Have it Your Way: Personalization of Network-Hosted Services

Richard Hull, Bharat Kumar, Arnaud Sahuguet

Network Data and Services Research
Bell Laboratories, Lucent Technologies
Murray Hill, NJ 07974
{hull,bharat,sahuguet}@lucent.com

Abstract. This paper surveys recent trends in network-hosted end-user services, with an emphasis on the increasing complexity of those services due to the convergence of the conventional telephony, wireless, and data networks. In order to take full advantage of these services, private and corporate users will need personalized ways of accessing them. Support for this personalization will involve advances in both data and policy management.

1 Introduction: The Challenge of Too Many Services

With the emergence of the World Wide Web the realm of end-user accessible, network-resident data services has become extremely rich and varied. Voice services have lagged behind the web in terms of their complexity, due to their heritage in monolithic circuit-based networks. But voice services are catching up, as packet-based voice becomes more common and gateways are being made between the telephony network and the internet. For example, voice access to web-resident information (e.g., weather, email, personal calendar) is now becoming available through several service providers. And through standards such as Parlay [Par], internet-based services will have broad access to a variety of telephony network information and services (e.g., presence and location awareness, billing, call control), enabling third-party application developers to create a multitude of services that freely mix web-style and voice-based functionalities. Add to this the variety of mobile networks (2G, 3G, WAP, 802.11, ...), the increasing use of instant messaging, and the emergence of SIP [SIP] and intelligent terminal devices, and we see that most end-users will soon be deluged with a broad variety of services and ways to combine them, along with a multitude of details about all the different ways to contact their friends and associates.

Due to the wide-spread use of mobile devices the "collective network", or more precisely, the family of circuit-based, wireless, and data networks, is truly becoming a *ubiquitous* feature in our daily lives. Through a combination of mobile and fixed devices, a substantial portion of the world's population is reachable in an instant, and they can access the network whenever they want. Further, when a mobile device is on, both its status (idle, on-call, etc.) and approximate location can be tracked. And many of us spend one or more hours working with a network-connected computer. As a result, the collective network has a tremendous wealth of up-to-date information about where people are and whether they are available for communication.

So what will communication-oriented services look like in the future, and why do they need to be personalized? We mention here four key trends.

Making it easier to request services: An important trend is to make the collective network to be more "intelligent" with regards to providing typical services. A simple example of this is to support so-called "call by name" as opposed to "call by number". With call by name a user identifies who they want to call by that person's name, and lets the network determine which number or numbers to actually call, in order to make the connection. More generally, an "intelligent" network could keep track of end-user address books, preferences, appointment books, and habits, and permit the user to request services through high-level directives (e.g., "call Jerome" or "arrange a flight to Chicago after my last appointment on this Wednesday"), rather than the explicit style of service requests used today. The network should also be aware of the device or devices that the end-user is currently using or has available, and tailor information delivery to those devices appropriately. It should be noted that as technology surrounding the web services paradigm matures (e.g., to incorporate automated reasoning for service discovery and composition), then both end-user requests and automated notifications will be able to trigger increasingly sophisticated combinations of services, which implies the increasing complexity of taking individual profile data and preferences into account.

Automated notifications: We are already seeing automated notification services in limited areas (e.g., flight information, financial market events, and in e-commerce notification if your order gets shipped). These will proliferate as it becomes easier for service providers and businesses to provide them, and for end-users to launch and customize them. In the future we should be able to register for a broad variety of event notifications, specifying also under what conditions a notification should be issued, and how and when the notifications should be delivered (e.g., urgent ones to a pager or cell phone, less urgent ones to voice mail or email). (As an example of notifications gone awry, a well-known bank offered the service of automatically paging clients when their personal monthly 401K statements had been emailed. In general, those emails were processed in bulk during slow periods in the bank's schedule, i.e., at around 2 in the morning!)

