

A Management Framework for Service Personalization

Govindan Ravindran^{1,2}, Muhammad Jaseemudin² and Abdallah Rayhan²

¹ SOMA Networks, Inc, 312 Adelaide St W, Toronto, ON M5V 1R2, Canada

² Dept. of Electrical and Computer Engineering, Ryerson University, Toronto, Canada
gravin@somanetworks.com, jaseem@ee.ryerson.ca, rayhan@ee.ryerson.ca

Abstract. Content Network, also referred to as Content Distribution Network or Content Delivery Network, is an overlay network of caches and web servers between content providers' origin servers and end users. It allows content providers to move content closer to end users thereby improving content availability and user access latencies. Recently, value-added content delivery has become both economically and technologically feasible with the advent of last-mile broadband access. Of particular interest is *Service Personalization* that refers to the process of delivering personalized services, that operate on and provide value addition to the basic content based on end-user service and device profile information. Examples of personalized services include virus scanning, content adaptation based on subscriber bandwidth and device capability, request and content filtering, and localization services. In this paper, we propose a framework for managing service personalization in a content network. We believe such a service management framework is essential in increasing the scale and reachability of service personalization and in improving the reliability and availability of content services. The framework builds on the IETF-proposed OPES (Open Pluggable Edge Services) model by adding two components, namely, a Service Manager and an Authorization Server, to automate the service personalization process. The service manager is involved in all phases of service personalization management including subscriber management, authorization of content service requests, service layer fault management, and service layer accounting. The authorization server, on the other hand, collects and maintains subscriber profile, generates accounting records and performs service authorization for end-users.

Keywords. Service Management, Content Network, Service Personalization, Subscriber Management, Open Pluggable Edge Services, Content Services Network.

1.0 Introduction

Content Network (CN), also referred to as Content Distribution Network or Content Delivery Network (CDN), is an overlay network of caches and web servers between content providers' origin servers and content consumers or end-users. The content network allows content providers to move content closer to end-users by replicating content in intermediate cache servers [7]. This results in i) improved content availability, ii) decreased user perceived latencies during content access, and iii) reduced bandwidth demands and web processing loads on origin servers. Traditionally, web-caching proxies [6] are used to cache content closer to end-users. The introduction of content networks, however, helped content providers to meet the growing demands without placing cache management and distribution burden on

them. This is achieved by delegating the authority to content networks to act on behalf of and in close co-operation with content providers.

The content network is traditionally associated with caching and delivery of raw content to end-users without any modification or adaptation [7], [10]. Considering the potential limit on the revenue of basic content delivery, the content network operators look for ways to value add services (on behalf of content providers), such as adapting content to suit end-user device capability, in an efficient and scalable manner. With recent technological advances in last-mile broadband access, the delivery of enriched and value-added content has become both economically and technologically feasible. An overlay network on top of a traditional content network infrastructure can realize the delivery of enriched content, by content adaptation and modification. Such a content services overlay has been referred to previously as *Content Services Network* (CSN) and *Open Pluggable Edge Services Network* (OPES). In a CSN, a set of Application Proxy Servers collaborate among themselves and with content providers' origin servers and user agents to deliver enriched content [8]. In an OPES network, *surrogates* interact with external *call-out servers* to provide value-added services to end-users¹ [3]. With content services overlay the content provider is no longer directly involved in managing content services and delegates the authority to deliver enriched content to the content networks.

One emerging content service is *Service Personalization* that refers to the process of delivering personalized services to end users based on individual service and device profile information [1]. Examples of such personalized services include virus scanning, content adaptation based on subscriber bandwidth and device profiles, request and content filtering and localization services.

In this paper, we build on the proposed OPES model by adding a *Service Manager* and an *Authorization Server* components to automate the process of delivering personalized services to end users. The service manager and authorization server interact with OPES intermediaries and content provider origin servers to perform service layer fault, performance and accounting management.

