Explaining Scenarios for Information Personalization

Naren Ramakrishnan, Mary Beth Rosson, and John M. Carroll
Department of Computer Science
Virginia Tech, VA 24061
Email: {naren,rosson,carroll}@cs.vt.edu

November 5, 2001

Abstract

Personalization customizes information access. The PIPE ('Personalization is Partial Evaluation') modeling methodology represents interaction with an information space as a program. The program is then specialized to a user's known interests or information seeking activity by the technique of partial evaluation. In this paper, we elaborate PIPE by considering requirements analysis in the personalization lifecycle. We investigate the use of scenarios as a means of identifying and analyzing personalization requirements. As our first result, we show how designing a PIPE representation can be cast as a search within a space of PIPE models, organized along a partial order. This allows us to view the design of a personalization system, itself, as specialized interpretation of an information space. We then exploit the underlying equivalence of explanation-based generalization (EBG) and partial evaluation to realize high-level goals and needs identified in scenarios; in particular, we specialize (personalize) an information space based on the explanation of a user scenario in that information space, just as EBG specializes a theory based on the explanation of an example in that theory. In this approach, personalization becomes the transformation of information spaces to support the explanation of usage scenarios. An example application is described.

Word counts: Abstract (183 words), Main Text (10200), Bibliography (1092), Appendix (2375).

Keywords: Personalization, Partial Evaluation, Scenario-Based Design, Explanation-Based Generalization.

Contents

1 Introduction

Personalization constitutes the mechanisms and technologies required to customize information access to the end-user. It can be defined as the automatic adjustment of information content, structure, and presentation tailored to an individual user. With the rapid increase in the amount of information being placed online, the scope of personalization today extends to many different forms of information content and delivery [12, 32, 38], not just web pages. It is estimated that by the year 2003, personalization services will constitute the major component of the Internet industry [4].

There are undoubtedly 'personal views of personalization' [51]. This is evident both from the numerous ways in which the term is informally interpreted as well as the various choices available for designing, building, and targeting personalization systems [42, 50, 55]. A simple form of personalization is where a web portal such as `myCNN.com` allows a user to customize newsfeeds, colors, and layouts to create a personal gateway to the Internet [39]. This example abstracts the personalization problem to a point where the burden of completing the personalization task is shifted to the user, who must specify the settings. Another form of personalization involves a web browser that automatically 'hides' hyperlinks that will not lead to interesting pages. This example relies on more sophisticated user modeling; for example browsing history may be used to predict pages of interest. A third example is the recommendation facility at `amazon.com` that suggests books according to similarities in purchase behavior.

Not withstanding this variety, a core body of personalization algorithms and techniques have emerged. For instance, the mining of web user logs to identify browsing patterns has matured into a well-abstracted data mining problem [41]. Similarly, algorithms for determining similarities between buying patterns have been studied and scaled to realistic dimensions. However, the process of analyzing and specifying requirements for personalization and designing a system that achieves the desired functional goals is still an ill-understood and under-emphasized research issue. In fact, the *lifecycle* underlying design and deployment of personalization systems has not been articulated well enough to enable the investigation of these issues.

It is difficult to capture specifications of requirements independent of particular personalization algorithms or techniques. Beyond the familiar cognitive gap between specifying and implementing requirements, this is due to the dynamic nature of Internet technologies, where a new development (e.g., cookies [6]) enables a form of personalization that was not possible before. Consequently, the lifecycle for personalization systems tends to reflect the solution-first strategy of inventing a specific technique and then implementing it in a demonstration system. The hazards of such an approach are well documented [9].

Our goal in this paper is to begin building a bridge from the high-level design goals and functional requirements for personalization on the one hand, and the specific techniques and algorithms used to realize these goals and requirements on the other. Our approach is motivated by the recent development of a modeling methodology for personalization systems — PIPE ('Personalization is Partial Evaluation') [47, 49]. Personalization systems are designed and implemented in PIPE by modeling an information seeking interaction in a programmatic representation. PIPE helps realize a variety of individual personalization algorithms and enables the view of personalization as specializing representations. However, PIPE currently supports only the interaction modeling required of a personalization system; it does not address earlier stages in the lifecycle of personalization system design (such as requirements analysis) or later stages (such as verification and validation).

We elaborate how to integrate the use of PIPE with the early stages of the personalization lifecycle, in particular capturing requirements and translating them into the design of software. In extending PIPE, we employ scenario-based methods for analyzing and representing usage tasks. Scenarios and scenario-based methods are ideally suited for our purposes because they help identify personalization opportunities and organize design rationale for a system in terms of its constituent facilities. Two key contributions emerge from our approach. First, we relate PIPE to the context in which personalization scenarios are envisioned, abstracted, and realized in an information system, and thus contribute to a better understanding of the lifecycle underlying personalization system design. In particular, we relate personalization to the ability to operationalize the *explanation* of a scenario of (intended) usage. Second, we provide new techniques for managing and reasoning with scenarios that not only aid in personalization system design but also find applications in other situations that involve transformation of representations. For instance, the construction of simplified views of systems for training and demonstration purposes can be expressed using these methods.

Reader's Guide

The balance of the paper is organized as follows. In Section 2, we introduce the PIPE modeling methodology for personalization. We describe its capabilities, shortcomings, and relate PIPE to other projects that represent and reason about information seeking. Section 3 takes the first steps toward reasoning from scenarios (as representations of requirements) to modeling opportunities in PIPE. Section 4 further describes how scenario-based methods can extend PIPE to apply to the earlier stages in the lifecycle of personalization system design, such as requirements analysis and high-level specification of goals. Finally, Section 5 identifies opportunities for future research in both scenario-based methods and personalization systems design. A case study that illustrates many of the ideas introduced in this paper is provided in the Appendix.

2 PIPE: Personalization by Partial Evaluation

As a methodology, PIPE [47] makes no commitments to a particular algorithm, format for information resources, type of information seeking activities or, more basically, the nature of personalization delivered. Instead, it emphasizes the modeling of an information space in a way where descriptions of information seeking activities can be represented as partial information. Such partial information is then exploited (in the model) by *partial evaluation*, a technique popular in the programming languages community [30].

2.1 Example: Personalizing a Browsing Hierarchy

It is easy to illustrate the basic concepts of PIPE by describing its application to personalizing a browsing hierarchy. Consider a congressional web site, organized in a hierarchical fashion, that provides information about US Senators, Representatives, their party and state affiliations (Fig. 1 (left)). Assume further that we wish to personalize the site so that a reduced or restructured hierarchy is made available for each user. The first step to modeling in PIPE involves thinking of information as being organized along a motif of interaction sequences. We can identify two such organizations — the site's layout and design that influences how a user interacts with it, and the user's mental model that indicates how best her information seeking goals are specified and realized. In Fig. 1 (left), the designer has made a somewhat arbitrary partition, with type of politician as the root level dichotomy, the party as the second level, and state at the third. However the user might think of politicians by party first, a viewpoint that is not supported by the current site design. Site designs that are hardwired to disable some interaction sequences can be called 'unpersonalized' with respect to the user's mental model.

