# A Framework for Learning to Personalize Converged Services Involving Social Networks

Robert Dinoff, Tin Kam Ho, Richard Hull, Bharat Kumar, Daniel Lieuwen, Haobo Ren, and Paulo Santos
Bell Labs/Lucent Technologies, Inc.
700 Mountain Ave., Murray Hill, NJ 07974
e-mail: {rdinoff, tkh, hull, bharat, lieuwen, ren, paulo}@lucent.com
Phone: 908-582-5612

***Abstract*— The convergence of the wireline telecom, wireless telecom, and internet networks and the services they provide offers tremendous opportunities in services personalization. The Privacy-Conscious Personalization (PCP) framework, developed previously at Bell Labs, uses a high-speed rules engine during call processing to interpret a combination of incoming requests, user data, and user preferences in order to provide context-aware, requester-targeted, and preferences-driven responses to those requests (e.g., deciding whether to share a user's location with a given requester, what to show as the end-user's availability to a given requester, where to forward an incoming call). This paper describes key aspects of a new initiative at Bell Labs, called Intuitive Network Applications (INA), which will combine human factors and automated learning techniques, in order to gather and apply user data and preferences, to enable effective adaptation to end-user needs with minimal disruption to her. The focus of this paper is on applications which involve the interaction of an end-user with her social network, i.e., family, friends, colleagues, customers, etc. The paper describes (i) key requirements, (ii) a high-level architectural framework, and (iii) some specific directions currently under exploration for filling out the framework.**

## I. Introduction

Driven by innovations in the World Wide Web, and in the evolution of telecom networks towards an all-IP core, the wireline telecom, wireless telecom, and internet networks are converging. Increasingly, services that were traditionally available on one network or another will become available in a more ubiquitous manner, on any device (or devices) that the end-users happen to have at their disposal at a given time. At a more fundamental level, profile and other data (including presence, location, address books, preferences) will be available across network and service boundaries, which suggests that *in principle* services can incorporate rich, context-aware personalization based on the full wealth of information available. Achieving this goal, in the context of telecom networks with their requirements for high reliability, performance, and scalability, involves a host of issues beyond those typical of a single, isolated web-hosted application. During call and/or request processing, the Privacy-Conscious Personalization (PCP) framework [1], [2] developed at Bell Labs, uses high-speed data mediation [3] and the high-speed Vortex rules engine[1] to interpret a combination of incoming requests, user data, and user preferences in order to provide personalized responses to those calls/requests. A key challenge in the PCP framework concerns the issue of how to gather user data (e.g., buddy relationships) and preferences (e.g., under what circumstances is Michael "available" to Sally, and by which means). Expecting the end-user to fill in numerous web forms with this information, all in one sitting, is obviously untenable. This paper describes selected elements of a new initiative at Bell Labs, called Intuitive Network Applications (INA), which will combine human factors and automated learning techniques, in order to gather (and apply) user data and preferences in order to adapt effectively and with minimal interruption to the end user. The focus here is on describing (i) key requirements, (ii) a high-level architectural framework, and (iii) some specific directions currently under exploration for filling out the framework.

The PCP framework and INA approach are relevant to a broad variety of converged applications and services. In this paper we focus on services that involve relationships and interactions between an end-user and people in his social network, e.g., family, friends, colleagues, etc. A representative service is *availability*. This service is a natural extension of current presence services, such as IM presence (in which end-users or an end-user device explicitly sets a single presence indicator, such as 'present' or 'away', which is typically shared to all members of a buddy list) and network *presence* (in which raw connectivity to the network is indicated, again to all members of a buddy list). In contrast, in our view, an availability service should be (i) context-aware, (ii) requester-targeted, and (iii) preferences-driven. By context-aware, we mean that the service can take into account the end-user's current context, including dynamic information such as presence on various devices, location, and current usage of the devices, and more static information, such as buddy lists and relationships, calendar entries, perhaps corporate directory. By requester-targeted, we mean that the answer provided may depend on the person or service that is making the availability request; it is also useful to indicate degrees of availability to different requesters. By preferences-driven, we mean that the answer provided should reflect the preferences of the end-user in connection with his current context, the request, and the requester.

There are a myriad of other services where context-aware, requester-targeted, preference-driven decisions are needed. These include location sharing (both whether to share location and to what granularity), call-forwarding, various kinds of pro-

[1]This engine, now a Lucent product, is called the 'Lucent Vortex™ rules engine'; in [1], [2] the engine was referred to as the 'Houdini'.

