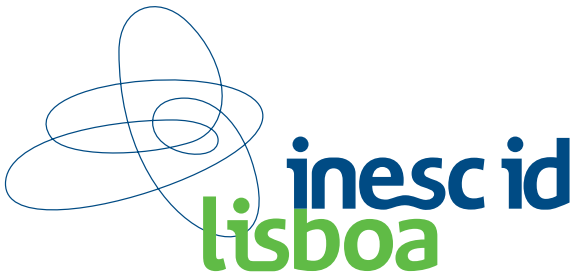

Technical Report RT/XX/2009

COGITARE: A Blueprint for A Cloud Infrastructure for Grid and Overlay Network Simulation Research

Luís Veiga João Nuno Silva João Garcia

INESC-ID/IST, Distributed Systems Group, Rua Alves Redol N. 9,1000-029 Lisboa, Portugal
luis.veiga@inesc-id.pt

Feb 2009

Abstract

This document presents a blueprint for COGITARE.

It describes a novel approach to grid and overlay network research leveraging distributed infrastructures and multi-core machines for increased simulation complexity and speed. It includes its motivation, background, current shortcomings and the core concepts of the novel research proposed.

COGITARE: *A Blueprint for*

A Cloud Infrastructure for Grid and Overlay Network Simulation Research

Luís Veiga, João Nuno Silva, João Garcia

luis.veiga@inesc-id.pt

Distributed Systems Group
INESC-ID Lisboa
Rua Alves Redol, 9
1000-029 Lisboa
Portugal

1. Introduction

This proposal presents a novel approach to grid and overlay network research leveraging distributed infrastructures and multi-core machines for increased simulation complexity and speed. We present motivation, background, current shortcomings and the core concepts of the novel research proposed.

Grid infrastructures and peer-to-peer overlay networks are areas of very active research within the distributed systems, middleware and network communities. Given the large number of elements in such topologies (grids and overlays), an important fraction of the research in this field is performed not on real systems but instead resorting to simulation tools (SimGrid, PeerSim, OverSim), hence avoiding machine cost and administration issues.

However, current grid and overlay simulation is encumbered by architecture, scale and performance limitations that cannot be solved simply by stacking more powerful computers. The performance of simulation tools is hindered because network and topology simulation code is mostly serial and manipulates a large global state assumed to be consistent. Simulations are run centrally, serially not drawing many of the advantages of increasingly prevalent multi-core machines or computing clusters. Thus, although the increased capability may be used to execute more grid and overlay simulations simultaneously, it is not possible to run each individual simulation faster. Moreover, one cannot leverage more CPUs to run more complex simulations (larger number of elements) within a given time frame.

Presently, Cloud Computing is considered the next step in large-scale computing infrastructures allowing integration of grids (with computing clusters distributed across virtual organizations) and utility computing farms (Amazon Elastic Clouds). In the Cloud, users have access to affordable computing power. Such infrastructures revolve around the instantiation of system-level virtual machines (Xen), of preset configurations, charged by the whole number of processing hours. However, with this model and current simulation tools, allocation of large number of virtual machines allows neither faster nor more complex simulations.

Existing free research testbeds (PlanetLab, Emulab) allow experiments of complete distributed systems by executing operating system, protocol/middleware and application code on each of the participating sites in a distributed computation. Interfaces for network (NS2) and overlay simulators (OpenDHT) are also offered.

Many researchers neither have access to clusters (dedicated or on the Grid), nor utility computing budget. Furthermore, they face a conundrum when testing algorithms and middleware. They either use higher-level grid/overlay simulators managing thousands of nodes but run serially, neither distributed nor parallel; or use lower-level system testbeds, parallel and distributed, but running complex OS/application code requiring computing power several orders of magnitude higher for experiments of comparable complexity.

When designing simulations, the models and representations used to describe topologies are often of low semantic level, ad-hoc and mostly programmatic. This forces researchers to actually write tedious code to instantiate nodes in the topology and intricate, tool-dependent code to implement basic data structures and behavior (routing, neighbor tables). Topology reuse, analysis and both scholarly and teaching dissemination is greatly limited.

We propose to further grid and overlay simulation by: 1) increasing scalability of simulation tools with a novel parallel, distributed and decentralized architecture; 2) unleashing the power of idle CPU cycles over the Internet harnessing them as a desktop grid (on a peer-to-peer overlay); and 3) ease grid and overlay research with a framework for topology definition, dissemination, evaluation and reuse.

In essence, COGITARE will provide researchers with a Grid4GridResearch.

