# Performance Assessment Architecture for Grid

Jin Wu and Zhili Sun

Centre for Communication System Research, University of Surrey,
Guildford, GU2 7XH, United Kingdom
{jin.wu,z.sun}@surrey.ac.uk

**Abstract.** For the sake of simplicity in resource sharing, Grid services only expose a function access interface to users. However, some Grid users want to know more about the performance of services in planning phase. Problem emerges since the Grid simply has no means in obtaining how well the system will cope with user demands. Current Grid infrastructures do not integrate adequate performance assessment measures to meet the user's requirement. In this paper, the architecture of Guided Subsystem Approach for Grid performance assessments is presented. We proposed an assessment infrastructure that allows the user to collect information to evaluate the performance of Grid applications. Based on this infrastructure, a user-centric performance assessment method is given. It is expected that this research will lead to some sort of extension in Grid middleware to facilitate the Grid platform the ability to handle applications with higher reliability requirements.

**Keywords:** Grid, performance assessment, reliability.

## 1 Introduction

The lack of performance indications becomes an obstacle for the continuously promotion of the Grid. User's sceptics in service quality significantly hold back the efforts of putting more applications onto Grid: it is hard to convince users to transfer valuable applications onto the Grid infrastructure before they have been clearly notified with the service quality they will receive. Such a problem is brought forward when the Grid middleware hides system details by allowing users to access service through a portal without recognising any detail of resource providers. On one hand, a standardised interface provides an easy access to heterogeneous resources. On the other hand, the system details are hidden behind for users to recognize the performance of Grid services. This paper studies the performance assessment architecture to solve the problem. It collects the requirements from the user, and feedbacks how well the system will cope with the user's requirement after examine related Grid components. This architecture does not help to improve the performance of individual component, but assist the users to select services more wisely by enabling the comparison among available services providers on like-for-like bases.

Two approaches can be used to obtain the performance assessment of an application: the user initiated assessment, and the infrastructure based assessment. The user initiated assessment approach evaluates the performance of an application by users'

P. Primet et al. (Eds.): GridNets 2008, LNICST 2, pp. 89 – 97, 2009.

own efforts. The services from all providers are trialled one by one, and their performances are documented for like-for-like comparisons. While this approach is easy to use, its utility is generally limited by its overhead, accuracy, and management concerns. For instance, a large number of users making independent and frequent assessment trials could have a severe impact on the performance of services themselves. Furthermore, the assessment abilities at user-ends are always limited which weaken the accuracy of assessment results. More important still, application providers might not agree to open their systems for trials based on economical and security concerns. Ideally, dedicated modules attached to service providers should be in place while its assessment results could be made available, with low overhead, to all perspective users. This is the basic motivation of infrastructure based performance assessment for Grid services. Our paper extends the research by combing the above two idea together - an assessment infrastructure is in place to assist the application selection. The novelties of this research lie on: a) the assessment infrastructure has been extended to support performance assessment other than network performance metrics assessment; b) a general architecture considering both application layer and transmission network performances are proposed; and c) Grid application users will be notified with possible performance before their executions. The reminder of the paper are organised as follows. Section 2 gives the works relate to this research. Section 3 gives an architecture level study of the performance assessment. Section 4 gives the detailed procedures for the performance assessment identified in Section 3. Finally, Section 5 summaries this work.

## 2 Related Works

Current researches have limited emphasises on Grid performance assessments. Previous researches have discussed the use of infrastructure to obtain data transmission performance between Internet hosts. A simple protocol for such a service, SONAR, was discussed in the IETF as early as February 1996 [2], and in April 1997 as a more general service called HOPS (Host Proximity Service) [3]. Both of these efforts proposed lightweight client-server query and reply protocols similar to the DNS query/reply protocol. Ref [4] proposed a global architecture, called "IDMaps", for Internet host distance estimation and distribution. This work propose an efficient E2E probing scheme that satisfies the requirements for supporting large number of client-server pairs. However, the timeliness of such information is anticipated to be on the order of a day and therefore, not reflect transient properties. Furthermore, this work can only assess the transmission delay among Internet host, this could be an underneath support of performance assessment, but this work did not touch the performance issue related to users. The topic of server selection has also been touched. Ref [5] proposed passive server selection schemes that collects response times from previous transactions and use this data to direct client to servers. Ref [6, 7] proposed active server selection schemes. Under these schemes, measurement to periodically probe network paths is distributed through the network paths are distributed throughout the network. Based on the Round Trip Time that is probed, an estimated metric is assigned to the path between node pair. Ref [1] study a scenario where multimedia Content Delivery Network are distributing thousands of servers throughout the Internet. In such scenario, the number of tracers and the probe frequency must be limited to the minimum required to accurately report network distance on some time scale.

