# Designing Next Generation Data-Centers with Advanced Communication Protocols and Systems Services

P. BALAJI, K. VAIDYANATHAN, S. NARRAVULA, H. -W. JIN AND D. K. PANDA

Ohio State University Technical Report OSU-CISRC-1/06-TR09

# Designing Next Generation Data-Centers with Advanced Communication Protocols and Systems Services*

P. Balaji K. Vaidyanathan S. Narravula H. -W. Jin D. K. Panda
Department of Computer Science and Engineering
The Ohio State University
{balaji, vaidyana, narravul, jinhy, panda}@cse.ohio-state.edu

## Abstract

Current data-centers rely on TCP/IP over Fast- and Gigabit-Ethernet for data communication even within the cluster environment for cost-effective designs, thus limiting their maximum capacity. Together with raw performance, such data-centers also lack in efficient support for intelligent services, such as requirements for caching documents, managing limited physical resources, load-balancing, controlling overload scenarios, and prioritization and QoS mechanisms, that are becoming a common requirement today. On the other hand, the System Area Network (SAN) technology is making rapid advances during the recent years. Besides high performance, these modern interconnects are providing a range of novel features and their support in hardware (e.g., RDMA, atomic operations, QoS support). In this paper, we address the capabilities of these current generation SAN technologies in addressing the limitations of existing data-centers. Specifically, we present a novel framework comprising of three layers (communication protocol support, data-center service primitives and advanced data-center services) that work together to tackle the issues associated with existing data-centers. We also present preliminary results in the various aspects of the framework, which demonstrate close to an order of magnitude performance benefits achievable by our framework as compared to existing data-centers in several cases.

## 1 Introduction

There has been an incredible growth of highly data-intensive applications such as nuclear research, medical informatics, genomics and satellite weather image analysis in the recent years. Sources such as nuclear physics research instruments, simulations, biomedical studies, network data analysis, online transactions and other instruments routinely generate multi-terabytes of data. With technology trends, the ability to store and share these datasets is also increasing, allowing scientists and institutions to create such large dataset repositories and making them available for use by others, typically through a web-based interface forming web-based data-centers. Such data-centers are not only becoming extremely common today, but are also increasing exponentially in size, currently ranging to several thousands of nodes.

Figure 1 shows the common components involved in designing such a web-based data-center. Requests from clients (over Wide Area Network (WAN)) first pass through a load balancer which attempts to spread the requests across multiple front-end proxies (Tier 0). These proxies perform basic triage on each request to determine if it can be satisfied by a static content web server or if it requires more complex dynamic content generation. The proxies also usually do some amount of caching of both static and dynamic content. Tier 1 is generally the most complex as it is responsible for all application-specific processing such as performing an online purchase or building a query to filter some data. At the back end of the processing stack (Tier 2) is the data repository/ database server with the associated storage. This is the prime repository of all the content that is delivered or manipulated.

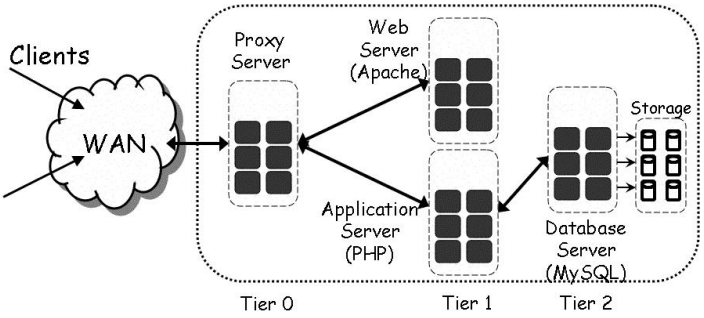


Figure 1: Web-based data-centers

With increasing interest in web-based data-centers, more and more datasets are being hosted online. Several clients request for either the raw or some kind of processed data simultaneously. However, current data-centers are becoming increasingly incapable of meeting such sky-rocketing processing demands with high-performance and in a flexible and scalable manner.

Current data-centers rely on TCP/IP for data communication even within the cluster-based data-center. These data-centers primarily use Fast or Gigabit Ethernet networks for cost-effective designs. The host-based TCP/IP protocols on these networks have high latency, low bandwidth, and high CPU utilization limiting the maximum capacity (in terms of requests they can handle per unit time) of data-centers. Together with raw performance, current data-centers also lack in efficient support for intelligent services that are becoming a quite common requirement today. For example, requirements for caching documents, managing limited physical resources, load-balancing, controlling overload scenarios, and prioritization and QoS mechanisms today are more stringent than ever before. Not only are current data-centers expected to handle these with high-performance, but also in a scalable manner to be utilized with minimal degradation on clusters ranging to thousands of nodes. However, currently there is no mechanism to achieve this. In summary, with exponentially increasing demands, the gap between what current data-centers can provide and what end-users demand is increasingly continuously; the primary rea-

*This research is supported by NSF grant #CNS-0509452 and NSF RI equipment grant #CNS-0403342.

sons being: (i) low performance due to high communication overheads and (ii) lack of efficient support for advanced features that are required today.

On the other hand, the System Area Network (SAN) technology is making rapid advances during the recent years. SAN interconnects such as InfiniBand (IBA) [3] and 10-Gigabit Ethernet (10GigE) [18, 16, 7, 15] have been introduced and are currently gaining momentum for designing high-end computing systems and data-centers. Besides high performance, these modern interconnects are providing a range of novel features and their support in hardware, e.g., Remote Direct Memory Access (RDMA), Remote Atomic Operations, Offloaded Protocol support, Quality of Service support and several others.

