Support Technology for Model-Based Design Targeted at a Cloud Environment

YANOO Kazuo, FURUMAKI Kazuhisa, IZUKURA Sayaka, SAKAKI Hiroshi

Abstract

Even with system integration in the cloud computing, platforms need to be designed that are customized with non-functional requirements to meet the specific needs of each customer while still providing combined infrastructure and service. The effectiveness of model-based design can be enhanced in a cloud environment as the design process can be automated using software. This paper discusses model-based platform design support technology (CARDO) targeted at system integration in the cloud computing.

Keywords

design support, non-functional requirement, model-based

1. Introduction

Over the past few years, demand for cloud-based solutions and services has skyrocketed. Even with system integration in the cloud environment, platforms must be designed that are custom-tailored with non-functional requirements to meet the specific needs of each customer while combining infrastructure and services. The cloud environment can be expected to provide many advantages to model-based design thanks to its ability to automate the design process using software. This paper discusses model-based platform design support technology (CARDO) targeted at system integration in the cloud computing.

By integrating this technology - together with automated construction technology and configuration management technology - into our computer-aided system integration automation plant (CASSIOPEIA) - an environment created to increase the efficiency and quality of system integration in the cloud computing - our new service is able to immediately provide customers with high-quality platforms.

2. Characteristics of Platform Design in Cloud-Based System Integration

System requirements can be classified into functional and non-functional (performance, availability, security, etc.). In conventional system integration, the work is typically divided with application designers being responsible for functional requirements while platform designers handle the non-functional requirements.

As cloud vendors provide the basic IT infrastructure and services, many of the difficulties related to platform design in cloud-based system integration are significantly reduced. However, it still remains necessary to incorporate non-functional features according to user specifications while integrating them with the provided infrastructure and services.

The use of an auto-scale function reduces the need for strict performance design, but does not eliminate the necessity for performance design. It is still necessary to take into consideration middleware such as RDBMS, which is difficult to scale, as well as shared resources such as networks and storage. Even deciding whether or not a project should be started at all is not possible until the total cost has been determined by estimating the number and type of servers required.

The three main characteristics of platform design for cloud-based system integration are as follows.

(1) Difficulty of non-functional evaluation

Cloud vendors do not provide complete details about their platforms, making it more difficult to determine whether the required non-functional items can be addressed or not. As a matter of fact, market research has revealed that one of the biggest obstacles to migrate the cloud-based environment to their business systems is precisely this lack of certainty regarding non-functional requirements such as security, availability and performance.

(2) Designing under restricted conditions

In conventional system integration, there are no constraints on design. In the cloud computing, however, non-functional attributes are designed under restricted conditions as they must be designed using the infrastructure and services supplied by the cloud vendor.

(3) Construction using software

Hardware resources are virtualized and can therefore be controlled with software, so the results of platform design are not network block diagrams and set-up instructions for the staff to operate but scripts for computers to execute construction processing. When designing a platform for cloud-based system integration, it is essential to take these characteristics into consideration.

3. Model-Based System Integration

Platform design for cloud-based system integration is highly compatible with methods for model-based development.

Model-based system engineering (MBSE)[1] is an approach that achieves high-quality and efficient system development by defining a "model" that systematically expresses the structure and behavior of systems. The introduction of MBSE makes it possible to confirm the validity of design in advance by evaluating customer requirements on the model and to mechanically derive an optimal design that meets customer requirements.

Compared to application design, there are few differences in platform design from one project to the next. This makes it possible to keep costs down and improve quality by creating models and templates that can be used in multiple projects. NEC is also developing model-based system integration in which MBSE is applied to system integration.[2, 3]

When this method is applied to the actual process of system integration, the question is whether or not the results will match man-hours for creation and maintenance of models. Conventionally, the benefits of model-based system integration can be roughly divided into two: 1) cost reduction and quality improvement by reusing designs and 2) support for the design of non-functional features by using non-functional evaluation on a model. In cloud-based system integration, the following benefits can also be expected.

- **Additional value of automated construction**
  Because the construction of infrastructure is done by software in the cloud computing, construction can be automated by generating construction scripts from the model.
- **Improved value of non-functional evaluation**
  Non-functional evaluation on a model solves the problem of determining whether or not non-functional requirements can be incorporated in cloud-based systems.
- **Minimizes man-hours required for creation and maintenance of models**
  Because design is performed under restricted conditions that are dependent on the infrastructure and services provided by the cloud vendor, there is a limit on the objects that can be made into models. As a result, man-hours for creation and maintenance of models can easily be minimized.

