Managing End-to-End Lifecycle of Global Service Policies

Daniela Rosu and Asit Dan

IBM T.J. Watson Research Center, 19, Skyline Drive, Hawthorne, NY, 10532, USA
{drosu, asit}@us.ibm.com

Abstract. Enterprise business services are often deployed over complex environments, managed by multiple service-management products. For instance, a business service may be configured as a three-tier environment with multiple services that run on different resource domains and span one or more tiers, and comprising service-management products such as workload managers, business resiliency managers, and resource arbiters. The objective policies of the enterprise business service, henceforth called Global Service Policies, determine the runtime policies used by the various management products. The lifecycle management of global service policies, including the deployment and enforcement stages, inherits the complexity of the enterprise IT environment. This paper proposes a novel framework for efficiently managing the deployment and enforcement lifecycle stages. The framework enables the complete automation of dissemination and translation of global policy for all service managers, for a low-cost, correct policy deployment. Also, the framework enables the runtime customization of resource arbitration components for using the actual business value models of the enterprise objectives global, for a high quality of policy enforcement. The proposed framework is prototyped and integrated with several IBM service-management products.

1 Introduction

In a service-oriented architecture, policies associated with business services define the business objectives under which the services are to be managed. Business objectives may be derived from Service Level Agreements (SLAs) [1, 2] established between provider and its customers. For instance, an SLA regarding a web-based application identifies the types of requests to be issued by the customer and the associated response time and availability objectives.

A typical enterprise business service consists of multiple software components, deployed over a complex environment managed by several independent service-management products. For example, a business service might be deployed as a three-tier configuration in an environment comprising web servers, application servers and data servers (see Fig. 1). Sample service-management products in this environment include i) workload managers [5] that prioritize and distribute service invocations in order to meet response time and throughput objectives, ii) business resiliency managers [4] that manage the backup nodes, and perform appropriate service reconfiguration in response to node failures such to satisfy recovery time and availability objec-

B. Benatallah, F. Casati, and P. Traverso (Eds.): ICSOC 2005, LNCS 3826, pp. 570 – 575, 2005.

tive, iii) resource arbiters [3] that dynamically change allocation of server nodes across tiers such that the service managers can satisfy their objectives. Therefore, multiple service-management components manage the same set of services under a common set of service level objective policies, referred to as Global Service Policies.

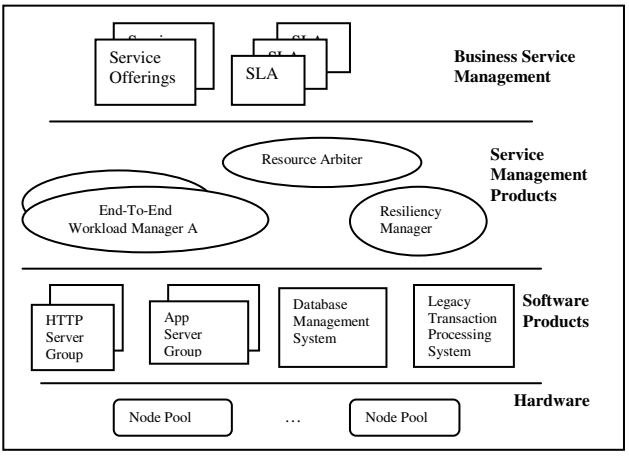

Fig. 1. Architecture of sample enterprise IT infrastructure

There are many challenges faced today in managing services in such an environment based on enterprise business objectives. Many challenges stem from the lifecycle management of Global Service Policies. The foremost challenge is the difficulty of deploying the enterprise business objectives in a consistent manner across all management components. The difficulty stems from multiple factors. First, the runtime policies used by each of these components are expressed in specific format which, most often, mixes business objectives with deployment details such as the domain in which the service is deployed, and groupings of service objectives into service classes used for management. Ensuring consistency is difficult because, many often, the conceptual elements expressed via policies do not match across these components. For example, a workload manager may manage service level objectives associated with a service endpoint, i.e., url or WSDL operation, while a resource arbiter may manage objectives associated with a node or cluster. As a result, existing solutions to policy deployment for complex SOA environments use manual, error-prone operations, involving multiple component-specific tools/GUIs.

Another challenge is the consistent enforcement of Global Service Policies across all service managers. Most prominent is the limitation of resource arbitration products to make decisions based on the actual business value model (i.e., value types and expressions) of the Global Service Policies and the related resource-allocation optimization objectives. Existing arbitration solutions use fixed value models and optimization objectives, such as a value model defined by component priority and an optimization model suitable for providing differentiated services [3]. However, many enterprises might use different value models, such as a benefit-driven model, based on

service fees and penalties for objective violations. Therefore, the actual models must be translated into the arbiter's model. Most often the translation results in an approximation of the actual models, leading to inconsistent policy enforcement.