One typical personalization solution involves anticipating every type of interaction sequence beforehand, and implementing customized interfaces (algorithms) for all of them [24]. For independent levels of classification (such as in Fig. 1 (left)), this usually implies creating and storing separate trees of information hierarchies. Sometimes, the site designer chooses an intermediate solution that places a prior constraint on the types and forms of interaction sequences supported. This is frequently implemented by directing the user to one of several predefined categories (e.g., 'to search by State, click here.'). It is clear that such solutions can involve an exponential space of possibilities and lead to correspondingly cumbersome site designs.

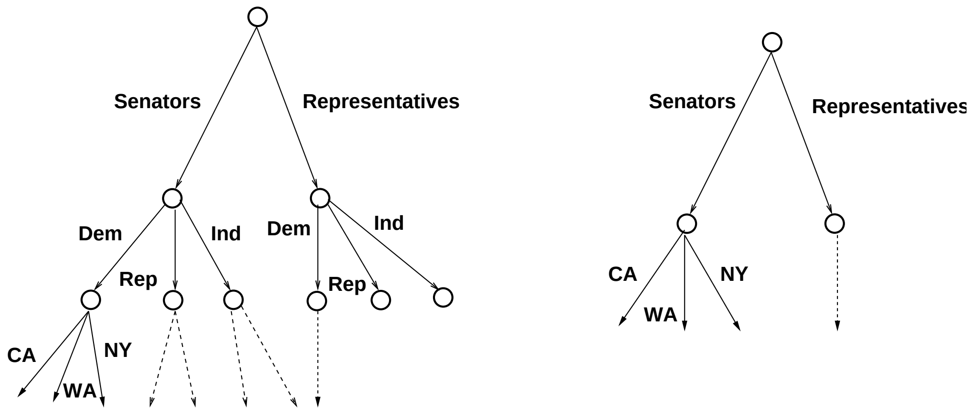

Figure 1: Personalizing a browsing hierarchy. (left) Original information resource, depicting information about members of the US Congress. The labels on edges represent choices and selections made by a navigator. (right) Personalized hierarchy with respect to the criterion ‘Democrats.’ Notice that not only the pages, but also their structure is customized for (further browsing by) the user.

```
int pow(int base, int exponent) {
  int prod = 1;
  for (int i=0;i<exponent;i++)
    prod = prod * base;
  return (prod);
}
```

```
int pow2(int base) {
  return (base * base);
}
```

Figure 2: Illustration of the partial evaluation technique. A general purpose `power` function written in C (left) and its specialized version (with `exponent` statically set to 2) to handle squares (right). Such specializations are performed automatically by partial evaluators such as C-Mix.

```
if (Sen)
  if (Dem)
    if (CA)
      .........
    else if (NY)
      .........
  else if (Rep)
    .........
else if (Repr)
  if (Dem)
    .........
  else if (Rep)
    .........
```

```
if (Sen)
  if (CA)
    .........
  else if (NY)
    .........
else if(Repr)
  .........
```

Figure 3: Using partial evaluation for personalization. (left) Programmatic input to partial evaluator, reflecting the organization of information in Fig. 1 (left). (right) Specialized program from the partial evaluator, used to create the personalized information space shown in Fig. 1 (right).

The approach in PIPE is to create a programmatic representation of the space of possible interaction sequences, and then to use the technique of partial evaluation to realize individual interaction sequences. PIPE models the information space as a program, partially evaluates the program with respect to (any) user input, and recreates a personalized information space from the specialized program.

The input to a partial evaluator is a program and (some) static information about its arguments. Its output is a specialized version of this program (typically in the same language), that uses the static information to 'pre-compile' as many operations as possible. A simple example is how the C function `pow` can be specialized to create a new function, say `pow2`, that computes the square of an integer. Consider for example, the definition of a `pow`er function shown in the left part of Fig. 2 (grossly simplified for presentation purposes). If we knew that a particular user will utilize it only for computing squares of integers, we could specialize it (for that user) to produce the `pow2` function. Thus, `pow2` is obtained automatically (not by a human programmer) from `pow` by precomputing all expressions that involve `exponent`, unfolding the for-loop, and by various other compiler transformations such as *copy propagation* and *forward substitution*. Automatic program specializers are available for C, FORTRAN, PROLOG, LISP, and several other important languages. The interested reader is referred to [30] for a good introduction. While the traditional motivation for using partial evaluation is to achieve speedup and/or remove interpretation overhead [30], it can also be viewed as a technique for simplifying program presentation, by removing inapplicable, unnecessary, and 'uninteresting' information (based on user criteria) from a program.

Thus we can abstract the situation in Fig. 1 (left) by the program of Fig. 3 (left) whose structure models the information resource (in this case, a hierarchy of web pages) and whose control-flow models the information seeking activity within it (in this case, browsing through the hierarchy by making individual selections). The link labels are represented as program variables and semantic dependencies between links are captured by the mutually-exclusive `if..else` dichotomies. To personalize this site, for say, 'Democrats,' this program is partially evaluated with respect to the variable `Dem` (setting it to one and all conflicting variables such as `Rep` to zero). This produces the simplified program in the right part of Fig. 3 which can be used to recreate web pages with personalized web content (shown in Fig. 1 (right)). For hierarchies such as in Fig. 1, the representation afforded by PIPE (notice the nesting of conditionals in Fig. 3, left) is typically much smaller than expressing the same as a union of all possible interaction sequences.

Since the partial evaluation of a program results in another program, the PIPE personalization operator is *closed.* In terms of interaction, this means that any modes of information seeking (such as browsing, in Fig. 3) originally modeled in the program are preserved. In the above example, personalizing a browsable hierarchy returns another browsable hierarchy. The closure property also means that the original information seeking activity (browsing) and personalization can be interleaved in any order. Executing the program in the form and order in which it was modeled amounts to the system-initiated mode of browsing. 'Jumping ahead' to nested program segments by partially evaluating the program amounts to the user-directed mode of personalization. In Fig. 3 (right), the simplified program can be rendered and browsed in the traditional sense, or partially evaluated further with additional user inputs. PIPE's use of partial evaluation is thus central to realizing a *mixed-initiative* mode of information seeking [48], without explicitly hardwiring all possible interaction sequences.