To achieve this, current simulation algorithms must be made parallel and distributed. We will pursue this by splitting simulated topologies across several computers and experimenting with less strict concurrency and consistency models (divergence bounding, vector-field consistency) to synchronize adjacent parts of the a topology being simulated by different computers. Simulated topologies will be represented and managed across COGITARE as recursive topologies, i.e., overlays where data items represent the very grid and overlay topologies whose simulation may span tenths or hundreds of computers.

To facilitate grid and overlay research and its dissemination, reuse, experimenting and teaching, COGITARE will provide a framework encompassing a domain-specific language to represent with detail topology structure and behavior, integrated with a grid portal for the creation of experiments, access to results and reuse of previous experimental data.

COGITARE will perform underlying resource discovery, storage, scheduling, execution, and monitoring services to allow efficient and balanced execution of large number of simulations simultaneously.

2. Related Work

This section briefly covers relevant work on the areas related with the scope of this proposal: A) Simulation tools/testbeds, B) Parallel/distributed computing, C) Cycle-providing infrastructures, D) Peer-to-peer overlays, E) Grid middleware.

A) Simulation tools/testbeds

Simulation facilitates design and improves productivity, reliability, avoiding equipment and personnel costs in deployment. Tools offer different semantics/abstraction levels in simulation configuration and execution. Lower-level semantics is used in electronics design [FFL05], computer architecture

(Simics[MCE+02], Archer[FBF+08]) and etwork simulation (NS2[IH09]). This allows fine-grained detail to define structure and behavior often with domain-specific languages. It provides precision in resource measurements and monitoring interactions, dealing with properties of the physical world (voltage, transmission delays).

Higher-level semantics imposes lower overhead and addresses more sophisticated behavior described with more abstract rules, yet defined programmatically, e.g., neighbor-sets and routing in simulation of peer-to-peer overlay topologies (Peersim, Oversim[BHK07]), and distributed computing running grid middleware (SimGrid[CLQ08] , GridSim[BM02]). Widespread tools use serial code to manage event queues limiting simulation size and complexity. Testbeds do allow parallel execution distributed over several nodes. Nonetheless, overhead is orders of magnitude higher (few rules vs. millions of instructions) in practice precluding simulation size and complexity. Our project differs from these by combining higher-level semantics with parallel simulation distributed over actual overlay infrastructures.

B) Parallel/distributed computing

Concurrent activities are implemented as processes, threads or tasks cooperating with data sharing and coordination. Coordination ranges from stricter shared-memory (and DSM), looser message-passing (MPI[F95]) to rarefied Bags-of-Tasks (OurGrid [BAV+07]). Activities may share data globally (DSM), partially in communication (MPI, MapReduce) or never in BoTs.

Sharing eases programming as it follows well-known programming models and avoids explicit communication. Still, it needs enforcement of consistency models, especially with caching/replication with distribution. Pessimistic approaches with locks/semaphores and DSM protocols [ZIS+97] have shown poor performance and little scalability. Optimistic approaches [SM05] allowing temporary inconsistencies have proven essential for performance and large scale. In Software Transactional Memory [HLM+03], consistency is enforced at commit. Eventual consistency [TTP+95] relies on ulterior reconciliation. Middle-ground approaches allow data divergence limited in time and scope [YV06], use locality-awareness [CWD+05], or both (our vector-field consistency [SVF07]). While cluster-based multiplayer online gaming is a key current scenario, network and overlay simulation is still unaddressed.

C) Cycle-providing infrastructures

Computing infrastructures have grown wider, both processing units and interconnect. Early parallel supercomputers comprised several CPUs (scalar, vectorial) linked by internal bus. Today, multi-cores equip cheaper, more compact multiprocessor computers. Previous increase of LAN speed fostered aggregating (even heterogeneous) computers into cluster systems (NOW[ACP95], PVM[S90]).

Beyond access to remote clusters or supercomputers, users access computational Grids [FK99] and shared/donated cycles (BOINC[AF06]). Grid access implies belonging to virtual organizations of institutions exchanging computational resources. Underlying middleware (Globus[F06]) handles all authentication, accounting and resource allocation. In cycle-sharing, idle CPU time of PCs over the Internet is used for data processing, e.g., biology, astronomy. Cycle-sharing is integrated with peer-to-peer techniques to federate clusters or build desktop grids (CCOF[LZZ+04]), OurGrid, our Ginger[VRF07]).

Denser powerful clusters were the advent of a utility computing business (Amazon EC2). infrastructures revolve around server clusters and virtualization techniques for on-demand execution

of several VMs in a single real computer. Users launch VMs and create virtual clusters. No infrastructure enables simulation tools for faster or more complex simulations.

