ACTiCLOUD: Enabling the Next Generation of Cloud Applications

DOI:
10.1109/ICDCS.2017.252

Document Version
Accepted author manuscript

Published in:
Proceedings of the 37th IEEE International Conference on Distributed Computing Systems (ICDCS)

ACTiCLOUD: Enabling the Next Generation of Cloud Applications

Georgios Goumas[1], Konstantinos Nikas[1], Ewnetu Bayuh Lakew[8], Christos Kotselidis[5], Andrew Attwood[3], Erik Elmroth[8], Michail Flouris[4], Nikos Foutris[5], John Goodacre[3], Davide Grohmann[7], Vasileios Karakostas[1], Panagiotis Koutsourakis[6], Martin Kersten[6], Mikel Lujàn[5], Einar Rustad[2], John Thomson[4], Luis Tomás[8*], Atle Vesterkjaer[2], Jim Webber[7], Ying Zhang[6], Nectarios Koziris[1]

[1]*ICCS, National Technical University of Athens* [2]*Numascale* [3]*Kaleao*
[4]*OnApp* [5]*The University of Manchester* [6]*MonetDB Solutions*
[7]*Neo Technology* [8]*Dept. of Computing Science, Umeå University*

***Abstract*—Despite their proliferation as a dominant computing paradigm, cloud computing systems lack effective mechanisms to manage their vast amounts of resources efficiently. Resources are stranded and fragmented, ultimately limiting cloud systems' applicability to large classes of critical applications that pose non-moderate resource demands. Eliminating current technological barriers of actual fluidity and scalability of cloud resources is essential to strengthen cloud computing's role as a critical cornerstone for the digital economy.**

ACTiCLOUD proposes a novel cloud architecture that breaks the existing scale-up and share-nothing barriers and enables the holistic management of physical resources both at the local cloud site and at distributed levels. Specifically, it makes advancements in the cloud resource management stacks by extending state-of-the-art hypervisor technology beyond the physical server boundary and localized cloud management system to provide a holistic resource management within a rack, within a site, and across distributed cloud sites. On top of this, ACTiCLOUD will adapt and optimize system libraries and runtimes (e.g., JVM) as well as ACTiCLOUD-native applications, which are extremely demanding, and critical classes of applications that currently face severe difficulties in matching their resource requirements to state-of-the-art cloud offerings.

***Keywords*-cloud computing, resource management, in-memory databases, resource disaggregation, scale-up, rackscale hypervisor**

I. Introduction

Since its emergence, cloud computing has become a dominant computing model enabling cost-effective and flexible access to a wide pool of resources. Large classes of applications, such as web services, big data analytics, distributed applications for social networking, storage and various other home and business applications, have been smoothly integrated into cloud services, leading to a rapid growth of the cloud market, expected to continue in the foreseeable future. To accommodate these demands and minimize the risk of losing the constantly increasing flow of customers, Cloud Service Providers (CSPs) have been relying on the continuous expansion of their IT facilities while employing simplistic policies to manage resources.

This approach though appears to be unsustainable for the next generation of mature cloud computing systems. The aggregation of IT infrastructures in large CSPs reduces the infrastructure and administration costs for small organizations and is more resource efficient due to economies of scale. However, it is the responsibility of the CSPs to properly manage these vast pools of resources.

Unfortunately, numerous studies have shown that CSPs fail to utilize the available resources efficiently, with their utilization ranging from 10% to 50% [1]–[7] for a number of reasons. First of all, the *"share-nothing"* PC system architecture used in data centers today, confines resource allocation to the physical bounds of servers, failing to express the fluidity of system resources that is inherent to the cloud paradigm, thus leading to resource *stranding* and *fragmentation*. Second, to guarantee availability during short time frames of peak demands and adhere to Service-Level Agreements (SLAs), CSPs compromise with severe *underutilization* of resources during non-peak periods [1]. Finally, both fragmentation and underutilization are exacerbated by the unpredictability and variability of workloads, as well as the lack of effective workload consolidation policies that would take into account resource availability, application demands and Quality of Service (QoS) levels [8]–[10].

On the application side, cloud computing has successfully encompassed applications that are either resource conservative or capable of requesting resources in the *"scale-out"* fashion, i.e., they can be easily decomposed to large numbers of loosely connected entities adhering to the *"share-nothing"* paradigm [10]. However, large classes of important applications are resource hungry and do not scale out gracefully for simplicity, correctness, programmability, and cost-effectiveness reasons. These applications are characterized as *"scale-up"*, since they need to execute on a single server under the *"share-anything"* paradigm. As such, they are typically executed on costly customized infrastructures under private administration and cannot take advantage of the cost-effectiveness and flexibility of the cloud [11].

To accommodate such classes of applications, ICT industries and research communities have started to realize new

*Currently working at Red Hat.

data center architectures that rely on the principles of hardware disaggregation and programmable infrastructure [12]–[16]. To efficiently manage though such disaggregated pool of resources, the current generation of cloud management hypervisors, scheduling algorithms, system libraries, and applications needs to be rethought and redesigned.

ACTiCloud proposes a novel cloud architecture that breaks the existing scale-up and share-nothing barriers and enables the holistic management of physical resources both at the local cloud site and at the distributed levels, targeting drastically improved utilization and scalability of resources. Our distributed, hyper-converged, "share-anything", resource scale-up and scale-out cloud platform will broaden the applicability of cloud technology across markets through richer and more cost effective deployments. This is achieved through redesign and enhancement of the entire cloud stack, including the hypervisor, the cloud manager, system libraries, language runtimes, and applications with a novel and holistic set of mechanisms and policies.

The remainder of the paper is structured as follows: Section II motivates the need for a novel cloud management architecture. Section III gives a high-level overview of the proposed architecture, while Section IV describes in detail the main ACTiCLOUD components. Section V presents the optimizations and adaptations needed from system libraries and applications to take advantage of the various features in an ACTiCLOUD-enabled infrastructure. Section VI presents the related work and current advancements in the area. Finally, Section VII summarizes the paper.

II. Motivation and Approach

Data-intensive applications are resource hungry and do not scale out gracefully for simplicity, consistency, correctness, programmability, and cost-effectiveness reasons. This class comprises numerous applications that are mainly database-driven. The rise of technologies such as column-stores [17]–[20]), NoSQL [21]–[25], and NewSQL [26]–[28] in the last decade, has caused a major paradigm shift in modern database systems [29]–[31], both in the use of hardware and in the developed software.

