ABSTRACT:

In a services marketplace where a particular service is provided by multiple service providers, service offerings have to be differentiated against competitor services in order to gain market share. Differentiation of services is also needed for different markets and for different consumer segments. Strategies to differentiate service offerings have to be unintrusive – without requiring major changes to the existing service realization mechanisms. In this paper, we present Service Flavors, a strategy for service providers to differentiate services. By using this strategy, it is possible to analyze and adapt various aspects of a service which help in differentiating it from that of the competitors. We model differentiating aspects as policies. We also provide a mechanism for enforcing these policies in the middleware.

KEY WORDS:
Service differentiation, Service policies, Non-functional aspects, Model-driven development

1 INTRODUCTION

Service-oriented computing paradigm deals with organizing and utilizing distributed capabilities under the control of different ownership domains (Oasis, 2005). A service represents an underlying capability of a provider which meets the goals of a consumer. In the services marketplace (Papazoglou & Georgakopoulos, 2003) context, a service could be a commoditized service, a specialized service or a monopolistic service based on the number of service providers providing that service.

Specialized services are provided by very few service providers in the services marketplace (e.g. payroll & benefits services). A service offered by a single service provider is a monopolistic service. Examples of monopolistic services are Apply for Driving License and File Tax Returns services offered by the state department (service provider). The citizen (service consumer) uses these services to apply for a motor vehicle driving license or to file his tax returns. Normally eGovernance services are monopolistic in nature as they are provided by a single service provider – the Government (Press, 2003).

In contrast, commoditized services are always provided by multiple competing service providers in a services marketplace. For example, consider a Shipping Service in the context of an e-marketplace such as eBay®. It is provided by multiple providers such as UPS®, USPS®, DHL®, OverniteExpress® or FedEx®. More often than not, the underlying capabilities represented by these services remain the same due to standardization of messages and interfaces. Standardization leads to business layer interoperability; efforts such as Universal Business Language (UBL)
(Meadows & Seaburg, 2004), ebXML (Kotok, 2001), RosettaNet (Damodaran, 2004) and UN/CEFACT (Hofreiter, Huemer, & Naujok) address business layer interoperability. The standardization of these competing services is a result of market compulsions. For customers, standardization supports easy migration from one provider to another. However, standardization takes away provider lock-in advantages for service providers. As a result, every service provider is faced with the dilemma of balancing standardization and differentiation of their service offerings. Given that standardization is a necessity, service providers of commoditized services must still differentiate their service offerings from that of the competition in order to sustain as well as gain market share. The differentiation strategy used to differentiate services in a services marketplace must be unintrusive i.e. without requiring major alterations to already existing service realization mechanisms.

The context of this paper is a service development and delivery platform that enables the service providers to differentiate their service offerings. Our main contributions in this paper are the following:

- We present Service Flavors - a strategy for unintrusive differentiation of service offerings.
- We provide means to identify and specify differentiating aspects of service offerings. We propose a way to document these aspects in a catalog.
- We propose a model-based approach for domain experts to specify differentiating aspects of service offerings as service policies.
- We describe how differentiation is achieved at runtime through policy enforcement during service invocation and execution.

1.1 Example: ShippingService

We use the example of a fictitious\(^1\) ShippingService – a commoditized service provided by FedEx®. A commoditized shipping service in an e-marketplace like eBay® would be provided by multiple service providers such as UPS®, USPS®, DHL® etc. The schemas of the messages could either be similar or different across service providers, but the underlying capability is the same. The service capability model below represents the ShippingService (fig. 1). The service capability model is based on a services metamodel described in (Harshavardhan Jegadeesan, 2008). The corresponding WSDL 2.0 (Chinnici, Gudgin, Moreau, Schlimmer, & Weerawarana, 2004) snippet (fig. 2) shows the functional service description of the fictitious shipping service. The shipping service defines a ShipItem operation which supports shipping a package from one place to another.

The rest of the paper is organized as follows: Section 2 introduces the concept of differentiating aspects (i.e. flavoring aspects); we identify specific flavoring aspects and document them in a catalog. Section 3 describes how the flavoring aspects are modeled as service policies using a generic policy metamodel. Section 4 describes how the policy models are translated to executable specification. Section 5 describes how these policies and thereby service flavors are enforced in the SOA middleware. Section 6 discusses related work. Section 7 discusses conclusions and future work.

\(^1\) all services and scenarios described in the paper are fictitious
Fig. 1. Service Capability Model

Fig. 2. WSDL 2.0 Snippet for Shipping Service
2 FLAVORING ASPECTS: DIFFERENTIATING ASPECTS OF A SERVICE

In order to differentiate service offerings, it is important to understand the changing parts across service offerings. By understanding the changing parts, we could arrive at aspects of a service that help in differentiating the service against competing services. Every service description has a functional part representing the underlying capability on offer and a non-functional part representing the terms in which the capability is offered – the terms of offer. The terms of offer represents aspects such as price, quality of service, discounts and promotions offered etc. The capability on offer satisfies a goal of the consumer under the constraints of the terms of offer. A service description must describe both the functional capability on offer as well as the non-functional terms of offer for automatic selection and consumption of a service (O'Sullivan, Edmond, & Ter Hofstede, 2002).

