Derivative pricing as a business grid application using NextGRID technology

A. Basermann, G. A. Kohring & C. Neff
NEC Laboratories Europe,
IT Research Division NEC Europe Ltd, Rathausallee 10,
D-53757 Sankt Augustin, Germany

Abstract

The focus of the option valuation application described in this paper is on complex derivatives like American style options on multiple underlyings. The MPI-parallel and vectorized pricing code developed is based on Monte Carlo and Quasi Monte Carlo simulation methods. Within the European Commission Project NextGRID, whose primary goal is to develop architectural principles for the Next Generation Grid, experiments with derivative pricing scenarios have been performed that test the viability and adequacy of the NextGRID security model as well as the interoperability of NextGRID components. The main benefit of the usage of NextGRID principles and components for the derivative pricing application itself is that it supported the transition from a mere high performance computing Grid application to a business Grid application much better reflecting the functional separation and different levels of expertise of the parties involved. Based on SLAs, it is now possible to make the selection of optimal hardware transparent to the users. The NextGRID security model ensures security and integrity of the data, in a framework that is flexible enough to be adapted to cases where Grid resources are added dynamically.

Keywords: derivative pricing, NextGRID, security, SLAs, business Grid application, performance.

1 Introduction

Both the steadily increasing number of traded financial derivatives and the demand for more and more complex derivatives tailored for special purposes require continuously increasing capacities for the numerical valuation of these

WIT Transactions on Information and Communication Technologies, Vol 41, © 2008 WIT Press
www.witpress.com, ISSN 1743-3517 (on-line)
doi:10.2495/CF080061

financial instruments. The price estimation for these complex products is already demanding, but their constant revaluation within the financial institutions' risk assessment, hedging and portfolio optimization processes can increase the computational resource requirements significantly. Regulation authorities amplify the need for fast and reliable computing capacities by imposing higher standards for availability, system integrity and security on financial institutions. At the same time, financial institutions feel the pressure to cut development times and costs for the introduction of new products and the need to reduce their IT costs. Grid-based solutions will become more and more important in this sector, since they offer a higher utilization of existing resources and offer the additional possibility to increase the computational capacity temporarily by adding external resources in the case of demand peaks.

For the realization of the derivative pricing service discussed in this paper NLE-IT developed a portable Monte Carlo and Quasi-Monte Carlo simulation [1] based pricing toolkit that enables the estimation of derivative prices and sensitivities for European- and American-style options. The parallel simulation toolkit has been tested on a variety of different architectures ranging from NEC SX vector computers to PC Clusters, providing a set of basic random number and low-discrepancy generators, underlying models and product templates [2]. It is designed to be easily extendable to new product templates and underlying models, which can be integrated internally or invoked by the toolkit.

The main requirements for the derivative pricing application in the context of a distributed computing environment are small response times, high service availability, interactive calculation modes for trading purposes, interfaces to sources for market data as well as privacy and integrity of the data.

Within the European Commission Project NextGRID [3], whose primary goal is to develop architectural principles for the Next Generation Grid, use cases with a range of likely financial business scenarios have been composed. NLE-IT´s portable derivative pricing code used for the investigations within NextGRID was controlled via an Excel Client on a standard PC. Excel was chosen as interface due to its wide-spread use in financial institutions and its capabilities in pre- and post-processing of the data. Experiments with derivative pricing scenarios have been performed that test the viability and adequacy of the NextGRID security model as well as the interoperability of implementations of NextGRID components on Windows/.NET and on Linux. At the same time, both the performance of available implementations of NextGRID components and potential bottlenecks in NextGRID specifications were examined.

2 NextGRID application architecture

The overall architecture of the derivative pricing application as mapped onto the NextGRID nomenclature is depicted in Figure 1 (cf. [4, 5]). The components coloured green are application independent NextGRID components, while those coloured white are application specific.

As NextGRID takes a business oriented approach to Grid computing, the package groups are geared towards delivering business needs. As can be seen

from Figure 1 the packages group components according to their expected use within a business relationship.