In this paper, we address the capabilities of these current generation SAN technologies in dealing with the limitations of existing data-centers. Specifically, we present a novel framework comprising of three layers, namely, communication protocol support, data-center service primitives and advanced data-center services. For the advanced data-center services, we further present two specific services, namely, dynamic content caching and active resource adaptation and reconfiguration. We also present preliminary results in the various aspects of the framework, showing the promise demonstrated by the proposed framework. Our results show close to an order of magnitude performance benefits achievable by our framework as compared to existing data-centers in several cases.

# 2 Proposed Framework

To satisfy the needs of the next generation data-center applications, we propose a three-stage research framework for designing data-centers as shown in Figure 2. This framework is aimed to take advantage of the novel features provided by advances in networking technologies.

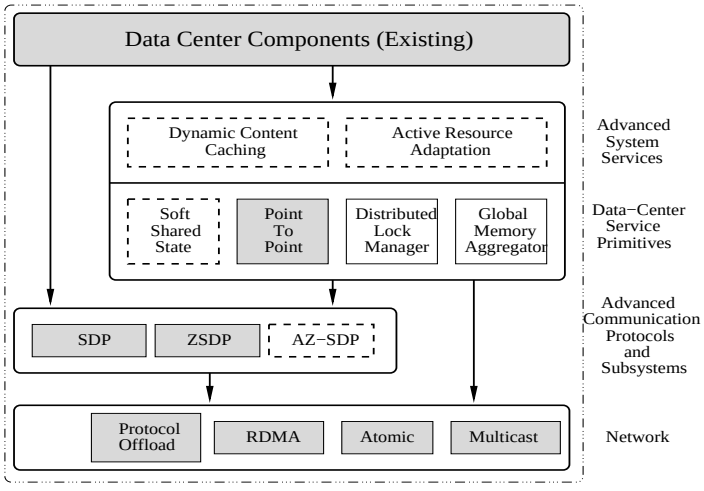


Figure 2: Proposed Framework

The framework is broken down into three layers, namely, communication protocol support, data-center service primitives and advanced data-center services as illustrated in the figure. Broadly, in the figure, all the colored boxes are the components which exist today. The white boxes are the ones which need to be designed to efficiently support next-generation data-center applications. Amongst these, for this paper, we concentrate on the boxes with the *dashed lines* by providing either complete or partial solutions. The boxes with the *solid lines* are aspects which are deferred for future work.

Existing data-center components such as Apache, PHP, MySQL, etc., are typically written using the sockets interface over the TCP/IP communication protocol. The advanced communication protocols layer aims at *transparently* improving the communication performance of such applications by taking advantage of the mechanisms and features provided by modern networks such as IBA and 10GigE. The goals of these advanced protocols are to maintain the sockets semantics so that existing data-center components do not need to be modified. More details about this layer are presented in Section 3.

The data-center service primitives and advanced data-center services layers aim at supporting intelligent services for current data-centers. Specifically, the data-center primitives take advantage of the advanced communication protocols as well as the mechanisms and features of modern networks to provide higher-level utilities that can be utilized by applications as well as the advanced data-center services. For the most efficient design of the upper-level data-center services, several primitives such as soft shared state, enhanced point-to-point communication, distributed lock manager, and global memory aggregator are necessary. In this paper, however, we limit our study to only the soft shared state primitive as described in Section 4.

The advanced data-center services are intelligent services that are critical for the efficient functioning of data-centers. For example, requirements for caching documents, managing limited physical resources and prioritization and QoS mechanisms are handled by these. In Sections 5 and 6, we discuss two of these services: (i) dynamic content caching and (ii) active resource adaptation and reconfiguration. Specifically, the dynamic content caching service deals with efficient and load-resilient caching techniques for dynamically generated content, while the active resource adaptation (used interchangeably with resource reconfiguration) service deals with on-the-fly and scalable management and adaptation for various system resources.

# 3 Communication Protocol Support

Several traditional applications used in the data-center environment such as Apache, PHP, MySQL, etc., have been developed over a span of several years using the sockets interface over the TCP/IP protocol suite. However, due to the high host processing overhead and copies associated with TCP/IP, this approach cannot be expected to give the best performance. Due to the inability of traditional sockets over TCP/IP in coping with the exponentially increasing network speeds, IBA and other network technologies recently proposed a new standard known as the Sockets Direct Protocol (SDP) [1]. SDP is a pseudo sockets-like implementation designed to meet two primary goals: (i) to directly and transparently allow existing sockets applications to be deployed on to clusters connected with modern networks such as IBA and (ii) allow such deployment while retaining most of the raw performance provided by the networks.

In our previous work [6], we revealed the benefits of SDP on sockets-based applications in a data-center environment. We showed that SDP can not only provide a better communication performance, but also significantly reduce the amount of host CPU