2.2 Modeling in PIPE

Modeling an information space as a program that encapsulates the underlying information seeking activity is key to the successful application of PIPE. For browsing hierarchies, a programmatic model can be trivially built by a depth-first crawl of the site. In addition, a variety of other information spaces and corresponding information seeking activities can be modeled in PIPE. In [47, 49], we have described modeling options for representing information integration, abstracting within a web page, interacting with recommender systems, modeling clickable maps, representing computed information, and capturing syntactic and semantic constraints pertaining to browsing hierarchies. Opportunities to curtail the cost of partial evaluation for large sites are also described in [49]. We will not address such modeling aspects here except to say that the effectiveness of a PIPE implementation depends on the particular modeling choices made *within* the programmatic representation (akin to [64]). We cannot overemphasize this aspect —- an example such as Fig. 3 can be made 'more personalized' by conducting a more sophisticated modeling of the underlying domain. For example, individual politicians' web pages at the leaves of Fig. 1 could be modeled by a deeper nesting of conditionals involving address, education, precinct, and other attributes of the individual. In other words, a single page could be further modeled as a browsable hierarchy and 'attached' (functionally invoked) at various places in the program of Fig. 3 (left). Conversely, the example in Fig. 1 can be made 'less personalized' by requiring categorical information along with user input. For instance, replacing `if (Dem)` in Fig. 3 with `if (Party=Dem)` implies that the specification of the type of input (namely that 'Democrat' refers to the 'name of the party') is required in order for the statement to be partially evaluated. Personalization systems built with PIPE can thus be distinguished by what they model and the forms of customization enabled by applying partial evaluation to such a modeling.

2.3 Reasoning about Representations

Not all information spaces (and information seeking activities) can be effectively modeled in PIPE. For example, a depth-first crawl of a site based on social network navigation (e.g., [`www.imdb.com`]) will result in spaghetti code. In such cases, we need a more complete understanding of the processes by which an online information resource is created, expressed, validated, and used. Even for sites that are easily personalized, PIPE requires that they be modeled so that all information seeking activities are expressible as partial inputs. For instance, consider the following three information seeking activities in the context of Fig. 1 (left).

User 1: I will specify a party name first; then I will specify the name of a state; finally, I will browse through any remaining links at the site.

User 2: I would like to see the list of possible states first. So, the top level of the site should present me links for all the possible states.

User 3: Show me information about the Democratic Senators of California.

The information seeking activity of User 1 can be easily realized in the representation of Fig. 3 (left), since we can partially evaluate the representation with respect to the user's choice of party and state. We say that the representation is well-factored for this activity and that it is *personable* for this activity. However, the activity of User 2 cannot be accomodated in Fig. 3 (left) since it requires restructuring operations that are not describable as partial evaluations. Applying partial evaluation to the representation of Fig. 3 (left) can simplify interactions and allow User 2 to make a choice of state out-of-turn. But it cannot change the default order in which the interactions are modeled, which is by a branch-of-congress-party-state hierarchy. In this case, we say that the representation is under-factored (for User 2's activity) and, equivalently, is *unpersonable* for it. The reader should note that this doesn't mean that User 2's request can never be satisfied in a PIPE model; see [49] for an alternate representation of the information space that is personable for User 2's activity (and is hence, well-factored for it).

Now, consider how we will satisfy User 3's request. This user has specified choices for all possible program variables — involving state, party, and branch of Congress. This amounts to a *complete evaluation*, rather than a partial evaluation. Complete evaluation in a PIPE model implies that every possible aspect of interaction is specified in advance, obviating the need for any interaction! Since PIPE emphasizes the specialization of interaction by partial evaluation, the representation of Fig. 3 (left) offers no particular advantages for User 3's activity. In this case, we say that the representation is over-factored and, again, is unpersonable (by partial evaluation) for User 3's activity. Thus, both under-factorization and over-factorization lead to unpersonable representations of information spaces. The interesting representations are in between.

In practice, it is acceptable to have a few situations that involve complete evaluation, as long as they are a small fraction of the total number of information seeking activities (that are to be accomodated in a PIPE model). More discussion about representations and their factorizations is available in [49]; for the purposes of this paper, it suffices to note that we have various possibilities for representing information spaces in PIPE and that it is important to choose a representation that is well-factored.

2.4 Related Research

In name and spirit, PIPE's personalization by partial evaluation is similar to RABBIT's [64] retrieval by reformulation. Both these approaches involve the modeling of information seeking in a setting that emphasizes (i) reconciling the mismatch between how an information space is organized and how a particular user forages in it, (ii) closure properties of the transformation operators, and (iii) the design of information systems in ways that highlight new evaluation criteria. Like RABBIT, PIPE assumes that 'the user knows more about the generic structure of the [information space] than [PIPE] does, although [PIPE] knows more about the particulars ([web pages]) [64].' For instance, personalization by partial evaluation is only as effective as the ease with which program variables could be set (on or off) based on information supplied by the user. As such, PIPE has no semantic understanding of the representation.

PIPE also differs from RABBIT in important ways. It emphasizes the modeling of an information space as well as an information seeking activity in a unified programmatic representation. Its single transformation operator (partial evaluation) provides a basis to reason about the design of personalization systems. Since partial evaluation works best for highly parameterized and structured spaces, the PIPE viewpoint relates the personalizability of an information resource to the factorizability of its representation. A well-factored information space is thus a personable one, since information seeking activities are expressible as partial inputs.

Research at the intersection of information systems and HCI has a strong tradition, with many other prominent examples. Both the Scatter/Gather [15] and Dynamic Taxonomies [57] projects rely on defining a set of operations under which transformations made on an information space are closed. These projects concentrate on retrieval and navigation, respectively. While there has been considerable research in web personalization [1, 2, 3, 26, 35, 40, 60], many of these algorithms/systems (or in some cases their results) are usefully viewed as modeling choices to be made in a PIPE implementation. For instance, the graph-theoretic recommendation algorithm described in [2] can be modeled as a function in PIPE, so that the results of the function are used to set values for program variables, which can in turn be 'linked' to more detailed information about the recommended artifacts. There have also been attempts at defining theories of information access, suitable for the design of personalization systems [44]. Pirolli [43] explains the idea of 'information foraging' and analyzes projects such as Scatter/Gather in this context.

Empirical research involving usage modeling and information capture is also relevant here. Drawing ideas from the ACT-R theory of cognition, Pirolli et al. [45] describe how a quantitative model of information foraging can be defined. Tools for capturing history of interaction in information foraging are also well studied [25, 63]. Mining web user logs has become a popular technique for obtaining models of site navigation [40, 41, 59]. While this strand of research has arrived at rich, quantitative models of site usage and navigation, there is a persistent gap between what could be mined from site usage and how the site could be automatically transformed to conform to any identified needs. Typical approaches to bridging from the results of site usage studies to opportunities for site restructuring are heuristic (see for instance [59]) and are limited in the transformations they employ [42].