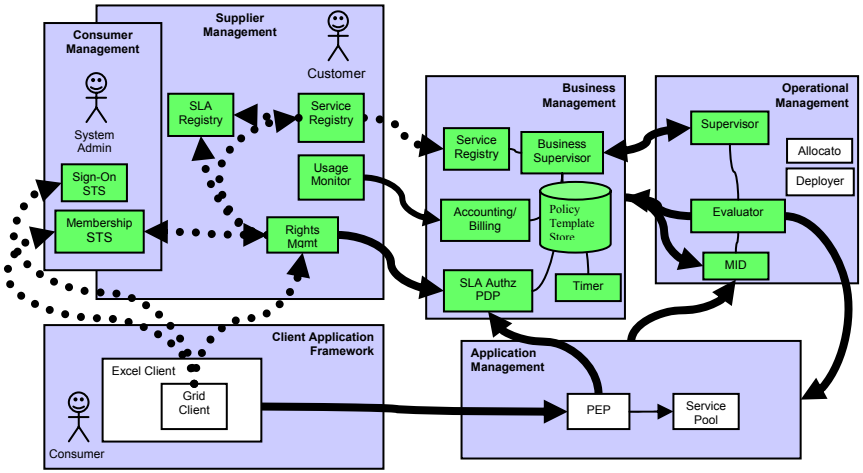

Figure 1: Schematic view of the derivative pricing application's architecture mapped on to the NextGRID nomenclature.

The Supplier Management package is used by customers to manage their relationships with one or more service suppliers. The components in this package allow the customer to keep a local registry of approved services and SLAs (Service Level Agreements), as well as offering the ability to monitor the consumption of resources. Additionally, the Rights Management component allows the customer to inform the service provider as to who in the customer's organization is allowed to use the services on offer; this fine grained resource allocation can be broken down to the level of individual operations on any given service. The components in this package are deployed at the consumer's site.

The Consumer Management package contains components which enable the consumer to control access to the Grid infrastructure to particular members of the consumer's organization.

The client application framework and the application management package contain the application specific components sitting at the consumer and provider sites respectively. While the NextGRID software provides support for these components, they were developed within the scope of this work to the particular demands of this application. In particular, .NET bindings were produced on the application side in order to integrate the Grid client into an Excel worksheet.

The Business Management package is deployed at the service provider's site and enables the provider to manage business relationships with multiple consumers. It contains a service registry detailing the services and their related SLAs which are on offer, as well as components for handling accounting and billing functionality. Since NextGRID provides an SLA driven Grid dynamic, a mapping is established between the terms of the SLA and policies governing

how resources are to be allocated and services deployed. These mappings are stored in the Policy Template Store component and provided to the Operational Management component through the Business Supervisor component.

It is the Operational Management package which provides the coupling between the SLAs governing the usage of the service and the services instances themselves. This is an event driven package, in which the MID (Management Information Distribution) component accepts events coming from other packages and distributes them to instances of the Evaluator component. Each Evaluator component examines the events and decides whether to allocate new computational resources and deploy new services corresponding to a given SLA, or whether to de-allocate previously allocated resources. The exact allocation and deployment mechanisms are site dependent. The Supervisors role is to interact with the Business Supervisor and to deploy the policy files associated with SLAs to the Evaluator and MID components. Basically, the policy files tell the Evaluator and MID how to react to particular events.

3 Derivative pricing experiments with NextGRID components

3.1 Security model

This experiment employed the basic NextGRID security model [5] as shown in Figure 2. Not every user in the consumer's organization is known explicitly by the service provider, rather a trust relationship is established between the Consumer Management component and the Business Management component. Through the use of the Rights Management component it is possible for the consumer to assign attributes to each user and instruct the provider how to interpret the attributes for use in making authorization decisions. Using this approach, the provider does not need to keep a list of which users are to be allowed access to the managerial interfaces and which are only to be allowed access to the normal compute interface. This system also allows a manager on the consumer side to decide which SLA any given user is allowed to use when accessing a given service.