Supporting a SOA demands novel solutions for lifecycle management of global service policies, which enable complete process automation and compliance with the actual enterprise business service model. Towards this end, we propose a novel framework for global policy management. The framework comprises an infrastructure for fully automated global policy dissemination and transformation into the runtime artifacts used by the individual service-management products. The infrastructure enables runtime, low-cost updates of the service infrastructure based on separation of business service model from service deployment and implementation details. Also, the framework includes a novel infrastructure for customization of the resource arbitration process for using the actual enterprise business service value models. The infrastructure uses the novel "optimization value model" abstraction that describes the relationship between orchestration objectives and the business value models of global policies and has methods that the arbiter can invoke in its decision procedures.

A large body of research has addressed the use of SLAs for the management of complex IT environments, composed of Web Services and computational grids. A detailed discussion of related work can be found in [6]. Our proposal distinguishes from related work in several ways. First, we consider the problem of SLA dissemination in which the service and objective specifications are decoupled from the service deployment details, which is a necessity in SOA environments. Second, we address the problem of resource arbitration for a dynamic business service environment, in which both SLA and optimization objectives can change at runtime, and require immediate, fully compliant, low cost integration. In the following, we briefly present the proposed infrastructure. For an extensive presentation see [6].

2 Automated Global Policy Dissemination

Enterprise business service management components produce Global Service Policy specifications used for the configuration of all of the service-management products in the IT environment. These specifications represent customer SLAs, enterprise-specific orchestration policies, and other types of policies. Global Policy specifications are described as XML documents, e.g, WS-Agreement schema, which can include policies that relate to multiple service managers (see Appendix A). In the policy dissemination process, one has to extract the related policy specifications for each of the service managers, transform the content to manager specific runtime artifacts, and deploy them according to policy qualifying conditions.

The proposed infrastructure for automated global policy dissemination builds on the separation of the business service model from the service deployment and implementation details. Namely, a Policy Disseminator (Fig. 2) performs policy filtering and distribution to service managers based on a generic global policy model, and manager-specific Global Policy Adapters perform transformation and deployment of these specifications as service manager-specific runtime artifacts based on the deployment and implementation details.

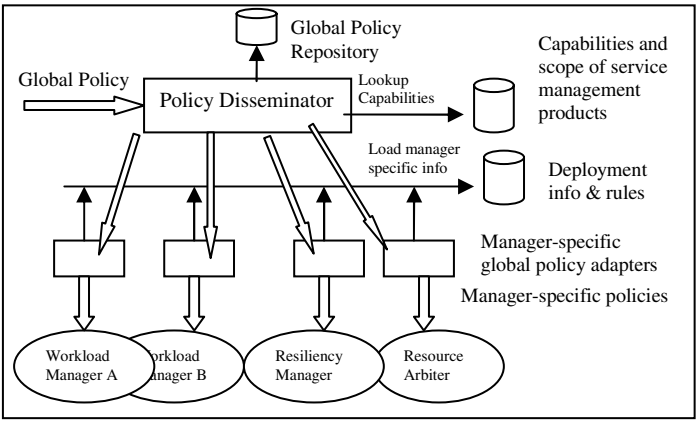

Fig. 2. Architecture for global policy dissemination

The global policy filtering is based on service manager capabilities, which are registered with the Policy Disseminator, and describe the manager's service scope and management objectives. The service scope identifies the set of enterprise services that the manager controls. The management objectives identify the type of SLO that the manager can enforce for a particular service. For filtering global policy specifications, one identifies the XML elements in the specification that define service scope and management objectives and matches them against the registered manager capabilities. For WS-Agreement specifications, these XML elements result from the content of wsag:ServiceReference and wsag:ServiceLevelObjective, respectively (see Appendix A).

The transformation of global policies into manager-specific runtime artifacts performed by Global Policy Adapters is based on (1) manager and service specific deployment information available in databases or configuration files, and (2) manager-specific rules for transformation of global policy abstractions. Adapters account for all of the global policy documents received from disseminator and for the related policy qualifying conditions, e.g., time interval when policy is applicable. Adapters handle various policy management elements, such as qualifying conditions, when the related service managers cannot handle them.

3 Customizable Resource Arbitration for Policy Enforcement

The proposed infrastructure for customizable resource arbitration is based on the novel "optimization value model" (OVM) abstraction. An OVM identifies an enterprise objective for optimization of resource allocation, such as "minimize the overall penalty value" or "maximize number of fulfilled objectives, in importance order". OVMs are defined by business service management components as orchestration policy. They are deployed at runtime to resource arbiters, which use them to customize

the decision method based on actual set of active policies. Multiple OVMs may be defined concurrently, each with specific qualifying conditions. The resource arbiter determines which OVM is applicable for a decision instance and uses the associated implementation in the decision process. Fig. 3 illustrates the main OVM components.