3 From Scenarios to Modeling Choices in PIPE

As mentioned earlier, PIPE's modeling methodology requires a programmatic representation (such as Fig. 3, left) for partial evaluation. Where do such representations come from? In this section, we analyze how personalization

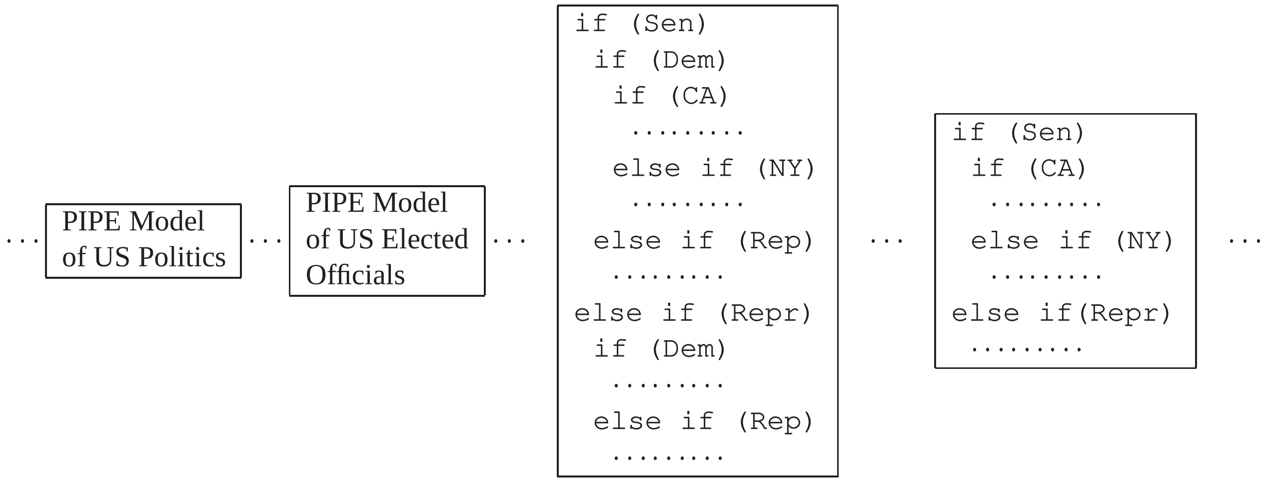

Figure 4: A space of PIPE models organized by the partial evaluation relation.

requirements originate in usage contexts, and how they can help to build the representations of information spaces needed in PIPE. In this sense, we are extending the PIPE methodology 'upstream' in the personalization system design life cycle, to include requirements analysis and specification.

While even a very general notion of requirements gathering applies in our situation [11, 34, 36, 58], personalization offers the unique viewpoint of interpreting a general, existing information resource in a specialized manner (and thus, indirectly improving it). Studies in traditional IR contexts (e.g., see [5, 64]) have shown that one way to achieve such specialized interpretation is to support the iterative reformulation of information requests. Besides reconciling the mental mismatch between user expectations and the facilities afforded by an information system, reformulation engages the user in an active dialog with the system, using both system features and user input to complete the information seeking activity.

We propose that such active search and reformulation episodes can be anticipated, revealed, and modeled by usage scenarios. These scenarios can then form the basis of a scenario-based analysis and design (SBD) process [8, 54]. Current practice is observed and described in scenarios, and such scenarios help analyze how designers and users think about complex information resources. By helping to reason about the tradeoffs and design rationale associated with system design decisions, scenarios can aid in identifying opportunities for personalization. How to systematically proceed from high-level goals and needs identified in a scenario to a programmatic model in PIPE is the subject of the balance of this paper.

Before we describe our approach, it is important to make some preliminary remarks. Let us revisit the two PIPE models in Fig. 3. The model in Fig. 3 (right) is the result of partially evaluating Fig. 3 (left) with respect to 'Democrats.' However, the model of Fig. 3 (left) can itself be viewed as the result of a partial evaluation (say, of a model that provides information about all US politicians, with respect to 'congressional officials'). In other words, Fig. 3 (left) is personalized for information about members of the US Congress. Likewise, the model of Fig. 3 (right) can be viewed as the starting point of interaction with an (unpersonalized) information system, one that is designed for people who are interested in only Democrats. It should thus be clear that there is actually a continuum of PIPE models (see Fig. 4), organized along a partial order (where the specialization relation is partial evaluation).

Given this observation, we can cast our requirements analysis problem as a search within a space of PIPE models, such as Fig. 4. But we can go further. Every model in this space can be thought of as the result of a partial evaluation or, equally, as a starting point for partial evaluation. This means that the task of selecting a PIPE representation for subsequent personalization can actually be viewed as a problem of partially evaluating (personalizing) a more general representation! Designing a personalization system is thus reduced to a problem of personalization (of a general, and perhaps ineffective, information space). A PIPE representation can be seen as 'freezing' some aspects of interaction and making available some other aspects to model users' information seeking activities. In Fig. 3 (left), the model is the result of partial evaluation with respect to congressional officials, but program variables pertaining to party, type, and state are available to represent users' personalization objectives. This viewpoint reinforces our idea that both designing and using personalization systems involve specialized interpretation of information spaces.

Clearly this cyclic argument has to end somewhere, so what is the 'starting model' for partial evaluation? And where does it come from? Our approach is to relate opportunities identified in usage scenarios to a characterization of the space of PIPE models, by qualifying a most-specific and the most-general elements of the space. We then define an evaluation function to express our preference for one model over another. Before we can formally describe our methodology, we must broaden our view of partial evaluation, moving from its algorithmic details as a specialization function, to larger contexts that recognize the space of models induced by the partial order.

One such context is the work on explanation-based generalization (EBG) in AI [17]. Just as partial evaluation addresses the specialization of programs, EBG addresses the specialization of domain theories. In fact, van Harmelen and Bundy have observed [61] that when programs and domain theories are both represented in Prolog notation, partial evaluation and EBG are essentially equivalent. With respect to our scenario modeling problem, EBG makes the important addition of recognizing the space of models induced by the partial evaluation relation: the space is first defined by a systematic process of 'explaining' observations and reasoning about features that are relevant to the observation. The vocabulary for conducting the explanation is provided by the domain theory; the relevant features thus identified help characterize the search space. Next, EBG provides a search criterion for evaluating models, one that emphasizes the utility and usefulness of the ensuing representations.

We can borrow this idea of explanation, using it to bridge from usage scenarios to the models and representational choices required by PIPE. Just as an existing domain theory supports the construction of an explanation for an example observation (which then guides the specialization of the theory), an existing information space can support the construction of an explanation for a scenario (of intended usage), which can then guide the personalization of the information space (in our case, thus helping to design a personalization system).

3.1 Explanation-Based Generalization

To understand how we can bridge the high-level requirements uncovered through scenarios and the programmatic modeling required by PIPE, we briefly review the basic ideas of EBG in an everyday context. Consider a non-native speaker of English (Linus) visiting the United States. He is attempting to learn conversational constructs for 'being polite.' The essence of EBG is that it is easier for Linus (at first) to verify or explain why a particular conversation is an example of politeness, than to describe or define politeness in a vacuum. Thus, Linus observes instances of politeness and generalizes from them by explaining why they appear to be polite. For instance, he witnesses the following dialog between two people:

> **Person 1:** Sir, I was wondering if you could point me in the direction of Central Park.
> **Person 2:** Sure. Make a right two blocks after the gas station.