D) Peer-to-peer overlays

Peer-to-peer is effective in file-sharing, audio/video streaming, content/update dissemnination, anonymity and cycle-sharing [ATS04]. P2P overlays are split in two dimensions: structure and centralization, with hybrids in both. Structured (Chord,Pastry,CAN,Viceroy) map content to node identification with locality gains, load-balancing, ensured and range-based lookup queries. Unstructured (Gnutella,FreeNet) allow unrestricted content placing and arguably tolerate node entry/exit churn (partially refuted in [CCR05]). Most P2P are mostly or fully decentralized.

Resource discovery [TTP+07] in P2P cycle-sharing finds computational resources: fixed (capabilities, applications) and transient (CPU/memory availability). Structured approaches adapt topologies to map resource descriptions/locations as searchable content either flat, layered or N-dimensional [CFC+03]. Unstructured also centralize in super-peers [PMB+05], trees to speed search and minimize message flood, gossip. Our Ginger currently is a hybrid of: structured for store, result caching, capability matching; unstructured for CPU availability, gridlet task forwarding in vicinities; super-peers for reputation and accounting. Peer-to-peer desktop grids have not been harnessed for larger-scale overlay and grid simulation.

E) Grid middleware

Grid initiatives have not only the goal to mediate access to remote computing resources but also try to integrate into portals the tools needed by researchers on specific scientific areas. For instance the LHC computing grid [L04] not only allows access to scientific data, processing power but also provides interfaces to simulation and analysis tools used by the high-energy physics community. Moreover, a relevant area is the development of user interfaces (Ganga[HLT+03]) to interact with the available tools.

In other scientific areas, other initiatives (Archer, NanoHub[KMB+08]) develop the simulation tools and provide portals for easy access to them, simulation configuration, execution results and public data. Usefulness for the scientific community of such grids (integrating tools and data under common interfaces) fosters a large educational potential. A corresponding initiative will have similar impact in research reproducibility in overlay [NLB+07] and grid communities.

3. Architecture

This proposal aims at attaining the overall goal of laying foundations that will contribute to step up research in peer-to-peer overlay and grid topologies to a new level. To achieve this, the project will tackle existing limitations regarding scale, efficiency and expressiveness by providing a new Grid4GridResearch, as described in this research plan.

Most present-day research in these areas is accomplished resorting to simulation tools. Simulation allows researchers to carry out experiments with more elements (e.g., participating sites, applications) and more sophisticated behavior than would be possible by resorting exclusively to manually deployed and user-driven applications. The results of these stress-test experiments enable researchers to argue and defend claims about their proposed overlay protocols, grid middleware and schedulers. In such competitive research areas, new ideas without strong results to back them up

simply will not deserve full credit, regardless of their intrinsic value. As the Internet grows bigger, thus must simulations increase in size and, consequently, also in complexity.

Nonetheless, there are three important challenges that correspond to present shortcomings causing drag in these areas, regarding not only research but also even teaching: i) serial nature of simulation tools, commonly involving global state and event queue manipulations, prevent the scale-up that could be provided by the increasing democratization of access to multi-core machines (e.g., dual-core laptops and quad-core desktops); ii) impossibility to execute simulations in distributed manner thereby preventing the scale-out that could be reached by engaging platforms for voluntary cycle-sharing, utility computing and grid infrastructures (in testbeds as PlanetLab full execution imposes an overhead that greatly limits simulation size), and iii) dominance of programmatic descriptions of researched and tested topologies, locked in specific programming languages and simulator API idioms.

The aforementioned shortcomings cause a number of problems that hinder the expansion of current overlay and grid research, both in depth, breadth and community impact. Examples of such include: limitation to size and complexity of the simulations; limited scope of results; inefficient employment of resources involved in experiments; lack of architecture/simulator-agnostic descriptions of protocols, middleware, and schedulers; very restricted ability to perform independent repeated research [NLB+07]; no uniform repository for reusability of such research; global absence of accessible teaching platforms.

To address the problems identified, we propose COGITARE attacking them both in breadth and depth. We intend simultaneously to contribute with novel scientific contributions providing solutions specific to the problems described, as well as to provide the research community with a both useful and usable simulation infrastructure integrated with a portal for easy access, configuration and sharing. Hopefully, contributions and results of the project will be also valuable for other simulation fields and communities.

The primary substrate for COGITARE is an extendable peer-to-peer overlay platform (the mesh) comprising possibly asymmetrical participant nodes, aggregating individual desktop peers as well as efficiently leveraging available time on server clusters, grid sites and utility computing time.

The COGITARE mesh will be extendable. It provides fundamental support for simulation deployment, result caching, data storage, and resource discovery and management for cycle-sharing and distributed scheduling. On top of these fundamental mechanisms, more high-level services will be deployed such as retry-based fault-tolerance, replication-based reliability, check-pointing and migration to ensure progress in long simulations, and more global evaluation aspects such resource recycling, accounting and quality-of-service. Though challenging, it must be noted that such high-level services will be implemented at the overlay/simulator-level. Therefore, the associated software development effort is much lower than implementing similar mechanisms within an operating system or virtual machine runtime.