In the case of the ShippingService, the underlying capability is to ship items from one place to another. Given that the capability on offer is the same across competing services, how does a service provider differentiate his shipping service from that of the competition? On what basis does a service consumer choose a particular shipping service? Consider the examples of websites such as www.redroller.com and iShip™ that compare the shipping services provided by various providers such as USPS®, DHL® and OverniteExpress®. It is interesting to note the dimensions in which these services are compared – delivery date (quality of service) and shipping rates (price). Therefore, given that the capability on offer is the same, consumers would choose a particular service based on attractiveness of the terms of offer. Certain aspects of the terms of offer – aspects such as cost (pricing), discounts (promotions), availability, quality of service, convenience of use, packaging etc (Toma, Roman, & Fensel, 2007) – that make service offerings attractive to consumers become differentiating aspects, we call these Flavoring Aspects. The important criteria to determine if an aspect is a flavoring aspect or not is to answer this question – ‘would the aspect make the terms of offer attractive to the consumer and differentiate the service offering from that of competition?’ We call this the attractiveness of terms of offer criterion. It is important to note that, there are other aspects such as service reputation (Maximilien & Singh, 2002), market perception and service rating (by rating agencies) etc. which also significantly impact the choice of a service by a consumer. However, these are not under the direct influence of the service provider and are not considered in this paper. Our service flavors strategy supports two use-cases to differentiate service offerings:

UC#1: Creation of competitive service offerings by unintrusively varying flavoring aspects and attractively positioning services to consumers.

UC#2: Creation of targeted service offerings by offering the same capability on-offer under different terms of offer. Each targeted offering represents a discrete variation of a service or a Service Flavor.

2.1 A Catalog of Flavoring Aspects

The flavoring aspects identified during the early-stage services development have to be precisely documented in a catalog as they are reusable assets in services development. We propose a standard schema to document the flavoring aspects. The standard schema would facilitate better communication among stakeholders during the design and modeling activities. The schema is presented (see table 1) along with a pictorial XML Schema (Fallside, 2000) representation below (see Fig. 3):
<table>
<thead>
<tr>
<th>Name of Concern</th>
<th>The name of the concern</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of Aspect</td>
<td>Denotes the aspect type</td>
</tr>
<tr>
<td>Related Aspects</td>
<td>Denotes related aspects for this aspect</td>
</tr>
<tr>
<td>Context</td>
<td>Denotes the context in which the aspect applies</td>
</tr>
<tr>
<td>Rationale &amp; Discussion</td>
<td>Provides a brief description of the aspect and its application</td>
</tr>
<tr>
<td>Quantification</td>
<td>Denotes applicability of the aspect. It could be: 1. List of services in the services portfolio, a random list of services 2. Select services / service operations or service interaction points (end-points) 3. Ownership Domain (a physical or administrative partition for services). 4. A particular customer or a customer segment.</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>Vocabulary defines a set of vocabulary items which are used to describe the aspect</td>
</tr>
</tbody>
</table>

Table 1. Standard Schema for Flavoring Aspects Catalog

Fig. 3. XML Schema (pictorial representation) for Flavoring Aspects

We use this schema to catalog flavoring aspects. The advantage of documenting these flavoring aspects in the early-stage services development is that it helps in separation of concerns. Most importantly, note that the catalog presented below (fig. 4) is not complete. It is extensible to support specification of various other flavoring aspects. New flavoring aspects could be added to the catalog, for example Dispute Resolution could be another flavoring aspect that could be added to the catalog. Dispute Resolution mechanisms between the provider and consumer could be different in different markets and geographies (ombudsman, courts, special appellate tribunal) etc.
Further we document selected flavoring aspects below using the schema described in Table 1.

<table>
<thead>
<tr>
<th>Name of Concern</th>
<th>Service Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of Aspect</td>
<td>Availability</td>
</tr>
<tr>
<td>Related Types</td>
<td>Temporal Availability, Spatial Availability</td>
</tr>
<tr>
<td>Context</td>
<td>Service consumer trying to consume a service should be aware of the time (temporal) and location (spatial) the service would be available.</td>
</tr>
<tr>
<td>Rationale and Discussion</td>
<td>Service Availability in terms of time (temporal) and location (spatial) determines when and where a service consumer interacts with the service provider to accesses the service offering. Availability of a service is determined by a host of factors such as provisioning of the service, the availability constraints of the service provider.</td>
</tr>
<tr>
<td>Quantification</td>
<td>Any service consumption scenario is affected by the availability concern of the service.</td>
</tr>
</tbody>
</table>