The rest of the paper is organized as follows. In Section 2, we discuss related work in general and the OPES model and its components in particular. In Section 3, we discuss the service personalization process. In Section 4, we present and discuss management framework for service personalization and present scenarios for subscriber management, service authorization, service layer fault, performance, and accounting management. In sections 5, we discuss scalability and performance of the service personalization network in general, and a broadcast notification mechanism to improve the scalability in our management framework in particular. We conclude in Section 6.

2.0 Related Work

2.1 Content Service Network

Content Services Network (CSN) [8] was proposed to allow content transformation and processing as an infrastructure service accessible to end-users. A CSN layer can

¹ *Surrogate* and *call-out servers* are explained in Section 2.

be considered as an overlay built around CNs that interacts collaboratively with user-agents (end-users), content origin servers, and other network intermediaries in the content delivery process to provide value-added services. Both the pre-distribution and post-distribution services in the CSN model are static in nature and lacks dynamic content adaptation and service delivery based on subscriber and content profiles as in the service personalization model. Furthermore, the CSN model does not describe service layer management in general, and service layer fault, performance, and accounting management in particular.

2.2 Open Pluggable Edge Services Network

The Open Pluggable Edge Services (OPES) model, proposed by the IETF OPES Working Group is a form of content services overlay network [3]. The OPES model defines the surrogate and the call-out server intermediaries in the content path. The OPES also defines an *Admin Server* whose primary purpose is to authenticate and authorize policy rule authors and service module authors. Once the authors are authenticated and authorized, policy rules and service modules can be downloaded into admin server and then into surrogates and/or call-out servers. To prevent unauthorized content processing and service delivery, OPES is constrained to provide services that are only authorized either by a content provider or an end-user.

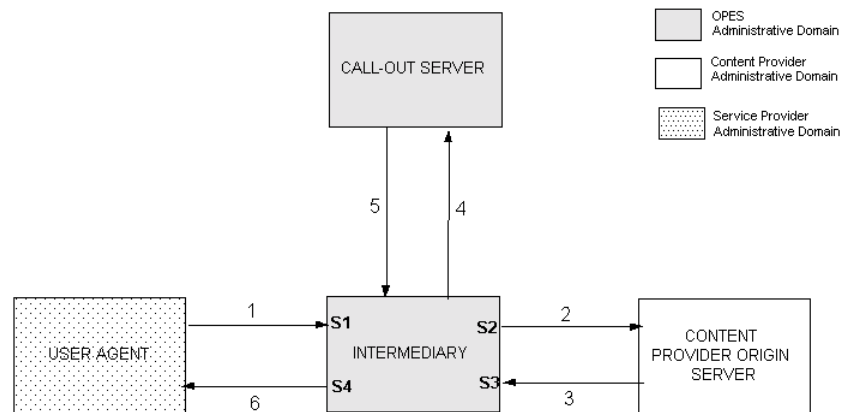


Figure 1: Service execution points in an OPES network

Figure 1 illustrates a typical data flow for an HTTP content service request:

1. The request for content from a user agent arrives at a surrogate through a requesting routing mechanism² (1).
2. The surrogate retrieves content from its local store if it has the valid copy of the requested content. Otherwise, the surrogate establishes a session with content provider's origin server to retrieve the content (2), (3).

² The user request is routed to a surrogate that most likely has the requested content and/or is located closer to the end-user. Sometimes content requests are routed to a surrogate based on additional factors such as surrogate load factor, surrogate administrative state, etc.

3. The retrieved content is then vectored to an external call-out server for processing (4), (5).
4. The enriched content is delivered to the user agent (6).

Figure 1 also shows the four common processing points in the OPES surrogate. The processing points 'S1' to 'S4' represent locations in the round trip message flow where policy rules can be evaluated against content request and response messages that trigger service module executions [4]. Depending on the service type, rules can be evaluated at any of these processing points. For example, for a pre-distribution service³, rules are evaluated at 'S3' to add value-added services to content before it is stored in the cache. For a post-distribution service⁴, rules are evaluated at 'S4' to add value-add services just before the content is delivered to end-users. Though content provider rule modules can be evaluated at any of the processing points, there are constraints placed on which service execution points end-user rule modules can be evaluated [3].