Simulations of peer-to-peer overlay and grid topologies will be parallelized and distributed. What is now mostly a single main task in current simulation tools must be decoupled in order to allow concurrent progress of (partial) simulation of different regions of the simulated topologies. This allows speed-ups due to parallel execution and increases scalability by enabling the mesh to host simulations of possibly unbounded size and interaction complexity. Most events localized in the vicinity of a simulated node cannot interfere with other simulated nodes many hops afar.

Nonetheless, consistency must ultimately be enforced to ensure simulation accuracy and correctness. In order to achieve this, we plan to partition the global simulation state and investigate the adaptation of optimistic consistency algorithms, such as divergence bounding and more recent vector-field consistency [SVF07] employed. We will incorporate notions of locality-awareness, present in most of current middleware architectures for massive multiplayer games, into the overall consistency enforcement of the decoupled partial simulations. Hard synchronization needs only to occur when events interfere with two or more of such simulated regions.

Over time, execution of simulations over the mesh, together with the expectable fluctuations in the mesh membership, will trigger higher-level services such as migration in order to restore load-balancing, and to improve performance by aggregating simulated regions with more intensive intercommunication within neighboring mesh nodes, or even within same node if sufficient capability is available there.

To overcome the lack of expressiveness in current simulation tools, regarding the definition of simulations, we propose a domain-specific language that allies greater expressiveness with the ability of reuse for teaching, study, and repeated research. Such topology modeling language (TML) will allow the specification of simulation requirements, flat, layered, multidimensional, hierarchical and novel recursive topologies to simulate arbitrarily large systems.

TML will encompass rules to specify: entry, exit, routing and recovery protocols; neighbor tables; indexing and data stored at nodes and how/where to retrieve it.

Simulated topologies will not be limited to content sharing. They could themselves be simulated cycle-sharing topologies as it is common in grid infrastructures. Therefore, TML will allow definition of policies regarding resource discovery, reservation, scheduling, accounting, recycling, redundancy, versioning, etc. to be enforced within each simulation. At its core, COGITARE will be a recursive overlay topology comprising its substrate and all the simulated topologies in a true Grid4GridResearch. TML and its templates will be the empowering means to allow simulated topologies to define and explore new structures, behaviors, strategies and their easy widespread sharing.

To ease the deployment of COGITARE, its clients will be distributed using a number of alternatives all of them incorporating the same message and simulation protocol to ensure portability and integration. Though clients could be stand-alone applications to be run on desktops or as jobs on a Grid (for maximum performance), we intend instead to develop different versions of space-efficient virtual appliances thereby easing deployment and taking advantage of concurrent simulation activities: middleware targeting multi-core equipment (MPI, STM), cluster infrastructures (distributed virtual machines as Terracotta, MPI), grid middleware and utility computing infrastructures (bundled virtual appliances to run as a virtual cluster). To keep the project within scope, the main emphasis will be on fundamental parallelization and distribution of simulations over essentially voluntary peer-to-peer cycle-sharing overlays. However, we consider relevant to COGITARE's acceptance the ability to interface with other relevant current infrastructure models to harness all resources made available to it. Regarding proper valuation of both free and paid computing cycles, we will ponder the definition of exchange rates to properly account for grid memberships and utility computing time.

Finally, we want to extend available open-source groupware and content management systems to create a portal, for groups and communities to interface with COGITARE and interactively share topologies, simulations, results. An XML-RPC or REST-based interface enables automatic deployment and retrieval of results promoting embedding of COGITARE in web pages and mash-ups.

4. *Mesh* - Overlay Infrastructure Design

This task will address the design of the primary substrate for COGITARE: the mesh. The mesh is a peer-to-peer overlay able to harness idle computing cycles from asymmetrical participant nodes. Thus, it will incorporate specific mechanisms and interfaces to engage peers, being them individual desktop, server clusters, grid sites and utility computing time slices. Each type of peer has different capabilities and a dedicated module will be able to access them (invoking corresponding middleware) and exploit them (representing them as higher capacity nodes) by giving higher simulation loads.

The COGITARE mesh will be extendable. It serves only to provide fundamental support, as mentioned, for: simulation deployment, result caching, data storage, and resource discovery and management for cycle-sharing and distributed scheduling. The mesh will obey to a hybrid and hierarchical structure. In this sense, the mesh behaves as recursive overlay topology.