Table 2. Flavoring Aspect: Service Availability

<table>
<thead>
<tr>
<th>Name of Concern</th>
<th>Service Pricing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of Aspect</td>
<td>Pricing &amp; Payment</td>
</tr>
<tr>
<td>Related Aspects</td>
<td>Payment Mode, Charging Style</td>
</tr>
<tr>
<td>Context</td>
<td>A Service offering from a provider could have a cost associated with it.</td>
</tr>
<tr>
<td>Rationale and Discussion</td>
<td>Payment is a concern during access of a paid service. Payment for a service is determined by cost of service access, the charging style and the payment modes.</td>
</tr>
<tr>
<td>Quantification</td>
<td>Payment is a concern across a set of paid services.</td>
</tr>
</tbody>
</table>

Table 3. Flavoring Aspect: Service Pricing
Our approach enables identification and specification of flavoring aspects irrespective of how the service would eventually be realized – completely automated implementation, human task or a hybrid. Our ShippingService is an example of hybrid service realization consisting of two parts – the shipping request, which is fully automated and the physical shipment of goods, which is a
human task. While flavoring aspects such as Pricing and Promotion impact the automated part (i.e. shipping request), a quality of service aspect such as Delivery Time impacts the human task (i.e. physical shipment of goods). However, it is important to model all flavoring aspects irrespective of the impacted part(s) because these aspects capture the terms of offer of the service irrespective of the realization.

3 MODELING FLAVORING ASPECTS AS SERVICE POLICIES

After identifying and documenting the flavoring aspects, we need to specify semantics and associated operational behavior of flavoring aspects. We need to define a vocabulary for each flavoring aspect with a set of vocabulary items along with types and domain of applicable values for each of these vocabulary items.

3.1 Specifying Vocabulary for Flavoring Aspects

As a first step in specifying vocabulary, vocabulary items associated with each flavoring aspect has to be identified and defined. Each vocabulary item would have a type and applicable values (domain) associated with it. The set of vocabulary items for a flavoring aspect provide a controlled vocabulary to define that flavoring aspect. For example, the pricing flavoring aspect can have vocabulary items such as type of price (applicable values: absolute price, proportional price, dynamic price), price, credit period etc. We define vocabulary items for six selected flavoring aspects we identified and defined above. The vocabulary items could also be extended to suit specific scenario and business needs. We present vocabulary items, their type and applicable values below (Table 8):

<table>
<thead>
<tr>
<th>No.</th>
<th>Aspect</th>
<th>Vocabulary Items</th>
<th>Type</th>
<th>Applicable Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Service Availability</td>
<td>Availability Period</td>
<td>Validity</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Geographical Location</td>
<td>Location</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Service Pricing</td>
<td>Pricing Period</td>
<td>Validity</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Applicable Location</td>
<td>Location</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Price Type</td>
<td>String</td>
<td>Absolute price, Proportional price, Dynamic price</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Price</td>
<td>Amount</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Additional Price Type</td>
<td>String</td>
<td>Tax, Shipping, Commission, Octoroi, Surcharge</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Additional Price</td>
<td>Amount</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Credit Period</td>
<td>Duration</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Payment Mode</td>
<td>String</td>
<td>Cheque, Cash, Credit Card, Bank Transfer, Gift Coupon</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Charging Style</td>
<td>String</td>
<td>Pay-per-use, Rental,</td>
</tr>
<tr>
<td></td>
<td>Service Promotion</td>
<td>Subscription</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>------------------</td>
<td>--------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Promotion Period</td>
<td>Validity</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Applicable Location</td>
<td>Location</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reward Type</td>
<td>String</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reward Value</td>
<td>Integer</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Discount Percent</td>
<td>Integer</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Discount Value</td>
<td>Float</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Service Privacy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Use of Personal Information</td>
<td>String</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Is Shared with Third-Party Partners</td>
<td>Boolean</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Service Security &amp; Trust</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>isIdentificationRequired</td>
<td>Boolean</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Identification Type</td>
<td>String</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>isdigitalCertificateRequired</td>
<td>Boolean</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Digital Certificate Type</td>
<td>String</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>isEncryptionRequired</td>
<td>Boolean</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Encryption Type</td>
<td>String</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Service Quality</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Execution time</td>
<td>Duration</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Rating</td>
<td>Integer</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>rating type</td>
<td>String</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>rating agency</td>
<td>String</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 8. Vocabulary Items, their type and Applicable Values for Flavoring aspects

The vocabulary of flavoring aspects could standardize across service providers as an industry standard over time. For example, cost (price) and execution time (delivery time) are now standardized vocabulary items in the package shipping industry. Consensus on vocabulary of flavoring aspects could also be achieved using ontologies (Noy & McGuinness, 2001). QoSOnt is an effort to achieve consensus on quality of service vocabulary for service-centric systems (Dobson, Lock, & Sommerville, 2005). Creating ontologies for flavoring aspects will support automatic inference, reasoning and semantic interoperability of vocabulary items across flavoring aspects and service providers in the marketplace. However, presently, we have defined flavoring aspects vocabulary as simple XML schemas (See Service Pricing.xsd in a pictorial form) (see fig. 5). Nevertheless, it is possible to document these aspects as separate ontologies, using a web ontology language like OWL.