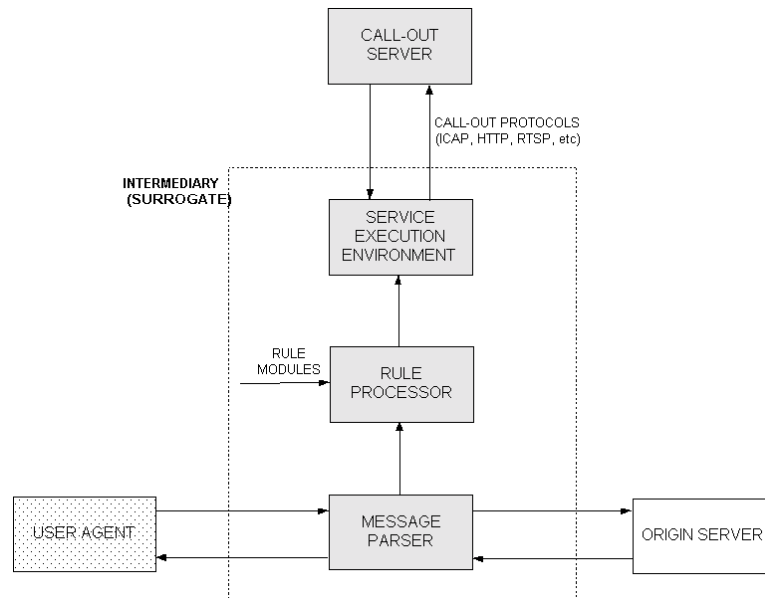


Figure 2: Rule-based service execution in an OPES surrogate

Figure 2 presents the core components of an OPES surrogate. The surrogate vectors content flow to service modules through a message parser and rule processor. The message parser traps content for processing and sends it to the rule processor, which evaluates parsed content against policy rule modules⁵ and invokes execution of

³ A pre-distribution service is performed on content before it is cached in the surrogate and distributed to end-user.

⁴ A post-distribution service is performed on content just before it is delivered to end-user.

⁵ The rule modules are written in a standard policy rule language such as the proposed Intermediary Rule Mark-up Language.

service module(s) when certain rules are fired. The service modules run either in surrogate's local service execution environment (referred to as *proxylets*) or in a remote service execution environment in an external call-out server. In the latter, the content requests and/or responses are encapsulated in an HTTP or ICAP (Internet Content Adaptation Protocol) PDU and forwarded to the call-out server for processing [4], [5].

The admin server, on the other hand, primarily provides authentication and authorization services for policy rule authors. The admin server is not located in the content path and therefore is not involved in any real-time service delivery management. The policy rule authors, once authenticated and authorized, download policy rule modules in the admin server from where they are distributed to surrogates and/or call-out servers.

3.0 Service Personalization

The goal of service personalization is to provide the means for retrieving the optimal variant of content (as response to content requests) based primarily on (i) subscriber profile including service preferences and end-device capabilities, (ii) content profile, and (iii) content provider policies.

There are several advantages in delivering personalized services from a content services overlay infrastructure such as the OPES network:

1. It mitigates the reach and scalability concerns of providing service personalization from centralized servers since the content provider has to collect and maintain subscriber and device profiles for potentially a very large number of subscribers.
2. The content provider can delegate certain functions such as collection and maintenance of subscriber information, service authorization, and generation of service detailed records to content network operators.
3. The content provider can simply act as a policy decision point for service execution. For instance, the content provider, through content and service policies, can place constraints on the processing of content. The content services network such as OPES can act as a policy execution point for content provider's content and service policies.
4. Because surrogates are delegated to operate on behalf of and often in close co-operation with one or more content providers, they provide an ideal platform for content aggregation from a select set of content sources. This allows for generation of rich variety of dynamic content in a distributed and scalable fashion.
5. Surrogates provide the widest possible audience of subscribers for a given set of content when compared to caching proxies that are often located at ISP domains.