Support for collaborative activities: Appropriate use of the collective network can substantially enhance the effectiveness of long-term (e.g., multi-day or multi-month) collaborations. This can range from enhancing simple conference calls into flexible multimedia/multi-device conferencing, to supporting persistent "chat sessions" that provide multiple modes of interaction through time, including the abilities to share data files easily, to register for event notifications relevant to the collaboration, to track whether participants are available, and to contact participants immediately when this is deemed appropriate. This notion of collaboration extends also to relationships between enterprises and customers, e.g., between a bank and someone applying for a mortgage loan, where a variety of information must be passed, sometimes under time constraints. More generally, network-hosted services can become as rich as the collaborations that people and/or enterprises participate in.

Keeping control in the hands of the end-user: As noted earlier, the collective network has access to a vast amount of information about individual users. This information should be revealed to others only by permission. Further, the individual user will typically be willing to share certain information with certain other individuals, based on a variety of circumstances (e.g., business colleagues might be given access during business hours while family might be given access at essentially any time). It is important that users be able to express their preferences accurately, and that the collective network honor their right to privacy.

Achieving this vision of a more intelligent collective network involves advances in a number of diverse technologies, including data management; preference and policy management; workflow and collaborative systems; distributed systems; standards for sharing information and control across networks and devices; new forms of call models and session management; and continued work on natural language interfaces. The current paper is focused on challenges arising in data and policy management.

2 Managing User Profile Information

Network-hosted services revolve around providing information to end-users, providing connections between end-users with each other or with automated systems working on behalf of enterprises, and enabling end-users to invoke commercial, financial or other services outside of the network proper. Simply providing these services entails the storage and use by the collective network of a broad array of information about end-users, including, e.g., the correspondence between users and their devices, awareness that mobile devices are on data and/or circuit paths for establishing connections to them, and awareness of the billing model (e.g., pre-paid vs. post-paid, and the billing plan). A key element of simplifying and personalizing these services is for the collective network to store and use information specific to individual end-users and relevant to the services being provided, and to flexibly share this information across the components within networks, and between different networks.

Currently, end-user information is scattered across the networks, and typically organized by device rather than end-user, with primary emphasis given to the performance of individual services rather than enabling sharing of information between services and networks. More specifically, in the circuit telephony realm information about subscribers and how to bill them is maintained in so-called "central offices", which are generally opaque to internet-based applications. In the wireless domain a network of "Home Locator Resources" (HLRs) hold real-time data about device location and status, and have ready access to pre-paid billing information (so that service can be terminated if the account becomes empty). Only recently is HLR data becoming more freely available, through standards such as Parlay. In the internet user data is managed in an extremely loose and distributed manner. The end-user relevant data managed by the various networks is largely governed by standards bodies (e.g., ITU-T [ITU] and more recently Parlay for the wireline telephony network, 3GPP [3GPa] and 3GPP2 [3GPb] for the wireless networks, and IETF [IET] for the internet).

What data needs to be maintained and shared for simplifying and personalizing network-hosted services? Web portals such as MyYahoo! provide a simple form of personalization, allowing users to specify what locales they would like weather information for, or what kinds of news stories should be presented. Services involving realtime communication can involve more intricate personalization. For example, for "call by name" the network needs to hold a mapping from end-user abbreviations (e.g., "my boss") to unique person identifiers, and from these to the family of devices used by the identified person. Realtime information about presence, availability, and location should be factored into the "call by name" service, to help gain access to the callee. And this realtime information must be filtered against the callee's preferences, e.g., based on her schedule (e.g., working hours vs. personal hours), the caller, and the current situation (e.g., working against an urgent deadline, on vacation, in transit, open to interruption). Over time, the preferences that can be stored will become more and more intricate.

Convenient sharing of profile and preference information between networks, portals, and services is not currently supported. An important pre-cursor to this sharing is to provide support for secure and controlled sharing of such information across networks and between enterprises. The Liberty Alliance consortium [Lib] is working to develop standards in this area. A key paradigm is to support the notion of "network identity", enabling a single sign-on to the network, which can be used transitively to authenticate user-specific access to services across the collective network. The development of this paradigm must take into account issues around enterprise firewalls, since a large population is based within enterprises and use the network for conducting enterprise-directed activities. Importantly, a single sign-on mechanism can provide the vehicle for passing user-specific profile and preference information to services, although care must be taken to share with a given service only information that the user has permitted for that service.