Regarding content and routing, the mesh is a structured overlay. Each top-level data item stored in the mesh is in itself either: topology description, description of resource discovery and management policies, simulation setting, or results of simulation execution. Topology descriptions encompass node population, routing and communication protocol, and content placement rules. Resource and management policies to be enforced include: description of resource discovery, reservation, scheduling, accounting, recycling, redundancy, and versioning. Simulation setting encapsulates a topology and resource and management policy. Simulation results include the outcome of simulations and aggregate properties such as average and total messages sent, etc.

Regarding base-level services such as resource discovery (mainly w.r.t, CPU, memory, bandwidth) and scheduling of (partial) simulations, the mesh will employ automatically designated super-peers to act as hubs of information about resources of groups comprising their neighboring nodes. Super-peers will aggregate into a structured overlay only to exchange resources when there are not enough within each group.

The result of this task will be the base peer-to-peer cycle-sharing overlay structure and protocol to store and execute tasks of overlay and grid topology simulations. It will contribute with a novel structure and resource discovery mechanism tailored to its goal.

5. *Simulator* - Parallel and Distributed Topology Simulator

This task will handle the parallelization and distribution of peer-to-peer overlay and grid simulators that are currently of serial execution, i.e., a single main task manipulating a global state and managing a master event queue. After electing a code base from a current simulator of each type, we will decouple its main task, partition its global state and incorporate concurrency in the simulator's code.

Each concurrent simulator thread will be in charge of executing a number of simulation steps to simulate a specific region of the topology, i.e., it will perform a partial simulation. This way, events localized in the vicinity of a simulated node cannot interfere with other simulated nodes at a distance preventing them from being influenced by it (i.e., out of the event – message – horizon for a given number of hops) until some time has passed. Partial simulations may run in the same or different nodes. This will favor migration later.

This is intended to leverage notions of locality-awareness employed in most massive multiplayer games that while guaranteeing global consistency, employ region-based approaches to speed up most of the game interaction. The amount of time or rounds that mandates synchronization among regions to ensure safety when interactions cross region boundaries will be enforced resorting to novel optimistic consistency algorithms, adapted from existing divergence bounding and vector-field consistency [SVF07].

Each node of the mesh will execute a component of the distributed simulation engine. Communication among nodes running partial simulations of one topology simulation will interact via a standard API that will either manage shared memory or exchange messages via the mesh. The simulation engine component will have different versions in accordance to the type of underlying infrastructure providing computing cycles. All versions of the simulation component will perform parallel (i.e., multi-threaded) execution targeting multi-core machines as even desktop computers can benefit from it. All versions will be deployable as virtual appliances.

Simulation components to execute on server clusters, grid sites and utility computing infrastructures will leverage widespread middleware, libraries and virtual machine support (e.g., shared-memory, STM, Terracotta, MPI, Globus, and Xen). We intend to investigate all of them but will select for implementation preferably one example representative for each of the categories.

In order to maintain proper balance with respect to scheduling across the mesh, in spite of the different capabilities of the nodes, we will use special template simulations in order to benchmark average simulation performance on each platform. With respect to interaction with hard currency (namely utility computing and possible grid membership fees), we will employ a simple arbitrage mechanism based of updatable exchange rates.

The result of this task will be a faster and more scalable overlay and grid topology simulator able to run in parallelized and distributed manner. It will run on top of the mesh and will allow executing simulations of unbounded complexity regarding size and interactions among simulated participating nodes.

6. *TML* - Topology Modeling Language Design and Implementation

This task will define a new domain-specific language (TML) designed to express everything in COGITARE: data, structure, behavior and policies. We will research how graph description languages can include behavior and data structures to be manipulated at each simulated node.

TML syntax will include overall definition of simulations and topology structure (e.g., flat, layered, multidimensional, hierarchical and possible novel recursive topologies). This way, TML will allow easy definition of arbitrarily large systems in comparison with today's cumbersome programmatic approach.

Regarding data structures inside simulated nodes, TML syntax will encompass rules to specify neighbor and routing tables. Concerning behavior, TML will be used to express protocols (e.g., entry, exit, routing, and recovery). TML also allows the definition of pre-existing data content placed at simulated nodes prior to simulation start.

At a higher level, TML will allow enrichment of simulations by separating topology structure and routing from higher-level policies such as resource discovery, reservation, scheduling, accounting, recycling, redundancy, versioning. Once defined, the description of a policy should be reusable in

other simulations. This way, definitions of topology-related structure, data, behavior and policy can be mixed-and-matched to increasing reusability, productivity and observability of studied properties.

The development of language syntax will follow an incremental approach to ensure overall soundness, completeness and minimality.

The result of this task will be TML parsing and interpreting environment tools. We intend to explore approaches based on byte-codes to speed up simulation execution. Such an approach will ensure compatibility across platforms hosting the mesh and the topology simulator. Furthermore, we will design a number of pre-built templates to function as a starting kit to easily create simulated topologies exploring new structures, behaviors, and strategies.