## 3 Architectural Level Study to Performance Assessment

An architectural level study of performance assessment is given in this section. Performance assessment is used to find out the degree of likeliness by which the performance of a Grid service fulfils the user's requirements. In many cases, multiple component services from different administration domains are federated into a composite service which is finally exposed to users. Needless to say, the performances of related component services affect the performance of the composite service that is directly visible to users.

The accuracy, efficiency, and scalability are main concerns for the performance assessment of Grid services. Generally, Grid could be large scale heterogeneous systems with frequent simultaneous user accesses. Multiple performance assessment requests must be employed to assess performance in a timely and overhead efficiency manner. Apparently, the larger and more complex the system, and the more varied its characteristics, the more difficult it becomes to conduct an effective performance assessment. Subsystem-level approach is a solution to overcome the scalability problem to performance assessment over large scale systems. As its name suggests, the Subsystem-level approach accomplishes a performance assessment by developing a collection of assessments subsystem by subsystem. The rationale for such an approach is that the performance meltdown that user experienced will show up as a risk within one or more of the subsystems. Therefore, doing a good job at the subsystem level will cover all the important area for the whole system. The following observations identify the limitations of the approach. Independent subsystem performance assessment tends to give static performance and assume in the way that the upcoming invoking request does not carry out a meaningful impact to the subsystem performance. But a subsystem can exhibit differently when the additional invoking request is applied. Without a dynamic performance assessment which takes the characteristics of both invoking request and subsystem, the performance assessment result has a systematic inaccuracy. Independent subsystem performance assessment is prone to variance in the way performance metrics are characterised and described. Without a common basis applied across the system, subsequent aggregation of subsystem results is meaningless or impossible to accomplish. A consequence is the inevitable misunderstanding and the reduced accuracy of performance assessment. Subsystem A's analysts can only depict the operation status and risk of Subsystem A, but it may not have an accurate understanding of Subsystem A's criticality to the overall service performance. As viewed from the perspective of the top level performance assessment, the result from subsystem can be wasted effort assessing unrelated performance metrics, or subsystem performance metric that crucial to the overall performance assessment is not measured.

Given the weakness of the conventional subsystem performance assessment approaches, we propose the Guided Subsystem Approach to performance assessment as an enhancement to conventional approaches.

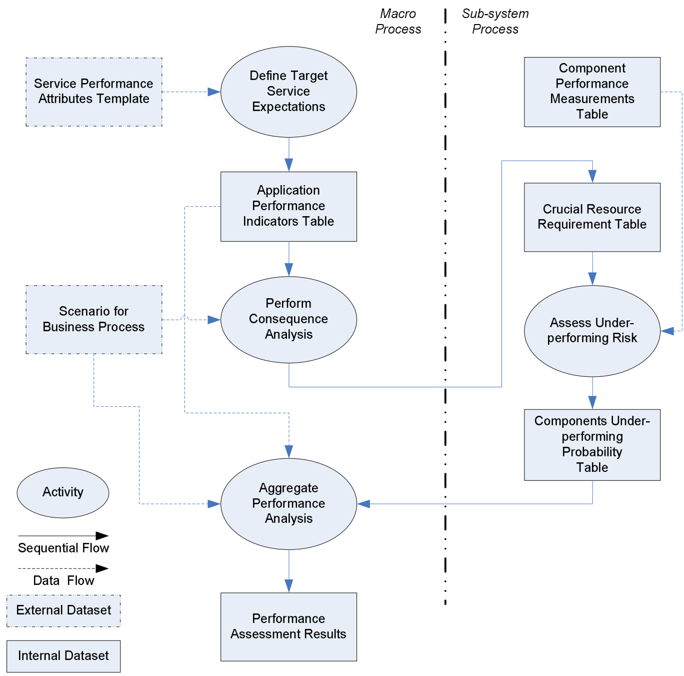


**Fig. 1.** Architectural Level Process for Guided Subsystem Assessment

The main idea of the Guided Subsystem Approach is to introduce a user-centric component that interprets the user's assessment requests and assessment function of subsystems. It conducts macro-level pre-analyse efforts. The top-down efforts are enforced to translate those performance assessment requests and transmit to the subsystem assessment modules. Such the dynamically generating of assessment requests to subsystem helps to prevent mismatching between user's assessment requirements and the assessment actions applied onto the subsystems. The Guided Subsystem Approach takes the advantage of both top-down and subsystem approaches while presenting better efficiencies, consistencies, and systematic accuracies. Furthermore, the Guided Subsystem Approach has top-down characteristics together with characteristics from subsystem. The user centric interpretation components and subsystem performance assessment components can be designed and implemented separately.