What mechanisms and paradigms should be used to share user profile and preference information across the various networks, and across a multitude of services? The basic answer is to generalize the federated database architecture [HM85,SL90] that was developed in the 70's and 80's. That architecture, developed primarily for data sharing within an enterprise context, acknowledges that multiple relatively autonomous sub-organizations will maintain their own databases, and proposes essentially that a *virtual* global schema be created for sharing data between those organizations. The global schema can be viewed as providing a conceptual single-point-of-access for all of the data, without requiring the different sub-organizations to store their data in a specific format or technology.

Two generalizations of the federated model are needed to effectively support sharing of user profile information. The first arises in the context of a single service provider, where data is often replicated in network components to achieve satisfactory performance. Unlike the usual federated paradigm, these network components are embedded devices that do not provide data management support typical of modern DBMSs. Further, updates may happen in both directions – a change to call-forwarding information might be received from a cell phone through an embedded copy of data, while a change to a billing plan will typically come from a database-resident copy of data and be passed to relevant embedded components. As user profile information becomes more sophisticated, the inherent distribution and complexity of the storage architecture will make it increasingly difficult and expensive for applications and services to access it. For this reason, following the spirit of the federated architecture, an abstraction layer should be introduced, to support a conceptual single-point-of-access for (realtime and static) user profile information. Services and applications can then access user profile information in a uniform and consistent manner. Because performance is critical in network-hosted applications, new techniques will be needed to provide adequate performance underneath this abstraction layer.

The second generalization of the federated model involves allowing for the graceful interaction between multiple federations. Each host must be able to access and use data that is coming from "outside", i.e., from other hosts and entities in the collective network. One approach to support such sharing is illustrated by the HLR. Sharing of data between HLRs is crucial for support of cell-phone "roaming", and standards have been developed to support this data sharing. But standards will be more elusive in other domains. A wireless service provider (WSP) may want to support the ability of an end-user to maintain an address book in the network, and to synchronize it with the address book on their cell phone. While some users will be satisfied with having the WSP hold the primary copy of their address book, other users will prefer that an independent third party, e.g., a web portal, hold the address book. In this case, the portal and the WSP may both hold federated schemas for their internal data, but also need to share data between those federated schemas. Unlike the original federated architecture, there is no over-arching organization in place to maintain a federated schema between these two entities. Richer examples arise when data relevant to providing a single service is spread across multiple hosting organizations, e.g., when the wireless network holds presence information, a web portal holds preference information, an enterprise is providing a particular service for a fee, and another enterprise (or a network operator) is providing billing services.

What about the data models that are and will be used? While most network-hosted data continues to reside in relational DBMSs, there is increasing usage of an LDAP representation, motivated by the fact that LDAP is rich enough and efficient for many current user profile applications, and provides flexible support for physical distribution. As user profile information becomes richer and as XML becomes more pervasive, we expect a gradual transition to the use of XML for passing profile information between network hosts. Indeed, adopting XML offers the advantages of a unified data model and unified query language, that easily encompass the relational and LDAP models and languages. Further, a standard (XACML) is being developed for access control against XML data.

Schema management will emerge as a key research area in profile management. Historically, schemas for profile databases have developed in an *ad hoc* way by different providers. More recently, standards groups (e.g., DMTF [DMT] DEN User Information Model, 3GPP Generic User Profile) have proposed or are developing standardized schemas for holding profile information. Looking forward we expect network operators to take a hybrid approach – starting essentially with a standardized schema for profile information, but adding to it as new services arise that require the incorporation of new profile information. In many cases, the new services will be developed by third-party application developers. How does the network operator ensure that the growth and evolution of the profile schema is reasonably conherent? More specifically, what tools can be developed, so that developers that need to extend the profile schema hosted in a network can (a) easily learn what is already in the profile schema, (b) avoid adding redundant elements (unless there is good cause), and (c) add new elements in a manner consistent with the rest of the schema. If the data is represented in XML format, then a possible direction is to annotate the profile schema (perhaps represented in XML Schema) with semantic descriptions of different nodes and subtrees.