A subscriber profile may include description of device capabilities, access rate, accounting information, and service subscriptions such as content filtering service, translation and localization services. Content profile, on the other hand, describes content type information and policies for acceptable content manipulation. For instance, a content profile may dictate policies on what type of services that can be applied on the content. Content provider service policy help to choose optimal transformation for a given content. Subscriber and content profile together with content provider service policies contain all information needed to make any personalization decision.

There are two ways of generating optimal or the most appropriate content variant for a subscriber:

1. Modifying or transforming content (retrieved from a content source) to completely or most closely fit subscriber's profile given the content profile and content provider's service policy. Service modules (in call-out servers) and/or proxylets (in surrogates) carry out the transformation of content. The service modules and proxylets take base content as input and generate transformed content suitable for delivery.
2. The content source may have multiple versions of the same content and therefore generating a content variant for a subscriber reduces to the process of selecting the most appropriate content version. For instance, in HTTP Web sites, authors are allowed to store multiple versions of the same information under a single URL. *Transparent Content Negotiation* (TCN), a proposed mechanism to select the best appropriate variant of the content, is layered on top of HTTP and provides a mechanism for automatically selecting the best content variant when the URL is accessed [11].

4.0 Management Framework for Service Personalization

We present in this section a management framework for the service personalization model described earlier. The management framework enables and automates the service personalization process thereby increasing its reach and scalability. The main functions of the proposed framework include:

1. *Subscriber Management* involving collection and maintenance of subscriber service and device profile information, authentication and authorization services for end-users, and communicating with AAA servers in other administrative domains for end-user authentication and accounting purposes.
2. *Fault Management* involving anticipating and reacting to call-out servers failure in real-time.
3. *Performance Management* involving real-time collection of load and usage statistics from call-out servers and using the data to decide primary and alternate servers for service delivery mainly for load balancing and for resource optimization of call-out servers.
4. *Service Accounting* involving generation of service detailed records that are forwarded to the content provider's billing server for processing and invoice generation.

The service personalization network we describe builds on the OPES model by adding service manager and authorization server components. Figure 3 describes various components and data flow (for a content adaptation service) in a service personalization network:

1. A user agent request for a personalized content is routed through a request-routing mechanism and arrives at a surrogate at the service execution point S1 (1).
2. The surrogate evaluates the content request against policy rule modules authored by the content provider to determine whether the content service requested requires authorization. If an authorization is required, the content is forwarded to the service manager for authorization (2). For instance, the surrogate could send an HTTP

GET or POST request to the service manager with the original content request passed in either on the URL string or as an HTTP payload.

3. The service manager in turn sends an authorization request to the authorization server (3). The authorization server consults subscriber profile repository to determine services the end-user is subscribed to and authorized to receive. Optionally, the authorization server can contact the AAA server in end-user's access network for user authentication. It then sends an authorization response back to the service manager along with user's service and device profiles (4).
4. The service manager combines user's service preferences, content profile information, content provider's service policy, along with call-out servers' load and usage statistics to generate a policy rule module for service execution in call-out server(s). The service manager then uploads the policy rule module as part of the authorization response (5).
5. The surrogate after receiving service manager's authorization response retrieves content either from the local store or from the content provider's origin server (6), (7). The content will then be evaluated against the uploaded policy rule module at the service execution point 'S4'. The policy rule module will trigger service module executions at one or more call-out servers for content processing before it is delivered to the user agent (8), (9), and (10).