Traditionally, database systems assume that data primarily reside on disks, and data pages are frequently brought into or out of the main memory for processing. Therefore, one of the main focuses of those systems has been to minimize the disk I/Os. However, enabled by the price decrease of main memory, and to meet the demand of the rapid growth in the amount of data and computations to handle, both existing (e.g., traditional relational databases) and emerging (e.g., column stores, NoSQL, and NewSQL) database systems have shifted their focus to in-memory processing. In this setting, a database system is heavily tuned to work most efficiently with data that are already in the main memory. As long as all processing data are kept in memory, disk I/O is a secondary or even a non-issue. Consequently, to cope with the ever increasing performance demands of the data-intensive applications, modern databases consume steadily more main memory.

However, state-of-the-art cloud offerings are not able to match these demands, thus keeping this class of applications outside the cloud. Since database systems are often the cornerstones of business critical applications (e.g., online transaction processing, decision making, and predictions), to ensure their position in the market, it is extremely important to embrace the current and anticipate the future evolution within cloud offerings.

Recently, a number of cutting-edge industrial products and research projects have emerged targeting resource disaggregation and rack-scale computing [12]–[15], [32]. These approaches envision breaking the physical boundaries of the long lasting and traditional PC architecture by making hardware resources managed as independent units. Until now, in cloud computing systems resource disaggregation has only been realized for storage systems. As in typical configurations, storage data is shared across the cloud hosts, providing both data resilience and increased IO throughput.

Resource disaggregation opens up new opportunities as well as challenges. On the one hand, the demands of data-intensive and resource-hungry applications, called *ACTiCLOUD-native* applications hereafter, can be matched by such infrastructures. On the other hand, the management of the infrastructure and the applications themselves are becoming more complex, requiring redesign and enhancement of the entire cloud management stack. ACTiCLOUD's vision is to architect, design, and develop a novel cloud management stack to tackle this challenge and enable the cloud model to embrace the next generation of applications.

ACTiCLOUD targets enhanced resource efficiency by creating flexible pools of resources that are aggregated from the available resources across multiple servers hosted at a single data center. Additionally, it utilizes the existing distribution of a cloud configuration to further balance loads between collaborating cloud sites that are geographically distributed. In this way, CSPs can resort to much more cost efficient capacity planning, utilizing support from collaborating cloud sites to accommodate the demands of peak periods. Such collaboration can be used both between separate public cloud vendors, and between an enterprise private cloud and their public cloud partners.

Fig. 1 shows the core concept of ACTiCLOUD. Resources collected from multiple servers from wide pools at the rack-scale can be assigned to applications minimizing fragmentation and enabling the execution of applications with large resource requests. When resources at a site come to end, applications can be serviced by sibling sites that reside in remote geographical locations.

ACTiCLOUD aims to break the two critical barriers that currently hinder true fluidity of cloud resources: the *server barrier* and the *data center barrier*. Although cloud

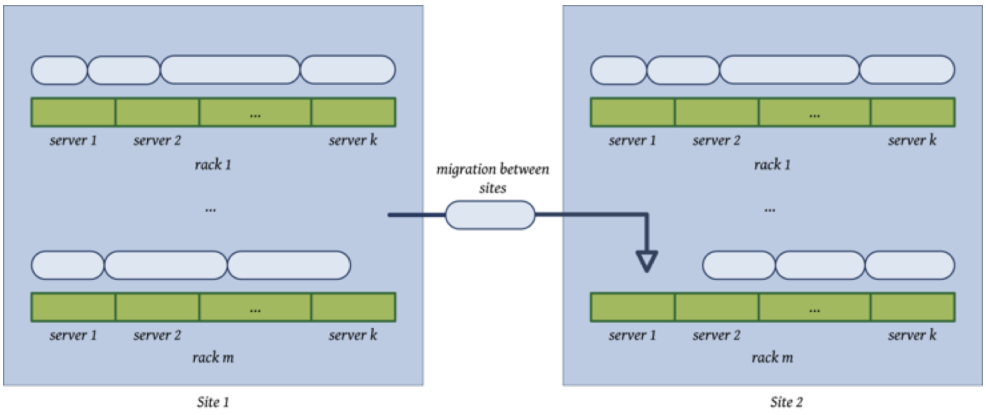

Figure 1: Core ACTiCLOUD concept.

technologies advertise their ability to provide flexibility and elasticity in resource provisioning, this is true so far only for storage resources. These resources can be shared between multiple servers within a single data center via either centralized Storage Attached Network (SAN) solutions or distributed storage platforms (such as Ceph [33] or OnApp Integrated Storage [34]), and horizontal (i.e., "scale out") scalability where additional Virtual Machines (VMs) are allocated to cover increased demands. The critical resources of processing cores and main memory remain constrained to what a physical server can provide, limiting the ability of the infrastructure to host resource-demanding applications (especially large-scale in-memory processing) and increase resource utilization. ACTiCLOUD aims to extend resource disaggregation beyond storage, to cores and main memory.

Additionally, typical cloud configurations involve sites that are geographically distributed and disjointly managed either due to operational needs or scalability restrictions. Solutions such as WL2 [35] have been proposed recently, that leverage Software Defined Networking to provide a scalable, high-performance, multi-datacenter layer-2 network architecture, thus enabling tighter cross-site interactions. ACTiCLOUD builds on top of this technological trend to provide a set of mechanisms and policies that will support application migrations between cloud sites that are geographically distributed [36]. In this way, ACTiCLOUD-enabled platforms will be able to support policies enforcing geo-based load balancing, improved resource utilization and service locality (e.g., when traffic comes from various places, or if there are strict quality/latency requirements, or if there are regulatory constraints), cost optimization (e.g., when resources are cheaper in another place), and service continuity and geo-redundancy.

By utilizing the ACTiCLOUD approach, we intend to provide the necessary technological mechanisms to support resource-demanding, ACTiCLOUD-native applications in ACTiCLOUD-enabled cloud systems. To this direction, we optimize system software and managed runtime systems that are heavily utilized by ACTiCLOUD-native applications, and two cutting-edge representatives of ACTiCLOUD-native applications, i.e., a columnar in-memory analytical database (MonetDB) [18] and a graph database (Neo4j) [37], will be adapted. At the same time, an ACTiCLOUD-enabled cloud system continues to support traditional cloud applications without requiring any adaptation.