3 Managing Preferences and Policies

Maintaining profile and preference information in a (distributed, federated) profile database can provide a solid foundation for extensive personalization of network-hosted services. In this section we argue that incorporating rules-based preferences and policy management into the network can provide substantially increased flexibility and ease of maintenance for supporting this personalization. We also describe several research issues raised by this approach.

The notions of policy and policy-enabled have taken many meanings. In broad terms, *policy* in connection with a computerized system refers to the logic that governs choices that the system can make; these choices may be based on a variety of factors including personalization, optimization, quality of service, etc. A system is *policy-enabled* if it provides a mechanism to change the policy in a dynamic manner, without recoding or stopping the system. There are a range of approaches to policy-enablement in current systems. One approach, called here *data-structure-driven* policy enablement, is to provide a (presumably rich) data structure for holding preference information, and provide application code that interprets the data structure. The data in the data structure can be updated at will, thereby providing dynamically changable behavior. For example, to support a "call by name" application in this manner there would be a data structure provided so that users can specify when and how they should be reachable, and when they should not be reachable. Another approach, called here *rules-based*, is to permit the specification of families of rules to define the policy, and to provide a rules engine for executing on those rules. Within this approach there is considerable variety, because there are several paradigms for rules-based systems, ranging from simple condition-action languages that do not permit chaining (e.g., as found in the IETF standards), to production system style systems (e.g., OPS5 [For81], CLIPS [CLI], ILOG [ILO]), to various styles of logic programming (e.g., Prolog, stratified, etc.) [Apt91] and other paradigms (e.g., PDL [LBN99], Ponder [DDLS01], Vortex [HLS$^+$99]). The rules-based approach, especially if chaining is supported, is typically more flexible than the data-structure-driven approach, because in practial applications, a rules language has the potential to express more than a fixed program that is interpreting values of a fixed data structure. At the extremely expressive end of policy enablement would be the use of more advanced logical systems (e.g., description logics [BBMR89] or first-order logic) to specify policy. For the remainder of this paper we focus primarily on rules-based policy enablement.

With regards to personalization of network-hosted services, it is useful to keep four distinct architectural aspects of policy enablement in mind. One aspect concerns the presentation of the policy-enablement to end-users. In most cases, because end-users are coming from all walks of life, they will be presented with a table-driven GUI for entering, viewing, and modifying their preference information. In the case of data-structure-driven policy enablement, the user-entered information will be used to populate the underlying data structure; indeed, there is typically a close relationship between that data structure and the choices presented to the user. In the case of rules-based policy enablement, the user-specified preferences will be translated into a collection of rules, which are held in a *policy database* (following here the terminology used by Parlay and other standards). A *policy execution point* is a location in the system where a collection of rules can be executed, in order to render a decision based on stored policy. And a *policy enforcement point* is a location in the system where the result of policy decisions can are used to dictate actual system behavior.

It is natural to ask at this point: if end-users are typically presented with a table-driven GUI for entering preferences, then why in practice is the increased flexibility of the rules-based approach really needed? One answer is that over time, if a rules-based approach is used then extensions to the kinds of preferences users can specify can typically be made by modifying the GUI and translation into rules, but without modifying the underlying infrastructure. Another answer is that a rules-based approach permits a common underlying infrastructure to support many different classes of users (e.g., road warriors, corporate executives, students, retirees, home-makers, emergency workers), with their respective GUIs that are based on substantially different kinds of preference information.

An important issue in policy management for personalization concerns the choice of rules (or richer logic) paradigm for specifying policies. The DMTF and IETF policy bodies, which focus on managing network and system components, have favored rules systems with no chaining: if the condition of a rule is true, then a specific action is taken at the appropriate policy enforcement point. In connection with personalization, however, the assignment of intermediate variables and the use of rule chaining can be beneficial. As a simple example, consider the "call by name" application (or any kind of calling application), and the preferences that the receiving party might specify to guard their privacy. It may be useful for the receiver to use rules to assign values to intermediate values such as "time category" (e.g., working hours, family hours, sleep hours) and "caller category" (e.g., colleague, family member, high priority caller). These variables might be used in rules that determine whether callers are routed to voice mail instead of making one or more phones ring. Set-valued intermediate variables can also be useful, so that the rules can consider a set of possibilities simultaneously (e.g., devices for contacting a person). The use of intermediate values can enable rule sets to be much more succinct than the case where intermediate values are not permitted. Finally, while intermediate variables and chaining seem useful, it is unclear how important recursive rules are for personalization applications. Restricting attention to acyclic rule sets has benefits in terms of performance, and simplifies other issues mentioned below.