3.2 Use of Service Policies

While the service description languages like WSDL specify service capability, service interfaces and messages, the service policy languages are used to specify the non-functional terms of offer. In our Service Flavors approach, we use service policies to define the terms of offer of a service offering. The service policy mechanism is based on prepositional logic and allows specifying...
assertions (constraints) on vocabulary, an ability to combine these assertions using conjunction or disjunction into service policies. A service policy represents the terms of offer of a service offering. The terms of offer is associated with the capability on offer by attaching the service policy to the service description using policy attachment mechanisms, offering separation of concerns. The ‘terms of offer’ of the service is essentially a set of assertions – constraints on the vocabulary items of different flavoring aspects.

Technically, competitive service offerings (UC#1) are created by altering the terms of offer by varying the assertions in the corresponding service policy. Targeted service offerings (UC#2) are created by conditionally attaching different terms of offer (different service policies) to service offerings.

```xml
<?xml version="1.0" encoding="utf-8" ?>
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:include schemaLocation="Common.xsd" />
  <xs:element name="Pricing">
    <xs:complexType>
      <xs:sequence>
        <xs:element xmlns:q1="www.fictitious.com" name="ApplicablePeriod" type="q1:Validity" />
        <xs:element xmlns:q2="www.fictitious.com" name="ApplicableLocation" type="q2:Location" />
        <xs:element name="CreditPeriod" type="xs:duration" />
        <xs:element name="Price">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="PricingMechanism">
                <xs:simpleType>
                  <xs:element>
                    <xs:complexType>
                      <xs:restriction base="xs:string">
                        <xs:enumeration value="pay-per-use" />
                        <xs:enumeration value="subscription" />
                      </xs:restriction>
                    </xs:simpleType>
                  </xs:element>
                </xs:simpleType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="ChargingStyle">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:enumeration value="pay-per-use" />
              <xs:enumeration value="subscription" />
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

**Fig. 5.** Vocabulary Items for Service Pricing Flavoring Aspect
3.3 Service Policy Metamodel

Normally domain experts (business experts) independently define the terms of offer (service policies) and associate them to service offerings. Domain experts prefer a visual syntax to specify terms of offer. We provide a service policy metamodel to support model-driven development of service policies. The service policy metamodel must support modeling of four functional layers of a generic policy model (Anderson, 2005):

- **Vocabulary Specification**: Specification of vocabulary items associated with various flavoring aspects and their applicable values which would then be used in service policies.
- **Constraint Specification**: Specification of assertions (constraints), which would normally be constraints on the applicable values of vocabulary items.
- **Policy Specification**: Specification of acceptable combinations of the constrained vocabulary items.
- **Bindings Specification**: Specification of application of the service policies on various policy subjects (services).

The service policy metamodel we present in this paper is a MOF2 (OMG, 2006) based metamodel. The ‘Core’ in the figure represents the UML2:: Infrastructure (OMG) package. We explain the key classes, associations and constraints if any, in the metamodel (fig.6):

![Service Policy Metamodel Diagram](image)

**Fig. 6. Service Policy Metamodel**
Service Policy: A service policy defines a set of enforceable constraints which would be applied on a policy subject (Oasis, 2005). It reflects the point of view of a service participant who is the owner of the policy.

Service Participant: A service participant could be a service provider or a consumer. A service provider policy is communicated to the consumer along with the service description. In our case, where we represent terms of offer through policies, the service participant is the service provider.

Policy Subject: A policy subject represents an entity on which a policy is applied. A policy subject extends the Core: Element. The policy subjects could be Ownership Domain (supports physical or administrative partition of services); Service, Service Interface, Service Operation, Message and Interaction Point (end point), Consumer and Consumer Segment. If a set of policies are applicable on a single policy subject, these are reconciled and represented as an ‘effective policy’.

Policy Scope: A policy scope represents a set of policy subjects on which a policy could be applied. It is a mechanism to group policy subjects together in order to apply the same policy on them. More than one policy could also be applied on the policy scope.

Policy Alternative: Each policy has a set of policy alternatives out of which at least one has to be honored. The policy alternative which is honored is called the ‘chosen alternative’. Every policy alternative would have more than one policy assertion.

Policy Assertion: Every policy alternative would have one or more policy assertions. A policy assertion is a constraint applied on a vocabulary item (constrained element) of a particular domain. The policy assertion specifies the allowable range, range of values, or set of values for a vocabulary item. The policy assertion could be optional in nature. It also has an operator associated with it. The operator is a predicate operator used to describe constraints.

Policy Domain: A policy domain represents a grouping of assertions belonging to a particular aspect such as pricing, availability, security & trust etc. It is representative of a flavoring aspect. A policy domain is identified by a name and a namespace URI and it extends the Core: Namespace.

Vocabulary Item: A vocabulary item represents semantics associated with a particular flavoring aspect and belongs to a policy domain. Every vocabulary item has a set of applicable values. The vocabulary items for a particular domain (aspect) are defined by the domain expert. It extends the Core: DataType.

Service Consumer: A service consumer is a service participant. A consumer can be a policy owner and the consumer policy has to have a suitable intersection with the service provider policy for the consumer and provider to collaborate. The consumer also belongs to a particular consumer segment. It could also be a policy subject. By associating a service policy to a consumer we could model consumer specific SLAs.

Consumer Segment: A consumer segment is used to group related consumers together. It represents a market segment of customers. A consumer segment could be a policy subject on which segment specific policies can be applied. That applied service policy would hold good for consumers belonging to that consumer profile. Service offering can be targeted to a specific customer segment.
We use the metamodel to model the service pricing policy (fig. 7) and apply it to our shipping service. Consider the following Subscription Pricing Policy the provider adopts to differentiate against competing shipping services and also increase subscription customers in order to gain predictability in monthly revenues. Assertions are created on vocabulary items credit period, price and charging style of the pricing flavoring aspect, represented as a policy domain.