III. The ACTiCLOUD architecture overview: Enabling Resource Disaggregation and Beyond

We conceptualize the architecture into two logical dimensions. The first dimension contains management operations that provide access to the virtualized computing infrastructure which is the *ACTiCLOUD IaaS*. The second dimension contains operations related to optimizations and adaptation of system libraries and applications in order to enable applications to run on ACTiCLOUD IaaS. Schematically, these dimensions provide a layered foundation for the architecture.

Fig. 2 depicts an abstract overview of the ACTiCLOUD architecture. At the bottom level, namely *Disaggregated Hardware resources*, we employ two cutting-edge European technologies brought in the project by Numascale [12], and Kaleao [13]. These technologies are capable of supporting resource disaggregation for typical cloud servers and microservers respectively, thus providing the mechanisms to build resource pools at the rack-level by unifying the resources of multiple servers. The *Aggregation Layer* presents a cache coherent system to the *Rackscale Hypervisor* that takes advantage of that via kernel modules, system libraries or virtualization elements.

On top of this server architecture, *Rackscale Hypervisor* extends a state-of-the-art hypervisor, provided by ONAPP's MicroVisor [38], which operates at the rack-scale to provide a unified management of resources and a single system image. On top of the rack-scale hypervisor lies the *Holistic Resource Management*, which extends the de facto open-source cloud management software OpenStack [36], [39] to provide a holistic, autonomous management of resources both locally and across distributed clouds, thus enabling both "scale-up" and migration of ACTiCLOUD-native applications. Those three layers (server, hypervisor, and cloud

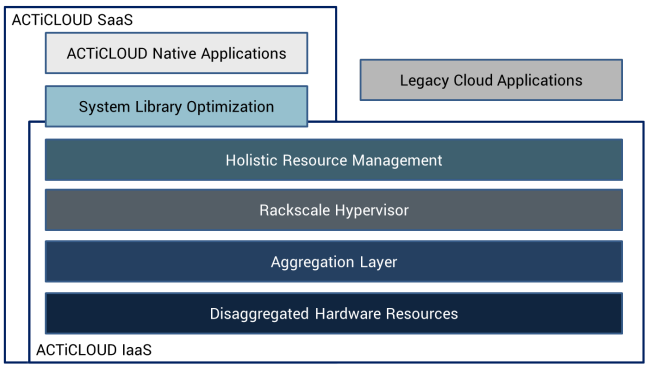

Figure 2: High Level ACTiCLOUD Architecture.

manager) provide a novel substrate for *resource-efficient distributed IaaS clouds*.

To enable ACTiCLOUD-native applications such as the next generation business-critical, resource-hungry application frameworks (e.g., DataBase-as-a-Service (DBaaS)), different adaptations and optimizations of system libraries and runtimes are required. First, at the *System Library Optimization* layer, critical system libraries that manage resources (e.g., Linux libNUMA) and the language runtime (e.g., JVM) will be optimized. Then, at a higher level, namely *ACTiCLOUD-Native Applications*, in-memory, column-store OLAP databases (e.g., MonetDB) and graph databases (e.g., Neo4j) will evolve to take advantage of the features provided by ACTiCLOUD IaaS.

IV. The ACTiCloud IaaS Main Building Blocks

This section presents the different building blocks that make up the underlying ACTiCLOUD IaaS to address the challenges and opportunities and realize the concepts presented in the earlier sections.

Fig. 3 shows the different ACTiCLOUD components and their interactions. The disaggregated hardware resources and aggregation layer expose themselves and provide monitoring features to the rackscale hypervisor. The rackscale hypervisor provides transparent, isolated, and abstract view of the hardware resources for each running application. It allows VMs to effectively share resources at the rack-level, effectively eliminating the physical server boundaries. On top of the rackscale hypervisor, the holistic resource manager dynamically and efficiently schedules resources for the incoming as well as running applications according to their requirements. The details of each building block are presented in the next sections.

A. Disaggregated Hardware Resources and Aggregation Layer

The ACTiCLOUD architecture will be implemented on top of two novel European server technologies, i.e., Numascale and Kaleao, that support resource disaggregation. However, the proposed architecture, the concepts, and the majority of the software modules (apart from those that are specific and optimized for the embraced technologies) will be compatible to alternative solutions for resource disaggregation.

In the Aggregation Layer the Numascale platforms link together commodity servers to form a single unified system, in which all processors can coherently access and share all memory and I/O, breaking through the scale-up limitations within existing data centers. The combined system runs a single instance of a standard operating system such as Linux. At the heart of the Numascale platforms is NumaConnect, implemented with the NumaChip [12] - a single chip that combines the cache coherent shared memory control logic with an on-chip high-performance network switch. The system size can be scaled up to 4K nodes, where each node can contain multiple processors. Effective memory size is limited only by the 48-bit physical address range provided by state-of-the-art processor technologies resulting in a record-breaking total system main memory of 256TiB.

The Kaleao KMAX platform [13] is a new-generation server system architecture based on the principles of true convergence with the hardware design creating a "share-anything" resource scale-out platform. Its scalable architecture is constructed from compute units, each defined by their processing, memory, and IO-bandwidth capabilities. A node is then created from multiple compute units with local access to the global and distributed pools of IO resources, part of which are instantiated locally and thus creating the physical node circuits. Each node then defines network, and storage compute units which dynamically create and assign IO device resources to each virtual machine instantiated through a technique known as physicalization. Four nodes are then grouped into a blade and up to 12 blades can be installed in a single 3U chassis.The KMAX platform today utilizes its “share-anything” capabilities across storage and networking, with the ability to expose other resources, such a memory, into the global pools expected within the roadmap of Kaleao products. A single 42U rack of such a system could therefore scale to provide applications access to over 20,000 cores with access up to a 350 TB pool of paged memory, 14 Tb/s of network access bandwidth and 6 PB of flash storage.