We turn now to key research issues in policy management in support of personalizing network-hosted services. One key issue, of course, is performance. For many services in the telephony network users will expect response times of well under a second, which can be a challenge for typical rules engines operating in, e.g., typical production system applications. In the case of some network hosted services, at least in the near term this may be mitigated because the size of the rule set used for a particular decision is limited (e.g., for "call by name", only the preferences of the caller and callee need be examined). However, a basic issue in policy management for personalization concerns the interaction of rules with databases. For example, in connection with "call by name" a user might have rules stating that a call from any employee of the same company will be received during working hours – this might mean that an incoming call will have to be checked against a corporate database holding information about 50K or more fellow employees. Mechanisms need to be developed for executing rule sets in a way that avoids expensive database dips whenever possible. From an architectural point of view, it is important to understand a variety of tradeoffs in this area, including whether to place policy execution points "near" associated databases, and if that cannot be done, how much data to pass to the policy execution point when requesting a decision vs. having the execution point query a database for information on an as-needed basis.

Another research area concerns dynamic combinations of rules. For example, in policy languages such as Ponder and standards from IETF and Parlay, rules can have associated roles, and the roles can be used at runtime to gather together a set of rules to be used for a particular decision. In connection with "call by name" there might be separate rule sets for the caller and the callee, which are combined at runtime. And there might be additional rules that the caller's or callee's employee wish to be enforced, or that a network operator will insist upon. In cases where there is no rule chaining, it is relatively straightforward to interpret the meaning of an essentially arbitrary collection of rules. If there is rule chaining, and hence potentially complex interdependencies between rules, it may be difficult to successfully combine rules dynamically, since unexpected interactions between the combined rule set might arise. A promising direction here, at least in the restricted case of acyclic rule sets, might be to support the construction of targeted rule sets, that essentially map from selected "input" variables to selected "output" variables. We might permit combining of two rule sets if the "output" variables of one set correspond to the "input" variables of the other set. Generalizations of this are easily imagined, to permit multiple rule sets to be combined in a structured manner.

A final challenge concerns providing the ability to specify global policies in essentially one place, and to somehow manage to map them to appropriate network components to acheive maximum performance while remaining true to the semantics of the global policy. In more detail, when end-users specify preferences, they should be able to conceptualize the network as one unified system. But the rules that are created from their preferences will typically reside on, or provide decision-making for, multiple components in the network. As one example, in the Parlay standards several different components can be policy-enabled, including call control, presence awareness, location, and charging; a "call by name" capability might entail several or all of these. More generally, user preferences might be mapped to policy enforcement points at different levels of the network, e.g., to presence awareness at the level of the Parlay standards, to a softswitch for executing policy around call control, and to edge routers for enhancing the performance of streaming data associated with a multimedia call. The emergence of SIP and intelligent endpoints raises another form of distribution, e.g., the possibility that my SIP-enabled personal phone and my SIP-enabled business phone could both execute policies on my behalf – but how can these policies be coordinated? Managing policy in these kinds of distributed environments is a relatively new topic for policy research. Recent work includes [PFW$^+$02,AMN02], which develop approaches to provide forms of hierarchical structuring on rule sets, with a form of inheritance of rules from a super-entity to a sub-entity. Adapting these approaches to support personalization of network-hosted services will by challenging, in part because of the heterogeneity of the policy paradigms used by different network components, and the peer-to-peer nature of SIP endpoints.

4 Conclusions

We have seen that supporting personalization in network-hosted applications raises several research issues in both data and policy management. In data management this includes developing extensions of the federated model, new tools to support schema management and evolution, and of course, performance. In policy management this includes performance of rules engines in a new context, mechanisms for dynamic combining of rule sets, and mechanisms for working with distributed rule sets.