As we have seen, cloud-based system integration can make a model-based design platform much more cost effective. Thus, we expect that model-based system integration will be more widely used in cloud-based system integration.

4. Model-Based Platform Design Support Technology

4.1 Outline

The CARDO model-based platform design support engine further advances conventional technology incorporated in the non-functional design support technology featured by facilitating non-functional evaluation on a model. This makes it possible to achieve semi-automation of non-functional designing.[4]

Platform design using models includes many configuration parameters. These can include the number of clusters, the specifications of virtual machines (the number of cores, etc.), redundancy methods, back-up methods, etc. The settings of these parameters can help determine which non-functional features can be implemented on the platform.

In-depth knowledge and skill is critical to assessing which non-functional features will be required to meet the specific characteristics of each project and to choose the optimum combination of setting parameters (the most cost-effective).

Once the non-functional requirements have been input, the CARDO calculates the combination of optimal setting parameters that meet those requirements. As a result, the platform designers can concentrate on evaluating the non-functional requirements.

4.2 Configuration

Fig. 1 shows the overall configuration of the CARDO. Based on the non-functional requirements input by the user, the lowest-cost configuration which meets the non-functional requirements is output after searching the setting parameters of the models stored in the repository.

Included in the models of platforms are non-functional parameters. These include specifications for non-functional evaluation, price information, and system configuration definitions, raw performance of servers and storage machines (SPECint values, maximum IOPS values, etc.) and MTTF (mean time to failure).

4.3 Non-functional Evaluation Engine

The non-functional evaluation engine evaluates performance and availability based on the platform model and the provided design parameters. After converting the performance evaluation into a layered queuing model and the availability evaluation into a fault tree or stochastic reward net model, evaluation values are worked out using a simulation or analytical calculation (**Fig. 2**).[2)]

4.4 Automatic sizing engine

The automatic sizing engine finds the lowest-cost configuration that meets the non-functional requirements. It generates setting parameters, inputs them to the non-functional evaluation engine, and processes these results while reevaluating whether or not those results will actually meet the non-functional requirements. This is generally called an optimization problem, and various solutions have been researched and developed.

The CARDO uses a method to search for the optimal solution by adopting either constraint programming (CP) or a branch and bound (BB) algorithm on a case by case basis. The availability requirements resulting in a fault tree uses CP because these requirements can be formulated in the form of a constraint logic programming algorithm. On the other hand, calculation of layered queuing uses BB because simulation is generally required for the calculation. When BB is used, appropriate heuristics are introduced, and approximate solutions found by analytical calculation is used as relaxation problems. Thus, a substantial increase in search speed can be achieved.

Table 1 shows the results of a test aimed at a relatively simple Web3 in which the numbers of simulations performed until an optimal solution was obtained were compared. For the comparison, the non-functional requirements were changed each time. The numbers of simulations were reduced significantly by the proposed methods.[5)]

4.5 Execution Examples

Now let's take a look at examples of platform design using the CARDO in a virtual cloud service. **Table 2** shows a list of

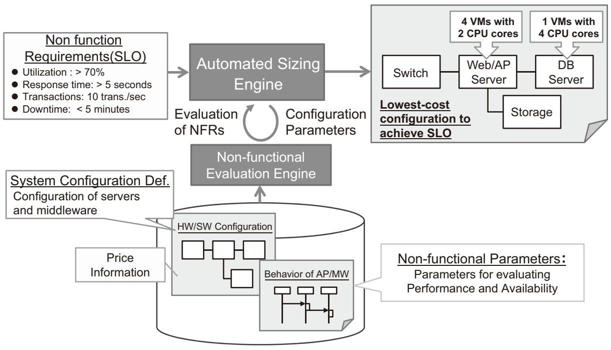

Fig. 1 Overall configuration of CARDO.

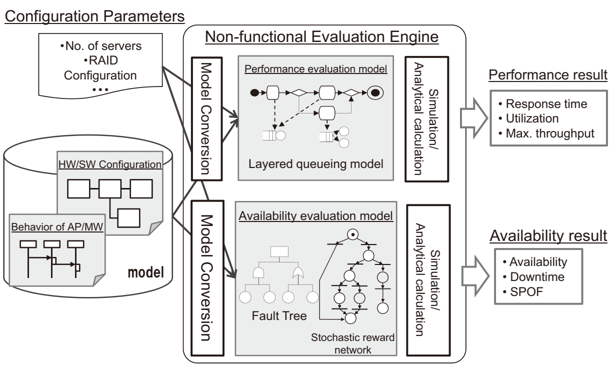

Fig. 2 Non-functional evaluation engine.