B. Rackscale Hypervisor

The rack-scale hypervisor implements a thin virtualization layer for the unified pool of resources that are aggregated from all the physical servers in the rack and provides virtual machines that are dynamically adapted to any application resource requirement. The hypervisor layer enables an efficient multi-tenant environment, increasing resource utilization and allows smart resource management in a power-efficient manner. To this direction, the hypervisor implementation needs to: a) scale efficiently at rack-level, transcending resource limits (e.g., core counts or memory sizes) that state-of-the-art hypervisors currently support, b) incorporate awareness

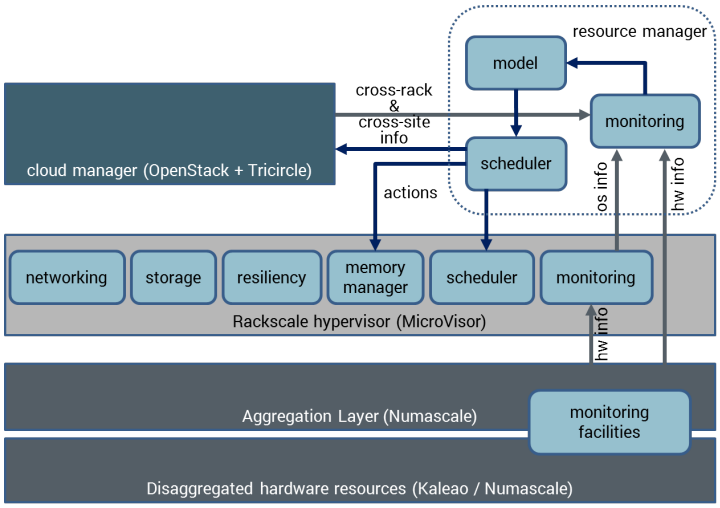

Figure 3: ACTiCLOUD IaaS Building Blocks.

of resource locality and network topology, and c) extend the reliability mechanisms at the rack scale to cope with failures and errors across all components at the rack-level.

Hyper-convergence is the software-defined approach to resource management that allows storage, compute and networking resources to be dynamically allocated, resized and relocated through software virtualization into multiple application-specific elastic units. Embracing hyper-convergence, the ACTiCLOUD hypervisor layer is rearchitected to allow distributed storage resources to be used by any CPU core in the rack with low overhead, while it allows for fine-grain control of each set of CPU and memory resources used, depending on their capacity, access characteristics and network topology.

For rack-scale systems to work efficiently, they need to be equipped with high bandwidth, low-latency, reliable network links between the servers. Various emerging interconnects [40], [41] aim at improving communication performance at the rack level, such as optical interconnects. The ACTiCLOUD platform is agnostic to the interconnect used and the rack-scale hypervisor can operate on future optical interconnects, determining the best links to use at any time.

Networking, *storage*, *memory*, and *CPU cores* in the whole rack are managed centrally from a distributed management and control layer that uses hypervisor-level mechanisms to present a rack-level system view and implement rack-wide management policies for resources and applications. The hardware resources located in the servers of the rack will be tied together in a pool and used for the deployment of VMs based on their resource needs, regardless of whether the resources are physically located on a single server or not. The hypervisor enables upper layer awareness of the location and connection characteristics (latency, bandwidth) between pairs of resources via efficient *monitoring* APIs. This knowledge combined with the detection of the workloads' characteristics, allows the holistic resource manager to provide optimal workload placement.

Finally, when working at the rack-level to provide a single system view there is an increased chance of a partial system failure. The ACTiCLOUD platform must be able to detect and gracefully recover from component failures, either at a hardware or software level, according to failure semantics and redundancy levels. Given the large number of interconnected components, failures should be expected and the platform will need to incorporate appropriate recovery mechanisms. Automatic failure detection is the responsibility of the hyper-converged hypervisor layer. However, correcting failures may require manual hardware replacement or operator action to restart software components, without compromising the existing workloads.

C. Holistic Cloud Resource Management

Management of resources in cloud environments has attracted significant industrial and academic attention. Large industrial players in cloud resource provisioning, such as Amazon, Microsoft, Google, and VMware, typically rely on in-house, commercial software solutions to manage their own clouds. The main free, open-source competitor to the above proprietary solutions is OpenStack, which has managed to become the choice of solution in numerous cloud installations worldwide, including both private and public clouds at various scales. OpenStack promotes open standards within the industry and is supported by a wide network of industrial and academic partners. ACTiCLOUD aims to design and implement an hierarchy of resource schedulers that will operate dynamically at the rack, site, and cross-site levels by extending existing hypervisor policies collaborating with the hypervisor layer and OpenStack, in order to maximize our impact and promote cloud interoperability and openness.

Current state-of-the-art research on cloud resource allocation focuses on the server and site levels to optimize

workload consolidation, i.e., the placement of multiple virtualized workloads in the same physical host, a key concept of cloud computing. Despite the significant benefits, workload consolidation increases the risk of performance anomalies compared to executions in an isolated environment, especially for latency-critical workloads [1]. Performance degradation mainly occurs due to interference of VMs in the same physical host [42]. Although isolated in its own virtual hardware sandbox, each VM competes with its neighbors for access to hardware components that cannot be exclusively provisioned by the host operating system. The colocation of applications on multi-tenant systems can present a challenge whereby one or more applications suffer from significant performance degradations. To make things even worse, resource contention may arise unexpectedly at any time between co-located VMs, severely affecting the offered QoS.

Several recent research works [1], [5], [8], [42]–[45] propose scheduling schemes that attempt to optimize workload coexistence, while avoiding application interference based on profiling of the considered applications and their interactions. However, these approaches have not been incorporated in cloud resource managers that either are conservatively avoiding co-locations of latency critical workloads [1], need to rely on user information that has doubtful preciseness [5] or does not even exist, or employ simplistic approaches based on decisions that are taken randomly, in a round-robin fashion, or taking into account only CPU utilization [8].

Resource allocation in OpenStack is static and straightforward. The nova-compute module interacts with the nova-scheduler service to determine how to dispatch compute requests. For example, the nova-scheduler service determines on which physical machine (host) a VM should launch, and once launched, the VM executes on that machine until the system administrator decides to migrate it manually. The filter scheduler is the default scheduler for scheduling virtual machine instances. It supports filtering and weighting to make informed decisions on where a new instance should be created. When the filter scheduler receives a request for a resource, it first applies filters to determine which hosts are eligible for consideration when dispatching a resource. OpenStack includes several filters that consider the proper matching of requested resources with available ones.