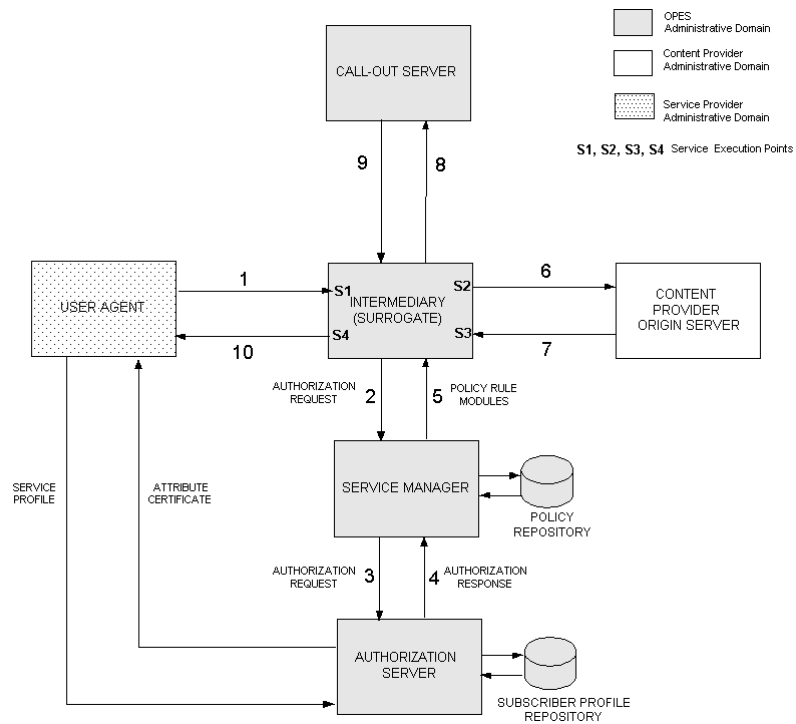


Figure 3: Components of a service personalization network

4.2 Policy Rule Modules

A rule processor on the surrogate matches rules by evaluating rule conditions against content properties and system and environment variables. A service module or proxylet is invoked based on the specified rule actions in all matching rules. The rule modules can be written using a policy specification language such as the proposed *Intermediary Rule Mark-up Language* (IRML) [2]. The policy rule modules processed in our framework include the *per-user* and *request-authorization* rule modules.

The per-user rule module is generated at the service manager and authorized by the end-user and/or the content provider whereas the request-authorization rule module is generated and authorized by the content provider. The per-user policy rules specify services that reflect the intent of the end-user and content provider. The content provider, through content profile and service policies, identifies conditions under which content is transformed. In addition, the content provider specifies optimal content transformation processes for various services. The end user, on the other hand, specifies preferred services through service subscriptions. Figure 4 shows an example per-user policy module written in IRML [2].

```
<?xml version="1.0"?>
<rulemodule xmlns="http://www.rfc-editor.org/rfcxxxx.txt">
  <author type="delegate">
    <name>service-personalization</name>
    <contact>rule-info@cdn.com</contact>
    <id>www.cdn.com</id>
  </author>
  <ruleset>
    <authorized-by class="content-consumer" type="group">
      <name>GR</name>
      <id>www.comcast.com/irml-groups/vs-subscribers</id>
    </authorized-by>
    <protocol>HTTP</protocol>
    <rule processing-point="4">
      <execute>
        <service name="McAfee Virus Scanning Service" type="primary" >
          <uri>icap://mcafee.spn.com/vscan</uri>
        </service>
      </execute>
    </rule>
  </ruleset>
</rulemodule>
```

Figure 4: Service Manager generated per-user policy module.

The rule module shown in Figure 4 is authored by the delegate on behalf of an end-user. It has a single rule-set authorized by the end-user. It specifies content request/response protocol, service execution point (on the surrogate), and the service module(s) to run. In this example, invoking a service module at the call-out server 'mcafee.spn.com' provides a virus scanning service. We can see that a second rule-set (authorized by the content provider) can be easily added to our example that overrules the service module specified in the end-user ruleset by specifying another service module in a different call-out server.

The request-authorization rule-set is processed at the service execution point 'S1'. It contains the property 'name' attribute that specifies the message property to match and the 'matches' attribute that specifies the message property value to match against.

The content request URI that matches the specified property value is forwarded to the service manager for authorization.

4.1 Service Manager Interaction

In order to perform service layer management, the service manager interacts with several server components located both within and outside its administrative domain. Figure 5 illustrates service manager interaction with various server components located both in its local administrative domain and in other administrative domains:

1. The service manager generates per user policy rule modules based on the subscriber profile information it receives from the authorization server and the content and service policy from content provider's administrative/policy server. The service manager responds to authorization requests from surrogates by uploading per-user policy rule modules. The per-user policy rule module specifies service module invocations for content processing to meet end-user's service profile.
2. The service manager receives load and usage statistics, and administrative state information from call-out servers. The data is used in computing the primary and alternate call-out servers for service module executions.
3. Call-out servers send 'registration' messages to the service manager when they come up for the first time and when new service modules are loaded from the admin server.

