

# Topology-aware Fault-Tolerance in Service-Oriented Grids

Paul Townend, Jie Xu  
School of Computing,  
University of Leeds,  
Leeds, UK

{pt,jxu}@comp.leeds.ac.uk

## Abstract

*A promising means of attaining dependability improvements within service-oriented Grid environments is through the set of methods comprising design-diversity software fault-tolerance. However, applying such methods in a Grid environment requires the resolution of a new problem not faced in traditional systems, namely the common service problem whereby multiple disparate but functionally-equivalent services may share a common service as part of their respective workflows, thus decreasing their independence and increasing the likelihood of a potentially disastrous common-mode failure occurring. By establishing awareness of the topology of each channel/alternate within a design-diversity fault-tolerance scheme, techniques can be employed to avoid the common-service problem and achieve finer-grained control within the voting process. This paper explores a number of different techniques for obtaining topological information, and identifies the derivation of topological data from provenance information to be a particularly attractive technique. The paper concludes by detailing some very encouraging progress with integrating both provenance and topology-aware fault-tolerance applications into the Chinese CROWN Grid middleware system.*

## 1. Introduction

A traditional way to increase the dependability of distributed systems is through the use of fault tolerant techniques [AND90]. The approach of design diversity - and especially multi-version design - lends itself to service-oriented architectures (SOAs) and hence Grids, as the potential availability of multiple functionally-equivalent services should allow a multi-version system to be dynamically created at much lower cost than would traditionally be the case. This is because a developer can pay to license/use pre-existing services rather than develop and maintain their own. Service-orientation promises to reduce the cost of developing and maintaining any in-house services, as well as the cost of integrating multiple services together [LIN03].

However, the provision of design-diversity fault-tolerance in Grid environments brings with it a new problem not encountered in traditional systems, whereby multiple disparate but functionally-equivalent services may - dynamically or statically during the course of their workflow - invoke identical 'common' services, thus reducing channel diversity, and potentially increasing the likelihood of common-mode failure occurring (whereby multiple channels fail in a similar way, producing identical but incorrect results) [TOW05a]. We refer to this

problem as the *common-service problem*. A potential solution to this problem is through the analysis of topological information of constituent channels/alternates within a design-diversity system. Methods of deriving such topological information are analysed within this paper, with the use of provenance information shown to be particularly effective. Furthermore, this paper demonstrates the successful integration of an existing provenance Grid system to an existing fault-tolerance Grid system, in order to allow for topology-aware fault-tolerance in a Grid environment. The feasibility of topological awareness is further enhanced by the integration of this system into the CROWN Chinese Grid middleware system.

## 2. Topological analysis

Services which are composed of other services are known as *composite services*, whereas services which only access their local system are known as *basic services*. Whether or not a service is composite or not is usually hidden from the perspective of a service requestor; indeed, the distinction between a basic service and a composite service is often seen as unimportant; for example, in the context of Web services, [ALO04] states "...whether a Web service is basic or composite is irrelevant from the perspective of the clients, as it is only an implementation issue."

However, the concept of a service being potentially composite takes on much greater importance when considering the common-service problem, due to the lack of knowledge about the workflow of alternates/channels within a design diversity scheme. This results in design diversity becoming less attractive in the context of SOAs, despite the great opportunities for cheap construction of such systems. It is therefore essential to investigate ways in which to reduce the impact of such common services between channel/alternate workflows. One such method is to exploit awareness in system topology to achieve this goal, as through knowledge of system topology it may be possible to devise techniques for reducing the impact of the common service problem, such as through the use of weighted voting algorithms.

The topology of a software system is essentially the set of dependencies between the components that constitute that system. In the context of the common service problem, it is important that the dynamic dependencies that can occur due to the ultra-late binding capability of service-oriented Grids are properly represented within the topological view of the system, and therefore we are particularly concerned with building up a view of system topology that represents the concrete workflows of every channel/alternate within a design diversity based system that are enacted for a given input, and – recursively – the concrete workflow of each component within those workflows.

We define topology as “*the set of dependencies between all components of a system that are used to produce an outcome based on a given input*”, where a system consists of a number of components, which cooperate under the control of a design to service the demands of the system environment. In order to provide such topology information, there needs to be mechanisms to extract it.

## 2.1 Methods of achieving topological awareness

At the current moment in time, there are no widely known methods of gaining information about the topology of a given service within a Grid system, and very little research has been performed in this area, the notable exception being that performed by [PER05]. However, there are a number of technologies that can potentially be used to provide such information.

One such technology is that of SOAP messaging. SOAP is the commonly used XML-based protocol used when passing messages between services within Grids. The body element of a SOAP message is concerned with the main information that the sender wishes to transmit to the receiver, whilst the header element (if present) contains additional information necessary for intermediate processing or

added value services. One possible method of recording topology information is to require the hosting environment of each service within a Grid to append topology information within the header of every outgoing SOAP message it sends. In this way, the topology of a given request can be reconstructed from the individual blocks within the SOAP header element, once the final SOAP message containing a result is received by the fault tolerance mechanism. This is similar to the approach used in [PER05], which attempts to gather topological information by prepending dependency information onto a CORBA request as it passes from stage-to-stage.