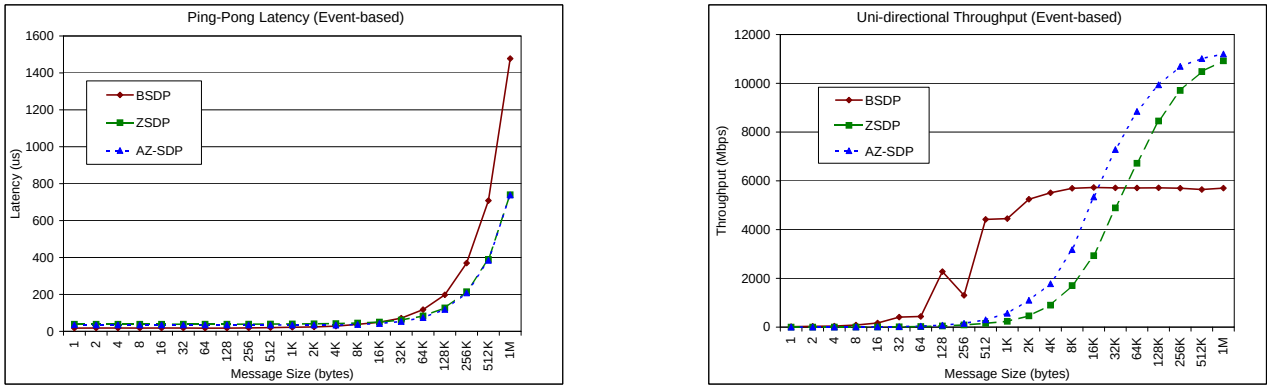


Figure 3: AZ-SDP Performance: (a) Latency and (b) Unidirectional Throughput

cycles used for protocol processing. However, the basic implementation of SDP itself has several disadvantages. For example, overheads such as memory copies that are associated with the SDP implementation can result in severe limitations in its capabilities for achieving high performance. Thus, a *zero-copy* implementation of SDP which tackles these overheads allowing various sockets applications, including those in the data-center environment, to take advantage of the benefits of high-speed networks is highly desirable.

The SDP standard supports two kinds of sockets semantics, viz., Synchronous sockets and Asynchronous sockets. In the synchronous sockets interface, the application has to *block* for every data transfer operation whereas, in the asynchronous sockets interface, the application can *initiate* a data transfer and check whether the transfer is complete at a later time providing a better overlap of the communication with the other computation going on in the application. Due to the inherent benefits of asynchronous sockets, the SDP standard also allows several intelligent approaches such as *source-avail and sink-avail based zero-copy* for these sockets. However, most of these approaches that work well for the asynchronous sockets interface are not as beneficial for the synchronous sockets interface [4]. Further, due to its portability, ease of use and support on a wider set of platforms, the synchronous sockets interface is the one used by most sockets applications today. Thus, a mechanism in which the approaches proposed for asynchronous sockets can be used for synchronous sockets would be very beneficial for such applications.

In this paper, we propose one such mechanism, termed as *AZ-SDP (Asynchronous Zero-Copy SDP)* which allows the approaches proposed for asynchronous sockets to be used for synchronous sockets while maintaining the synchronous sockets semantics. In order to transparently provide asynchronous capabilities for synchronous sockets, two goals need to be met: (i) the interface should not change; the application can still use the same interface as earlier, i.e., the synchronous sockets interface and (ii) the application can assume the synchronous sockets semantics, i.e., once the control returns from the communication call, it can read or write from/to the communication buffer. In our approach, the key idea in meeting these design goals is to memory-protect the user buffer (thus disallow the application from accessing it) and to carry out communication asynchronously from this buffer, while *tricking* the application into believing that we are carrying out data communication in a synchronous manner. More details about the design of the AZ-SDP scheme can be found in [4].

We evaluate the AZ-SDP implementation and compare it with the other two implementations of SDP, i.e., Buffered SDP (BSDP) and Zero-copy SDP (ZSDP). BSDP and ZSDP are the copy-based SDP and synchronous zero-copy SDP implementations, respectively. We present ping-pong latency and uni-directional throughput micro-benchmarks results.

Figure 3(a) shows the point-to-point latency achieved by the three stacks. As shown in the figure, both zero-copy schemes (ZSDP and AZ-SDP) achieve a superior ping-pong latency as compared to BSDP. However, there is no significant difference in the performance of ZSDP and AZ-SDP. This is due to the way the ping-pong latency test is designed. In this test, only one message is sent at a time and the node has to wait for a reply from its peer before it can send the next message, i.e., the test itself is completely synchronous and cannot utilize the capability of AZ-SDP with respect to allowing multiple outstanding requests on the network at any given time, resulting in no performance difference between the two schemes.

Figure 3(b) shows the uni-directional throughput achieved by the three stacks. As shown in the figure, for small messages BSDP performs the best. The reason for this is two fold: (i) Both ZSDP and AZ-SDP rely on control message exchange for every message to be transferred. This causes an additional overhead for each data transfer which is significant for small messages and (ii) Our BSDP implementation uses an optimization technique known as reverse packetization to improve the throughput for small messages. For medium and large messages, on the other hand, AZ-SDP and ZSDP outperform BSDP because of the zero-copy communication. Also, for medium messages, AZ-SDP performs the best with about 35% improvement compared to ZSDP. More micro-benchmark results such as communication and computation overlap and impact of page faults on the performance of AZ-SDP can be found in [4].

# 4 Data-Center Service Primitives

As mentioned in the proposed framework (Figure 2), multi-tier data-centers need efficient support for many higher level data-center primitives. These data-center service primitives are used to build more advanced services such as dynamic content caching, active resource adaptation and reconfiguration, etc. Current data-centers can benefit from several higher level system primitives such as soft shared state, distributed lock manager and global

memory aggregator.

The soft shared state primitive deals with efficient sharing of information across the cluster by creating a logical shared memory region using IBA's RDMA operations. The global memory aggregator integrates system wide memory and provides applications with free memory from other nodes to utilize. The distributed lock manager provides for efficient locking capabilities allowing for access arbitration and managing sharing of data and resources across the data-center. While all these aspects are important for the efficient functionality of the advanced data-center services, in this paper, we limit our scope to the discussion about just the soft shared state primitive.