7. *Services* - High-Level Resource Management for the Mesh

This task is required to design the mechanisms necessary to ensure overall balance over time across the mesh.

Since the mesh is mostly comprised of voluntary desktop machines, node exit and failure will be frequent, not the exceptional event. Therefore, to ensure fairness and balance in resources used by running simulations and to prevent wasted efforts on node failure, a number of higher-level services are to be deployed on top of the basic mesh, although embodied simply by modules plugged-in the basic client bearing mesh protocol and simulator engine.

To ensure fairness, these modules will provide migration of partial simulations to perform load-balancing. Migration is also useful to attempt at co-locating sibling partial simulations in the same mesh node or neighboring nodes. This will prevent network communication among simulation engines and increase speed. Overall balance and fairness will be helped by global resource recycling and accounting to enforce similar quality-of-service across users.

To prevent wasted work, these modules will provide fundamental fault-tolerance and reliability support. Each partial simulation may be scheduled several times to different nodes in the mesh, its results gathered and compared in order to confirm their correctness. To ensure very long running simulations make any progress, partial simulations will be checkpointed periodically and their state stored in the mesh as regular simulation result data.

We intend to minimize software development effort by reusing and combining basic services offered by the mesh. The result of this task will be an extended mesh with better service with the properties mentioned.

8. *Integration* - Optimization of TML Simulations on Parallel Distributed Simulator deployed over the Mesh

This task defines an additional development phase in order fully integrate TML simulations over the mesh executed by the parallel and distributed simulation engines. It includes extensive analysis, profiling and benchmarking and will result in adjustments to simulator code, finetuning of the mesh overall structure and adaptations on the Topology Modeling Language.

We expect the effort related with this task to be reduced but we choose to honestly inscribe it in the timeline. The result of this task will be an improved version of COGITARE with better performance, availability and scalability.

9. *Grid4Grid* - Portal Development

In this task, we will extend open-source content management systems and group-ware to create the Grid4GridResearch portal.

The portal will support content sharing (topologies, simulations, policies, results) among users, groups, and simulation-related communities. The main part of the development effort will focus on allowing web interactive and automatic (scripted) interface with COGITARE. For that, we intend to develop XML-RPC or REST-based interfaces.

This will have the interesting outcome of allowing embedding of COGITARE in web pages and mash-ups, and automatic deployment and retrieval of results. The result of this task will be the creation of a truly topology simulation community portal providing integrated and universal access to content and resources to perform simulations.

Conclusion

This document describes a novel approach to grid and overlay network research leveraging distributed infrastructures and multi-core machines for increased simulation complexity and speed. We presented its motivation, background, current shortcomings and the core architectural concepts of the novel research proposed. This is an on-going effort to further our peer-to-peer overlay cycle-sharing topologies by providing a scalable, efficient and reliable simulation substrate for the very grid and overlay topologies developed by the research community.

Thus, grid and overlay simulation are improved due to: 1) increasing scalability of simulation tools with a novel parallel, distributed and decentralized architecture; 2) unleashing the power of idle CPU cycles over the Internet harnessing them as a desktop grid (on a peer-to-peer overlay); and 3) ease grid and overlay research with a framework for topology definition, dissemination, evaluation and reuse.

The infrastructure, simulation engine, topology modeling language, management services, and portal comprise a Grid4GridResearch platform.

References

[FFL05]
J. A. B. Fortes, R. J. Figueiredo, M. S. Lundstrom Virtual Computing Infrastructures for Nanoelectronics Simulation Proceedings of the IEEE 2005

[MCE+02]
Peter S. Magnusson, Magnus Christensson, Jesper Eskilson, Daniel Forsgren, Gustav Hållberg, Johan Högberg, Fredrik Larsson, Andreas Moestedt, Bengt Werner: Simics: A Full System Simulation Platform. IEEE Computer 35(2): 50-58 (2002)

[FBF+08]
R. J. O. Figueiredo, P. O. Boykin, J. A. B. Fortes, T. Li, J. Peir, D. Wolinsky, L. K. John, D. R. Kaeli, D. J. Lilja, S. A. McKee, G. Memik, A. Roy, G. S. Tyson Archer: A Community Distributed Computing Infrastructure for Computer Architecture Research and Education 2008

[IH09]
T. Issariyakul, E.Hossain Introduction to Network Simulator NS2 ISBN: 978-0-387-71759-3 Springer Books 2009

[BHK07]
Ingmar Baumgart, Bernhard Heep, Stephan Krause OverSim: A Flexible Overlay Network Simulation Framework, Proceedings of 10th IEEE Global Internet Symposium (GI '07) in conjunction with IEEE INFOCOM 2007 2007