1-4244-0166-6/06/$20.00 @2006 IEEE

active alerting (e.g., traffic, mailbox status – whether, when and how to alert the end-user), and more, broadly feature interaction resolution. Although not a focus of the current paper, another area for such decisions is in pro-active, targeted advertising.

While the focus of the PCP framework was on the development of network infrastructure to enable near-realtime, personalized decisions, the focus of the INA effort is on *how* to gather or learn the end-users' preferences. This issue is examined from four perspectives (i) automated learning techniques to gather preferences in ways largely transparent to the user; (ii) human factors techniques to minimize the (perceived) inconvenience to users of explicit gathering of data and preferences; (iii) security and privacy concerns, so that end-users clearly understand and control what data is gathered, how it is used, and when it is destroyed; and (iv) network-level architectural realization, so that the required data gathering and learning algorithms can be achieved in the context of highly scalable, highly reliable, and near-realtime performance. An additional challenge, in the context of converged networks, is achieving the correct balance between learning and decision-making at the edge of the network (e.g., in a laptop or cell-phone) vs. in a more centralized location (e.g., where data from multiple devices and networks is more readily available).

The INA project lies at the intersection between personalization research and ubiquitous computing research, and brings new aspects to each of these. As with ubiquitous computing (*e.g.,* [4]), there is an emphasis on the holistic view of end-users, the several networks they employ, and the many devices they use. A key challenge in INA is to effectively (i) collect information from these networks and devices, (ii) synthesize this information, both in bulk and in realtime, and (iii) deliver personalization decisions at the right times to the right places (network, device). In this regard, both PCP and INA can be viewed as a layer on top of the raw ubiquitous communication abilities provided by the emerging converged networks. As with much of the research on services personalization (*e.g.,* [5], [6]), a key goal is to (a) learn end-user preferences and/or behaviors as effortlessly as possible, and (b) find efficient ways to adapt based on what was learned, possibly in different parts of the network and on diverse devices. A key factor in many cases is our focus on requester-targeted decisions, which is at the level of binary relationships between end-users, rather than the more unary relationship between the network and individual end-users (e.g., in connection with content download behaviors). In particular, while similarities between different users are still important, in the case of requester-targeted services personalization, the level of differentiation between end-users and their relationships to other end-users is more intricate than in traditional services personalization.

## II. Example: Personalized Availability

This section illustrates key aspects of context-aware, requester-targeted, preferences-driven personalization, through a concrete example and some general remarks about it. The example is focused on a personalized availability service, as introduced in Section I. The example can be viewed as an extension of currently available products that enable one end-user to view, from a single application (*e.g.,* on their cell-phone), the presence of friends and colleagues on their multiple devices (cf., [7], [8]). The focus here is on providing to end-users the ability to protect and control the presence/availability information that is shared with others, through the use of automatic reasoning inside the network (as opposed to repeated explicit, manual presence settings through the day). We stress that the PCP approach [1], [2] provides a carrier-grade (ultra-reliable, near-real-time) framework for supporting the kinds of personalization described here.

The concrete portion of the example presented here is based on an extension of the "Friends Night Out" suite of prototype demos developed by Lucent [9], which are used to illustrate the kinds of blended services that can be created in the emerging converged network. The extension, called FNO-PCP, was developed in late 2005 and has been shown at various trade shows since.

### A. A day in Michael's life

We introduce here some aspects of the FNO-PCP demo, and provide more detail in Section III below. The focus is Michael, an office-worker with two devices – an office phone, a mobile phone with IM and voice. The demo also involves his co-worker Tom and a non-co-worker friend Sally. The demo highlights how Michael's availability to Tom and Sally changes as he goes through his day. The demo is focused on discrete choices (e.g., available by cell phone or not available by cell phone), and does not incorporate levels of availability (e.g., available by cell phone but only for urgent matters).

According to Michael's preferences, when he is in his "working" context, he is available to co-workers via any of his devices, but not available to non-co-worker friends. If Michael is working, and then receives a phone call (which might be known through event notifications from a switch or PBX), he becomes available to his co-workers only by IM. This might be refined, so that if Michael is on a call with a superior (according to a corporate directory), then he becomes unavailable even by IM. If Michael goes into a meeting (as might be indicated through a calendar entry), then Michael might become completely unavailable to co-workers, but remain available to Sally via cell phone. After work, and perhaps during lunch break, Michael may become available by cell phone to friends, including Sally and including perhaps co-workers who are also personal friends.