ACTiCLOUD builds on top of the virtualized pools of resources at the rack-level, and implements an hierarchy of *scheduling* modules at the rack, site and cross-site levels. The goal of ACTiCLOUD is to a) better utilize the available resources in the local ACTiCLOUD-enabled site, b) distribute load among the distributed cloud according to the specified policies (e.g. load balance, resilience, and energy efficiency), and c) enable dynamic decision making and actions. The hierarchical scheduling module collaborates with the hypervisor and OpenStack.

Within the rack level, the scheduler considers more elaborate allocation schemes for workload consolidation that minimize application interference. The scheduler takes rack-scale decisions on application consolidation using fine-grained information collected from all critical rack-level components (CPUs, memory modules, interconnection network etc.) and application resource footprints via lightweight *monitoring* tools. At the site-level, current scheduling policies are extended to increase the scheduling awareness of the underlying, rack-scale server platforms. On top of that, it implements *workload characterization & modelling* techniques to model the relationship between the data center's resources and applications in order to decide upon more efficient application collocations that avoid interference and act accordingly by workload redistributions, in order to better meet the objectives of the high-level scheduling policies enforced.

To realize distributed cross-site resource management, ACTiCLOUD extends the capabilities of current VM migration mechanisms with: a) prediction techniques, based on applications' memory access patterns and network usage that will enable a more accurate estimation of migrations duration as well as the amount of network traffic required, b) migration scheduling algorithms that take advantage of the improved estimations about migration times and sizes not only to decide on the migration order, but also to better choose the target VMs to migrate as well as the preferred migration mechanisms (pre-copy, post-copy, migration data compression, memory pages prioritization, etc.).

V. Enabling ACTiCLOUD-Native Applications: DBaaS

This section discusses optimizations and modifications required in both system libraries and applications to allow the exploitation of the ACTiCLOUD architecture's features. Fig. 4 shows the expected optimization and modifications in the system libraries and applications. We focus on the changes expected in the Linux OS kernel and the Java Virtual Machine (JVM) for the reasons stated below. We also use two representative in-memory databases to enable DBaaS and use the features exposed in an ACTiCLOUD-enabled system.

A. Optimizing system software and language managed runtimes

To better facilitate large databases in cloud ecosystems, ACTiCLOUD needs to optimize substrate software libraries and runtime systems that also significantly affect the management of resources of the databases in the upper layers. Regarding software libraries, we need to focus on the way resources (especially memory) are managed by the guest operating system, provide optimized extensions of the libNUMA library to make efficient use of the extended NUMA capabilities of ACTiCLOUD systems, and also adapt kernel scheduling and OS services within Linux. Moreover, as

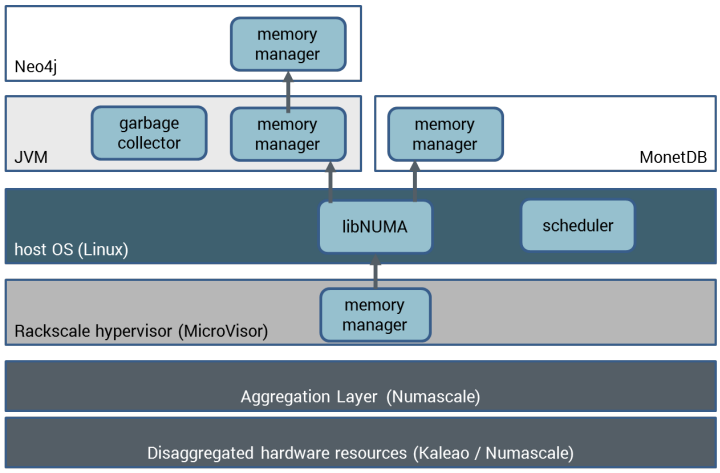

Figure 4: Enabling ACTiCLOUD-native applications: DBaaS.

ACTiCLOUD works on a deep stack of resource managers (hypervisor, cloud management, guest operating system, language runtimes and database frameworks), we need to vertically homogenize their operations, resolve inefficiencies and incompatibilities so as to ensure smooth optimization across the stack.

Regarding managed runtime languages, these are highly dependent on virtualization technologies that have seen an impressive growth since mid-1990s. The emergence of Big Data has even further increased the importance of virtualization. In particular, application virtualization, such as in the JVM or Microsoft's .NET CLR (Common Language Runtime), has been especially powerful. The vast majority of big data processing frameworks, such as Hadoop [46], Spark [47], Flink [48], Neo4j [37], Storm [49], Samza [50], and many others, rely on JVMs. Some of them (e.g. Samza) support only JVM-based languages (e.g., Java or Scala), while other frameworks (e.g., Spark) provide also support for languages such as Python.

The increase in compute processing power and memory size has lead to applications operating over increasingly larger data sets, resulting in high memory requirements. This stresses the memory system and consequently, in the context of managed runtime languages, affects the memory management system offered by the runtime. Garbage collection, the process of automatic memory management inside a JVM, is employed by numerous distributed applications in cloud data centers. Google (AppEngine), Microsoft (Azure), Twitter, and Facebook totally or partially base their software on them.

Garbage collection enables a VM to exploit high-level features of programming languages (e.g. Java, Python) while increasing productivity and guaranteeing safety. However, it suffers from many inefficiencies on cloud-based architectures. A major problem of such systems is that they execute a high number of independent JVMs in order to scale to the large number of nodes that are used. These JVMs behave independently of each other, which can have severe performance consequences. For example, it has been identified that the lack of coordination between JVMs regarding when to perform garbage collection results in significant performance slowdowns [51] in Apache Spark and Cassandra – often, a pause in one JVM to perform garbage collection propagates to the rest due to synchronization requirements, stalling the whole system. Finally, in latency-critical applications (e.g., web servers or databases), these idle intervals can cause requests to take unacceptably long times; and thus make a node's data unavailable.