Subscription Pricing Policy

“The service provider of the fictitious shipping service (FedEx®) provides a one-month credit period for its subscription customers.”
3.3 Consumer Context-aware Service Policies

Segmenting customers and targeting those customer segments with promotions is a good service differentiation strategy. We create targeted offerings (UC#2) – Service Flavors – by offering the same service with different terms of offer. For example, a pricing or promotion policy could be different for members of an alliance from that of other consumers., say, like members of Star Alliance in the aviation sector can redeem frequent flyer miles (reward points) across alliance partners Terms of offer for specific consumer segments can be created by creating different service policies for each of those segments. The service consumers could be segmented based on various customer segmentation schemes – based on customer characteristics such as small businesses, business partners, or members of an alliance or based on qualitative characteristics such as gold, silver or platinum customers derived based on previous engagements or revenues from the customer. These schemes are based on existing business imperatives and are different across industries and businesses. The domain expert defines consumer segments and associates specific service policies to each segment. Consider – for example – the USFSB Member Discount Policy that provides a flat 10% discount to members of the US Small Businesses federation (customer segment). A new consumer segment (USFSB: Consumer Segment) is created to represent the small business customers belonging to the USFSB. A service policy is created which has a promotion alternative with a constrained value for vocabulary item (Discount Percent = 10%) for the service promotion policy domain. This example is modeled using our service policy metamodel in fig. 8.

![Service Promotion Policy Model for USFSB Customer Segment](image-url)

**Fig. 8.** Service Promotion Policy Model for USFSB Customer Segment
4 CONVERTING SERVICE POLICY MODELS TO EXECUTABLE POLICY SPECIFICATIONS

Once the service policies are modeled to represent the terms of offer, they have to be converted to appropriate executable standards for them to be incorporated into the service description. In the ShippingService example, the service capability model (in fig. 1) was converted to a standard WSDL 2.0 service description (in fig. 2). In the same manner, the policies described using our service policy models have to be transformed to appropriate industry accepted open-standards. Since there are multiple and sometimes competing standards, we look at different standards available in each of the four layers of the generic policy model (Table 9.).

<table>
<thead>
<tr>
<th>Generic Policy Model Layer</th>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vocabulary Specification</td>
<td>Domain-Dependent Specification</td>
</tr>
<tr>
<td></td>
<td>WS-Security (Atkinson et al.), WS-Trust (Anderson, Bohren, &amp; Boubez) etc., WS-ReliableMessagingPolicy, domain-dependent assertions for pricing, promotion etc.</td>
</tr>
<tr>
<td></td>
<td>Domain-Independent Specification</td>
</tr>
<tr>
<td></td>
<td>XML Schema (Fallside, 2000), Web Ontology Language (OWL) (Smith, Welty, &amp; McGuinness, 2004) to support specification of domain ontologies for flavoring aspects</td>
</tr>
<tr>
<td>Constraint Specification</td>
<td>Domain-Dependent Specification</td>
</tr>
<tr>
<td>(Specification of Assertions)</td>
<td>WS-Security, WS-Trust, WS-ReliableMessagingPolicy</td>
</tr>
<tr>
<td></td>
<td>Domain-Independent Specification</td>
</tr>
<tr>
<td></td>
<td>WS-PolicyConstraints, XACML (Anderson, 2005)</td>
</tr>
<tr>
<td>Policy Specification</td>
<td>WS-Policy (Bajaj et al., 2004), WSPL (Anderson, 2004)</td>
</tr>
<tr>
<td>Binding Specification</td>
<td>WS-Policy Attachment (Bajaj et al.)</td>
</tr>
</tbody>
</table>

Table 9. Standards relevant to Generic Policy Model Layers

Technically the policy models created using our service policy metamodel could be transformed to any of these specifications using MOF2 Model to Text Transformation Language (MTL) (Oldevik, Neple, Grønmo, Aagedal, & Berre, 2005) standard mappings. But we have made certain choices for the standards.

For the purpose of vocabulary specification, we use domain-independent XML schemas. While specifying vocabulary as ontologies has obvious advantages, we prefer XML schemas for simplicity. For specifying assertions, we prefer domain-independent assertion specification using WS-Policy Constraints (see section 4.1). For the purpose of policy specification, we choose to use the industry accepted standard WS-Policy. The assertions that are part of every alternative in the WS-Policy specification would be domain-independent assertions specified using WS-PolicyConstraints language. For the purpose of binding, we use the WS-Policy Attachment specification. Currently we support intrinsic attachment of policies to WSDL 2.0 elements. Attaching policies to a service consumer or a consumer segment is outside the scope of WS-PolicyAttachment and is realized in the SOA middleware.
4.1 Domain-Dependent vs. Domain-Independent Specification

Vocabulary and assertions (constraints) can be defined using either domain-dependent or domain-independent specifications (Anderson, 2004). Industry standard specification languages such as WS-Security, WS-Trust, WS-ReliableMessaging (WS-RM) (Bilorusets, Bosworth, & Box, 2004) etc. provide domain-specific assertions on a rigid and restricted vocabulary for the technical domains such as security, trust and reliable messaging. Moreover, it is possible to define domain-specific vocabulary and assertions for domains such as availability, pricing etc. based on the policy author guidelines (Yendluri & Yalçinalp).

However, we prefer domain-independent specifications to specify vocabulary and assertions for flavoring aspects. Our preference is due to the flexible vocabulary requirements presented by flavoring aspects. Especially, the vocabulary for certain flavoring aspects such as pricing and promotions not only evolve, but also significantly vary across industries – in contrast to the rigid vocabulary of technical domains such as security. For example, promotions could be based on reward points in the logistics domain, but based on flyer miles in the aviation domain. Hence the vocabulary items (reward points / flyer miles) for promotions domain vary when compared to a rigid vocabulary item (securityToken) in the security domain. In order to support flexible vocabulary specification, we prefer to use domain-independent specifications. Consequently, we choose a domain-independent policy assertion language (WS-PolicyConstraints) to specify assertions. WS-PolicyConstraints uses XACML (Godik & Moses, 2003) based functions to specify constraints on the vocabulary items. Even technical domains such as security can be expressed using WS-PolicyConstraints (Anderson 2004). In summary, our decision to use domain-independent specification is based on the following:

- To support flexibility of vocabulary for flavoring aspects.
- Absence of domain-specific specifications for aspects such as availability, pricing, promotions etc.
- Advantage of using a single generic policy handling logic for parsing policies in the SOA middleware. Domain-specific assertions require domain-specific policy handling in the SOA middleware to enforce these policies.

Going back to our Subscription Pricing Policy modeled in fig 7, the constraint specified on the vocabulary item creditPeriod in the model is converted to a WS-PolicyConstraints below (fig. 9):