Of course, no system for automatically inferring an end-user's requester-targeted availability will be 100% correct. In the FNO-PCP demo this is addressed in two ways. First, Michael can override network inferences, e.g., by "telling" the network that he will work beyond normal hours, or "telling" the network to permit a form of availability to Sally not normally permitted (e.g., because they are planning a rendezvous for that evening). Second, Michael can adjust his preferences at any time; these are reflected immediately in subsequent decisions made by the network about availability values.

## III. PCP: A Foundation for Personalization

This section briefly reviews the Bell Labs Privacy-Conscious Personalization (PCP) framework [1], [2], that can support

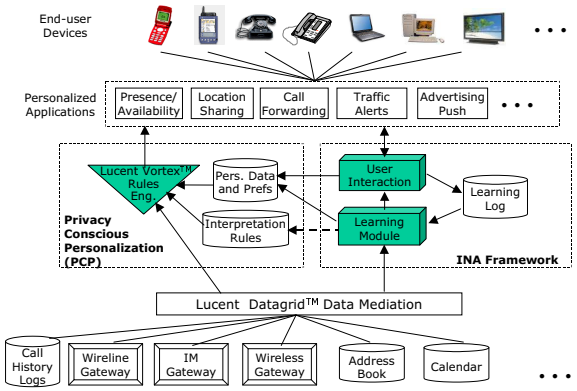



Fig. 1. High-level architecture of PCP and INA frameworks

carrier-grade (ultra-reliable) context-aware, requester-targeted, preferences-driven personalization. The PCP framework provides a versatile basis for highly refined services personalization, and is thus an appropriate foundation upon which to develop learning technologies in support of services personalization.

Figure 1 depicts the key components of the PCP framework. (For now, ignore the box in the center-right of the picture, labeled "INA Framework".) The figure shows end-users gaining access, via various devices, to a variety of personalized applications such as Availability, Location Sharing, etc. These rely on the PCP framework for the personalized decisions, and PCP itself relies on the high-speed data mediation capabilities of the Lucent Datagrid™ product [3] to obtain needed data from the network (and possibly from trusted corporate entities, etc.) (This could be extended to obtain data from less-trusted entities, e.g., using the Liberty Alliance Data Service Template (DST) standard).

Three primary components are shown within the PCP framework. One of these is the store of Personalization Data and Preferences. We illustrate these notions with reference to the FNO-PCP demo example of Section II. First, we note that a broad variety of profile data about Michael may be relevant to deciding his availability. The more static information includes Michael's address book with some buddy relationship information (e.g., Sally is a friend), corporate directory (to infer co-worker relationships, which the address book might not contain), and Michael's calendar. The dynamic information includes the status of different devices (e.g., on-call and to whom), and also the location of Michael's cell phone). We note that some of this data can be easily gathered from the network or other places (e.g., many office workers record most of their meetings in an electronic calendar), whereas other data might be relevant only to certain personalized applications (e.g., buddy relationship info). We refer to the latter as *personalization data*.

The FNO-PCP demo also illustrates the use of end-user (*personalization*) *preferences*. In the example, Michael's preferences include things "if it's between 9 and 5 and I'm in Murray Hill, then assume that I'm in working context"; "when I'm in working context then make me available to co-workers but not to non-co-worker friends"; and "if I'm working and on the phone to my boss or in a meeting, then make me completely unavailable to others".

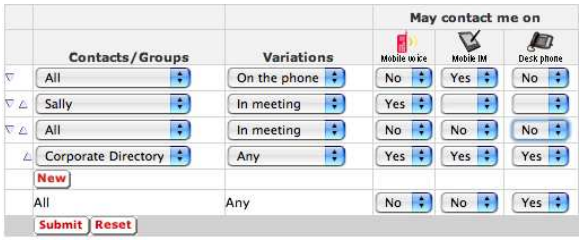



Fig. 2. Fragment of web form; part of a preferences palette

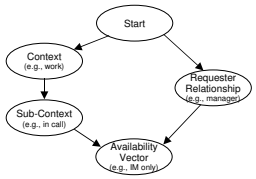



Fig. 3. Example of Decision Flow in a Preference Palette

## A. Preference Palettes

An underlying principle of the PCP framework is that there is no "one-size-fits-all" approach to personalizing an application involving social networks. This is because there is such diversity among people. Instead, we propose focusing on multiple market segments of end-users, e.g., students, road-warriors, office workers, home-makers, etc. We expect that the level of technical sophistication and the criteria used for personalization will vary from segment to segment. In PCP, the notion of *preference palette* (called "preferences template" in [1]) refers to the collection of artifacts used to provide personalization for a single market segment. A preference palette will include the web (and cell-phone) forms used to gather end-user personalization data and preferences, the database schema for holding that information, the family of associated data used in the decision making, the Lucent Vortex™ ruleset used to interpret the available data when requests needing personalization arrive, and a Decision Flow (described below).