At this point, Linus can infer that this is a valid example of politeness (from Person 2's helpful response) and proceeds to explain the observation. By analyzing the structure of Person 1's query and using his knowledge of how English sentences are constructed, Linus constructs an explanation of this observation. DeJong [17] shows that an explanation can be viewed as a tree where each leaf is a property of the example being explained, each internal node models an inference procedure applied to its children, and the root is the final conclusion supported by the explanation (namely, that the above was an example of politeness). The explanation tree proves that the conversation is polite and helps separate out the relevant and incidental parts of the above conversation; any attribute of the conversation that does not participate in the 'proof' does not contribute to politeness.

Most general:	$\prec$polite conversational construct$\succ$.
	....
	$\prec$well-mannered phrase$\succ$ $\prec$Linus's desire$\succ$.
	....
	$\prec$address$\succ$, I was wondering if $\prec$Linus's desire$\succ$.
	....
	Sir, I was wondering if $\prec$Linus's desire$\succ$.
	....
Most specific:	Sir, I was wondering if you could point me in the direction of Central Park.

Figure 5: Example generalizations for the Central Park conversation.

Using this structure (and his knowledge of English), Linus can then study how it can be generalized to other situations. For instance, he can reason that the phrase 'Sir, I was wondering if' is what confers politeness onto the whole sentence. He can also conclude that 'Central Park' is not a property of politeness per se, but a feature of the particular request. Notice that Linus could have arrived at the phrase 'Sir, I was wondering if' himself (without the above example), but that would have required a lot of imagination (computation, for an AI system [56]) on his part. This is the essence of EBG — namely that we don't 'actually learn anything factually new from the instance [56]' but such examples point us in the direction in which to specialize our unmanageable domain theory (in this case, Linus's rules of grammar and knowledge of how English sentences are constructed).

The reader might notice a disconnect between the G in EBG (which stands for *generalization*) and our statement that EBG is really a technique for *specializing* domain theories. This can be understood by noticing that the most common usage of EBG is in learning concept descriptors from individual examples [17]. Thus, while it is the domain theory that is being specialized, the example is being generalized by throwing away parts of its explanation structure. In other words, the domain theory constitutes the prior knowledge that is useful for generalization [56]. Linus can then exhibit his newly acquired politeness in a different situation such as: 'Sir, I was wondering if you could hold open the elevator for me.' Or even further, he might generalize 'Sir' to include 'Madam' and 'Lady.'

In concept learning, the level to which Linus generalizes a particular explanation is influenced by *operationality*. For instance, if he doesn't generalize beyond 'Sir, I was wondering if you could point me in the direction of,' then his learning can only be applied to, say, situations when he is lost. At the other extreme, Linus might reason that there are many other ways of being polite (such as 'Could you please ...?') and conclude that any well mannered phrase prefixed to a request constitutes an instance of politeness. Such an over-generalization is however less operational, since it assumes that Linus has some other way of deciding what makes a phrase 'well mannered.' Operationality is thus related to the utility of the induced generalization.

3.2 Using EBG in Personalization

Keller [33] shows how we can think of EBG as a search through a concept description space such as Fig. 5. The operationality consideration is then the objective function used to evaluate entries in the concept description space. The most specific construct simply records the conversation and can only be replayed in an exactly similar situation. The effort to instantiate the construct is thus minimal but many such constructs will likely be needed to cover a realistic set of situations. The most general construct involves no learning on Linus's part and merely restates his desire to learn polite constructs. If Linus adopts this construct, he can have one single explanation structure to support all situations but he has to expend the effort to instantiate it (effectively, constructing the proof) every time he needs to be polite.

Analogously, the goal of obtaining a PIPE model is viewed as search through a space of possible PIPE models,

Inputs	—- Functional Description of Personalization Problem (i.e., a non-operative definition) —- Domain Theory (describing site layout, task models, browsing semantics; must support the construction of an explanation) —- Usage Scenario (showing successful realization of personalization goal; typically a sequence of interactions which achieves goal) —- Operationality Criterion (helps evaluate alternative PIPE models; outlines which aspects of explanation structure can be fixed, and which should be available for capturing interactions with users)
Output	—- PIPE model (that defines a personalization system)

Figure 6: Formalizing requirements analysis in the personalization lifecycle as an EBG problem.

ordered by the partial evaluation operator. Explaining a user's successful interaction at a site (or collection of sites) with respect to a domain theory (more on this later) will help identify the parts of the interaction that contribute to achieving the personalization objectives. The explanation tree thus serves to define the search space of PIPE models. The operationality boundary is then the point at which certain parameters are fixed in our representation of the information space and certain others are available to model users' information seeking interactions.

Adapting from [18], we can formalize requirements analysis for PIPE as shown in Fig. 6. Our methodology requires the specification of four inputs. A functional description of the personalization problem is assumed so that we can distinguish between scenarios where the user was successful in achieving his objectives from those where he was not successful. The domain theory is the most critical aspect of the methodology and encodes knowledge about site layout, browsing semantics, task models, and any other information that is relevant for reasoning about the user's personalization objectives. In addition, the domain theory language should support inference procedures (e.g., deduction, rewriting) that enable the construction of explanations. Usage scenarios, narratives, and think-aloud records constitute the third input and together with the domain theory, drive the explanation construction process. Finally, operationality serves as the criterion for evaluating the space of models induced by generalizing an explanation.

Procedurally, our methodology consists of a three-stage approach (see Fig. 7): (i) constructing explanations from scenarios of use; this reveals how a given site (or collection of sites) helps the user to complete his information seeking activity, (ii) operationalizing the explanations in terms of site facilities; this allows us to assess the most relevant aspects of the explanation structure that we would like to retain and express in a personalization system, and (iii) expressing the operationalized explanation in a PIPE model (which will allow its subsequent personalization for future users). We now proceed to study these steps in greater detail.

4 Explaining Personalization as Partial Evaluation

4.1 Constructing Explanations from Scenarios

The SBD process begins with an analysis of current usage practices. This would typically take place through field work that includes observation of work sessions, interviews or surveys of domain experts, and collection and analysis of work artifacts. The goals and concerns of current use are synthesized and contextualized as problem scenarios.