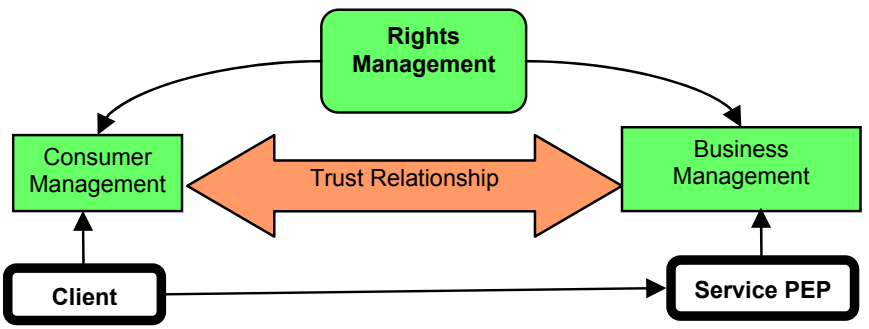

Figure 2: The security model employed for this experiment.

The sequence of events for sending and processing a request is as follows: When the consumer is ready to send a request to the service, the client software first obtains a security token from the Consumer Management component through the NextGRID Endpoint Security infrastructure [5]. This token is attached to the request inside the federation header, which contains information pertaining to the SLA under which the request should be processed along with other billing information, and then sent to the server. At the server side, the federation header is examined, the necessary information is extracted and passed to the Business Management component for verification and validation and only if the Business Management component returns a positive response the request is allowed to proceed to the pricing service for evaluation. The call to the Business Management component is a NextGRID specific feature and represents the main innovation in the security architecture compared with other state-of-the-art frameworks.

3.2 Experiment setup

The physical setup for the derivative pricing experiment depicted in Figure 3 was chosen to be as realistic as possible in that there are firewalls between the user and the service provider; furthermore the application server is protected from the internet gateway machine via a second firewall.

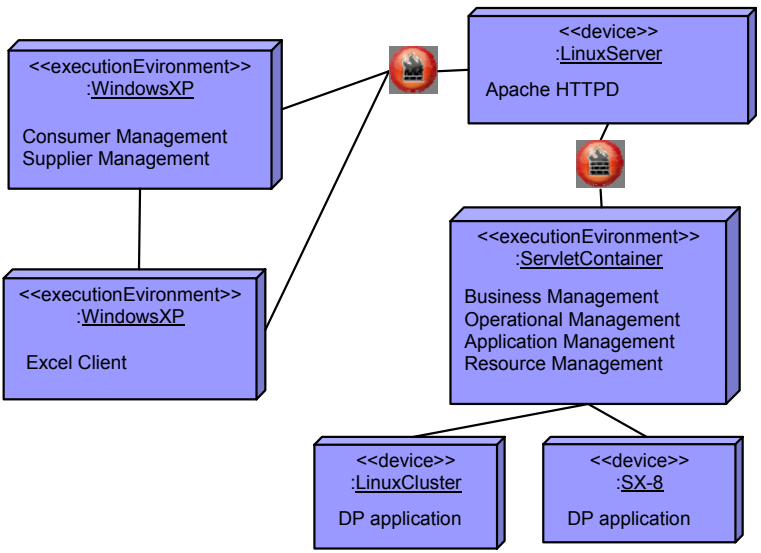

Figure 3: The physical setup for the derivative pricing experiments.

In a more complete scenario there would also be a firewall between the end-users machine running the Excel Client and the machine running the client side Consumer Management services; however, resource limitations for this experiment did not allow an exploration of such a scenario.

There were two compute systems used for the experiments, a Linux cluster consisting of 32 dual AMD Athlon Processors interconnected through Myrinet network and an NEC SX-8 vector computer with 4 processors. Two different compute nodes were used since certain types of models run more efficiently on vector machines while other models are more efficient on scalar machines.

The Excel Client (cf. Figure 4) reflects the setup properly. The interaction with NextGRID components is based completely on .NET (implemented in C#). The client additionally makes use of VBA (Visual Basic for Applications) code serving for pre- and post-processing of the data. A number of callback routines and interfacing objects makes the interaction of the VBA legacy code with the NextGRID components possible. That this worked without problems is an important lesson, since VBA code running in Excel is widely used in financial institutions.