As just one example, Figure 2 shows part of a web form that might be used in a preferences palette for office workers such as Michael in the FNO-PCP demo scenario. In this form, which is relevant to situations when Michael is in the Work "Activity", Michael can input (or modify) preferences concerning how his availability should be managed when in different "Sub-Activities". Each row of the table can be viewed as a personalization preference, and in this preferences palette, they are to be interpreted in a top-to-bottom fashion. To illustrate briefly, e.g., if he is in a call he is only available by mobile IM; and if Michael is in a meeting, then he is available to Sally by cell phone, but not by other means. We note that each row of this table, while corresponding to a "preference", can be stored as a tuple in a relational database; more generally, in PCP each preference can be represented as some data structure (tuple, nested tuple, etc.), which is then interpreted by associated Lucent Vortex™ rules.

## B. Interpreting data and preferences

Returning to Figure 1, the other two components of the PCP framework are the Lucent Vortex™ Rules Engine, and

a family of Interpretation Rules. (The arrows here indicate the principle direction in which information flows.) When a personalized decision is needed, the Lucent Vortex™ Rules Engine uses the Interpretation Rules to guide the processing of the incoming request (including identifying information about the requester), the network (and other) context information available from Lucent Datagrid™, and the Personalization Data and Preferences. In this manner, the PCP framework can provide high-speed, on-demand interpretation of the incoming request according to context, user data and preferences.

The Lucent Vortex™ Rules Engine is based on a production-system style rules language (see [10]), similar in some respects to ILOG [11]. But, with Lucent Vortex™, an engineering trade-off was made, between expressive power and the response-time requirements typical of the telecom environment. In particular, the Lucent Vortex™ rules language supports forward chaining but no cycles, which enables rapid execution. Indeed, Lucent Vortex™ was engineered to execute typical decisions within a millisecond or so [2], so that the overall time it takes to process data via Lucent Vortex™ is comparable to the time taken to make a single database dip. Through prototyping efforts, we have found the Lucent Vortex™ language to meet the needs for personalizing several converged services, including most of those mentioned in Section I.

### C. Decision Flows

The use of chaining and intermediate variables in Lucent Vortex™ provides the basis for a natural structuring of rulesets, where the rules defining each intermediate variable form a natural module. Because of the acyclicity property on rules, there is a natural directed acyclic graph (DAG) that connects all of the intermediate variables.

Figure 3 depicts the Decision Flow associated with the preferences palette used in the FNO-PCP demo of Section II. Here, one thread of reasoning determines *Context* and then *Sub-Context*, and a separate path determines *Requester-Relationship* (e.g., spouse, manager, family, friend, co-worker). Finally, these are combined to obtain the *Availability Vector*. As discussed in Section IV below, the Decision Flow provides a natural vehicle for providing end-users with a high-level explanation of why the network made a certain decision (e.g., why is Michael available to a co-worker when he is in a meeting?) (A Decision Flow might include all of the variables used in a Lucent Vortex™ ruleset, but in practice we expect that a more abstracted Decision Flow will be more useful to end-users, for the purpose of explaining how the network made a decision.)

## IV. Learning of Facts and Preferences

A key goal of the INA project is to apply human factors and machine learning techniques to dramatically reduce the quantity of personalization data and preferences that end-users must self-provision, while still gaining the advantages of richly personalized decisions in the network. The key components of the proposed INA system are illustrated on the right side of Figure 1. There is a Learning Log, and a Learning Module, which can access both the Learning Log and additional data from the network (e.g., additional log data). There is also a User Interaction module, which can be used to query the end-user about preferences or behaviors. Of course, this is highly generic. What makes INA unique is the way that we will focus the learning tasks, so that existing techniques might be applied, and so that users will be able to understand, at least at a coarse level, why decisions are made, and also be able to adjust the system behavior in a variety of circumstances.

### A. Using PCP to focus the learning tasks

In the general case, learning about a user's preferences concerning converged services such as Availability is vastly open ended. Further, it may be difficult to quickly accumulate learning sets that are sufficiently large to do adequate learning. Finally, no learning system is 100% accurate, and so it will be useful to solicit guidance from the end-user at various times. For these reasons, we feel that it is useful to let human designers provide some initial structure for the personalization task, so that learning algorithms will be able to reach conclusions more quickly and accurately. In particular, when using INA in connection with personalization of a particular application, it is assumed that various market segments have been identified, and that for each segment a team of human designers has created an appropriate preferences palette, including the web forms to gather personalization data and preferences, the Decision Flow, and the Lucent Vortex™ ruleset for interpreting all of the relevant data.

