6 Related Work

Despite much research and development on load balancing in middleware for distributed systems, we believe that there is no other work that leverages the benefits of DDS on a load balancing solution. Hence, this thesis seems to be the first proposal of dynamic load balancing of data stream processing for DDS-based systems.

A common load balancing solution applied on Web Servers, cloud computing and clusters is based on centralized dispatcher [58] [59] [60] [61] [62] [63] [64] [65] where all data stream or requests go through the dispatcher, which chooses one server node to process a set of the data stream or to accept the client request. It is important stress that this approach has a centralized load balancer that is a bottleneck and is not reasonable on Pub/Sub systems.

While in Pub/Sub systems nodes with the same subscription receive the same data, in the proposed load balancing solution *Processing Nodes* – which are homogeneous and have the same subscription – do not receive the same data, instead each data is delivered to a single *Processing Node* in order to produce a data stream flow. To do so, the proposed solution manages how data are routed by DDS to *Processing Nodes,* which is simpler than routing problem in Pub/Sub systems.

However the novel proposal, [66] [67] [9] are related to this work, as they propose load balancing mechanisms for distributed Pub/Sub and DDS systems, either for the routing layer or the data processing layer.

6.1 Dynamic Load Balancing in Distributed Content-based Publish/Subscribe

The work by Cheung and Jacobsen [66] has proposed and developed a load balancing mechanism to balance the subscription load among Brokers on the Padres Pub/Sub system [68], where publishers and subscribers may freely migrate among *Brokers* aiming a load balance among the brokers. In Padres, Brokers (*edge brokers*) are organized into clusters, and each cluster has one special Broker, named *Cluster-head-broker*, as shown in Figure 31. Subscribers are connected with *Edge brokers* while publishers are connected with *Cluster-head brokers*, which in turn, also interconnect the *Edge brokers* in the same cluster.

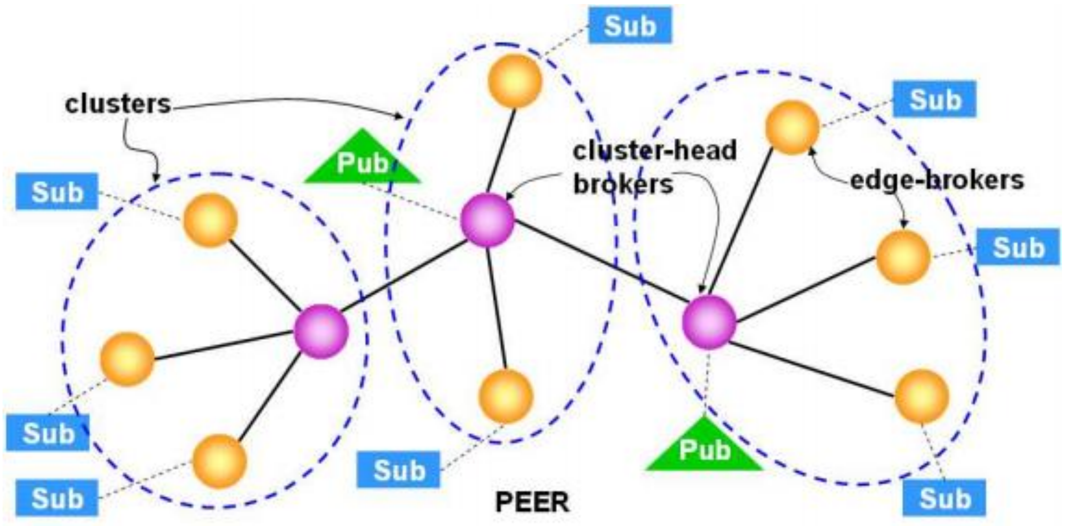

Figure 31 – Padres architecture [27]

The work supports two levels of load balancing, *local* and *global*. In local load balancing, edge-brokers from the same cluster exchange information in order to balance their loads, and in global load balancing, neighboring clusters exchange summarized load information about their edge-brokers to perform load balancing between edge-brokers from different clusters.