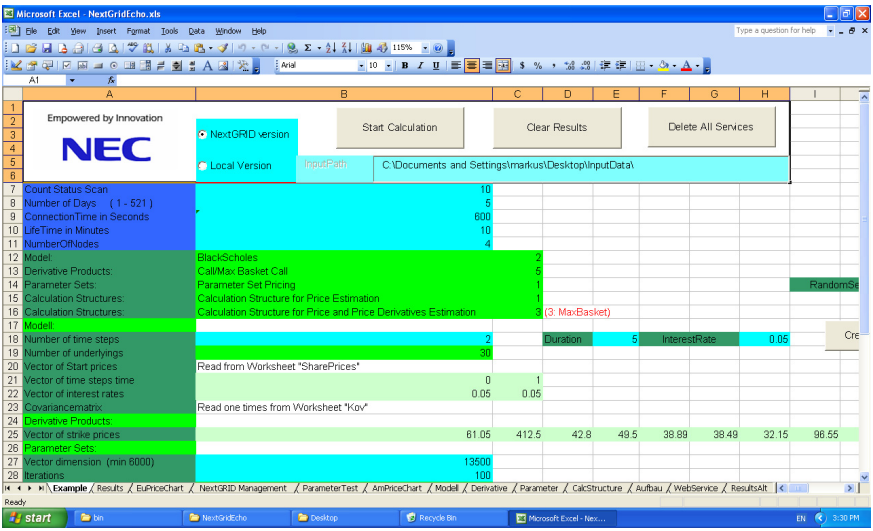

Figure 4: Control sheet of the Excel client.

3.3 SLAs

The first step in creating the new infrastructure involved designing the SLA templates by which the services would be governed. Initially three SLA templates were created, “bronze”, “silver” and “gold”. The “bronze” and “silver” levels represented relative performance increments on the Linux Cluster, while the “gold” level represented a service running on the SX-8. Later a fourth SLA was added which included a performance model to automatically determine the best computing platform for any given data set.

Regarding the NextGRID SLA schema [5–7], one can distinguish between functional and non-functional terms. The functional terms are mainly guarantees given by the provider covering performance and service availability, while the non-functional terms apply to both parties. The most important performance

criterion is the minimum raw computational power to be applied to processing the request, while the most important availability criterion is the daily percentage of up-time. One very important non-functional term concerns IPR (Intellectual Property Rights). The consumer agrees to indemnify the provider in the case where the consumer sends requests containing data for which he/she has not secured the proper permissions and the original data holder takes legal action against the provider. On the other hand, the provider agrees to indemnify the consumer in the case it has not secured the proper license for its pricing algorithms and the proper owner takes legal action against the consumer.

The next step in setting up the experiments involved installing the Business Management and Operational Management packages and configuring the two to work together. Doing so involved the creation of SLA policy files for the MID and Evaluator components (cf. Figure 1). These policy files are distributed by the Supervisor component when the SLA is deployed.

3.4 Service management

The use of the Operational Management package enabled a new approach to managing the application. Instead of using a one-user-one-service technique, a group of service pools was created, with one pool for each of the predefined SLAs. Figure 5 depicts the newly developed service management architecture.

When a new pricing request arrives the SLA under which it should execute is extracted from the message and a request for a corresponding service is made at the "Service Pool". If a suitable service exists and is not busy, then the service, or rather a proxy, providing an interface to the service is returned and the request executes normally. If no suitable service is available, or all services are currently busy executing requests for other users, then a "Service Not Available" response is returned to the "Switchboard". Where upon, the "Switchboard" puts the request on hold, issues a "Waiting" event to the MID component and after a waiting period which is SLA dependent, inquires once again at the "Service Pool" whether or not a suitable service instance is available.

If, when the "Switchboard" inquires after a service supporting a particular SLA, the "Service Pool" is unable to find such a service, it will issue a "Deploy Service" event to the MID. Upon receipt of a "Deploy Service" event, the MID will forward it to the Evaluator component which will trigger an "Action" causing the "Service Manager" to deploy a new service instance operating under the desired SLA. The next time the "Switchboard" inquires whether or not a service is available, the "Service Pool" will return the newly created instance.