More generally, the area of personalization leads to a blurring of the traditional roles of databases and rules engines. In the context of personalization, a query such as "is Joe available for a call right now?" can not be viewed as a database access. Rather, it is a context- and time-specific request that will be answered by an interaction of rules engine and database, working with preferences stored in a combination of rules and data. As such, policies are used to control access to personal data according to user preferences. The converse also holds – that smart policies often require access to a variety of data. This confluence of data and policy management gives rise to a broad range of new questions in optimization, in models for distribution, and in providing structure for combining preference information from different sources.

Acknowledgements

The authors thank many of their colleagues at Bell Labs for numerous conversations that have formed the basis for much of the perspective presented here. Special acknowledgement is due to Peter Patel-Schneider, Prasan Roy, Jerome Simeon, Ming Xiong, and the late Javier Pinto. The Network Data and Services Research Department at Bell Labs is pursuing a number of research projects related to the themes described in this paper.

References

[3GPa] 3GPP. The 3rd Generation Partnership Project. www.3gpp.org.

[3GPb] 3GPP2. The 3rd Generation Partnership Project 2. www.3gpp.org.

[AMN02] X. Ao, N. Minsky, and T. D. Nguyen. A hierarchical policy specification language, and enforcement mechanism, for governing digitual enterprises. In *Proc. of IEEE 3rd Intl. Workshop on Policies for Distributed Systems and Networks (Policy2002)*, Monterey, California, 2002.

[Apt91] Krzysztof R. Apt. Logic programming. In J. Van Leeuwen, editor, *Handbook of Theoretical Computer Science, Vol. B*, pages 493–574. Elsevier, 1991.

[BBMR89] A. Borgida, R. J. Brachman, D. L. McGuinness, and L. A. Resnick. CLASSIC: A structural data model for objects. In *Proc. ACM SIGMOD Symp. on the Management of Data*, pages 59–67, 1989.

[CLI] CLIPS. CLIPS: A tool for building expert systems. www.ghg.net/glips/CLIPS.html.

[DDLS01] N. Damianou, N. Dulay, E. Lupu, and M. Sloman. The Ponder specification language. In *Proc. of IEEE 2rd Intl. Workshop on Policies for Distributed Systems and Networks (Policy2001)*, HP Labs Brisol, UK, 2001.

[DMT] DMTF. The Distributed Management Task Force. www.dmtf.org.

[For81] C. L. Forgy. OPS5 user's manual. Technical Report CMU-CS-81-135, Carnegie Mellon University, 1981.

[HLS$^+$99] R. Hull, F. Llirbat, E. Simon, J. Su, G. Dong, B. Kumar, and G. Zhou. Declarative workflows that support easy modification and dynamic browsing. In *Proc. of Intl. Joint Conf. on Work Activities Coordination and Collaboration (WACC)*, pages 69–78, February 1999.

[HM85] D. Heimbigner and D. McLeod. A federated architecture for information management. *ACM Trans. on Office Information Systems*, 3(3):253–278, July 1985.

[IET] IETF. The Internet Engineering Task Force. www.ietf.org.

[ILO] ILOG. ILOG business rules. www.ilog.com/products/rules.

[ITU] ITU-T. International Telecommunication Union – Telecommunications Standardization Sector. www.itu.int.ITU-T/.

[LBN99] J. Lobo, R. Bhatia, and S. Naqvi. A policy description language. In *AAAI*, 1999.

[Lib] Liberty Alliance. Liberty Alliance project. www.projectliberty.org.

[Par] Parlay. The Parlay group. www.parlay.org/.

[PFW$^+$02] L. Pearlman, I. Foster, V. Welch, C. Kesselman, and S. Tuecke. A community authorization service for group collaboration. In *Proc. of IEEE 3rd Intl. Workshop on Policies for Distributed Systems and Networks (Policy2002)*, Monterey, California, 2002.

[SIP] The SIP Forum. Session Initiation Protocol. www.sipforum.org.

[SL90] Amit P. Sheth and James A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys*, 22(3):183–236, September 1990.