Another potential means for storing topological information is through extensions to WSDL documentation. A WSDL document describes the interface of a Web service and how to interact with that interface; however, an extension to WSDL could allow for information about the workflow of a given service/operation to be expressed within the service description of a service. A service requestor could thus parse this information in order to ascertain the workflow of a given service operation.

A similar technique to this is to store information about the topology of a given service’s workflow (or alternate negotiable topologies) in a metadata document (such as a *WS-Policy* document) related to that service; this data can then be retrieved via a technology such as *WS-MetadataExchange* and parsed accordingly.

The advantage and disadvantages of each of these techniques are summarised below:

- *Appending topology information to SOAP headers.* This method shows promise, as it allows for the capture of dynamic system topology; this is a particularly useful property in the context of service-oriented Grids, due to their ability to enact late-binding schemes. However, the method also requires that each hosting environment within a virtual organisation be adapted to perform this task in a syntactically and semantically equivalent way; as there is currently no existing technology that provides this functionality, this is a non-trivial task. Also, the approach could – depending on the size of a workflow (which may be recursive) – result in very large SOAP headers that require increased time to parse and transmit, thus resulting in increased system overhead.

- *WSDL.* Embedding topological information within the WSDL document of a service has the advantage that topology can be derived before the service is invoked; this results in increased choice for any topology-aware fault tolerance mechanisms as decisions can be made ahead of invocation. However, this method cannot express the topology of a service that exploits dynamic, ultra-late binding techniques, and requires that any fault tolerance

mechanism that wishes to exploit topological data to always have access to the latest version of the service's WSDL document. Additionally, such a method would require extensions to the standard WSDL schema that currently do not exist, and would have to be adhered to by each service within a workflow recursively.

- *Metadata.* The notion of embedding topological information within the metadata of a service has similar advantages to that of embedding the information within a WSDL document; namely that topology can be extracted in advance of the invocation stage. However, the approach shares a similar disadvantage in that dynamic ultra-late binding services (in any part of the set of workflows enacted) cannot be modelled in this way. Unlike embedding topological information within a WSDL document, however, there are already standard ways – such as WS-Policy and WS-MetadataExchange – for expressing and extracting metadata, and so at least from a syntactic point of a view, the approach is more feasible.

## 2.2. Use of Provenance

In addition to the methods expressed above, we propose a new technique to deriving topological information about service workflows: analysis of the provenance information of a given task. The provenance of a piece of data is the documentation of the process that led to a given data element's creation. This concept is well established, although past research has referred to the same concept with a variety of different names (such as lineage [LAN91], and dataset dependence [ALO97]).

The recording of provenance information can be used for a number of goals, including verifying a process, reproduction of a process, and providing context to a piece of result data. Provenance in relationship to workflow enactment and SOAs is discussed in [SZO03]; in a workflow based SOA interaction, provenance provides a record of the invocations of all the services that are used in a given workflow, including the input and output data of the various invoked services. Through an analysis of interaction provenance, patterns in workflow execution can thus be detected. Using actor provenance, further information about a service, such the script a service was running, the service's configuration information, or the service's QoS metadata, can be obtained. Through an analysis of interaction provenance, system topology can thus be derived.

This approach to deriving topological information has a number of advantages over all three of the methods described in section 2.1. Methods of deriving topology from WSDL descriptions or service metadata are attractive in that the entire

topology of a process can be known ahead of invocation (prospectively); however, these schemes assume a static topology and thus cannot be used with dynamic, ultra-late binding services. Conversely, recording topology within SOAP header elements can be used to record dynamic topology, but is strictly retrospective – the topology of a process can only be analysed upon the return of a result.

By deriving topology information from a provenance system, a neat compromise can be made between these properties; topology information can be derived at run-time during the course of an invocation (and subsequent workflow enactment) by querying data contained within the provenance store; this allows some analysis to be performed before a result is received, and also supported dynamic, ultra-late binding services.

Additionally, provenance technologies are already in existence, and are being increasingly used in real-world hosting environments, and so the infrastructure on which topology information can be extracted does not need creating, unlike in the case of appending information to SOAP messages.

**Table 1. Methods for topology derivation.**

Scheme	Dynamic	Data Availability	Feasibility
SOAP headers	Yes	Retrospective	Requires implementation of middleware support.
WSDL	No	Prospective	Requires extensions to specification.
Metadata	No	Prospective	Technology already exists.
Provenance	Yes	Run-time.	Technology already exists.

A comparison of all four schemes is shown in table 1. Due to the advantages it provides over other possible methods, the view of this paper is that provenance is a particularly attractive technique to consider when deriving topological information.