The major challenge faced by existing JVMs is that they must evolve to meet the needs of cloud computing. ACTiCLOUD will extend and optimize state-of-the-art JVMs to harness the benefits of the ACTiCLOUD architecture. To this direction we will change and optimize JVMs in the following manner: a) JVMs will be able to utilize the extended amount of resources (both computational and memory) to ensure peak performance by mitigating the long garbage collection pauses, b) the garbage collector will be able to manage memory that is not physically located in the underlying hardware but in a remote location, c) JVMs will exploit the unified architecture proposed by the ACTiCLOUD project to improve node communication and synchronization, and d) the aforementioned functionalities will be transparent to the application code.

B. Designing ACTiCLOUD-Native Application: DBaaS

Data warehouse and OLAP applications, such as Business Intelligence and data analytics, form a major class of critical applications hosted by cloud systems. Those applications assist enterprises to gain insights from various business data at maximal efficiency while for a minimal cost. The rapid growth of the various types and size of business data and the importance of assisting business critical decisions timely have imposed unprecedented demands on the data warehouse applications. Since the last decade, column-based Database Management Systems (DBMSs) that are highly

optimized for in-memory query processing have been generally acknowledged as one of the best underlying technology to support high-performance data warehouse and OLAP applications.

MonetDB is an open-source, columnar, SQL:2003 compliant DBMS with full-fledged support for ACID (Atomicity, Consistency, Isolation, Durability) properties of transactions. With its in-memory optimized technology, MonetDB focuses on data-intensive data warehouse analytics, which typically require scale-up to 100s GiBs of memory and tens to hundreds of processing cores.

ACTiCLOUD enables MonetDB to advance from a static DBMS to a dynamic elastic cloud DBMS, a major step forward in its evolution in the cloud computing era. First of all, MonetDB will strengthen its scalability both vertically (i.e. scale-up) and horizontally (i.e. scale-out). First, new features will be added into MonetDB to thoroughly exploit the massive computation power and memory volume a single ACTiCLOUD-enabled rack-scale system can provide to transparently scale up applications requiring complex data analytics. Second, MonetDB will collaborate with the hypervisor and the resource manager of ACTiCLOUD to quickly react to workload changes by dynamically increasing or decreasing resource allocations. Finally, MonetDB will improve its resilience by capitalizing on the geographical distribution of the underlying cloud platform. We will extend the naive replication scheme of MonetDB to place database replicas at strategic locations for better fault-tolerance, performance and availability.

Graph databases are the fastest growing category of data storage and query technology today. With easy modeling and powerful query capabilities, graphs are becoming a common tool for exploitation of big data and are also being increasingly used as primary operational databases. Graph data structures can be used to effectively store context-rich data sets while preserving their complexity. Consequently, applications that make use of graph databases to mine real-time insights from their data are able to pose questions not previously possible, partially due to the high dimensionality of the data itself (the context-preserving characteristic of graphs) and partially due to the unique ability of graph databases to deeply query these graphs. Due to their graph-optimized storage and query engines, graph databases are able to not only mine complex patterns from graphical data with lower latency than traditional and competing technologies, but to do so with a fraction of the resources.

Neo4j is the world's first and leading graph database. Neo4j is optimized for high-performance graph traversals and safety and its users can query many millions of edges and vertices per second in a graph, while enjoying the safety of ACID transactions when writing data into the graph.

Neo4j is optimized for in-memory operation; its cache subsystem is optimized for graph storage and retrieval under highly concurrent load. In the absence of sufficient memory Neo4j will fall back to using slower levels of tiered storage (disk), permitting its performance to gracefully degrade. That said, the performance cost of using persistent storage is significant, users are advised to provide Neo4j with adequate memory to cache their entire data set whenever possible. Today's cloud providers, however, do not offer machine instances with very large (TiB) RAM at cost-competitive rates.

To take advantage of ACTiCLOUD-enabled systems, Neo4j needs to evolve its internal architecture to be aware of resource access costs. In doing so, the query planner should, in general, be able to optimize for the cheapest routes through the graph based both on the graph structure (as per today) and the cost of accessing pages or subgraphs (by the end of the project). At single-rack scale, the ACTiCLOUD-enabled system enables large graph analyses that exceed the capacity of a single machine to take place. Specifically, using the ACTiCLOUD stack to provide scale-up capabilities, Neo4j will be equipped to process graphs that far outstrip main memory constraints and bring many cores into the analysis. This obviates the costly need to export graphs to third-party system (e.g. Hadoop, Spark) for iterative analysis and instead enables the database to perform that work. At multi-rack scale, Neo4j will be optimized to adapt to load by requisitioning hardware resources dynamically from the underlying cloud management platform.

VI. Related Work

In recent years, several efforts have been undertaken to improve the control and management resources in cloud environments by focusing on different but complimentary issues. DOLFIN [52] and RESERVOIR [53] projects base their approaches on VM migrations to attain energy efficiency, while PANACEA [54] and ORBIT [55] projects aim at fault tolerance. Projects such as CACTOS [56] and HARNESS [57] support application-specific resource provisioning mechanisms for heterogeneous cloud infrastructures. RAPID [58] also focuses on resource management but its solution is specific for an heterogeneous edge computing cloud architecture that involves mobile devices, edge servers and cloud servers. ACTiCLOUD goes beyond the state-of-the-art by working on an holistic resource management scheme that embraces hierarchically three critical levels: the rack, local site, and cross-site levels.

MIKELANGELO [59] and IOSTACK [60] propose new software architectures for cloud systems. MIKELANGELO's focus is on the software stack on a single cloud server with a focus on I/O and enabling HPC, without considering resource disaggregation. IOSTACK on the other hand embraces the concept of storage disaggregation and supports computation disaggregation for Map-Reduce style Big Data applications. Beyond these technologies, ACTiCLOUD supports full computation and memory disaggregation applied to the entire ecosystem of data-intensive and

resource-hungry applications (e.g., relational, graph, Map-Reduce style).

A number of projects focus on cloud applications with a special objective to improve their management of resources in terms of elasticity and scalability. This family includes CLOUDSCALE [61], CONPAAS [62], SEACLOUDS [63], CELAR [64], and INPUT [65]. These projects support application-specific resource provisioning and optimization mechanisms incorporated within specific application's software layers and programming environments. ACTiCLOUD is agnostic to programming frameworks and to that extent our approach is orthogonal to the aforementioned projects.