```xml
<Apply
  FunctionId="&wspc;function:is-less-than-or-equal">
  <AttributeValue
    DataType="&xsd;duration">P1M</AttributeValue>
  <ResourceId>AttributeValue
    AttributeId="creditPeriod"
    DataType="&xsd;duration" />
</Apply>
```

**Fig. 9.** Specification of constraints using WS-PolicyConstraints

In fig.10, we show the enhanced service description for the Shipping Service. The WSDL 2.0 is enhanced with the Subscription Pricing Policy. The SubscriptionPricingPolicy is specified with a single alternative which has domain-independent assertions on vocabulary items ‘creditPeriod’, ‘priceType’, and ‘chargingStyle’ specified using WS-PolicyConstraints. The policy is intrinsically referenced in the <service/> using <wsp: Policy Reference />.

In summary, our decision to use domain-independent specification is based on the following:

- To support flexibility of vocabulary for flavoring aspects.
- Absence of domain-specific specifications for aspects such as availability, pricing, promotions etc.
- Advantage of using a single generic policy handling logic for parsing policies in the SOA middleware. Domain-specific assertions require domain-specific policy handling in the SOA middleware to enforce these policies.
4.2 Flavored Service Consumption

The enhanced service description describes the combination of the capability on offer (service interface, types and messages) and the relevant terms of offer (service policy). The enhanced service description represents a discrete variation of the service – a Service Flavor. For a Service Consumer, an appropriate service flavor (i.e. an enhanced service description with a specialized service policy) is presented by the services registry based on the consumer segment, during service look-up. If no specific service flavor is available (i.e. if no special terms of offer are available for the consumer segment) the enhanced service description contains the default terms of offer. The consumers use the enhanced service description to invoke the service.

5 ENFORCING SERVICE POLICIES IN THE SOA MIDDLEWARE

To support service differentiation, the terms of offer expressed as service policies have to be enforced in the SOA middleware during service invocation. The most important criterion for policy enforcement is that it has to be unintrusive – without requiring any changes to the existing service realization mechanisms. Earlier we mentioned that our service flavors approach is agnostic to service realization mechanisms i.e. our flavoring aspect could impact the automated part of service realization, the human task or both wherever applicable. In case of a fully
automated realization, complete policy enforcement happens in the middleware. Suppose the realization is through a hybrid model (partly automated, partly human task). Then partial policy enforcement happens in the middleware. In our Shipping Service example, the realization is a hybrid model—the shipping request is automated, whereas the physical shipment of goods is a human task. Consider a Quality of Service flavoring aspect such as delivery time that impacts the human task. In this case, the enforcement of delivery time cannot be done in the middleware during service invocation.

The service consumer constructs a SOAP request message based on the enhanced service description. The policy preferences are expressed in the SOAP header using SOAP header extensions for each flavoring aspect. Fig. 11 presents a sample SOAP request for the ShippingService (the SOAP body is not presented for brevity).