Now it can also happen that all service instances are busy, in which case the "Switchboard" will send out repeated "Waiting" events. At some point the waiting time will exceed an SLA dependent threshold, and the next event will then trigger an action causing the "Service Manager" to deploy a new service instance (provided sufficient hardware resources exist).

Finally, it may happen that a particular service instance is idle. After a length of time, an "Idle" event will be sent by the "Service Pool" to the MID. Once the length of the idle period exceeds a given threshold, the next "Idle" event will trigger an action causing the "Service Manager" to retire the idle instance.

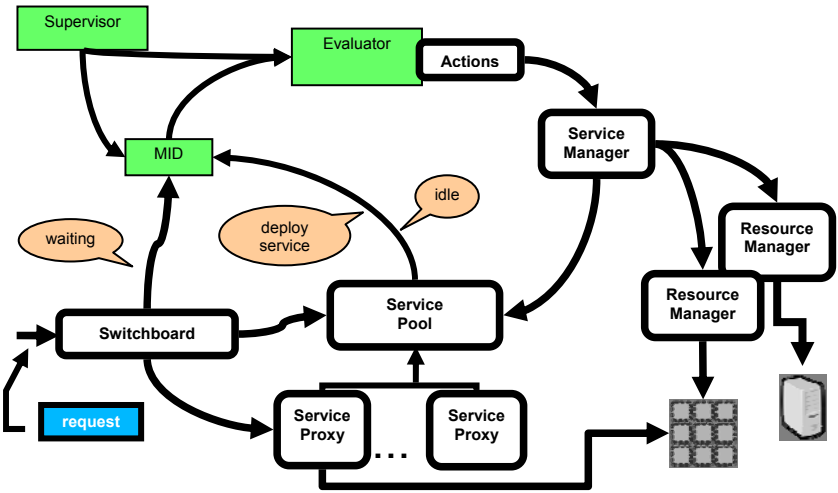

Figure 5: A view of the service management architecture.

3.5 Performance measurements

In [2], the scalability and performance of the MPI-parallel and vectorized derivative pricing code developed was discussed. As an example, Figure 6 shows the scaling behavior of the code for pricing of an American Max. Basket call option (Black Scholes model) on the maximum of five underlying stocks.

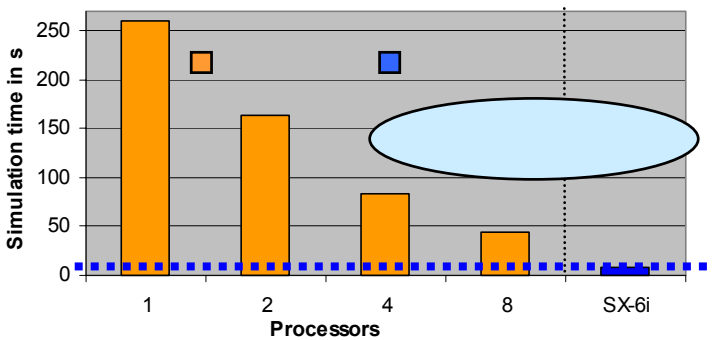

Figure 6: Scalability of the derivative pricing code.

The measurements were performed on a PC cluster with 2-way AMD Athlon 1900+ SMP nodes and on an ultra-compact NEC SX-6i vector processor ("micro-supercomputer"). Beside a good parallel efficiency due to the Monte Carlo approach, an advantageous cache exploitation on RISC processors and a high vectorization degree on vector machines provided an excellent per-

processor performance of the code. The latter explains the superior performance of the pricing application on the SX-6i in Figure 6.

One of the important criteria for the application scenarios discussed in this paper was performance in a distributed environment, in particular the penalty paid when accessing the application remotely. Remoting overheads were measured using NextGRID compliant security components and compared to the case where no security was in force. The results are shown in Figure 7.

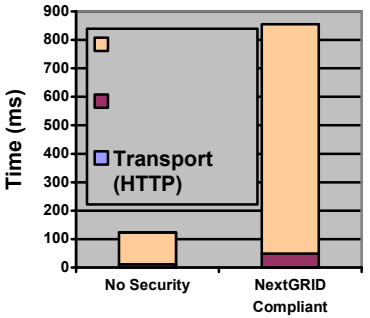

Figure 7: Remoting overheads per request for the derivative pricing service.