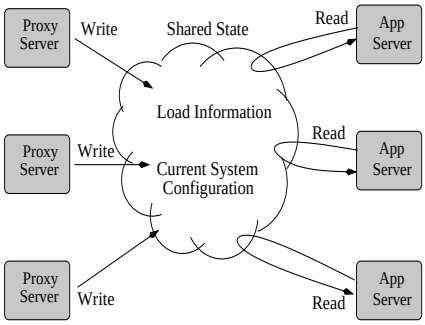


Figure 4: A Soft Shared State Scenario

The soft shared state primitive relies on two basic ideas: (i) Avoiding to maintain strict consistency to minimize overhead (applications need to explicitly maintain this) and (ii) Asynchronous reads and writes to the shared state without involving other CPUs. Figure 4 shows a sample soft shared state scenario with proxy servers writing certain information into the soft shared state and application servers reading this information from the soft state shared. All the operations shown are asynchronous operations.

Typically in a multi-tier data-center, the performance of the servers in the proxy tier and applications server tier is dependent largely on the processor load. In our earlier work [24] we had shown that one sided operations like Remote Direct Memory Access (RDMA), perform better than two sided operations under high load conditions. Hence, protocols based on RDMA can help in achieving our goals of a load resilient soft shared state.

Leveraging the benefits of one sided RDMA operations of InfiniBand, we have designed efficient mechanisms to share information [34]. In particular, our RDMA based *Notice Board* like mechanism for sharing of information has been very effective in providing an efficient load resilient soft shared state primitive. In this approach, we mark a region of memory as the primary means of sharing of information between the different data-center processes. There are two primary operations used on this memory region: (a) *put* operation to update state information and (b) *get* operation to access state information. These operations use RDMA when accessing remote memory and direct memory accesses when accessing local memory.

In our experiments, we have observed that the benefits of RDMA in simple get or put operations is better than the two sided sockets (over IPoIB) based operations. Figure 5 shows RDMA read and corresponding sockets based get operation with increasing load on the remote server. We clearly observe that both the latency and the throughput of the sockets based get operations is significantly affected by remote side load where as RDMA can sustain its performance.

In our work elaborated in the following sections, we utilize the soft shared state primitive to enable smooth and efficient cooperation of different nodes in the data-center.

# 5 Dynamic Content Caching

Trends in current generation data-centers show that computation and communication overheads impact the performance and scalability of data-centers significantly. Caching dynamic content, typically known as *Active Caching* [11] at various tiers of a multi-tier data-center is a well known method to reduce the computation and communication overheads within the data-center. However, it has its own challenges; primarily due to issues such as cache consistency and cache coherence. In the state-of-art data-center environment, these issues are handled based on the type of data being cached. For dynamic data, for which relaxed consistency or coherency is permissible, researchers have proposed several methods like TTL [17], Adaptive TTL [13], and Invalidation [21] in the literature. However, for data like stock quotes or airline reservation, where old quotes or old airline availability values are not acceptable, strong cache consistency and coherency is essential.

Providing strong consistency and coherency is a necessity for *Active Caching* in many web applications, such as on-line banking and transaction processing. In the current data-center environment, two popular approaches are used. The first approach is pre-expiring all entities (forcing data to be re-fetched from the origin server on every request). This scheme is similar to a no-cache scheme. The second approach, known as *Client-Polling* [24], requires the front-end nodes to inquire from the back-end server if its cache entry is valid on every cache hit. Both approaches are very costly, increasing the client response time and the processing overhead at the back-end servers. The costs are mainly associated with the high CPU overhead in the traditional network protocols due to memory copy, context switches, and interrupts [29, 14, 6]. Further, the involvement of both sides for communication (two-sided communication) results in performance of these approaches heavily relying on the CPU load on both communication sides. For example, a busy back-end server can slow down the communication required to maintain strong cache coherence significantly.

Recent trends have seen rapid growth of content composed of multiple dynamic objects. Documents of this nature are typically generated by processing one or more data objects stored in the back-end database, i.e., these documents are dependent on several persistent data objects. These persistent data objects can also be a part of multiple dynamic documents. So in effect these documents and data objects have several many-to-many mappings between them. Thus, any change to one individual object can potentially affect the validity of multiple cached requests.

Simple architectures are sufficient to provide strong cache coherency which only deals with a file level granularity for coherency, i.e., each update affects an object which can be a part of one or more cached requests. However, most data-centers allow and support more complex web documents comprising of multiple dynamic objects. These additional issues necessitate more intricate protocols to enable dynamic content caching and make the design of strongly coherent caches extremely challenging. Further, since an updated object can potentially be a part of multi-

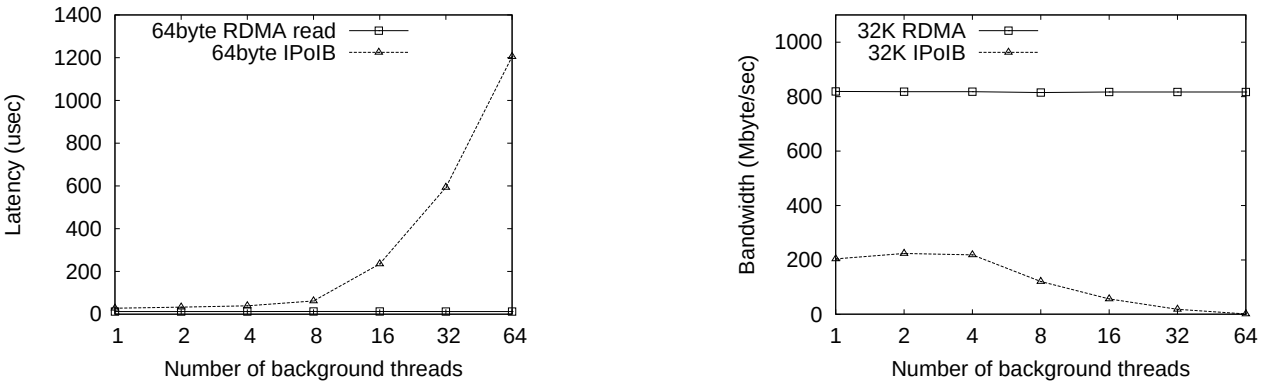


Figure 5: Performance of IPoIB and RDMA Read with background threads: (a) Latency and (b) Bandwidth

ple documents across several servers, superior server coordination protocols take a central role in these designs.