```xml
- <soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
                    soap:encodingStyle="http://www.w3.org/2001/12soap-encoding">
  - <soap:Header>
    xmlns:p="http://fictitious.com/servicepricing"
    soap:role="http://fictitious.com/role/policyEnforcement" soap:mustUnderstand="true">
                                   xmlns:common="http://fictitious.com/common"
                                   soap:role="http://fictitious.com/role/policyEnforcement" soap:mustUnderstand="true">
        <consumer:reference scheme="UUID">00300571-1c6b-1dec-978d-559058888227</consumer:reference>
        <consumer:FormattedName>Generic Co.</consumer:FormattedName>
        <consumer:accessTime>2008-03-29T13:20:00.000</consumer:accessTime>
      - <common:Location>
        <common:Country>US</common:Country>
      </common:Location>
    </consumer:ConsumerProfile>
  </soap:Header>
</soap:Envelope>
```

Fig. 11. Sample SOAP request for the Shipping Service

The header extension element Consumer Profile provides enough consumer information to associate the consumer with a consumer segment. The header extension Pricing provides the charging style and the price type from the pricing flavoring aspect. For each service invocation, the SOAP request has to be processed in stages: first, the consumer has to be associated to a segment using the Consumer Profile header element, then each of the header extensions have to be processed to enforce the terms of offer. To support extensibility, each flavoring aspect is associated with a handler, which will process related header extensions. We assume that our flavoring aspects are orthogonal and can be handled in any order.

We realize policy enforcement using an active SOAP intermediary (Mitra, 2003) which we call the Flavoring Intermediary (fig.12). The flavoring intermediary is configured to play the role (soap: role="http://fictitious.com/role/policyEnforcement") of a Policy Enforcement Point (PEP) for flavoring service policies. The flavoring intermediary works on the SOAP header extensions
associated with flavoring aspects specially tagged with the \textit{soap: role} attribute (see fig. 11). We use Apache Axis 2.0 (hereon Axis2) SOAP engine (Perera et al., 2006) as the SOAP intermediary.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{flavoring_intermediary.png}
\caption{Flavoring Intermediary to enforce flavoring service policies}
\end{figure}

We take advantage of the extensible SOAP processing model of Axis2 that allows user-defined Phases and Handlers. A flavoring handler is defined for the enforcement of each flavoring aspect. Enforcement of policies takes place in a flavoring phase (a user-defined phase in Axis2) that invokes the flavoring handlers. Fig. 13 describes the flavoring phase and the associated handlers packaged in a flavoring module.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{flavoring_module.png}
\caption{The Flavoring Module with Flavoring Phase and Flavoring Handlers in Axis2}
\end{figure}

We also have a Consumer Profiling Handler (CPH), which is configured to be the first handler that gets invoked in the flavoring phase. The CPH deals with profiling the consumer and associating the consumer with a consumer segment. The consumer segment information may be useful for other flavoring handlers to enforce their specific policies. In our example (fig. 11), the Pricing Handler would need to know the consumer segment to apply the Subscription Pricing Policy. We use the Axis 2 \texttt{MessageContext} to share the consumer segment information from the CPH with the other handlers. In our example, the header elements \texttt{ConsumerProfile} and \texttt{Pricing} are handled by CPH and Pricing Handler respectively.

When a new flavoring aspect is needed, a new instance of a policy domain is added in the policy model and a corresponding flavoring handler is generated and automatically added to the end of the flavoring phase. Since the policies are based on domain-independent assertions the semantics
of any policy could be understood by a generic processing logic (a standard XACML policy processor).

6 RELATED WORK

Service Flavors presents a strategy to differentiate services in the service marketplace from the perspective of the provider. The closest related work is the Web Services Offering Language (WSOL) (Tosic, Patel, & Pagurek, 2002). The WSOL presents the concept of ‘class of service’ which relates to our discrete service variation, the service flavor. The WSOL specification presents a WSDL1.1-compatible XML language to describe a service offering which is a class of a web service. The class of service concept could be used to differentiate services. An important thing to note is that WSOL is not specifically intended for differentiating services, but is designed to support management of services, expressing constraints (pre-, post-conditions) etc.

In comparison, Service Flavors is a complete strategy and not just a language like WSOL. It supports the entire life-cycle from early-stage identification, definition and documentation of differentiating (flavoring) aspects as well as the vocabulary items associated with them. The service flavors approach is also open-standards compliant, whereas the WSOL is not an approved standard. Service Flavors use open-standards such as WS-Policy, WS-PolicyAttachment and WS-PolicyConstraints which have industry backing to define discrete variations of a service. The service flavors approach also prescribes a model-driven approach to create discrete variations of a service. Our approach also uses standard SOAP header extensions to specify terms of offer during service invocation. This is common practice in domain-dependent specifications such as WS-RM, WS-Trust, WS-Security etc. Others have also used SOAP headers to specify non-functional properties in the context of Grid Services [Foster et. Al. 2005] and in the in the context of Web Services [Ortiz, & Hernandez 2007]