The "WS Overhead" is the complete overhead required for processing the Web Services related parts of the message, excluding any time required for the pricing calculation itself. As can be seen, the remoting overheads for using the fully NextGRID compliant security architecture are significant. In many usage scenarios, these overheads are acceptable, however there may be usage scenarios where they would not be. There is still room for optimization in this scenario. For example, if a security context was employed then one could avoid transmitting the SAML (Security Assertion Markup Language) token (which is approximately 18 KB) with every request. Hence, the NextGRID security model could be improved by integration of a more efficient token exchange approach such as that provided by the WS-Secure Conversation specification [8].

4 Conclusions

The Integration of NextGRID components into the derivative Pricing application enabled the transition from an initially purely high performance computing Grid application (with excellent scaling behavior and per-processor performance) to a business Grid application properly reflecting organizational structures.

The NextGRID SLA orientation inspired the implementation of new service levels where, based on a performance model, the optimal platform is selected from the input data. The Grid service can then be utilized by users without the technical background necessary to decide whether a pricing problem would run faster on how many nodes of a PC cluster than on a vector computer or special purpose hardware. For service providers this offers the additional advantage that they can adapt and optimize their resources to the needs documented in the SLAs and make better use of technological advancements. Other service levels may

allow access to specific hardware, such that this solution is flexible and can be easily adapted to specific needs in an organization.

Security is of vital importance in financial applications, and even intra-organizational access to data is subject to rigid restrictions controlled by regulation agencies. The NextGRID security model can be mapped to these restrictions, and NextGRID components allow secure implementations keeping the security issues mostly transparent to end users of the Excel Client. The performance of the NextGRID components used for secure transport is sufficient for many time-critical applications (market situations can change within seconds, and options in particular are very sensitive to changes). Approaches such as the use of WS-Secure Conversation could further reduce overheads.

Acknowledgements

This work was partly supported by the European Commission's IST research program under the contract 511563-NextGRID [3]. This paper expresses the authors' opinions and not necessarily those of the European Commission. The Commission is not liable for any use that may be made of the information contained in this document.

References

[1] Boyle, P., Options: A Monte Carlo Approach, *J. of Financial Economics*, **4**, pp. 323–338, 1977.

[2] Schumacher, J, Jaekel, U. & Basermann, A., Parallelization and Vectorization of Simulation Based Option Pricing Methods, *Proc. of ICCSA 03*, Springer, LNCS 2669, pp. 139–147, 2003.

[3] NextGRID, http://www.nextgrid.org/.

[4] Snelling, D., Anjomshoaa, A., Wray, F., Basermann, A., Fischer, M., Surridge, M. & Wieder, P., NextGRID Architectural Concepts, *Proc. of the CoreGRID Symposium in conjunction with the Euro-Par 2007*, Rennes, France, 2007, to appear (http://www.nextgrid.org/download/publications/NextGRID_Architectural_Concepts.pdf) .

[5] Snelling, D., Fischer, M., Basermann, A., Wray, F., Wieder, P., & Surridge, M., NextGRID Vision and Architecture White Paper V5, December 2005 (http://www.nextgrid.org/download/publications/NextGRID_Architecture_White_Paper.pdf).

[6] Mitchell, B. and McKee, P., SLAs A Key Commercial Tool, published in: *Innovation and the Knowledge Economy: Issues, Applications, Case Studies*, eds. P. Cunningham & M. Cunningham, IOS Press Amsterdam, 2005.

[7] Hasselmeyer, P., Koller, B., Schubert, L. & Wieder, P., Towards SLA-Supported Resource Management, *Proc. of HPCC-06*, Lecture Notes in Computer Science, Volume 4208, Springer, Munich, pp. 743–752, 2006.

[8] WS-Secure Conversation 1.3, http://docs.oasis-open.org/ws-sx/ws-secureconversation/ 200512/ws-secureconversation-1.3-os.html.