A preferences palette can be used in two ways to structure the learning and the interaction with the end-user. In the first approach, the network attempts to learn values for filling in the web forms of the preferences palette. This might include inferring the significance of various relationships (e.g., via how often an end-user answers phone calls from another end-user, of determining the values and sequencing of preference tuples, such as those illustrated in Figure 2). We note that different known learning techniques might be more suitable for different kinds of personalization data and preferences. A topic for investigation is to determine which learning algorithms work best for filling out different parts of the preferences palette.

In this approach, the results of the learning will be written, by the learning module, into the Personalization Data and Preferences store, as indicated by the solid arrow. (We also show a dotted arrow to the Interpretation Rules; however, we view the learning of those rules to be much more ambitious.) It is assumed that end-users know or will learn about the different web forms and how their data is combined. In this case, the Decision Flow of the preferences palette may be useful in providing a high-level view of how the decision-making goes.

We mention here a second, slightly more open-ended approach to using machine learning in the context of PCP. Here, for each node of the Decision Flow, the value for the variable associated to that node is determined by a set of rules (acting on the input and previously inferred values). As an alternative, imagine that for some of the intermediate variables, we abandon the associated Lucent Vortex™ rules, and simply apply a known machine learning technique. At this "micro" level of the decision process it may be acceptable to use a learning approach, such as Bayesian nets, for which it is

difficult to explain how decisions are made. Implementation of this approach could be done with the Lucent Vortex™ Rules Engine, but might involve the use of highly expressive "support" functions to act on the results of the learning. However, it might be more natural to modify the architecture of PCP, to have all the decision-making be conducted by an interoperation of Lucent Vortex™ and various learning algorithm decision procedures.

### B. Pre-population of Preferences

When users subscribe to a service, especially paid services, they will expect those services to be useful to them from the moment they subscribe. This is an interesting conundrum for a system based on learning through observation. What do we do on day 1, when we have no observations available?

A natural solution is to try to pre-populate the needed data with reasonable defaults, and evolve those defaults through learning. However, there is not one single set of defaults that will provide a level of customization to everyone. A solution to this problem involves realizing that different populations of users have related yet individualized preferences, as discussed in III-A. Then, if the learning system could know that a user was, for example, a high school student, then it could start with templates of preference palettes and fill in the values faster as they become available. In our work, we are observing and studying the communications habits of different populations, gathering data from several population groups to create their default palettes [12]. At installation time, the system would then ask users to self-identify as associating with a specific population, and set preference palettes based on those inputs. This would give users a significant level of customization from day one, as well as accelerate the learning process.

### C. Obtaining user data

Multiple sources need to be queried to obtain all the required factors to base learning on. Some places where data might come from can be the call session manager (*e.g.,* call logs, location), enterprise data (*e.g.,* calendar), subscriber profile data (*e.g.,* address book), and even the end-user's device (*e.g.,* ring/vibrate settings, location in case of GPS-enabled phones). Hence, efficient mechanisms are needed to monitor/query/aggregate this data.

The data required for analysis (*i.e.,* learning a user's behavior) can also be broadly categorized into bulk data and real-time data. Bulk data refers to data that is can be queried in an offline mode with no real-time performance penalty except that required for generating the data. Examples of this kind of data include call logs. Real-time data is user or network data that is being queried in real-time, in order to generate decisions for that user. Examples include information about the current call a user is involved in, the participants of the call, the relationship of the participants to the caller, user location, device presence, etc. Monitoring or querying this data has a direct correlation to the scalability and responsiveness of the given application.

### D. Privacy of user data

There is increasing social awareness of the importance of privacy of user information, along with a need for service providers to have explicit privacy policies on how the user information can be used and shared. Privacy requirements can vary widely depending on the user population; different market segments may have different privacy constraints; e.g., enterprise worker vs. residential consumer, different age groups, different countries.

The user should be allowed to determine which (if any) information can be queried and logged, and under what circumstances (*e.g.,* only between 9-5pm for the first 4 weeks of activating a service). The system must present these choices ("privacy SLA") to the user in a manner that the user will be comfortable with. The user will have to relinquish some privacy by authorizing the service provider to monitor, record and analyze some activities, while obtaining from the service provider some assurances as to how the information will be used, and when it will be deleted. We are currently surveying users in different populations to identify the features that must be included in a privacy SLA.