For defining individual Service-Level Agreements (SLAs), there are languages such as the IBM’s Web Service-Level Agreement (WSLA) (Ludwig, Keller, Dan, King, & Franck, 2002) and HP’s Webservice Management Language (WSML) (Sahai, Durante, & Machiraju). These SLAs support mere QoS guarantees than really define discrete variations. These languages are mature, high-quality and proven, but inherently lack the capability to express discrete service variations as they were not meant for flavoring. WS-QoS (Tian, 2005) is a QoS specification language which has a notion of class of service; however it is more centered on network-level QoS and is not useful to flavor services.

The flavoring aspects such as pricing, availability etc. are essentially non-functional properties of services. (O’Sullivan et al., 2002) has done extensive work on non-functional properties in service description; (O’Sullivan, Edmond, & Ter Hofste, 2005) also presents formal syntax of service properties. Though some of our non-functional aspects are inspired from this work, our focus is a subset influencing flavoring of a service.

A striking feature of these flavoring aspects is that they represent crosscutting service-level concerns from the perspective of the service provider. Aspect-Oriented Software Development (AOSD) (Filman, 2005) offers an elegant way to handle cross-cutting concerns in software development by modularizing these concerns as aspects. The flavoring aspects represent crosscutting concerns such as availability, quality of service, pricing and promotions which are largely service-level provider concerns. There could be other crosscutting concerns in services.
development such as domain-level concerns, technical middleware concerns, service realization concerns (implementation and composition concerns) which are not addressed by flavoring aspects. The focus of flavoring aspects is to create service flavors and support differentiating service offerings in the services marketplace.

Modularizing cross-cutting concerns is also addressed in the industry through Business Rules in the context of business process management [Narayanan 2009, WebSphere 2008]. Business Rules are conceptually similar to Aspects – in the AOSD sense – but the use case scenarios are typically restricted to process management i.e. to modularize the changes in a process implementation so that a change in a business rule does not change the process. In contrast flavoring aspects typically capture service properties based on the attractiveness of terms of offer criterion.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we presented a strategy to support unintrusive differentiation of services in a service marketplace. We identified terms of offer of a service to be the changing part in a service description in the context of differentiation. The terms of offer was represented by the non-functional aspects which we identified as flavoring aspects. These flavoring aspects were expressed as service policies defined by the domain expert using a service policy metamodel. These flavoring service policies were enforced by a flavoring SOAP intermediary acting as a policy enforcement point. By adopting the service flavors strategy, a service provider can unintrusively differentiate his services from that of competing services in the marketplace. The service flavors strategy could also be effectively employed to target specific customer segments. It could also be used to provide services to business partners or consumers based on custom SLAs.

The future work would involve the following:

- Understanding the dependency of one flavoring aspect on another (e.g. impact of promotions over pricing) and providing a way to specify this dependency.
- Developing MOF2 to Text Language (MTL) based templates for converting policy models to open-standard specifications.
- In addition to using service flavoring to differentiate services, providers could also package service offerings with other related services to augment their value and convenience of use. The related services could be either from them or from their business partners. For example, the Ship Item service could be packaged with a free Shipment Pickup service, which picks up shipment from the premises of the customer to create a Service Bundle. In our future work, we would enhance the service flavoring strategy with the service bundling strategy to support service differentiation.

REFERENCES


22


Yendluri, P., & Yalçınalp, Ü. *Web Services Policy 1.5-Guidelines for Policy Assertion Authors.*

ABOUT THE AUTHORS

**Harshavardhan Jegadeesan** works in the Research & Breakthrough Innovation group of SAP Labs, India. His areas of interest include enterprise service-oriented architectures, enterprise systems and business process platforms. He can be reached at harshavardhan.jegadeesan@sap.com

**Sundar Balasubramaniam** is an Associate Professor of Computer Science at BITS, Pilani. His current research interests include Service Oriented Architectures, Formal Methods and Real Time Systems. His past research and development experience includes design and development of Compilers, research and development on File System Synchronization, Mining and analysis of User Behavior on the web, research on Information Retrieval techniques for web portals, as well as research, design, and development of software for Automatic Management / Provisioning of large scale IT infrastructure. He can be reached at sundarb@bits-pilani.ac.in