As an example, consider the analysis and development of scenarios for personalizing information about US

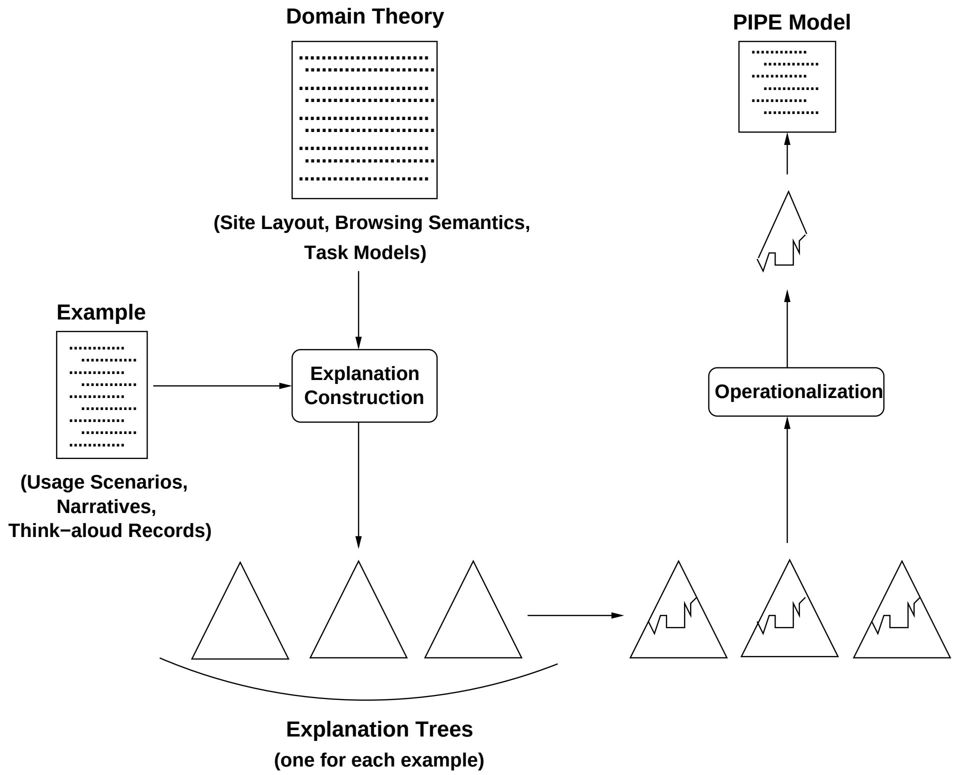

Figure 7: From scenarios to modeling choices in PIPE: toward a lifecycle of personalization system design.

Nancy's scenario : Nancy, a resident of North Carolina, is interested in determining the committees that the junior senator from her state is a member of. She uses the PoliticalInfo web site to perform her information seeking activity. The top level of the site features various categories of political information, organized according to the offices of government (such as 'President,' 'Congress,' and 'State Offices'). She clicks on 'Congress' and reaches a page that prompts her to choose a state. Upon selecting 'North Carolina,' the site refreshes to a selection involving branch of Congress (Senate or the House of Representatives). Nancy selects 'Senate,' and the system now requests information on whether the senator occupies the junior or senior seat. By mistake, she clicks on an advertisement banner for campaign finance reform, which causes a new browser window to be opened up, soliciting information from Nancy for an opinion poll. She hurriedly closes this new window, goes back to her browsing session in progress, and clicks on 'Junior Seat.' Nancy scrolls down the displayed homepage of the individual, eyeballs the various headings, and finally spots the information about committee memberships. She notes that there are four committees that the senator is a member of. Satisfied with the results of her information seeking, Nancy exits from her browser program.

Figure 8: Fictitious narrative of a problem scenario synthesized from field observation.

R1:	politicalinfo(x)	$\Leftarrow$	complete(x)
R2:	complete(x)	$\Leftarrow$	officeselect(x,“Congress”) $\wedge$ member(x) $\wedge$ aspect(x)
R3:	complete(x)	$\Leftarrow$	officeselect(x,“President”) $\wedge$ aspect(x)
...			
R25:	member(x)	$\Leftarrow$	representative(x)
R26:	member(x)	$\Leftarrow$	senator(x)
...			
R32:	senator(x)	$\Leftarrow$	branchselect(x,“Senate”) $\wedge$ stateselect(x,s) $\wedge$ seatselect(x,“Junior Seat”)
R33:	senator(x)	$\Leftarrow$	branchselect(x,“Senate”) $\wedge$ stateselect(x,s) $\wedge$ seatselect(x,“Senior Seat”)
...			
R48:	aspect(x)	$\Leftarrow$	aspectselect(x,“Education”)
R49:	aspect(x)	$\Leftarrow$	aspectselect(x,“Committee Memberships”)
R50:	aspect(x)	$\Leftarrow$	aspectselect(x,“Home City”)
...			
S1:	stateselect(x,s)	$\Leftarrow$	clickedon(x,p,s) $\wedge$ congresslevel(p) $\wedge$ hyperlink(s)
S2:	adselect(x,a)	$\Leftarrow$	clickedon(x,p,a) $\wedge$ advertisement(a)
...			

Figure 9: An example domain theory for reasoning about interactions at the fictitious PoliticalInfo site. All variables are assumed to be universally quantified (syntax adapted from [17]).

F1:	officeselect($x47$,“Congress”).
F2:	stateselect($x47$,“North Carolina”).
F3:	branchselect($x47$,“Senate”).
F4:	adselect($x47$,“Campaign Finance Reform Adverstisement”).
F5:	seatselect($x47$,“Junior Seat”).
F6:	aspectselect($x47$,“Committee Memberships”).

Figure 10: Description of Nancy’s scenario for subsequent explanation.

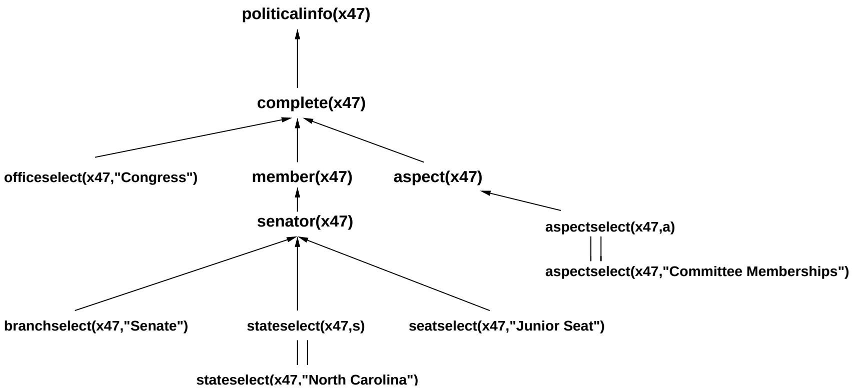

Figure 11: Constructing an explanation for Nancy’s scenario. Following DeJong [17], the arrows indicate influence of rule antecedents on consequents and the parallel lines indicate unification constraints.

political officials. We might observe users' interactions with various web sites for several hours, then interview them and analyze their decisions or other artifacts that they generate. Empirical techniques for analyzing web logs and for the automatic sessionizing of user access patterns might also be used (see [14] for methods of data preparation and gathering for this activity). The analysis might point to a recurring scenario in which users select a particular site, navigate the pages in the site by making various selections on the category of the political official and finally, lookup individual web pages to obtain specific information about a particular official. For instance, Fig. 8 describes a positive instance of our modeling goal (i.e., personalizing information about political officials at the PoliticalInfo web site).