6.2
A DDS-compliant infrastructure for fault-tolerant and scalable data dissemination

REVENGE [67] is a DDS-compliant infrastructure for news dispatching among mobile nodes and which is capable of transparently balancing the data distribution load within a DDS -based system. It implements a P2P routing substrate, shown in Figure 32, that is fault tolerant and self-organizing. More specifically, it is able to detect crashed nodes, and to re-organize the routing paths from any source node to any mobile sink nodes.

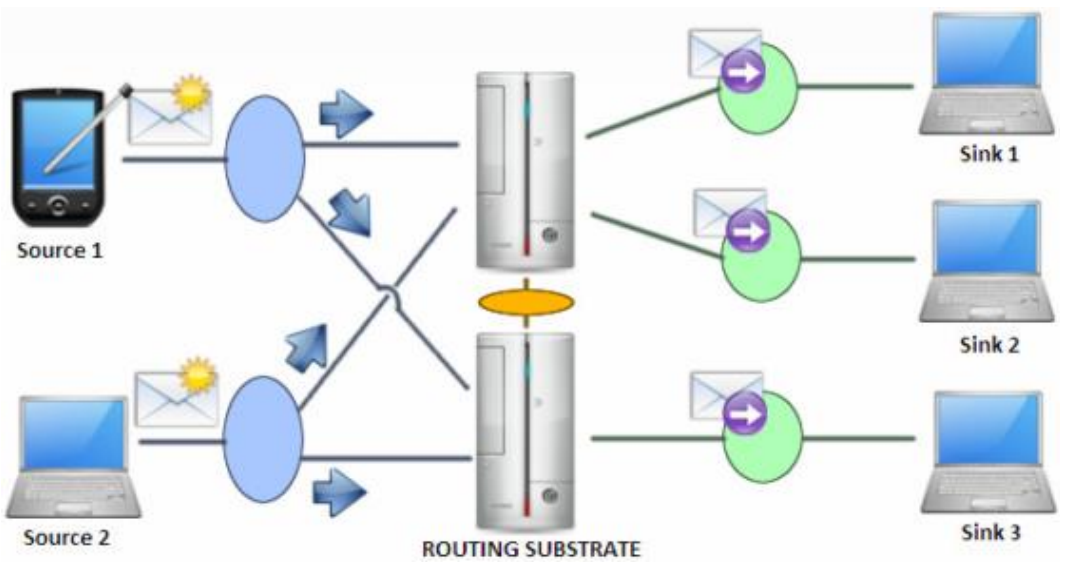

Figure 32 – REVENGE high-level architecture [67]

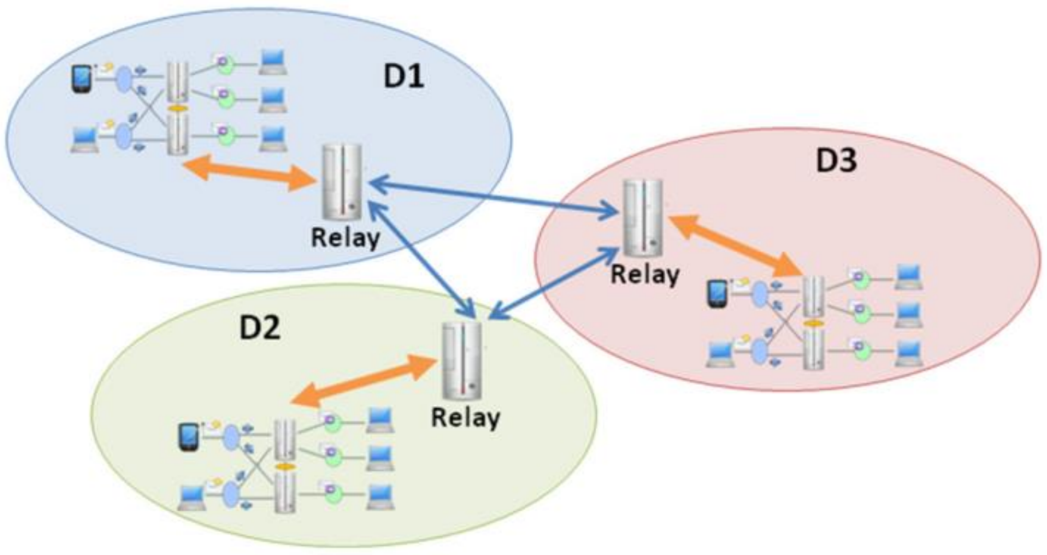

Figure 33 – Multi-domain REVENGE distributed architecture [67]