## 4 Guided Subsystem Performance Assessment

For an assessment infrastructure, it needs to give performance indications after users express their expectations to the target Grid service. Users can have diverse performance expectation for a Grid service. Their performance expectations should be expressed in a standardised way to let the assessment infrastructure learn the need of users. A performance indicator is defined to notify users the result of performance assessment. The performance indicator should have pragmatic meaning and allow user to perform like-with-like comparison between similar Grid services delivered from different providers.

For the purpose of measuring the ECU of a target Grid service, three types of inputs need to be identified:

- **Set of Performance Malfunctioning (SPM)** is a set containing any possible performance malfunctioning feature by which Internet application users might possibly be experienced. It is a framework of discernment attached to applications.
- **Consequence of Performance Malfunctioning (CPM)** measures the damages every particular performance malfunctioning feature could possibly cause. It depicts the users' expectations of performance towards the application.
- **Probability of Performance Malfunctioning (PPM)** measures the possibility of performance malfunctioning appears under a given framework of discernment. This input depicts the how the underneath system reacts to the invoking requests.

The ECU of a target Grid service can be given upon the collection of the above three aspects of inputs.

To obtain the *SPM* is the first step for *ECU* measurement. The performance malfunctioning is a feature of an application that precludes it from performing according to performance specifications, a performance malfunctioning occurs if the actual performances of the system are under the specified values. *SPM* is usually defined when a type of Grid service is composed. When a type of Grid service is identified, its standard *SPM* can be obtained. Different types of Grid services could lead to different *SPM*s. Basically, *SPM* is authored by system analysis, and could be, nevertheless, updated thanks to users' feedbacks. However, the management and standardisation of SPM is outside the scope of this research. Without loss of generality, we simply consider standard *SPM*s can be given by a third party generalised service when a use-case is presented, where the *SPM* of service *u* is denoted as $SPM_u$. For a service *u* with a standard *SPM* including *n* malfunctioning type, there exists $SPM_u = \{s_1,...,s_n\}$, which can also be denoted as an n-dimensional vector $\vec{SPM}_u, \vec{SPM}_u = (s_1,...,s_n)$. Identifying the severity of performance malfunctioning is another important aspect of *ECU* measurement. *CPM* is a set of parameters configured by users in representing the severity each performance malfunctioning feature could possibly cause. Define a function $p_{<user_x|SPM_u>}$ representing the *CPM* configuration process for the user $user_x$ on the framework of discernment of use-case *u*, $SPM_u$. Denote this *CPM* as a n-dimensional vector $\vec{CPM}_u = (c_1,..,c_x,..,c_n)$, where $c_x$ is the cost of damages when the $x^{th}$ performance malfunction exists. Then, for $\forall s_x \in SPM$, $\exists c_x \in [0,+\infty)$ satisfying $p_{<user_x|SPM_u>} : s_x \to c_x$. It is users' responsibility to configure the *CPM*. In many cases, policy based automatic configurations are possible in order to make this process more friendly to users. The third part of performance assessment is to find out the performance of the invoked system in terms of under-performing possibility when an application scenario is applied. The performance of the invoked system relates to the ability of physical resource that the system contains, and their managements. An *n*-tuple, PPM, is defined to measure the performance malfunctioning probability of a given function under a scenario. Denote $s_x$ as a performance metric satisfying $s_x \in SPM_u$.

For a use-case $u$, suppose the specified performance values as $SPM_u'$, where $SPM_U'=\{s_1',\ldots,s_h'\}$. The scenario $e$ is executed to apply the use-case $u$. $p_x$ is defined as the possibility of performance malfunctioning that exhibited by the $x^{th}$ performance metric, where $p_x=\Pr(s_x<s_x')$. Then, we have the probability of performance malfunctioning for scenario $e$, where $P\vec{P}M_e=(p_1,\ldots,p_n)$. The PPM related to three factors: a) the ability and usability of components; b) importance of components to overall performance; and c) invoking frequency of components. SPM and CPM can be given by the user, while PPM needs to be obtained from sub-systems that compose the target Grid service.