We now turn our attention to the domain theory which consists of: a modeling of the information seeking goal in terms of an underlying schemata; our understanding of PoliticalInfo's layout (its site structure and how link choices specify information seeking attributes), and; aspects that capture how browsing interactions take place at the site (such as 'clicking on advertisement banners cause new browser windows to be opened up'). While EBG is sometimes proposed as a possible computational model of human concept formation and cognition, it is important to note that the purpose of a domain theory in our methodology is *not* to explain users' behavior at this level. Rather, we seek to encode information in the domain theory that helps relate interactions with information systems to the realization of information seeking objectives. This requires that the domain theory be sufficient to reason deductively why an observed sequence of interactions achieves the desired personalization objectives. Our experience is that for a variety of information spaces that support a goal-oriented view of information seeking (e.g., browsing hierarchies involving taxonomic relationships), this assumption of the availability of a domain theory can indeed be satisfied.

For instance, a domain theory for interactions at PoliticalInfo can be organized as shown in Fig. 9. The first part of the theory describes an underlying schema for achieving the politicalinfo information seeking goal. Rule R1, in particular, is our functional specification of the personalization problem. It is non-operative and merely states that an interaction session (x) that is complete can help satisfy the personalization objectives. Rules R2 and R3 describe two specific ways in which an interaction can be complete, namely by either concentrating on aspects dealing with members of the US Congress or on the President. Members of the US Congress, in turn, are defined as either senators or representatives (rules R25 and R26). Notice that there are many possibilities for defining member(x) and these are encoded in the domain theory. The second part of the domain theory describes how primitive actions carried out by the user correspond to the specification of information seeking attributes. For instance, the selection of a state (rule S1) can be made by clicking on an available hyperlink from the congresslevel page. Similarly selections of advertisements can be made by clicking on advertisements from any page (rule S2).

Using the predicates in the domain theory, we can represent Nancy's scenario ($x47$) as shown in Fig. 10. For ease of presentation, Fig. 10 describes only the high-level selections inferred from Nancy's interactions instead of assertions at the level of clicks and hyperlinks. We now proceed to show that Nancy's scenario is a 'correct' example of accessing information about political officials. In other words, we attempt to prove that politicalinfo($x47$) is true. The explanation tree constructed by resolution is shown in Fig. 11 and identifies the salient aspects of the scenario that contribute to realizing Nancy's information seeking goals. In particular, rules R26 and R32 have been used to prove that Nancy's selection of attributes defines a particular political official. The explanation tree also reveals that the interactions relating to the advertisement for campaign finance reform do not contribute to Nancy's objectives. We exclude the aspect of Nancy recording the details of the committee memberships; her satisfaction with the information seeking activity is assumed to be implicit in the completion of the proof.

4.1.1 A Note about Domain Theories

Before we describe the next stage in our methodology, it is pertinent to make some observations about the domain theory. First, we are not constrained to a predicate logic representation for the domain theory. The only requirement is that 'the representation language support the construction of an explanation' [17]. Second, alternate domain theories might permit the explanation of the same scenarios, but in qualitatively different ways. This is a useful feature since it prevents overgeneralizing from the observed features of the scenario. In addition, it allows us to compare and contrast domain theories and determine if the resulting explanations are acceptable. Third, we can start with a 'coarse' domain theory and revise it by focusing on particular scenarios and situations [19]. Such *theory revision* research is an active area of EBG, where explanation-based techniques are augmented with more empirically-based methods to address the problem of imperfect prior knowledge. Finally, while the availability of a general schemata aids in the construction of a domain theory, external guidance (from users and think-aloud records) can support the construction of an explanation and augment domain theories that are incomplete.

This last feature is especially useful when we extend our approach to more complex situations, such as multiple information resources. Consider a scenario where Nancy is seeking information about financial investments. During analysis and design, some sub-goals and decisions can be inferred by think-aloud protocols while Nancy forages in various web sites to address her financial analysis goal. For instance, Nancy might report that she conducted a mental calculation of dollar amounts from Euro currency in analyzing some merger stocks, and hence we might model a procedure for unit conversion as an intermediate goal in our domain theory. Similarly, Nancy might have performed a manual information integration by copying text from one browser window to another or might have performed a mapping from company names (e.g., 'Microsoft') to ticker symbols ('MSFT'). We would represent these as unification constraints or semantic mappings in our domain theory, respectively. By augmenting a domain theory in this manner, we can summarize scenarios collected as field data into appropriate explanation structures.

We also recognize the possibility that a domain theory might be ineffective in explaining scenarios. Consider the case when Nancy is seeking information about the 'Democratic senator from North Carolina' and is unsure if the senator occupies the junior or senior seat. If the site does not allow the direct specification of her request, Nancy might resort to trying both choices of seats to determine the one occupied by the Democrat. An ineffective domain theory might incorrectly infer that Nancy's information seeking was focused on both senators! We thus need to discount some steps in the scenario as being tentative or exploratory. This is a well studied problem in EBG and various strategies for reducing dependence on such 'brittle theories' (such as induction over explanations) have been proposed [17, 20].

The reader will also note that the explanation in Fig. 11 (or the domain theory) does not capture the order in which the attributes were specified in Nancy's scenario. In this particular instance, the temporal sequencing of subgoals is not critical to completing the explanation; Nancy could have selected 'Senate' first (if the site allowed it) before the choice of state was made. In a different application, the domain theory might need to support the construction of explanations that recognize the ordering of interactions.

4.2 Operationalizing Explanations

Fig. 11's explanation of Nancy's scenario, while identifying relevant parts of the domain theory, is too specific to be used as the basis of a personalization system. The next step in our methodology is thus to determine the parts of the explanation structure that we would like to retain and express in a PIPE model. A trivial step of identity elimination is first done to eliminate dependence on the particular scenario of Nancy (i.e., the $x47$ in Fig. 11 is replaced by just x). Operationalization can then be thought of as drawing a cutting plane through the explanation tree. Every node below the plane is too specific to be assumed to be part of all scenarios. The structure above the plane is considered the persistent feature of all usage scenarios and is expressed in the personalization system design. The user is then expected to supply the details of the structure below the plane so that the proof can be completed.

For instance, if we draw the cutting plane just below politicalinfo(x) then this is equivalent to no personalization at all. A single explanation tree can accommodate all possible information seeking activities but really provides no support as a personalization system.

If the cutting plane is drawn to include the leaves, then this amounts to freezing all aspects of Nancy's scenario so that it can be replayed in full. In such a case, it is unlikely that a personalization system modeled after one explanation tree will satisfy all users. We could freeze many more such explanation trees and the design of the

click [here]	if you are the user who seeks information about committee memberships of the junior senator from North Carolina.
click [here]	if you are the user who likes information about the educational background of the President.
click [here]	if you are the user who seeks details about bills proposed by the Republican representative from Littletown constituency of Virginia.
click [here]	...

Figure 12: Operationalizing multiple explanations at the leaf level leads to an over-factored representation in PIPE.