In order to optimize data distribution among different administration domains and improve system scalability, REVENGE adopts a federated DDS model of relay nodes, which are nodes that interconnect the DDS domains. Figure 33 illustrates the relay nodes that interconnect three DDS domains. The P2P routing substrate is fault tolerance

6.3 Scalability of Distributed Dynamic Load Balancing Mechanisms

Calsavara and Lima Jr [9] proposed a distributed load balancing mechanism that is based on the concept of *virtual magnetic fields*. The main idea of the approach is to attract to idle or underloaded nodes, messages (data) that were sent to overloaded nodes, so to distribute the load among the nodes. A *magnetization relationship* occurs when a node *A* attracts messages from a node *B*. This is a transitive relationship, that is, if a node *B* attracts messages from *C*, these messages are indirectly attracted to *A* since *A* attracts messages from *B*.

As a result of these relationships, the authors explain that the nodes and their magnetization relationships build an overlay network called *magnetization network*. Each magnetization relationship has a strength, or force, that depends on the current load information of each node, i.e., the lower is the load of a node, the higher is its attraction strength. *Magnetic fields* and *magnetization network* build a directional graph where the attraction strength is the cost between two vertices.

6.4 Discussion

One of the most remarkable differences between [66] and this thesis is that Cheung and Jacobsen work with heterogeneous client nodes that have to receive all data that match their subscriptions. In a appositive way, the *Data Processing Slice Load Balancing* works with homogeneous server nodes that have the same subscription but should not receive the same data. While [66] focuses on balancing the *Brokers´s* load by migrating clients (publishers or subscribers) to other *Brokers*, the proposed solution by this thesis relies on DDS' P2P architecture for balancing the data flow processing load among *Processing Nodes* rather than balancing the subscription and dissemination loads, which is transparently done by DDS and SDDL.

Similarly to Cheung and Jacobsen’s work, in REVENGE [67] load balancing is focused only in the routing of subscriptions and notifications, rather than load balancing the data processing load. Unlike REVENGE, this thesis proposes a solution to balance the load on *Processing Nodes* so as to enable the deployment of new services that require a great amount of computational resources. To achieve fault tolerance and a better load balancing on the routing layer, REVENGE works with the concept of multi-domain communication and “hot copies” of the routing substrate. This thesis neither works with multi-domains nor have capabilities to support fault tolerance on its load balancing.

Magnetic Field [9] approach is a decentralized and not coordinated load balancing where there is not a Load Balancer. Thus, the nodes communicate with each other to build the *magnetization network*. The *Data Processing Slice*, on the other hand, is a coordinated and global load balancing where *Load Balancer* is in charge of gathering load information about nodes and making and managing the load redistribution actions among nodes. Differently from the message attractions in magnetic fields, Data Processing Slice approach selects a single Processing Node that must process each data (message). To do so, the proposed solution manages the data routing done by DDS.

While Calsavara and Lima Jr's approach relies on the attraction of messages through the magnetization relationship, this thesis proposes an approach that relies on *Data Processing Slices*, which is expected to provide an efficient load balancing system that is able to directly delivery messages (data) to the appropriate nodes on DDS-based systems.

The main advantage of employ coordinated and global load balancing is that better analyzes and decisions can be done since the *Load Balancer* is able to gather information about all PNs after take any decision. Since the *Load Balancer* act also as a coordinator for the PNs during a Load Balancing Process, there is no need for complex autonomic algorithms and *leader election* at the PNs to decide which actions should be executed by each PN to realize the load balancing without conflicts. The clear disadvantage of this approach is that the *Load Balancer* may be a point of failure and bottleneck for the system´s scalability when there are hundreds of thousands of PNs since the *Load Balancer* has to analyze the load of all PNs.

Although there are many other load balancing approaches that are found both in academia and industry, none of them explores the capability of the node to receive all data published in a DDS Domain without the need of *Brokers* or a central dispatcher. In order to effectively realize a processing load balancing in a DDS Domain, a possible approach is simply managing the subscription filters so to control the data stream processing. Therefore, all data routing and its optimization is responsibility of DDS.