In our work [23], we present a complete architecture to support strong cache coherency for dynamic content caches. Our architecture, which is based primarily on the soft shared state primitive using one sided RDMA operations, is designed to handle caching of responses composed of multiple dynamic dependencies. We propose a complete architecture to handle two issues: (i) caching documents with multiple dependencies and (ii) being resilient to load on servers. In our work, we have explored mechanisms for maintaining necessary information on the application servers to achieve the above objectives.

We have designed two schemes to handle different data-center scenarios: (i) *Invalidate All* and (ii) *Dependency Lists*. Both the schemes use the protocols shown in Figure 6 for validating and updating of cache/caching information, as the basic mechanism for providing the caching support. Figure 6(a) shows the protocol the proxy caches use to verify the validity of a cache entity. Figure 6(b) elaborates the protocol used by the application servers to disseminate invalidation information among themselves upon receiving updates to the cached objects.

Figure 7(a) shows the performance of a data-center with our designs for dynamic content traces with increasing update rates. Our experimental results show more than 20 times improvement for the overall data-center throughput using our caching techniques (in particular *Dependency Lists*). Figure 7(b) shows that our designs can sustain high performance for overall data-center requests while maintaining strong coherency with multiple object dependencies even under heavy load. In addition, we also study the effects of varying dependencies on these cached responses. Detailed design descriptions and additional performance numbers can be found in [23].

# 6 Active Resource Adaptation

A multi-tier data-center is truly a collection of a vast number of system resources from different nodes connected over a high-speed network such as IBA or 10GigE. However, as described in Section 1, each data-center is logically broken down into several tiers or sub-clusters which handle different aspects of the data-center functionality. Further, several ISPs and other web service providers host multiple unrelated web-sites on their data-centers. The increase in such services and partitions results in a growing fragmentation of the resources available and ultimately in the degradation of the performance provided by the data-center.

While the large number of resources present in the data-center provide a great potential with respect to the performance achievable, harnessing their truly aggregate benefits requires them to function together without being fragmented by the various data-center imposed partitions. However, doing this in an unorganized and uncoordinated manner might result in further degradation in the performance. Thus, it is desirable that we have a technique to actively coordinate the various resources of the data-center so as to make the partitions *fuzzy*, i.e., while the resources are broken down into different partitions, they are not completely bound to their partition but instead are capable of migrating to other partitions on demand. Several researchers have focused on the design of adaptive systems that can manage clusters and/or react to changing workloads in the context of web servers [22, 19, 28, 10, 26, 12, 20, 30]. In order to achieve high performance, though, such coordination needs to be done with low overhead and high performance. Further, with the exponentially increasing sizes of data-centers (ranging to several thousands of nodes today), the solution provided needs to be highly scalable with the increasing data-center sizes.

In this section, we describe our approach for actively coordinating the usage of the various CPU resources in a data-center to achieve two broad goals: (i) to better utilize the limited resources in the data-center environment to improve the performance and capacity of current data-centers (in terms of number of requests they can handle per unit time) and (ii) to avoid unnecessary wastage of resources in order to provide resource guarantees to end users. In our previous work [8] we have shown the strong potential of using the advanced features of high-speed networks in designing such techniques. In this work, we extend the knowledge gained from our previous study in designing and implementing schemes to improve the utilization of resources and provide a better performance and at the same time allow for resource guarantees for end users.

**Over-provisioning of Resources:** Over-provisioning of resources in the data-center for each service provided is a widely used approach. In this approach, resources are alloted to each service depending on the worst case estimates of the load expected and the resources available in the data-center. For example, if a data-center hosts two web-sites, each web-site is provided with a


Figure 6: Design for (a) RDMA based Strong Cache Coherence Validation and (b) Coherent Invalidations

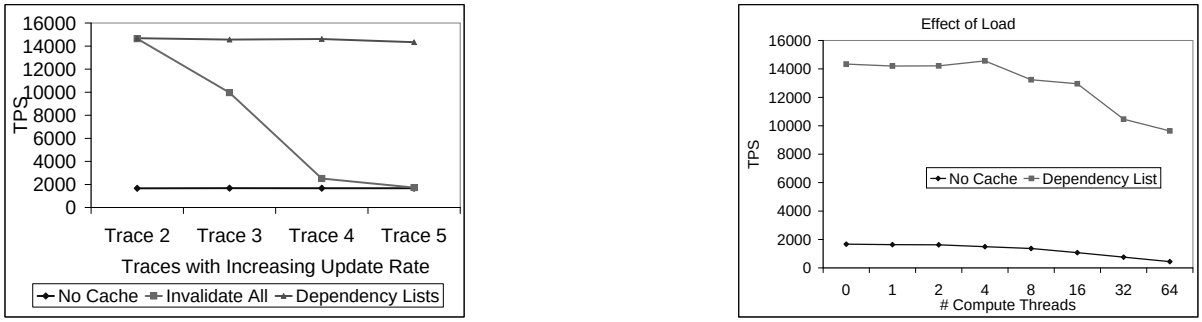


Figure 7: Performance of Data-Center with (a) Increasing Update Rate and (b) Increasing Load

fixed subset of the resources in the data-center based on the traffic expected for that web-site. It is easy to see that this approach would suffer from severe under utilization of resources especially when the traffic is bursty and directed to a single web-site.