The service should be able to degrade gracefully under the situation where some or all of the information required to do accurate learning is not available (*e.g.,* due to non-disclosure by the user). In such cases however, the user can also be queried for information at appropriate moments. For example, a restaurant-locator application can send an SMS to a user to check if he is willing to disclose his location this time, and the user can respond back via the same mechanism.

### E. How to represent the learned preferences

There are multiple issues to consider when determining how to represent the learned preferences. The requirements when considering a particular representation are: it should be easy to incrementally add or remove learned preferences; it should be easy to incorporate explicit user preferences; it should be possible to represent both absolute facts as well as probabilities; it should be efficient to interpret these preferences; and the system should be able to scale to (potentially) millions of users.

Based on past work, we think that a relational database can be an effective solution that addresses the above requirements. The model and semantics of learned preferences is then governed by the DB schema chosen. (This would be an extension of the database schema in the preferences palette.) Note that other languages for representing facts (*e.g.,* RDF) could be mapped to an RDBMS schema as well. Finally, we note that Lucent Vortex™ could execute the logic that interprets these preferences with probabilities.

### F. Interacting with the User

To increase the confidence in the learned rules, it will be useful to query the user for assistance. We propose setting two probability thresholds for querying the user, and possibly allowing the user some control over these thresholds. The user will be queried only when the belief in a preference lies between these two thresholds. We envision querying the user for the candidates that are most likely to be relevant (i.e., those whose preference rules are most likely to trigger) at times that are convenient for the user. In fact, one of the user's

preferences may be exactly "when is it OK to bother me with preference settings", and even that preference can be learned with continued observation.

In addition to the incremental presentation of new learned behaviors to the user, the user also needs to have the ability to examine and edit all the learned behaviors. The forms provided by the preference palettes are natural for this. These forms together with the Decision Flow will help the end-user understand decisions made by the network.

## V. Preliminary Results

While this paper is primarily to outline a new project, we are confident that statistical analysis and machine learning techniques can be used to derive useful, personalized preference information for telecommunications. A small pilot study is underway, using phone usage data collected from a Lucent site's PBX system. (All the phone numbers were one-way hashed to prevent loss of privacy.) The pilot study aims at understanding the commonality, persistence, and predictability of usage patterns for individual or groups of users. The findings can then drive the design of preference characterizations and continuous learning.

### *A. Recognition of common calling patterns in the population*

We first investigated whether users can be divided into groups (clusters) sharing a common calling pattern. This is especially useful when the data collected for some individuals are sparse, which is often the case for a typical employee's daily calling behavior. Clustering would allow "borrowing" information from other individuals. The idea is also well-known in recommendation algorithms [13].

We counted the number of calls initiated in every 5 minute period for one month (September 2005), obtaining the time series presented in Figure 4. Spectral analysis showed a strong daily cycle, with peaks every day at 10 a.m. and 2 p.m. This motivated our trial of clustering using both Singular Value Decomposition (SVD) and functional data analysis techniques. (Singular value decomposition is as an efficient and convenient dimension reduction technique that was used to analyze call center data in [14].) Both techniques revealed consistent clustering results – distinct clusters are noted for weekday and weekend/holiday calling patterns. We show the results for functional clustering analysis in Figure 5. (Note that Monday the 5th was Labor Day—hence, its similarity to the weekend.) This method can be extended to study cycles in other units such as a hourly pattern. This preliminary study leads significant plausibility to our hypothesis of Section IV-B that useful telecommunications patterns can be identified in groups of users (*e.g.,* students, office workers) that can help pre-populate user preferences.

Analysis of phone usage for different user groups has also been pursued in the Reality Mining project at MIT's Media Lab, where mobile communication activities are recorded and studied in detail for a group of 100 volunteers from the MIT-related academic community, resulting in interesting discoveries about the social behavior [15]. Techniques involved in the analysis include hidden Markov chain, entropy characterization, and eigen-decomposition. This prior work also showed

