Enabling Resource Access Visibility for Automated Enterprise Services

Kaushik Dutta, Department of Information Systems, National University of Singapore, Singapore, Singapore
Debra VanderMeer, College of Business, Florida International University, Miami, FL, USA

ABSTRACT
Organizations deliver on their mandates by executing a variety of services. Over the past few decades, service automation software systems, such as SAP and PeopleSoft, have enabled the automation of services. While much attention in the literature and in industry has been devoted to the implementation and functional correctness of automated services, little focus has been granted to ensuring responsiveness for services. As service automation platforms host larger and larger numbers of services, and services execute with greater and greater levels of concurrency, fault resolution becomes an important issue in ensuring expected responsiveness levels. In particular, two factors impact fault resolution in service automation platforms. First, each executing service requires access to specific data and system resources to complete its processing. As greater numbers of services execute concurrently, there is increasing contention for these data and system resources, leading to greater numbers of faults and SLA violations in service execution. Second, the black-box nature of service automation platforms provides little visibility into the nature of resource contention that caused a fault or SLA violation. This lack of visibility makes fault resolution difficult, and in many cases impossible, because it is difficult to trace the root cause of the problem. In this paper, we address the problem of system-level resource visibility for services through the design and development of a system capable of mapping abstract service workflows to their data and system impacts to enable resource visibility. Our system has been tested and demonstrated effective, as we demonstrate in a case study setting.

Keywords: Automated Services, Enterprise Applications, Monitoring, Quality of Service, Resource Impacts

INTRODUCTION
Firms are increasingly virtualizing manual processes (Overby, 2008) as automated services. In this context, service quality, defined by Xu et al. (2013) as “a customer’s global, subjective assessment of the quality of an interaction with a vendor, including the degree to which specific service needs have been met,” is an important concern. To date, much of the literature in service quality considers the challenges inherent in ensuring functional correctness when converting manual processes to automated services (Linton, 2003), improving service quality for physical processes (Mukherjee et al., 1998; Soteriou and Chase, 2000), or leveraging information

DOI: 10.4018/jdm.2014040101
technology to improve provisioning of a firm’s
customer service in general (Ray et al., 2005;
Karimi et al., 2001).

As enterprise service delivery platforms,
such as SAP and PeopleSoft, grow in scale, and
are assigned increasingly larger sets of responsi-
bility for service processing, the impact of
service interruptions has a proportionally larger
impact on a firm’s ability to function – when a
relatively small set of servers are tasked with
handling critical internal and external service
tasks, any issue that impacts the service de-

livery platform has the potential bring entire
departments, perhaps even an entire firm, to a
standstill (Pang and Whitt, 2009).

Creating effective service quality mainte-
nance processes is an active area of work (Tri-
enekens et al., 2004) and a billion-dollar industry
(Oracle, 2008). Quality of Service (QoS) models
do not mandate that problems must not occur
at all. In fact, modern enterprise technology is
complex enough that it is generally accepted
that issues will occur. QoS models, typically
codified in Service Level Agreements (SLAs),
take this as a given, and focus on guarantees of
recovery, i.e., how quickly issues are resolved,
and how much downtime a service will suffer.

Of primary importance is how quickly issues
are resolved, usually quantified by the time to
resolution (TTR) metric (Hiles, 2002). Organ-
izations prefer TTRs on the order of minutes,
not hours. However, for application support
staff, TTRs can be on the order of multiple
hours or longer, when particularly tricky issues
arise. In this work, we collaborated with an
application development and support team for
a major US media company. In the experience
the media company’s enterprise applications
manager, TTR for some production issues on
their PeopleSoft platform could run 5-8 hours.
The service downtimes that result from this
have significant impacts on the organization’s
bottom line in terms of lost productivity and
lost sales opportunities.

Problem resolution in general follows a
four-step workflow (Johnson, 2002), as depicted
in Figure 1. Here, the workflow begins when
the issue is logged as a new trouble ticket, and
alerts are sent to the appropriate support staff. In
the second step, the root cause analysis (RCA)
step, the application support staff member as-
signed to the ticket gathers information aimed
at determining why the problem occurred. In
the third step, the support staff cleans up any
partially completed processes and restarts
them to move the impacted business service(s)
forward. For example, a partially-completed
payroll calculation will need to be restarted with
its original inputs so that the next scheduled
steps can take place as designed. In the fourth
step, the application support analyst develops
recommendations for preventing the issue’s
recurrence in the future, and documents the
problem characteristics and resolution process
for future use.

In terms of problem resolution, the focus of
service management frameworks (such as ITIL
(APM Group, Ltd., 2009) and ISO 20000 (for
Standardization, 2011)) is to build knowledge
bases in the “Prevent” step in Figure 1, such
that IT staff can more easily recognize (based
on problem characteristics) and remediate the
same problem in future instances of the “Root
Cause Analysis” and “Remediate” steps (Ga-
lup et al.,2009; Rai and Sambamurthy, 2006;
McNaughton et al., 2010). In order to build the
underlying knowledge bases to support this
type of service management, it is critical that

---

**Figure 1. Problem resolution workflow**

![Problem Resolution Workflow Diagram](image-url)
Developing a Homegrown Course Management System: Community/Course Action/Interaction Management System (CAMS)
Cases on Database Technologies and Applications (pp. 154-170).
www.igi-global.com/chapter/developing-homegrown-course-management-system/6210?camid=4v1a