Active resource adaptation and reconfiguration alleviates this problem of wastage of resources by dynamically mapping applications to resources available inside the data-center. It enables the data-center resources to efficiently adapt their functionality based on system load and traffic pattern. In our work [5], we focus on the design of coarse-grained constraint-based active resource adaptation and reconfiguration techniques using the advanced features offered by high-performance networks. Tasks related to system load monitoring, maintaining global state information, etc., are handled using the data-center primitives like soft shared state as mentioned in Section 4.

Figure 8 shows the RDMA based protocol used by active reconfiguration. As shown in the figure, the entire cluster management and active reconfiguration is performed by the lightly loaded load-balancer nodes without disturbing the server nodes using the RDMA and remote atomic operations provided by InfiniBand. Some of the other major design challenges and issues involved in dynamic adaptability and reconfigurability of the system are: (i) providing a system-wide shared state, (ii) concurrency control to avoid live-locks and starvation, (iii) avoiding server thrashing through history aware reconfiguration and (iv) tuning the reconfigurability module sensitivity. Further details about the other design issues can be found in [5].

While capable of achieving high performance, basic active resource reconfiguration does not have any concept of service differentiation *per se*. Thus, it cannot be *directly* used in a data-center environment having different service requirements for different websites in terms of hard and soft QoS guarantees. Hard QoS guarantees require that the resources guaranteed be available to the website at all times. Soft QoS guarantees rely on the average load on the website and client workload pattern studies. To achieve this capability, we extend the basic active reconfiguration scheme to allow service differentiation in the shared data-center environment. In particular, we address the issues associated with the basic dynamic reconfigurability scheme and propose two extensions to it, namely (i) *dynamic reconfiguration with prioritization (reconf-P)* and (ii) *dynamic reconfiguration with prioritization and QoS (reconf-PQ)*.

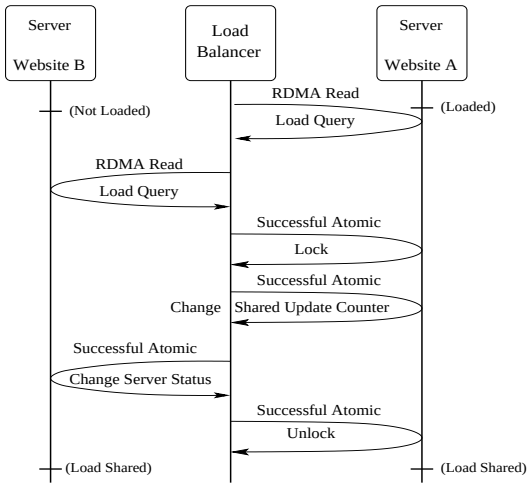


Figure 8: RDMA based Protocol for Dynamic Reconfigurability

Figure 9 shows the capabilities of the active reconfiguration schemes in two aspects, namely improved performance by better utilization of resources and capability to provide effective resource guarantees and prioritization.

**Performance of Active Reconfiguration:** Figure 9a shows the performance achieved by reconfigurability, rigid-small and

rigid-large scheme. Rigid-small considers a data-center with eight nodes and allots four nodes to each website (both websites are of equal priority). Rigid-large considers a data-center with fourteen nodes and allots seven nodes to each website. We see that for small burst lengths the dynamic reconfigurability scheme performs comparably with the Rigid-small scheme. As the burst length increases, its performance increases and converges with that of the Rigid-large scheme, i.e., for large burst lengths, a data-center having eight nodes can utilize active reconfiguration to achieve a similar performance as a data-center having fourteen nodes.

**QoS and Prioritization with Active Reconfiguration:** In order to evaluate different aspects of the three schemes (reconf, reconf-P and reconf-PQ), we create three test case scenarios. In the first case, a load of high priority requests arrives when a load of low priority requests already exists. In the second case, a load of low priority requests arrives when a load of high priority requests already exists. In the third case, both the high priority requests and low priority requests arrive simultaneously. Figure 9b compares the QoS meeting capabilities of each of the schemes for the three cases for a real-world WorldCup trace [2]. We see that the basic reconfigurability and the prioritization schemes perform well in some cases for the high priority requests and in some other cases for the low priority requests. However, these schemes lack the consistency in providing the guaranteed QoS requirements to both the websites. The prioritization with QoS scheme on the other hand meets the guaranteed QoS requirements in all cases for both the websites. Detailed analysis for the percentage of times the schemes are able to meet the soft QoS guarantees and other results are available in [5].

# 7 Discussion and Work-in-Progress

Our proposed framework mentioned in Section 2 builds multiple layers of efficient designs. Apart from the services mentioned in this paper, these different layers can also be utilized to design other data-center system applications and services as needed. More importantly, however, our designs have already been integrated into current data-center applications such Apache, PHP and MySQL and can be easily integrated to other applications as well. Also, though this work has been done in the context of InfiniBand and 10GigE, our designs rely on quite common features provided by most RDMA-enabled networks and can be easily extended to work with several other networks such as Myrinet [9], Quadrics [27], etc.

In the current context, we intend to focus on several aspects as described in this section. In addition to basic communication infrastructure and primitives such as soft shared state, multi-tier data-centers also need efficient support for many other higher level data-center primitives such as the distributed lock manager. Current approaches solve this problem by exchanging explicit two sided messages with the help of server threads on different nodes which incurs huge latencies, especially on loaded servers. Remote atomic operation provided by modern interconnects open up many interesting opportunities to implement a highly efficient distributed lock manager with minimal overhead. We are currently looking at several design alternatives in developing such a distributed lock manager primitive which can be utilized by data-center services such as resource adaptation and caching.