[CLQ08]
H. Casanova, A. Legrand, M. Quinson SimGrid: a Generic Framework for Large-Scale Distributed Experiments 10th IEEE International Conference on Computer Modeling and Simulation. 2008

[BM02]
R. Buyya, M. Murshed GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing Concurrency and Computation: Practice and Experience (CCPE), Volume 14, Issue 13-15, Wiley Press 2002

[F95]
I. Foster Designing and Building Parallel Programs (chapter 8 Message Passing Interface) ISBN 0201575949 Addison-Wesley 1995

[BAV+07]
F. Brasileiro; E. Araujo; W. Voorsluys; M. Oliveira; F. Figueiredo; Bridging the High Performance Computing Gap: the OurGrid Experience CCGRID 2007 Seventh IEEE International Symposium on Cluster Computing and the.. 2007

[ZIS+97]
Yuanyuan Zhou, Liviu Iftode, Jaswinder Pal Singh, Kai Li, Brian R. Toonen, Ioannis Schoinas, Mark D. Hill, David A. Wood Relaxed Consistency and Coherence Granularity in DSM Systems: A Performance Evaluation. sixth ACM SIGPLAN symposium on Principles and practice of parallel programming 1997

[SM05]
Y. Saito, M. Shapiro Optimistic replication. ACM Comput. Surv. 37(1): 42-81 2005

[HLM+03]
Maurice Herlihy, Victor Luchangco, Mark Moir, and William N. Scherer III. Software Transactional Memory for Dynamic-Sized Data Structures. Proceedings of the Twenty-Second Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing 2003.

[TTP+95]
Terry, D.B.; Theimer, M.M.; Petersen, K.; Demers, A.J.; Spreitzer, M.J.; Hauser, C.H. Managing update conflicts in Bayou, a weakly connected replicated storage system Proceedings of the fifteenth ACM symposium on Operating systems principles 1995

[YV06]
H. Yu, A. Vahdat The costs and limits of availability for replicated services. ACM Transactions on Computer Systems 2006

[CWD+05]
J. Chen, B. Wu, M. Delap, B. Knutsson, H. Lu, C. Amza Locality aware dynamic load management for massively multiplayer games. Proceedings of the tenth ACM SIGPLAN symposium on Principles and practice of parallel programming 2005

[ACP95]
T. E. Anderson, D. E. Culler, D. A. Patterson A Case for Networks of Workstations: NOW. IEEE Micro 1995

[S90]
V. S. Sunderam PVM: a framework for parallel distributed computing Concurrency: Practice and Experience archive Volume 2 , Issue 4 1990

[FK99]
Foster; C. Kesselman. The Grid: Blueprint for a New Computing Infrastructure Morgan-Kaufman 1999

[AF06]
D. P. Anderson and G. Fedak. The computational and storage potential of volunteer computing. In IEEE/ACM Intl. Symposium on Cluster Computing and the Grid May 2006.

[F06]
I. Foster. Globus Toolkit Version 4: Software for Service-Oriented Systems IFIP International Conference on Network and Parallel Computing Springer-Verlag LNCS 3779, pp 2-13, 2006.

[LZZ+04]
V. Lo, D. Zappala, D. Zhou, Y. Liu, and S. Zhao Cluster computing on the fly: P2p scheduling of idle cycles in the Internet. In The 3rd International Workshop on Peer-to-Peer Systmes (IPTPS'04) 2004

[ATS04]
S. Androutsellis-Theotokis, D. Spinellis A Survey of Peer-to-Peer Content Distribution Technologies ACM Computing Surveys, vol. 36 2004

[CCR05]
M. Castro, M. Costa, A. Rowstron Debunking some myths about structured and unstructured overlays Symposium on Networked Systems Design & Implementation - Volume 2 2005

[TTP+07]
P. Trunfio, D. Talia, H. Papadakis, P. Fragopoulou, M. Mordacchini, M. Pennanen, K. Popov, V. Vlassov, S. Haridi Peer-to-Peer resource discovery in Grids: Models and systems Future Generation Computer Systems archive Volume 23 2007

[CFC+03]
M. Cai, M. Frank, J. Chen, P. Szekely MAAN: A multi-attribute addressable network for Grid information services Proc. 4th Int. Workshop on Grid Computing, GRID 2003, 2003

[PMB+05]
D. Puppin, S. Moncelli, R. Baraglia, N. Tonelotto, F. Silvestri, A Grid information service based on Peer-to-Peer, Proc. 11th Euro-Par Conf. Euro-Par 2005 Springer 2005

[L04]
M. Lamanna. The LHC computing grid project at CERN. In Proceedings of the IXth International Workshop on Advanced Computing and Analysis Techniques in Physics Research 2004.