We consider a Grid service can be decomposed as many independent component Grid services, which actually contains software and hardware resources, and performs a set of functional activities. A set of metrics is used to measure the performance of a composite service. Let a data structure $<x,y>$ denote the $y^{th}$ performance metric of the component $x$. Let performance metrics $R_{<x,y>}(e)$ and $P_{<x,y>}$ measure the performance requirement by scenario $e$ and the delivered performance for $<x,y>$. $u_{<x,y>}$, $u_{<x,y>}\in\{0,1\}$, is the state value describing the healthy status of component $x$ with regards to the performance metric $<x,y>$, and satisfies

$$\begin{cases} u_{<x,y>}(e)=1, \text{ when } P_{<x,y>}<R_{<x,y>}(e) \\ u_{<x,y>}(e)=0, \text{ when } P_{<x,y>}\geq R_{<x,y>}(e) \end{cases}$$

An $n$-tuple, $U_x(e)$, is used to depict the performance status of component $x$ under scenario $e$.

$$U_x(e)=(u_{<x,1>}(e),\ldots,u_{<x,n>}(e))$$

where $n$ is the total number of all measurable metrics for the component $x$.

The under-performing of components surely affect the performance of scenario. A performance malfunctioning assignment function is used to represent the degree of influence to $SPM$ when a particular performance state exhibits in a particular component. A performance malfunctioning assignment function

$$m: SPM\rightarrow[0,1]$$

is defined, when it verifies the following two formulas:

$$\forall A\in SPM, m_{U_x(e)}(A)\in[0,1]; \text{ and } \sum_{A\in SPM} m_{U_x(e)}(A)\leq 1$$

where $m_{U_x(e)}$ denote the performance assignment function during the performance state $U_x(e)$. The higher value of $m_{U_x(e)}(A)$ denotes the higher influence of the state $U_x(e)$ to the performance malfunctioning $A$, and vice versa. The $m$ function can be co-authored by system analysis and simulations. However, how to obtain the $m$ function is not within the scope of this research. We consider the $m$ function can be generated by a third party service.

We then study how to assess the performance of a component. Let $R'_{<x,y>}$ denotes the predicted performance for $<x,y>$. For $\forall x, y$, there exist

$$v_{<x,y>}(e) = \Pr(R'_{<x,y>} < P_{<x,y>}(e)) \cdot$$

Once the component $x$ is invoked by scenario $e$, let $v_{<x,y>}(e)$ be the degree of likeliness of under-performance measured from $<x,y>$. The state of likeliness under-performance of component $x$ can be represented as an $n$-tuple,

$$V_x(e) = (v_{<x,1>}(e), \ldots, v_{<x,n>}(e)) \cdot$$

A similarity measurement function $sim : (X,Y) \to [0,1]$ is defined, where $X = (x_1, \ldots, x_n)$ and $Y = (y_1, \ldots, y_n)$ are two $n$-tuples, and

$$sim(X,Y) = \prod_{k=1}^{n} (1 - |x_k - y_k|) \cdot$$

Then, when $V_x(e)$ can be obtained, the performance assignment function can be given as

$$m_{V_x(e)}(A) = \sum_{U_x(e) \in \Phi_x} sim(U_x(e), V_x(e)) \times m_{U_x(e)}(A)$$

we also can have

$$m_{V_x(e)}(\vec{SPM}) = \sum_{U_x(e) \in \Phi_x} sim(U_x(e), V_x(e)) \times m_{U_x(e)}(\vec{SPM})$$

From this formula, the relationship between the performance of component $x$ and *SPM* is given.