## 3. PreServ Integration with CROWN

The ideas of exploiting topological information to aid in the provision of design-diversity software fault-tolerance in Grid environments have been explored in the FT-Grid tool, developed at the University of Leeds and discussed in [TOW05b]. This tool derives its topological data from analysis of provenance records generated by the PreServ provenance recording tool [GRO04], developed at the University of Southampton. Initial experimentation has been

very positive, and has demonstrated a significant reduction in the number of common-mode failures encountered during a large number of automated tests. [TOW05b].

One drawback of the approach is the assumption that every service within a Grid system has a compatible provenance recording facility; this is a rather strong assumption, and has been of questionable feasibility. However, recently, as part of the UK-Sino COLAB (Collaboration between Leeds and Beihang universities) project, the developers of the CROWN (Chinese Research environment over Wide-area Network – a middleware widely used in the Chinese e-Science programme) Grid middleware system have integrated the PreServ provenance system into the main CROWN middleware.

The intention of this is to allow provenance recording functionality to be transparently available on any service hosted on CROWN systems. From the perspective of using topological information to assist design-diversity fault-tolerance schemes, this is of great value, as the prospect of deriving complete topological information about channel/alternate workflows becomes much more feasible on systems developed within the CROWN Grid. Indeed, it is intention of the COLAB project to integrate a generalised FT-Grid application into future releases of the CROWN middleware, to take advantage of this situation.

This generalised FT-Grid application is intended to offer mechanisms for topological analysis that can then be fed into other fault-tolerance schemes, as well as offering to enact user-defined design-diversity fault-tolerance applications itself (such as Distributed Recovery Blocks and Multi-Version Design).

#### 4. Conclusions

There is a great need for methods for improving the dependability of applications within service-oriented Grid environments. A promising means of attaining such dependability improvements is through the set of methods comprising design-diversity software fault-tolerance, such as Multi-Version Design and Recovery Blocks.

However, applying such methods in a Grid environment requires the resolution of a new problem not faced in traditional systems. This paper investigates the *common service problem*, whereby multiple disparate but functionally-equivalent services may share a common service as part of their respective workflows, thus decreasing their independence and increasing the likelihood of a potentially disastrous common-mode failure occurring.

By establishing awareness of the topology of each channel/alternate within a design-diversity fault-tolerance scheme, techniques can be employed to avoid the common-service problem and achieve finer-grained control within the voting process.

This paper has explored a number of different techniques for obtaining topological information, and has identified the derivation of topological data from provenance information to be a particularly attractive technique.

The paper concludes by detailing some very encouraging progress with integrating both provenance and topology-aware fault-tolerance applications into the Chinese CROWN Grid middleware system.

#### 5. References

- [ALO97] G. Alonso, C. Hagen, "Geo-opera: Workflow Concepts for Spatial Processes", in Proceedings of the 5th International Symposium on Spatial Databases, Berlin, Germany, June 1997.
- [ALO04] G. Alonso, F. Casati, H. Kuno, V. Machiraju, "Web Services: Concepts, Architectures and Applications", Springer, 2004.
- [AND90] T. Anderson and P. Lee, *Fault Tolerance: Principles and Practice*. New York: Springer-Verlag, 1990.
- [GRO04] P. Groth, M. Luck, L. Moreau, "A protocol for recording provenance in service-oriented grids", in Proceedings of the 8th International Conference on Principles of Distributed Systems (OPODIS'04), Grenoble, France, December 2004.
- [LAN91] D. Lanter, "Lineage in Gis: The Problem and a Solution", Technical Report 90-6, National Center for Geographic Information and Analysis, UCSB, Santa Barbara, CA, 1991.
- [LIN03] Z. Lin, H. Zhao, and S. Ramanathan, "Pricing Web Services for Optimizing Resource Allocation - An Implementation Scheme," presented at 2nd Workshop on e-Business, Seattle, December 2003.
- [PER05] S. Pertet, P. Narasimhan, "Handling Propagating Faults: The Case for Topology-Aware Fault-Recovery", in DSN Workshop on Hot Topics in System Dependability, Yokohama, Japan, June 2005.
- [SZO03] M. Szomszor, L. Moreau, "Recording and reasoning over data provenance in web and grid services", Int. Conf. on Ontologies, Databases and Applications of Semantics, Vol. 2888 of Lecture Notes in Computer Science, pp. 603-620, Catania, Italy, November 2003.
- [TOW05a] P. Townend, P. Groth, J. Xu, "A Provenance-Aware Weighted Fault Tolerance Scheme for Service-Based Applications", in Proceedings of 8th IEEE International Symposium on Object-oriented Real-time distributed Computing, Seattle, May 2005
- [TOW05b] P. Townend, P. Groth, N. Looker, J. Xu, "FT-Grid: A Fault-Tolerance System for e-Science", in Proceedings of 4th U.K. e-Science All-Hands Meeting, 19th - 22nd Sept., 2005, ISBN 1-904425-53-4.