[HLT+03]
K. Harrison, W. T. L. P. Lavrijsen, C. E. Tull, P. Mato, A. Soroko, C. L. Tan, N. Brook, R. W. L. Jones GANGA: a user-Grid interface for Atlas and LHCb in Proceedings of Computing in High Energy and Nuclear Physics, La Jolla 2003

[KMB+08]
G. Klimeck,M. McLennan, S. P. Brophy, G. B. Adams III, M. S. Lundstrom. nanoHUB.org: Advancing Education and Research in Nanotechnology. Computing in Science and Engg. 10, 5 (Sep. 2008), 17-23. 2008.

[NLB+07]
S. Naicken, B. Livingston, A. Basu, S. Rodhetbhai, Ian Wakeman, Dan Chalmers The state of peer-to-peer simulators and simulations. Computer Communication Review 2007

Previous Work

[SVF08]
João Nuno de Oliveira e Silva and Luis Veiga and Paulo Ferreira, Heuristic for resources allocation on utility computing infrastructures (selected for special issue of Wiley Concurrency and Computation: Practice and Experience), ACM/IFIP/USENIX 9th International Middleware Conference (6th International Workshop on Middleware for Grid Computing - MGC 2008), Dec. 2008, ACM

[VPF07]
Luis Veiga and Paulo Pereira and Paulo Ferreira, Complete Distributed Garbage Collection using DGC-Consistent Cuts and .NET AOP-support, IET Software, 1(6), pp. 263-279, Dec. 2007, IET - The Institution of Engineering and Technology.

[SVF07]
Nuno Santos and Luis Veiga and Paulo Ferreira, Vector-Field Consistency for Ad-hoc Gaming (best-paper award), ACM/IFIP/Usenix International Middleware Conference (Middleware 2007), Sep. 2007 , Springer.

[VRF07]
Luis Veiga and Rodrigo Rodrigues and Paulo Ferreira, GiGi: An Ocean of Gridlets on a `Gridfor-the-Masses´, IEEE International Symposium on Cluster Computing and the Grid— CCGrid 2007, May. 2007, Rio Janeiro, Brazil, IEEE Computer Society Press

[GRF06]
Pedro Gama, Carlos Ribeiro, Paulo Ferreira: Heimdhal: A History-Based Policy Engine for Grids. International Symposium on Cluster Computing and the Grid (CCGrid 2006), 16-19 May 2006, Singapore. IEEE Computer Society

Authors Info

Our team includes researchers from the Distributed Systems Group at INESC ID Lisboa having the expertise and track record to pursue the goals set out in this proposal. The team has a combined publishing result spanning several areas involved in the proposal: peer-to-peer overlay and cycle-sharing, parallelized simulation (for Distributed Garbage Collection), parallel and concurrent processing, distributed resource management (algorithms for DGC) replication and optimistic consistency protocols, grid and utility computing middleware, and fault-tolerance.

Team members hold or are working towards Ph.D. degrees with theses proposals integrated in projects funded by FCT and international research labs (MS Research: OBIWAN and Rotor-DGC) whose main architect and developer was the Principal Researcher (Luís Veiga) during his Ph.D. studies, defended 2007.
This is illustrated by the list of recent selected publications (IEEE,ACM,IET,Wiley) by the team including: a journal paper on distributed garbage collection for application-level virtual machines in clusters[VPF07], an awarded paper describing novel vector-field consistency protocol[SVF07], a paper on peer-to-peer overlay cycle-sharing[VRF07], a paper on resource allocation in utility computing[SVF08] selected for journal issue, and a paper on declarative policies for grid resource management[GRF06].

Regarding education, the team includes current and previous lecturers of courses on Distributed Systems (replication, distribution, coordination and overlay topologies) and Operating Systems (parallel programming) supervising master and Ph.D. theses within the Distributed Systems Group: Luís Veiga, Paulo Ferreira, Carlos Ribeiro, Rodrigo Rodrigues. Luís Veiga has recently designed and chaired a novel master degree course on Virtual Execution Environments (VM principles, internals and deployment) submitted to and approved by the CS Department Coordinating Board at IST.

Full Team: Luís Manuel Antunes Veiga, João Nuno de Oliveira e Silva, João Coelho Garcia, Carlos Nuno da Cruz Ribeiro, João Pedro Faria Mendonça Barreto, Nuno Miguel Carvalho Santos, Paulo Alexandre Leal Barros Pereira, Jorge Pires Ferreira, Pedro Miguel Fernandes Sampaio, Pedro Miguel Silva Gama, Rodrigo Seromenho Miragaia Rodrigues, Rui Filipe Lopes Joaquim.