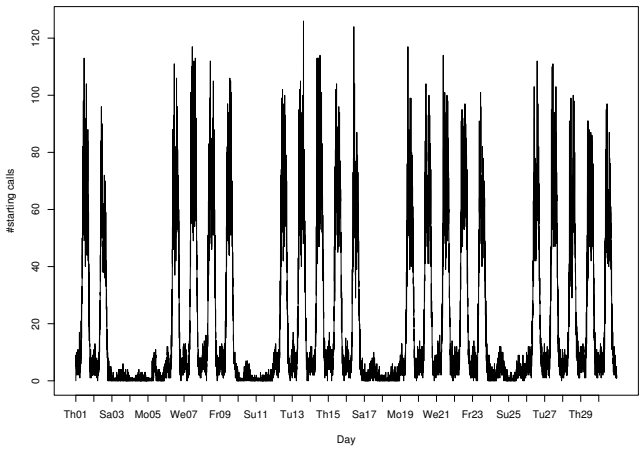



Fig. 4. Counts of calls for one month

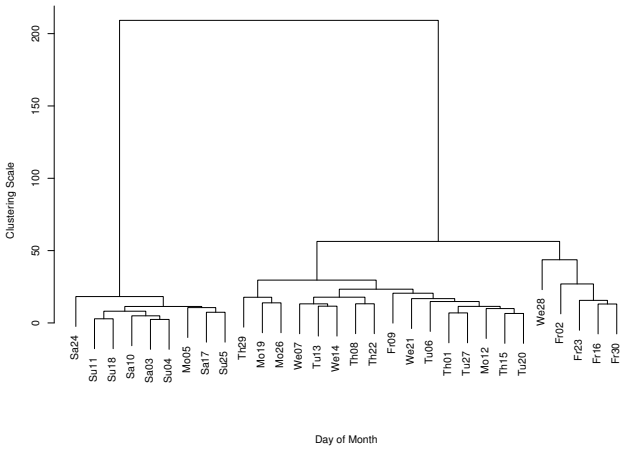



Fig. 5. Cluster Dendrogram generated from Functional Clustering Analysis

significant similarities in group behavior (*e.g.,* professors, undergraduates, graduate students)—leading further credence to our hypothesis of Section IV-B. We expect that these methods will also play an important role in our study.

### *B. Tracking an individual user's calling pattern*

While behavior pattern analysis for the whole population is helpful for bootstrapping, the framework of the INA project emphasizes calling pattern analysis for an individual user. In the PBX data, each calling record includes six variables: coded source number, call-start date, call-start time, call duration, coded destination number and call type which comprises six categories, INB (inbound), INFO (information), INT (internal), INTL (international), LCL (local), LD (long distance), and NC (no charge). A typical user's call pattern is presented on Figure 6. Figure 6 is a calendar plot starting on Thursday, September 1. Each large block stands for one day and is labeled with the date and the total number of calls for that day (in parenthesis). For instance, 33 calls were made on September 1. Within each block, the y-axis indicates the call type made, while the x-axis indicates the hour of the day. If two consecutive calls are of the same type, this will appear as a flat line segment, while a vertical jump indicates that the two consecutive calls are of different types. We can see that similar

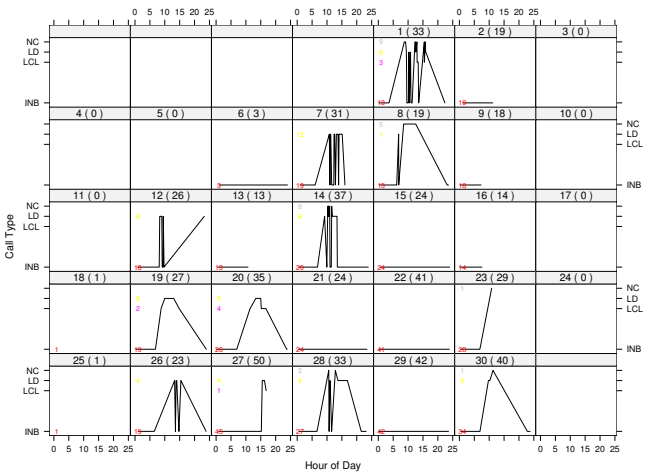



Fig. 6. A typical user's call pattern by call type and hour of the day

numbers and types of calls are made on the same weekday. We believe this lends significant credibility to our hypothesis that useful patterns can be identified for individual users. We are acquiring larger PBX data sets to further confirm our hypothesis and allow us to detect more sophisticated patterns (*e.g.,* most likely numbers to call next).

## VI. Related Work

The PCP framework and the example applications have been described previously [1], [2]. This work extends our prior work by initiating a program of learning the preferences that are necessary for PCP.

Numerous efforts exist on making cell phones more aware of their context, learning the user's routines and responding appropriately (*e.g.,* not ringing during certain times) [16]. The eWatch work performs unsupervised machine learning to independently cluster sensor quantities from multiple sensors (*i.e.,* light, skin temperature, microphone, accelerometer) into unlabeled contexts. We allow the system to question the end user, and the end user to view and manually edit learned rules. Other work [17] uses user calendar information to infer context and configure a cell phone to the appropriate ring/vibrate mode. We too use calendar information (and more) to determine a user's context.

