questa sezione verranno riportati i casi studio e i test effettuati sul framework. Tutti questi casi studio sono stati eseguiti su un sistema HPC Galielo-100 Cineca con le specifiche riportate nella tabella sotto

Parameter	**Value**
Number of nodes used	3
Processor	Intel CascadeLake 8260
Number of sockets per node	2
Number of cores per socket	24
Memory size per node	384 GB
Interconnect	Mellanox Infiniband 100GbE
OS	
Compiler	gcc
MPI	Open MPI 4.1.1

5.1 Sincronizzazione orologi su nodi diversi

Per ottenere risultati attendibili sulla metrica del tempo è stato necessario sincronizzare i nodi utilizzati prima di poter far partire i test. Per farlo è stato necessario implementare delle MPI_Barrier dopo il quale una *clock_getTime(MONOTONIC)* per più volte, andando successivaemtene a ripulire ed elaborare i dati riportati.


Boxplot of Time Differences
Time Difference
Difference of ns
0.3710
0.3705
0.3700
0.3695
0.3690
0.3685
0.3680
0.3675
Variation of Differences Between Nodes
Nanoseconds Difference
Data Point
0
200
400
600
800
1000


Distribution of Shifting Differences (Nanoseconds)
Frequency
Nanosecond Difference
500
400
300
200
100
0
0.3675
0.3680
0.3685
0.3690
0.3695
0.3700
0.3705
0.3710

Conclusioni

È stato dimostrato come un framework di comunicazione DDS può essere usato all'interno di un Power-Stack per la gestione di energia in sistemi HPC vincolati dalla potenza al fine di affrontare il problema della limitazione energetica.

Bibliografia

[1] Object Management Group. *Data Distribution Service*. 2004. URL: https://www.omg.org/spec/DDS/1.0.

[2] Object Management Group. *DDS Interoperability Wire Protocol*. 2008. URL: https://www.omg.org/spec/DDSI-RTPS/2.0.