Table 1 Comparison of number of simulations.

Model	NFRs		Number of simulations		
	Throu-ghput	Response time	Exhaustive Search	Naïve BB	Proposed Method
1	10	0.1	2500	100>	7
		0.3	2500	100>	4
	40	0.1	2500	100>	8
		0.3	2500	100>	11
2	50	0.1	2500	100>	10
		0.3	2500	100>	5
	100	0.1	2500	100>	7
		0.3		-	

Table 2 Fictional cloud service

Menu item	Content	Price		
		Advanced	Standard	Economy
CPU resource	[Basic] CPU x 1	120,000	44,000	37,000
	[Optional] CPU x 2	33,000	25,000	13,000
	[Optional] CPU x 4	90,000	57,000	-
	Allocation policy of physical CPU	1 core	0.5 core	0.25 core
	Allocation policy of physical CPU physical CPU	5	3	1
Standby system type	Cold standby	10,000		
	Warm standby	23,000		
	Hot standby	40,000		-
File backup cycle	Monthly	15,000		-
	Weekly	25,000		
	Daily	35,000		
	Online	40,000	-	

Table 3 Example of input non-functional requirements and output results.

	Non-functional requirements					Result	
	CPU Utilization	Average Response time	Throughput	Business continuity	Data recovery point	Plan	Option
1	80% or less	3 seconds or less	10 t/s	-	-	Economy	CPUx2
2			20 t/s			Standard	CPUx4
3				Processing continued in single failure	Up until 5 business days	Standard	CPUx4 Hotstandby Weekly backup
4		5 seconds or less	50 t/s		Up until 1 business day	Advanced	CPUx2 Hot standby Daily backup

plans and options available from the fictional cloud service, while **Table 3** shows examples of design parameters derived as the output of the CARDO when four types of non-functional requirements were input. As shown in these figures, the lowest-cost configurations that meet non-functional requirements can be automatically obtained from many combinations of plans and options.

5. Conclusion

This paper has introduced the model-based platform design support technology (CARDO) aimed at facilitating system integration in the cloud. By automating the processing to search for the optimal design parameters based on non-functional requirements, it is expected that this will significantly reduce the amount of time and labor required for platform design. Moreover, this approach effectively solves the biggest problem inherent to cloud system integration, i.e., the difficulty in determining whether or not the cloud's infrastructure can properly support the required non-functional features. Combining this with automated system construction technology makes it possible to rapidly design and generate a suitable platform.

While this technology is still in the research and development stage, we are proceeding with practicality validation using our cloud infrastructure services such as the NEC Cloud IaaS and are planning to incorporate this in our computer-aided system integration automation plant (CASSIOPEIA). Through our model-based platform design technology, we are committed to continuing our efforts to provide our customers with high-quality platforms customized to meet their needs in the most timely manner possible.

Reference

1) S. Friedenthal, et al. "INCOSE Model Based Systems Engineering (MBSE) Initiative", International Council on Systems Engineering, 2007.
2) M. Aizawa et al. "System Model for OMCS Platform Development", IPSJ Transations on CDS, vol. 2, no. 3, 2012 (in Japanese)
3) S. Izukura, et al. "Applying a Model-Based Approach to IT Systems Development Using SysML Extension," 14th International Conference on Model Driven Engineering Languages and Systems (MoDELS), 2011.
4) S. Izukura, et al, A Method for System Design Supporting in Cloud Environments, IEICE Tech. Rep., vol. 114, no. 50, SC2014-3, pp. 11-16, May 2014. (in Japanese)
5) S. Izukura, et al. "Determining appropriate IT systems design based on system models", 37th Annual International Computer Software and Applications Conference, 2013.

Authors' Profiles

YANOO Kazuo
Principal Researcher
Knowledge Discovery Research Laboratories

FURUMAKI Kazuhisa
Knowledge Discovery Research Laboratories

IZUKURA Sayaka
Assistant Manager
Knowledge Discovery Research Laboratories

SAKAKI Hiroshi
Assistant Manager
Knowledge Discovery Research Laboratories

Information about the NEC Technical Journal

Thank you for reading the paper.
If you are interested in the NEC Technical Journal, you can also read other papers on our website.

Link to NEC Technical Journal website

Japanese | English

Vol.9 No.2 Special Issue on Future Cloud Platforms for ICT Systems

Vol.9 No.2
June, 2015

Special Issue TOP

NEC Information