ClixSmart Navigator adapts the menu structure of a WAP portal to reduce the amount of time end users navigate looking for content. It keeps track of the links traversed for a given menu structure and uses a simple Bayesian model to estimate the likelihood of each possible site page being the one desired by an end user. [5] *Constrained Query Personalization* rewrites end user queries to take into account user preferences and context to help users find personalized content quickly via a mobile device [6]. They solve an optimization problem of maximizing degree of user interest in results while minimizing cost and bringing back an appropriate result size.

## VII. Conclusions

This paper lays out a structured framework for applying machine learning techniques for personalization of converged services that involve an end-user's social network. This framework enables existing, targeted learning techniques to be applied, while still being able to provide end-users with an explanation of why decisions were made and with the ability for them to adjust what is learned.

Many issues can be explored from this starting point. The primary thread, of course, is to identify specific learning techniques that are effective for learning different kinds of personalization data and preferences. Another interesting area is to develop techniques to "merge" preference palettes, e.g., to enable a combination of palettes for *Students* and *Young-Office-Workers*. This might be accomplished through some form of algebra on the palettes, or through a more run-time based blendings; in any case a key requirement is that the Decision Flow and personalization preferences of the merged palette be palatable to the end-user. Another area is to generalize the form of information to be learned, moving from preferences (which are typically tuples in a database) to actual rules or interpretation logic; associative data mining techniques might be relevant here. We are currently observing and studying the communication patterns of different groups of users to learn good default values for preference palettes for these groups.

## References


[1] R. Hull, B. Kumar, D. Lieuwen, P. Patel-Schneider, A. Sahuguet, S. Varadarajan, and A. Vyas, "Enabling context-aware and privacy-conscious user data sharing," in *Proc. 5th IEEE Intl. Conf. Mobile Data Management (MDM)*, 2004, pp. 187–198.

[2] ——, "Improving user experience through rule-based service customization," *Int. J. Cooperative Information Systems*, vol. 14, no. 4, pp. 469–502, December 2005.

[3] D. Lieuwen, T. Morgan, H. Raether, S. Ramamoorthy, M. Xiong, and R. Hull, "Subscriber data management in IMS networks," *Bell Labs Technical Journal*, vol. 10, no. 4, pp. 197–215, 2006.

[4] L. Kagal, V. Korolev, S. Avancha, A. Joshi, T. Finin, and Y. Yesha, "Centaurus: An infrastructure for service management in ubiquitous computing environments," *Wireless Networks*, vol. 8, no. 6, 2002.

[5] B. Smyth and P. Cotter, "Personalized adaptive navigation for mobile portals," in *Proc. Eur. Conf. on AI*, 2002, pp. 608–612.

[6] G. Koutrika and Y. Ioannidis, "Constrained optimalities in query personalization," in *Proc. SIGMOD*, 2005, pp. 73–84.

[7] Lucent Technologies, "Active phonebook for service providers," http://www.lucent.com/solutions/flash/activephonebook.

[8] FollowAp, "iFollow IM and Presence Client home page," http://www.followap.com/Index.asp?CategoryID=136&ArticleID=49.

[9] T. Chiang, A. Johnson, S. Kanwal, and F. Shaikh, "Friends night out – a working prototype of blended lifestyle service enabled through IMS," *Bell Labs Technical Journal*, vol. 10, no. 4, pp. 17–23, 2006.

[10] R. Hull, B. Kumar, and D. Lieuwen, "Towards federated policy management," in *Proc. IEEE Policy 2003*, 2003.

[11] ILOG, "ILOG Rules," www.ilog.com.

[12] R. Dinoff, R. Hull, B. Kumar, D. Lieuwen, and P. Santos, "Learning and managing user context in personalized communications services," in *Proc. Context in Advanced Interfaces (AVI2006 Workshop)*, 2006.

[13] J. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proc. 4th Conf. on Uncertainty in Artificial Intelligence*, July 1998.

[14] H. Shen and J. Z. Huang, "Analysis of call center data using singular value decomposition," *Applied Stochastic Models in Business and Industry*, vol. 21, pp. 251–263, 2005.

[15] N. Eagle, "Using mobile phones to model complex social systems," http://www.oreillynet.com/pub/a/network/2005/06/20/MITmedialab.html.

[16] N. Boyce, "Crafting a smarter, gentler cell phone," http://www.npr.org/templates/story/story.php?storyId=5201273.

[17] A. Khalil and K. Connelly, "Improving cell phone awareness by using calendar information," in *Proc. Int. Conf. Human-Computer Interaction (INTERACT)*, 2005.