All state-of-the-art research studies address different aspects of the issues that cloud environments face by looking at different levels of the cloud stack. In contrast, ACTiCLOUD goes beyond by rethinking and redesigning the entire cloud stack architecture, i.e. the hypervisor, the cloud manager, system libraries, language runtimes, and applications, to support the next generation cloud applications.

VII. Summary

ACTiCLOUD proposes a novel cloud architecture that breaks the existing scale-up and share-nothing barriers. It enhances resource efficiency by creating flexible pools of resources that are aggregated from the available resources across multiple servers hosted at a single data center, while also taking advantage of the existing distribution of a cloud configuration to further balance loads between collaborating cloud sites that are geographically distributed. Finally, ACTiCLOUD provides all the necessary technological mechanisms to support resource-demanding applications in ACTiCLOUD-enabled cloud systems, with a particular focus on in-memory database systems.

Acknowledgement

This research has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement no. 732366 (ACTiCLOUD).

References

[1] D. Lo, L. Cheng, R. Govindaraju, P. Ranganathan, and C. Kozyrakis, "Heracles: Improving resource efficiency at scale," in *Proceedings of the 42nd Annual International Symposium on Computer Architecture*, ser. ISCA '15, 2015, pp. 450–462. [Online]. Available: http://doi.acm.org/10.1145/2749469.2749475

[2] L. A. Barroso and U. Hoelzle, *The Datacenter As a Computer: An Introduction to the Design of Warehouse-Scale Machines*, 1st ed. Morgan and Claypool Publishers, 2009.

[3] M. Carvalho, W. Cirne, F. V. Brasileiro, and J. Wilkes, "Long-term slos for reclaimed cloud computing resources," in *Proceedings of the ACM Symposium on Cloud Computing, 2014*, 2014, pp. 20:1–20:13. [Online]. Available: http://doi.acm.org/10.1145/2670979.2670999

[4] J. M. Kaplan, W. Forrest, and N. Kindle, "Revolutionizing Data Center Energy Efficiency," McKinsey & Company, Tech. Rep., 2008.

[5] C. Delimitrou and C. Kozyrakis, "Quasar: Resource-efficient and qos-aware cluster management," in *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, 2014, pp. 127–144. [Online]. Available: http://doi.acm.org/10.1145/2541940.2541941

[6] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in *Proceedings of the Third ACM Symposium on Cloud Computing*, ser. SoCC '12, 2012, pp. 7:1–7:13. [Online]. Available: http://doi.acm.org/10.1145/2391229.2391236

[7] A. Vasan, A. Sivasubramaniam, V. Shimpi, T. Sivabalan, and R. Subbiah, "Worth their watts? - an empirical study of datacenter servers," in *16th International Conference on High-Performance Computer Architecture (HPCA-16 2010), 9-14 January 2010, Bangalore, India*, 2010, pp. 1–10. [Online]. Available: http://dx.doi.org/10.1109/HPCA.2010.5463056

[8] E. Angelou, K. Kaffes, A. Asiki, G. I. Goumas, and N. Koziris, "Improving virtual host efficiency through resource and interference aware scheduling," *CoRR*, vol. abs/1601.07400, 2016. [Online]. Available: http://arxiv.org/abs/1601.07400

[9] E. B. Lakew, C. Klein, F. Hernandez-rodriguez, and E. Elmroth, "Performance-Based Service Differentiation in Clouds," in *15th IEEE/ACM International, Symposium on Cluster Computing and the Grid (CCGrid '15)*, 2015, pp. 505–514.

[10] E. Lakew, A. Papadopoulos, M. Maggio, C. Klein, and E. Elmroth, "KPI-agnostic Control for Fine-Grained Vertical Elasticity," in *Proceedings of The 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2017)*.

[11] D. J. Abadi, "Data management in the cloud: Limitations and opportunities," *IEEE Data Eng. Bull.*, vol. 32, no. 1, pp. 3–12, 2009.

[12] "Numascale numaconnet." forbes, Accessed: 2017-01-22. [Online]. Available: https://www.numascale.com/numa_pdfs/numaconnect-white-paper.pdf

[13] "Innovative server platform architecture." kaleao. [Online]. Available: http://www.kaleao.com/

[14] "Intel rack scale architecture." intel. [Online]. Available: http://www.intel.com/content/www/us/en/architecture-and-technology/intel-rack-scale-architecture.htmll

[15] "Hp moonshot." hP. [Online]. Available: https://www.hpe.com/us/en/servers/moonshot.html

[16] "Software-defined infrastructure: the first step toward hyperscale." ericsson, Accessed: 2017-01-29. [Online]. Available: http://www.ericsson.com/res/docs/whitepapers/wp-hyperscale-cloud.pdf

[17] S. Idreos, F. Groffen, N. Nes, S. Manegold, K. S. Mullender, and M. L. Kersten, "Monetdb: Two decades of research in column-oriented database architectures," *IEEE Data Eng. Bull.*, vol. 35, no. 1, pp. 40–45, 2012.

[18] P. Boncz, M. Zukowski, and N. Nes, "Monetdb/x100: Hyper-pipelining query execution," in *CIDR*, vol. 5, 2005, pp. 225–237.

[19] M. Stonebraker, D. J. Abadi, A. Batkin, X. Chen, M. Cherniack, M. Ferreira, E. Lau, A. Lin, S. Madden, E. O'Neil, P. O'Neil, A. Rasin, N. Tran, and S. Zdonik, "C-store: A column-oriented dbms," in *Proceedings of the 31st International Conference on Very Large Data Bases*, ser. VLDB '05, 2005, pp. 553–564. [Online]. Available: http://dl.acm.org/citation.cfm?id=1083592.1083658

[20] R. MacNicol and B. French, "Sybase IQ multiplex - designed for analytics," in *(e)Proceedings of the Thirtieth International Conference on Very Large Data Bases, Toronto, Canada, August 31 - September 3 2004*, 2004, pp. 1227–1230.