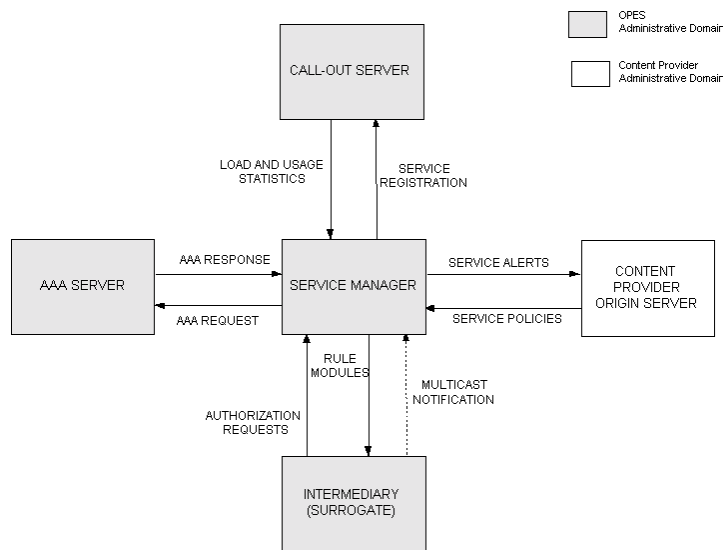


Figure 5: Service manager interaction in a service personalization network

4.2 Content Access and Interaction with AAA Services

The end-user is initially authenticated and authorized for network usage by the AAA server in his ISP's access network before his content requests are forwarded to

the service personalization network. For instance, in dial-up Internet access, the end-user provides AAA credential (e.g. user NAI, key, etc) to the Network Access Server (NAS) during the login process. The NAS acting as the AAA attendant communicates with the AAA server located in the access network to perform user authentication and authorization. Due to static trust relationship between the ISP and the service personalization network, content request from an end user arriving at a surrogate is not usually authenticated. Exception to this occurs when a content request arrives from a roaming user in which case the authorization server authenticates the user with the AAA server of the User Home Organization [12].

Content request identifies content by its URN embedded in the URL [9]. The users may subscribe to content services by using protocols such as the AAA protocol or the proposed ISDP protocol [8]. In the former, the user directly connects to a service personalization network's authorization server to input his service profile. As a token of subscription the user obtains *Attribute Certificate (AC)*, which he includes in his subsequent requests. The AC is used in part to authorize the user for services.

Another level of interaction may happen between the authorization server and the ISP's AAA server to retrieve relevant network information, which may be necessary for providing some services. For instance, to deliver location-based services the service manager may need user location information. Alternatively, if the host itself is equipped with GPS, the location information can be embedded in the request.