Moreover, simple caching methods are not very effective for multi-tier data-centers. Servers can achieve higher gains by sharing a common distributed cache (intra-tier, inter-tier) and maintaining meta-data information about the cached content on the cooperating nodes. As a part of our current study [25], we have designed and evaluated a remote memory based multi-tier distributed shared cache and studied the associated trade offs. We plan to extend the knowledge gained in this study to integrate and evaluate active and cooperative caching mechanisms proposed in this study.

Furthermore, the basic coarse-grained resource adaptation schemes can be enhanced to provide fine-grained resource dynamism in multi-tier data-centers. As part of the current study [33, 32], we have developed a load monitoring scheme that uses the RDMA operations for capturing the load information. Preliminary evaluations show that our scheme can report extremely accurate and fine-grained load information as compared to existing solutions. Also, we plan to extend the knowledge gained in our previous study [31] in utilizing the remote memory on a file system cache miss to avoid cache corruption in designing a full-fledged active reconfiguration for file management.

To achieve portability, several traditional applications used in the data-center environment are built over the sockets and interface and do not utilize advanced communication features (such as one-sided communication) provided by the networks. However, minor modifications to these applications can yield significant performance improvement. We plan to investigate, analyze these issues and study the associated benefits.

Several of the challenges and solutions described in the previous few sections are not completely independent. For example, the active resource adaptation schemes mentioned in Section 6 focus on reallocating resources and adapting the data-center environment to the varying load. However, blindly reallocating resources might have negative impacts on the caching schemes proposed in Section 5 due to cache corruption that can potentially occur. Thus, each of these proposed designs cannot be evaluated in a stand-alone fashion, but needs to be seen in an integrated environment. We plan to do carry out such integrated evaluation.

# 8 Concluding Remarks

In this paper, we presented a novel framework for addressing the two primary drawbacks of current data-centers: (i) low performance due to high communication overheads and (ii) lack of efficient support for advanced features such as caching dynamic data, managing limited physical resources, load-balancing, controlling overload scenarios, and prioritization and QoS mechanisms. Specifically, we presented a three-layer framework comprising of communication protocol support, data-center primitives and advanced data-center services and preliminary results in each of these components. Our experimental results demonstrate that this framework is quite promising in tackling the issues with current and next-generation data-centers and can provide close to an order-of-magnitude performance benefits as compared to existing solutions.

# References


[1] SDP Specification. http://www.rdmaconsortium.org/home.

[2] The Internet Traffic Archive. http://ita.ee.lbl.gov/html/traces.html.

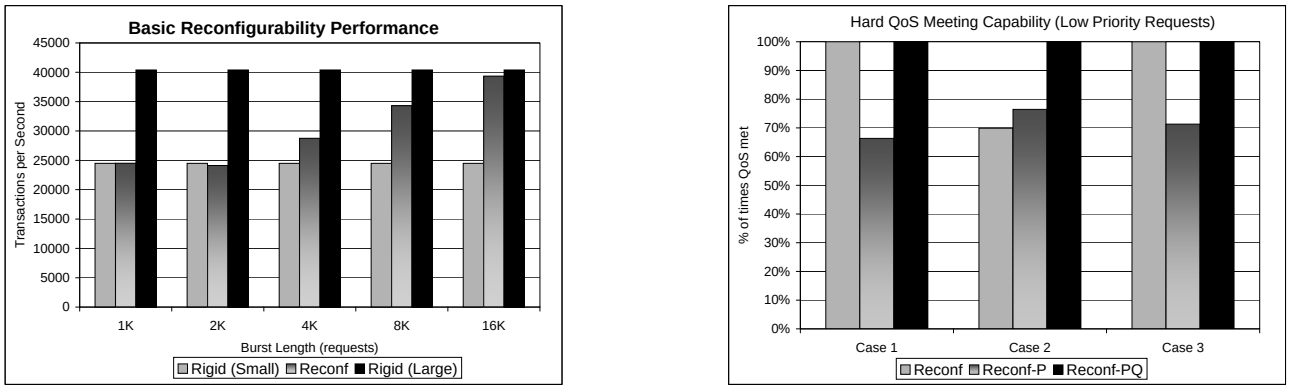


Figure 9: (a) Basic Reconfigurability Performance (b) QoS Meeting capability of Low Priority requests for a World Cup Trace


[3] Infiniband Trade Association. http://www.infinibandta.org.

[4] P. Balaji, S. Bhagvat, H. W. Jin, and D. K. Panda. Asynchronous Zero-Copy Communication for Synchronous Sockets Direct Protocol (SDP) over InfiniBand. In *CAC 2006; In Conjunction with IPDPS 2006*, 2006.

[5] P. Balaji, S. Narravula, K. Vaidyanathan, H. W. Jin, and D. K. Panda. On the Provision of Prioritization and Soft QoS in Dynamically Reconfigurable Shared Data-Centers over InfiniBand. In *Proceedings of IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS 2005)*, 2005.

[6] P. Balaji, S. Narravula, K. Vaidyanathan, S. Krishnamoorthy, J. Wu, and D. K. Panda. Sockets Direct Protocol over InfiniBand in Clusters: Is it Beneficial? In *ISPASS '04*.

[7] P. Balaji, H. V. Shah, and D. K. Panda. Sockets vs RDMA Interface over 10-Gigabit Networks: An In-depth analysis of the Memory Traffic Bottleneck. In *Workshop on Remote Direct Memory Access (RDMA): Applications, Implementations, and Technologies (RAIT 2004)*, 2004.

[8] P. Balaji, K. Vaidyanathan, S. Narravula, K. Savitha, H. W. Jin, and D. K. Panda. Exploiting Remote Memory Operations to Design Efficient Reconfiguration for Shared Data-Centers. In *RAIT '04*.