[21] "MongoDB." [Online]. Available: https://www.mongodb.org/

[22] "Apache cassandra." [Online]. Available: https://www.cassndra.org/

[23] "Redis." [Online]. Available: http://redis.io/

[24] "MarkLogic." [Online]. Available: http://www.marklogic.com/

[25] "Datastax." [Online]. Available: http://www.datastax.com/

[26] "Clustrix." [Online]. Available: http://www.clustrix.com/

[27] "NuoDB." [Online]. Available: http://www.nuodb.com/

[28] "VoltDB." [Online]. Available: http://voltdb.com/

[29] "Falling ram prices drive in-memory database surge," sAP, Accessed: 2017-01-25. [Online]. Available: http://www.digitalistmag.com/technologies/big-data/2012/10/24/ram-prices-drive-in-memory-surge-020097

[30] M. Vizard, "The rise of in-memory databases." posted on July 13th, 2012. [Online]. Available: http://insights.dice.com/2012/07/13/the-rise-of-in-memory-databases/

[31] H. 'Garcia-Molina ' and K. Salem, "Main memory database systems: An overview," *IEEE Trans. on Knowl. and Data Eng.*, vol. 4, no. 6, pp. 509–516, Dec. 1992. [Online]. Available: http://dx.doi.org/10.1109/69.180602

[32] "Bull connecting box." bull. [Online]. Available: www.scaleupservers.com/Bullion-S16-Server.asp

[33] S. A. Weil, S. A. Brandt, E. L. Miller, D. D. E. Long, and C. Maltzahn, "Ceph: A scalable, high-performance distributed file system," in *Proceedings of the 7th Symposium on Operating Systems Design and Implementation*, ser. OSDI '06, 2006, pp. 307–320.

[34] "OnApp Integrated Storage." [Online]. Available: https://docs.onapp.com/display/32AG/Integrated+Storage

[35] C. Chen, C. Liu, P. Liu, B. T. Loo, and L. Ding, "A scalable multi-datacenter layer-2 network architecture," in *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research*, ser. SOSR '15, 2015, pp. 8:1–8:12.

[36] "OpenStack Tricircle." [Online]. Available: https://wiki.openstack.org/wiki/Tricircle

[37] J. J. Miller, "Graph Database Applications and Concepts with Neo4j," in *Proceedings of the Southern Association for Information Systems Conference*, vol. 2324, 2013.

[38] X. Ragiadakou, M. Alvanos, J. Chesterfield, J. Thomson, and M. Flouris, "Microvisor: A scalable hypervisor architecture for microservers," 2015.

[39] "OpenStack ." [Online]. Available: https://wiki.openstack.org

[40] "Intel mxc," intel. [Online]. Available: http://www.intel.co.uk/content/www/uk/en/research/intel-labs-silicon-photonics-research.html

[41] "Delivering the datacenter of the future," open Commute Submit. [Online]. Available: http://www.opencompute.org/assets/OCP-Summit-V-Slides/Mainstage/OCP-Intel-Hooper.pdf

[42] C. Delimitrou and C. Kozyrakis, "Paragon: Qos-aware scheduling for heterogeneous datacenters," *SIGPLAN Not.*, vol. 48, no. 4, pp. 77–88, Mar. 2013.

[43] J. Mars, L. Tang, R. Hundt, K. Skadron, and M. L. Soffa, "Bubble-up: Increasing utilization in modern warehouse scale computers via sensible co-locations," in *Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO-44, 2011, pp. 248–259.

[44] A. Roytman, A. Kansal, S. Govindan, J. Liu, and S. Nath, "Pacman: Performance aware virtual machine consolidation," in *10th International Conference on Autonomic Computing, ICAC'13, San Jose, CA, USA, June 26-28, 2013*, 2013, pp. 83–94.

[45] A.-H. Haritatos, G. Goumas, N. Anastopoulos, K. Nikas, K. Kourtis, and N. Koziris, "Lca: A memory link and cache-aware co-scheduling approach for cmps," in *Proceedings of the 23rd International Conference on Parallel Architectures and Compilation*, ser. PACT '14, 2014, pp. 469–470.

[46] T. White, *Hadoop: The Definitive Guide*, 1st ed. O'Reilly Media, Inc., 2009.

[47] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," in *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing*, ser. HotCloud'10, 2010, pp. 10–10.

[48] P. Mika, "Flink: Semantic web technology for the extraction and analysis of social networks," *Web Semant.*, vol. 3, no. 2-3, pp. 211–223, Oct. 2005.

[49] A. Toshniwal, S. Taneja, A. Shukla, K. Ramasamy, J. M. Patel, S. Kulkarni, J. Jackson, K. Gade, M. Fu, J. Donham, N. Bhagat, S. Mittal, and D. Ryaboy, "Storm@twitter," in *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '14, 2014, pp. 147–156.

[50] "Apache samza." [Online]. Available: http://samza.apache.org

[51] L. Gu and H. Li, "Memory or time: Performance evaluation for iterative operation on hadoop and spark," in *10th IEEE International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing, HPCC/EUC 2013, Zhangjiajie, China, November 13-15, 2013*, 2013, pp. 721–727.

[52] "DOLFIN ." [Online]. Available: http://www.dolfin-fp7.eu

[53] B. Rochwerger, A. Galis, E. Levy, J. A. Cáceres, D. Breitgand, Y. Wolfsthal, I. M. Llorente, M. Wusthoff, R. S. Montero, and E. Elmroth, "RESERVOIR: management technologies and requirements for next generation service oriented infrastructures," in *Integrated Network Management, IM 2009. 11th IFIP/IEEE International Symposium on Integrated Network Management, Hofstra University, Long Island, NY, USA, June 1-5, 2009*, 2009, pp. 307–310.

[54] "PANACEA." [Online]. Available: /projects.laas.fr/panacea-cloud

[55] "ORBIT." [Online]. Available: www.orbitproject.eu

[56] "CACTOS." [Online]. Available: www.cactosfp7.eu

[57] "HARNESS." [Online]. Available: www.harness-project.eu/

[58] "RAPID." [Online]. Available: rapid-project.eu

[59] "MIKELANGELO." [Online]. Available: www.mikelangelo-project.eu

[60] "IOSTACK." [Online]. Available: www.iostack.eu

[61] "CLOUDSCALE." [Online]. Available: www.cloudscale-project.eu

[62] "CONPAAS." [Online]. Available: www.conpaas.eu

[63] "SEACLOUDS." [Online]. Available: www.seaclouds-project.eu

[64] "CELAR." [Online]. Available: https://en.wikipedia.org/wiki/CELAR

[65] "INPUT." [Online]. Available: www.input-project.eu