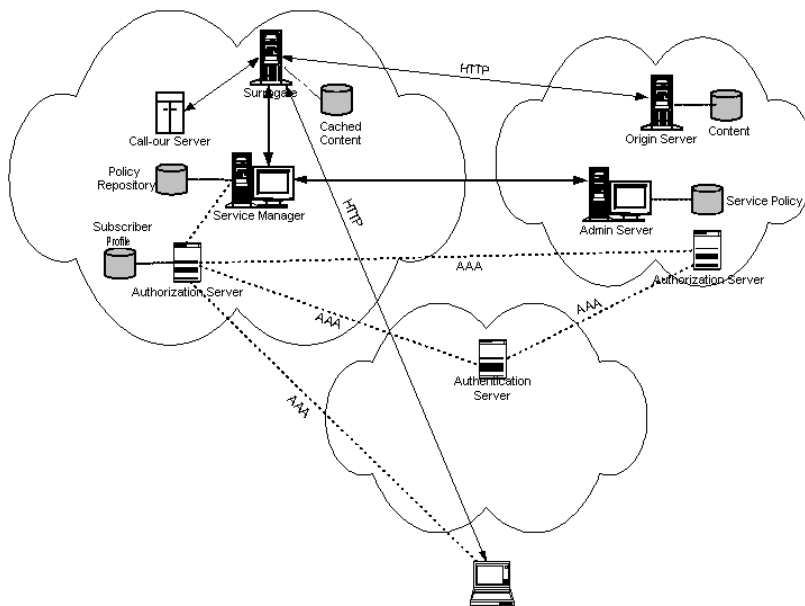


Figure 6: Service personalization network interaction

Figure 6 illustrates the interaction of the service manager with surrogates, the authorization server in the OPEs administrative domain, and the policy/admin server in the content provider's administrative domain. The authorization server communicates with AAA server in the access network to authenticate user and to

retrieve access network profile. The authorization server also communicates with the AAA server in the content provider domain to retrieve user service authorization information. Figure 6 also illustrates the end-user interaction with the authorization server to subscribe to various content services.

4.3 Subscriber Management

The authorization server primarily carries out subscriber management. The user before initiating requests for content services for the first time typically goes through a service registration process with the content provider. During this process he will be redirected to the authorization server of a service personalization network. The authorization server collects user's service preference, subscription, and accounting information on behalf of the content provider and stores them locally in the subscriber profile repository. For every subsequent change in user's profile he will be directed to the same authorization server to update his profile.

After the initial registration process, the user's content request arriving at a surrogate will be redirected to the service manager. The service manager acting as an attendant issues the authorization request. The authorization server responds by sending user's service, network, and device profile information to the service manager. The service manager uses this profile information along with content provider's service policy to generate per-user policy rule module(s). Optionally, the authorization server communicates with the AAA server of user's access network for user authentication and with content provider's AAA server for user service authorization. Refer to Figure 3 for the subscriber management data flow.

4.4 Fault Management

For each service, the per-user rule module specifies a primary call-out server (for the service module execution) and optionally one or more alternate call-out servers. There are two failure scenarios of the primary call-out server:

1. The primary server fails to execute the service application and returns an error code to the surrogate. The surrogate then invokes a similar service application on one of the alternate call-out server.
2. The primary server is down. In this case the surrogate times out waiting for the service module response and automatically invokes another service module in an alternate call-out server.

The service manager determines the primary and alternate call-out servers to optimize server resource usage and to balance the processing load across multiple call-out servers. The decision takes into account real-time data such as call-out servers' load and usage statistics, and their administrative state information.

4.5 Service Accounting and Content Provider Interaction

The authorization server is a natural place to log service accounting records as it has access to subscribers' service subscription and billing information as well as it is aware of subscriber content requests. This enables the authorization server to generate a *Service Detailed Record* (SDR) for each content request. The SDRs are then sent to content provider's billing server to generate invoices. The content provider also receives service alerts notifying any exceptions during service delivery.

5.0 Service Management Scalability and Performance

The OPES framework provides value added content services but adds call-out processing to the content response. In the proposed service management framework, the service manager is involved in all content requests that require authorization. This adds another processing step in the content request path. The latency introduced by the call-out processing and the service management framework amortizes over the content delivery time, but could add significant overhead to content delivery depending upon factors such as content size, available bandwidth, etc. One way to reduce the management overhead, especially for a large subscriber base, is to use a broadcast surrogate notification mechanism that prevents the involvement of service manager in most (if not all) content requests.

Figure 7 illustrates the broadcast notification mechanism. When a content request arrives first time from a user it is directed to the service manager for authorization and for downloading the per-user rule module. The authorization issued by the service manager is valid for a certain amount of time. Subsequent content requests that arrive within this time period do not require service manager's authorization and uses previously cached per-user policy module. Any future change in the user's service profile will be notified to all surrogates. Once notified, surrogates will retrieve the new rule module from the service manager. Since user profiles change infrequently, notifications are handled less often than content requests. Hence, a simple broadcast mechanism for notifications will work well. The service manager in this case merely acts as a dispatcher: it receives notification from the authorization server when a user profile is changed and broadcasts it to all surrogates. The surrogates having stale copy of the rule modules subsequently download them from the service manager.