OVM Descriptor: qualifying time and resource pool,
set of metrics (business value types and service KPIs)

OVM Method	Function
aggregateServiceForManager	Aggregate all objectives of a service managed by a service manager
aggregateServiceAcrossManagers	Aggregate all manager-level aggregates related to a service
aggregateAcrossServices	Aggregate all service-level aggregates related to analyzed allocation state
compareStateValue	Compare state-level aggregates

Fig. 3. OVM Model

An OVM is defined by a set of methods used by the arbiter for assessing which of the candidate allocation states is better to select for deployment, and a set of "metrics" used in this assessment, which can include global policy business value types, like penalty and importance, and arbitration-related service KPIs, such as 'distance from goal'. These methods enable the hierarchical aggregation of an allocation state "value" based on the values of the OVM "metrics" for each of the active global policy objectives in the given allocation state. The values of service objective metrics used in the aggregation of a state value are computed by the arbiter by interpreting the objective business value expressions extracted from global policy specifications, and using the service KPI values predicted by service managers or their adapters for the particular allocation state. The type of values produced by the OVM aggregation methods is specific to the OVM implementation. For instance, the value can be an array of pairs of objective importance level and maximum 'distance from goal', as needed for an optimization that maximizes objective compliance in importance order.

4 Conclusions

This paper introduces a novel framework for managing the lifecycle of global service policies in complex IT environments. The framework comprises novel architectures and techniques for performing global policy dissemination and transformation, and for enforcing global policy by enterprise-level resource arbiters. As a result, the deployment and enforcement stages of the global policy lifecycle can be fully automated while ensuring compliance with the enterprise business service objectives across changes related to the service deployment architecture, service objective business value models, and orchestration objectives. The prototype implementation integrated with IBM workload management and resource arbitration products demonstrates the feasibility of our proposals.

References

1. A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, J. Pruyne, J. Rofrano, S. Tuecke, M. Xu: Web Services Agreement Specification. Version 1.1, Draft 18, submitted to the Global Grid Forum, May 14, 2004.
2. A. Sahai, A. Durante, V. Machiraju: Towards Automated SLA Management for Web Services. *Hewlett-Packard Research Report HPL-2001-310 (R.1).* Palo Alto, 2002.
3. IBM Tivoli Intelligent Orchestrator http://www-306.ibm.com/software/tivoli/products /intell-orch.
4. BMC Software: http://www.bmc.com/products/proddocview /0,2832,19052_19429_31452409_ 124990,00.html
5. searchdomino.com: IBM Enterprise Workload Manager: http://searchdomino.techtarget.com/whitepaperPage/0,293857,sid4_gci1012290,00.html
6. D. Rosu and A. Dan: Managing End-to-End Lifecycle of Global Service Policies, IBM Research Report RC23661, July 2005.

Appendix A: Sample WS-Agreement-Based SLA

```
<wsag:AgreementOffer …>…
  <wsag:Terms>..
   <wsag:ServiceReference wsag:Name="Service0Ref" wsag:ServiceName="Catalog">
        <wsa:EndpointReference><wsa:Address>/CatalogShopping</wsa:Address>…
   <wsag:GuaranteeTerm wsag:Name="Goal-Performance">
    <wsag:ServiceScope wsag:ServiceName="Catalog" />
    <wsag:QualifyingCondition><aspNS:PeriodName>Primetime</PeriodName>...
    <wsag:ServiceLevelObjective> <aspNS:ResponseTimeObjective>
      <TimeSecs>2.0</TimeSecs>  <Percentile>98</Percentile></aspNS:ResponseTime..>
    <wsag:BusinessValueList>
        <wsag:Penalty> …
          <wsag:ValueUnit>USD</wsag:ValueUnit>
          <wsag:ValueExpression>
            <acel:Product>
               <acel:Minus><acel:PropertySensor name="aspNS:TransactionCnt" />
                   <acel:PropertySensor name="aspNS:OnTimeTransCnt"/></acel:Minus>
               <acel:FloatConstant><Value>1.00</Value></acel:Float…>
            </acel:Product> …
       </wsag:BusinessValueList>
   </wsag:GuaranteeTerm>
   <wsag:GuaranteeTerm wsag:Name="Goal-Availability">
    <wsag:ServiceScope wsag:ServiceName="Catalog" />
    <wsag:QualifyingCondition />
    <wsag:ServiceLevelObjective> <aspNS:AvailabilityObjective>
         <AccumulationIntervalDays>365</AccumulationIn…
         <PercentageAvailability>99.99</PercentageAvailability>
          </aspNS:AvailabilityObjective></wsag:ServiceLevelObjective>
    <wsag:BusinessValueList>
       <wsag:Penalty>
          <wsag:ValueUnit>Thousand USD</wsag:ValueUnit>
          <wsag:ValueExpression>
            <acel:Product>
              <acel:Max>
                 <acel:FloatConstant><Value>0</Value></acel:FloatConstant>
                 <acel:Minus><acel:PropertySensor name="aspNS:Downtime" />
                      <acel:PropertySensor name= "aspNS: DowntimeObjective"/>
                  </acel:Minus></acel:Max>
              <acel:FloatConstant><Value>1000.00</Value></acel:Flo…></acel:Prod..>
       </wsag:Penalty>
       <wsag:CustomBusinessValue><aspNS:RelativeImportance>High</aspNS..
    </wsag:BusinessValueList> …
```