[9] N. J. Boden, D. Cohen, R. E. Felderman, A. E. Kulawik, C. L. Seitz, J. N. Seizovic, and W. K. Su. Myrinet: A Gigabit-per-Second Local Area Network. *IEEE Micro '95*.

[10] C. Lu and T. Abdelzaher and J. Stankovic and S. Son. A Feedback Control Approach for Guaranteeing Relative Delays in Web Servers. In *the Real-Time Technology and Applications Symposium*, 2001.

[11] P. Cao, J. Zhang, and K. Beach. Active cache: Caching dynamic contents on the Web. In *Middleware Conference*, 1998.

[12] I. Chung and J. K. Hollingsworth. Automated Cluster-Based Web Service Performance Tuning. In *HPDC '04*, June 2004.

[13] M. Colajanni and P. S. Yu. Adaptive ttl schemes for load balancing of distributed web servers. *SIGMETRICS Perform. Eval. Rev.*, 25(2):36–42, 1997.

[14] E. V. Carrera, S. Rao, L. Iftode, and R. Bianchini. User-Level Communication in Cluster-Based Servers. In *the 8th IEEE International Symposium on High-Performance Computer Architecture (HPCA 8)*, Feb. 2002.

[15] W. Feng, P. Balaji, C. Baron, L. N. Bhuyan, and D. K. Panda. Performance Characterization of a 10-Gigabit Ethernet TOE. In *Proceedings of the IEEE International Symposium on High-Performance Interconnects (HotI)*, Palo Alto, CA, Aug 17-19 2005.

[16] W. Feng, J. Hurwitz, H. Newman, S. Ravot, L. Cottrell, O. Martin, F. Coccetti, C. Jin, D. Wei, and S. Low. Optimizing 10-Gigabit Ethernet for Networks of Workstations, Clusters and Grids: A Case Study. In *SC '03*.

[17] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP 1.1. RFC 2616. June, 1999.

[18] J. Hurwitz and W. Feng. End-to-End Performance of 10-Gigabit Ethernet on Commodity Systems. *IEEE Micro '04*.

[19] J. Carlstrom and R. Rom. Application-Aware Admission Control and Scheduling in Web Servers. In *Infocom '02*, June 2002.

[20] A. Jacob, I. Troxel, and A. George. Distributed Configuration Management for Reconfigurable Cluster Computing. In *ERSA '04*.

[21] D. Li, P. Cao, and M. Dahlin. WCIP: Web Cache Invalidation Protocol. IETF Internet Draft, November 2000.

[22] N. Bhatti and R. Friedrich. Web server support for tiered services. In *IEEE Network*, September 1999.

[23] S. Narravula, P. Balaji, K. Vaidyanathan, H. W. Jin, and D. K. Panda. Architecture for Caching Responses with Multiple Dynamic Dependencies in Multi-Tier Data-Centers over InfiniBand. In *IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID)*, 2005.

[24] S. Narravula, P. Balaji, K. Vaidyanathan, S. Krishnamoorthy, J. Wu, and D. K. Panda. Supporting Strong Coherency for Active Caches in Multi-Tier Data-Centers over InfiniBand. In *Proceedings of System Area Networks (SAN)*, 2004.

[25] S. Narravula, H. W. Jin, and D. K. Panda. Designing Efficient Cooperative Caching Schemes for Multi-Tier Data-Centers over RDMA-enabled Networks. Technical report, OSU-CISRC-6/05-TR39, The Ohio State University, 2005.

[26] P. Pradhan and R. Tewari and S. Sahu and A. Chandra and P. Shenoy. An Observation-based Approach Towards Self-Managing Web Servers. In *IWQoS '02*, 2002.

[27] F. Petrini, W. Feng, A. Hoisie, S. Coll, and E. Frachtenberg. The Quadrics Network (QsNet): High-Performance Clustering Technology. In *HotI '01*.

[28] S. Lee and J. Lui and D. Yau. Admission control and dynamic adaptation for a proportionaldelay diffserv-enabled web server. In *Proceedings of SIGMETRICS*, 2002.

[29] H. V. Shah, D. B. Minturn, A. Foong, G. L. McAlpine, R. S. Madukkarumukumana, and G. J. Regnier. CSP: A Novel System Architecture for Scalable Internet and Communication Services. In *the Proceedings of the 3rd USENIX Symposium on Internet Technologies and Systems*, pages pages 61–72, San Francisco, CA, March 2001.

[30] K. Shen, H. Tang, T. Yang, and L. Chu. Integrated Resource Management for Cluster-based Internet Services. In *OSDI '02*.

[31] K. Vaidyanathan, P. Balaji, H. W. Jin, and D. K. Panda. Workload-driven Analysis of File Systems in Multi-Tier Data-Centers over InfiniBand. In *Computer Architecture Evaluation with Commercial Workloads (CAECW-8), in conjunction with the International Symposium on High Performance Computer Architecture (HPCA)*, 2005.

[32] K. Vaidyanathan, P. Balaji, S. Narravula, H.-W. Jin, and D. K. Panda. Supporting Dynamic Fine-grained Resource Reconfiguration in InfiniBand-based Web Data-Centers. Technical report, OSU-CISRC-1/06-TR07, The Ohio State University, 2006.

[33] K. Vaidyanathan, H.-W. Jin, S. Narravula, and D. K. Panda. Accurate Load Monitoring for Cluster-based Web Data-Centers over RDMA-enabled Networks. Technical report, OSU-CISRC-7/05-TR49, The Ohio State University, 2005.

[34] K. Vaidyanathan, S. Narravula, P. Balaji, H. W. Jin, and D. K. Panda. Soft Shared State Primitives for Multi-Tier Data-Center Services. Technical Report OSU-CISRC-1/06-TR06, The Ohio State University, 2006.