Multiple components are involved when a scenario is being executed. To account for the probability of a component failure manifesting itself into user-end performance degradation, Dynamic Metrics (DM) is being used. Dynamic Metrics are used to measure the dynamic behaviour of a system based in the premise that active components are source of failure [2]. It is natural that a performance meltdown of component may not affect the remote instrumentation scenario's performance if not invoked. So, it is reasonable to use measurements obtained from executing the system models to perform performance analysis. There is a higher probability that, if an under-performance event is likely to exist in an active component, it will easily lead the scenario into malfunctioning. A data structure denoting the Component Status Description (*CSD*) is defined as:

$$CSD_x(e) = < i, x, start_x(e), duration_x(e), m_{V(x)}(\vec{SPM}) >$$

where $i$ and $x$ are the unified identification of invoking request and component; $start_x(e)$ and $duration_x(e)$ are, respectively, the start time and expected execution duration of component $x$. $Start_x$ can be given by analysing the scenario that invokes the component. We can assume the value of $duration_x$ can be given by an external service. When the CSD of all components that involve in a scenario are given, a mapping from the Time-Component Representation of a scenario to Time-Discrete Representation can be carried out. A data structure denoting the Scenario Status Description (*SSD*) is defined as:

$$SSD(e) = < j, \{x | x \in set(j)\}, start_j(e), duration_j(e), \bigcup_{x \in set(j)} m_{V(x)}(\vec{SPM}) >$$

where $j$ is the serial number for the component; $set(j)$ is the set of active components in the $j^{th}$ time fragment; $start_j(e)$ and $duration_j(e)$ are the start time and the time length for the $j^{th}$ time fragment when the scenario $e$ is applied. Since all components are independently operated, the accumulated performance malfunctioning assignment can be given by the following formula:

$$\bigcup_{x \in set(j)} m_{V(s)}(\vec{SPM}) = 1 - \prod_{x \in set(j)} (1 - m_{V(x)}(\vec{SPM}))$$

Therefore, the PPM of a scenario $e$ can be given as:

$$PPM(e, SPM_e) = 1 - \prod_{j \in scenario\ e} (1 - \frac{duration_j(e)}{T(e)} \cdot \bigcup_{x \in set(j)} m_{V(s)}(\vec{SPM}))$$

where $T(e)$ is total time required for scenario $e$.

Finally, the performance evaluation of a Grid service can be given as

$$Cost(e) = \sum_{A \in SPM} CPM(e, A) \cdot PPM(A)$$

where $\forall A \in SPM$ ,

$$PPM(e, A) = 1 - \prod_{j \in scenario\ e} (1 - \frac{duration_j(e)}{\sum_{j \in scenario\ e} duration_j(e)} \cdot \bigcup_{x \in set(j)} m_{V(x)}(A)),$$

$\bigcup_{x \in set(j)} m_{V(s)}(A) = 1 - \prod_{x \in set(j)} (1 - m_{V(x)}(A))$, and $m_{V_x(e)}(A) = \sum_{U_x(e) \in \Phi_x} sim(U_x(e), V_x(e)) \times m_{U_x(e)}(A) \cdot$

## 5 Summary

This paper describes the on-going research of the Grid service performance assessment. It explores the approach of assessing the Grid service performance by using Guided Subsystem Approach. The assessment approach given is an effective way to measure the composite Grid services which include multiple loosely coupled component services. Top-down approach is used to capture the characteristics of user demands. Performance requirements are automatically translated and transferred to component services. Subsystem approach locally analyses the local available resources and demands, and concludes the possibility of under-performing for the local module. Dynamic metric is used to combine the performance assessment result together taking into account the importance of the component. Future will adopt the assessment approach to specified applications to show its advantages.

## References

1. Akamai Corporation, `http://www.akamai.com`
2. Moore, K., Cox, J., Green, S.: Sonar - a network proximity service. Internet-Draft (February 1996), `http://www.netlib.org/utk/projects/sonar/`
3. Francis, P.: Host proximity service (hops) (preprint) (August 1998)

4. Francis, P., Jamin, S., Jin, C., Raz, D., Shavitt, Y., Zhang, L.: IDMaps: A Global Internet Host Distance Estimation Service. IEEE/ACM Trans. on Networking (October 2001)
5. Seshan, S., Stemm, M., Katz, R.: SPAND: Shared Passive Network Performance Discovery. In: USENIX Symposium on Internet Technologies and Systems (December 1997)
6. Bhattacharjee, S., Fei, Z.: A Novel Server Selection Technique for Improving the Response Time of a Replocated Service. In: Proc. of IEEE Infocom (1998)
7. Crovella, M., Carter, R.: Dynamic Server Selection in the Internet. In: Proc. of the 3nd IEEE workshop on the Architecture and Implementation of High Performance Communication Sybsystems (HPCS 1995) (1995)