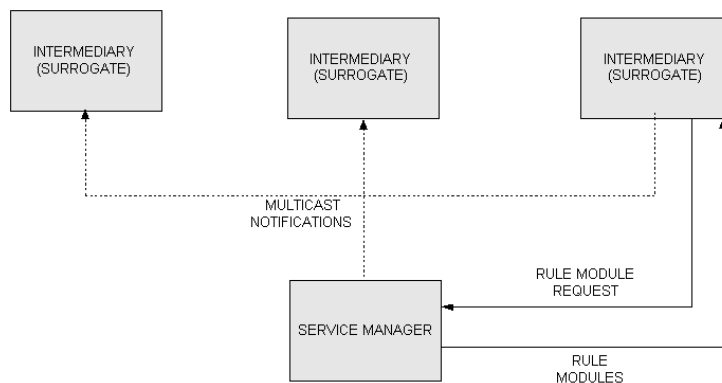


Figure 7: Multicast notifications in a service personalization network

6.0 Conclusion

We presented in this paper a management framework for enabling and automating the delivery of personalized services. The management framework builds on the proposed OPES framework by adding service manager and authorization server

components. The service manager generates per-user policy rule modules while the authorization server collects and maintains subscriber service and device profile information. The service manager interacts with content provider's admin server to retrieve content and service policies and with the authorization server to retrieve user profile information. The service manager then combines user profile with content and service profile to generate per-user rule modules that are evaluated against content response to deliver personalized services. We discussed the scalability and performance issues of the service management framework, and proposed a simple broadcast notification mechanism that avoids processing at the service manager for every content request. We believe such a service management framework is essential in increasing the scale and reach of service personalization and in improving the reliability and availability of content services.

In future, we would like to implement our framework to understand the applicability and viability of this approach. Another important issue worth investigating is the content delivery to mobile users, which we would like to pursue next.

References

- [1] A. Barbir, N. Bennett, R. Penno, R. Menon, J. Mysore, and S. Sengodan, "A Framework for Service Personalization", draft-barbir-opes-fsp-00.txt, <http://www.ietf-opes.org/>
- [2] A. Beck and A. Hoffman, "IRML: A Rule Specification Language for Intermediary Services", draft-beck-opes-irml-02.txt, <http://www.ietf-opes.org/>
- [3] G. Tomlinson, R. Chen, M. Hoffman, R. Penno, and A. Barbir, "A Model for Open Pluggable Edge Services", draft-tomlinson-opes-model-00.txt, <http://www.ietf-opes.org/>
- [4] L. Yang and M. Hoffman, "OPES Architecture for Rule Processing and Service Execution", draft-yang-opes-rule-processing-service-execution-00.txt, <http://www.ietf-opes.org/>
- [5] "ICAP the Internet Content Adaptation Protocol", draft-elson-opes-icap-02.txt, <http://www.ietf-opes.org/>
- [6] J. Wang, "A survey of web caching schemes for the Internet", ACM Computer Communication Review, no. 5, vol. 29, pp. 36--46, October 1999.
- [7] M. Green, B. Cain, G. Tomlinson, S. Thomas, P. Rzewski, "Content Internetworking Architectural Overview", draft-ietf-cdi-architecture-01.txt, <http://www.ietf.org/>
- [8] W. Ma, B. Shen, and J. Brassil, "Content Services Network: the Architecture and Protocols", In Proceedings of the 6th Int'l Web Caching Workshop and Content Delivery Workshop, June 2001.
- [9] R. Fielding, J. Gettys, J. Mogul, H. Frystyk and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2068, <http://www.ietf.org/>
- [10] B. Krishnamurthy, C. Wills, and Y. Zhang, "On the Use and Performance of Content Distribution Networks", ACM SIGCOMM Internet Measurement Workshop, 2001.
- [11] K. Holtman and A. Mutz, "Transparent Content Negotiation in HTTP", RFC 2295, <http://www.ietf.org/>
- [12] J. Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege, D. Spence, "AAA Authorization Framework", RFC 2904, <http://www.ietf.org/>