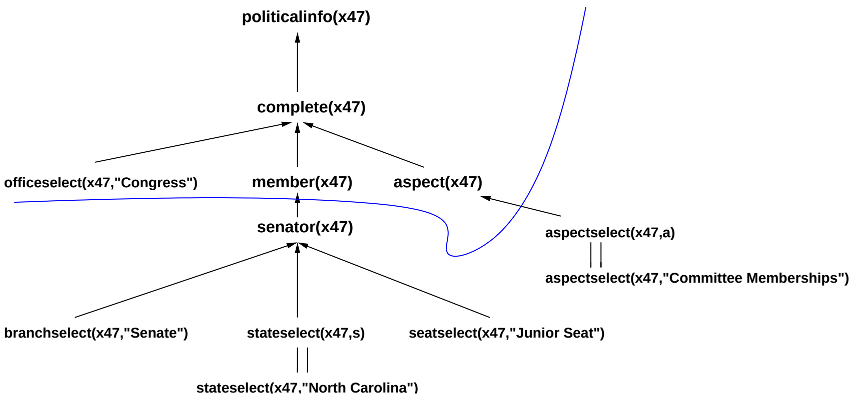

Figure 13: Operationalizing the explanation for Nancy's scenario.

personalization system then reduces to providing a top-level prompt for the correct explanation (see Fig. 12). This solution anticipates all forms of interactions and over-specifies the personalization problem. As is well-known in EBG, such a design is inefficient since a new user has to search for the correct explanation that is appropriate for his information seeking activity. From Section 2.3, we also know that such a design would be over-factored and unpersonable under PIPE. This is because the resulting PIPE model has only one argument (namely, the choice of the correct explanation) and all invocations of such a model have to involve complete evaluation!

Fig. 13 describes an intermediate solution where some aspects of the explanation are fixed but some other aspects are available for addressing users' interactions. This operationalization induces a system that personalizes information about congressional officials. It assumes that, like Nancy, a new user will invoke rule R2 from Fig. 9 but, unlike Nancy, could be interested in other members of Congress besides senators. In addition, the part of the tree specifying the aspect of interest is also available for specification by the user. The reader should note that such an operationalization will not cover scenarios where the user is interested in, say, information about the President. To accommodate this case, we could either move our operationality boundary or create and operationalize another explanation tree (for an appropriate scenario). Studying the tradeoff between these two possibilities constitutes the crux of operationality research. For the purposes of this article, it suffices to note that two dimensions of operationality are: how many explanation trees are operationalized, and where the boundaries are drawn in each tree.

EBG's ability to induce general constructs by explaining scenarios can be a drawback as well as an advantage. If we generate a lot of templates, then users' interactions with the personalization system can get burdened by a mushrooming of choices. At the same time, EBG provides a systematic way to cluster the space of users and to determine dense regions of repetitive interactions that could be supported. A case in point is a web site such as [amazon.com] that distinguishes between returning customers and new customers. A top-level prompt at the site makes this distinction (sometimes, this is automated with cookies) and transfers are made to different interaction

```
    .........
L1: if (Senator)
L2:  if (JuniorSeat)
L3:   if (NC)
L4:    if (CommitteeMemberships)
L5:     /* Details about committees */
        .........
       else if (Education)
        .........
     else if (SeniorSeat)
       .........
    .........
    else if (Representative)
    .........
```

Figure 14: Designing a PIPE representation from the operationalized explanation in Fig. 13.

sequences, based on the results of this choice. For returning customers, questions about mode of payment and mailing address are skipped because the parts of the proof dealing with those aspects are already subsumed in the design of the system. Another example is a web site that provides links from the top-level page to 'the top 10 frequently accessed pages at our site.' In this case, popular explanations have been operationalized at the leaf level and presented so that new users can directly access them.

Operationalization is only one way to generalize an explanation to other situations. A variety of other generalization approaches are prevalent in the EBG literature. There are techniques that conduct generalization across multiple explanation trees simultaneously, by identifying recurring subtrees [20]. To some extent, this can help overcome sensitivity to initially explained scenarios and also address shortcomings in the representation of the domain theory. There are also approaches that model and generalize temporal interactions and ones that help acquire iterative concepts. Iterative concepts are useful, for instance, when users to our PoliticalInfo site seek multiple aspects of information about a political official. Nancy was interested in only committee memberships but a different user could have been interested in committee memberships as well as the educational profile. Generalizing to n such aspects can be achieved by acquiring an iterative concept. We can also generalize to acquire recursive formulations; this is useful if information seeking has an exploratory nature to it. Linus might have visited a sports site four times in a single scenario, whose correct generalization could be 'keep visiting the page to see if the scores have been updated.' Finally, since the choice of operationality is primarily driven by empirical and usability concerns, a variety of existing methodologies for utility analysis and estimation can be employed here [17].

4.3 Designing a PIPE Representation

The last step is to express the operationalized explanation in a PIPE representation. We can think of this stage as designing an information system that provides all the necessary facilities to complete the proof. The part of the proof *above* the cutting plane is to be performed by the system, whereas the user has to supply the details of the proof *below* the cutting plane (in our case, for member(x) and aspect(x)).

Ideally, the user should be able to supply her part of the proof in as expressive a manner as possible. For instance, just saying 'North Dakota' and 'Representative' in the current political landscape defines a unique member of Congress. To achieve this effect in a PIPE model, we must ensure that all possible ways of completing the proof are describable in terms of interaction sequences. Alternatively, we might choose to support only certain possibilities of completing the proof. The PIPE model should thus be parameterized in terms of variables that help define member(x) and aspect(x). A representation in C is shown in Fig. 14. Lines L1, L2, and L3 help define member(x) for Nancy's scenario and Line L4 helps define aspect(x). Since PIPE representations can be partially evaluated, the user can specify the underlying attributes of the proof in any order.

It is important that we also model terminal code that gets triggered upon completion of the proof (or subproofs). In our running example, the terminal code (e.g., line L5) represents the results presented to Nancy upon successful completion of the proof (i.e., information about committee memberships).

The reader should recall that we have a variety of modeling options for creating the PIPE representation. Fig. 14 models the interaction as a browsing hierarchy, similar to Fig. 3. Instead we could have modeled the interaction as a sequence of forms to be filled by the user. Our representation also assumes that we have only one operationalized explanation structure. If we have multiple explanation structures, an extra program variable can be introduced that identifies the explanation a given user is interested in. The PIPE model in this case would be a sequence of representations such as Fig. 14, joined together by a top-level `switch` construct. A complete description of an example application developed by our methodology is given in the Appendix.

Programmatic PIPE models obtained by our methodology can be viewed as compact representations of all pertinent scenarios and, in this sense, are more expressive than scenario grammars [27] and scenario schema [46]. Using program compaction techniques [16, 49], we can further curtail the explosion of scenario possibilities. Any program analysis technique can then be applied to aid in scenario analysis. For example, the technique of *program slicing* [7] can help reason about the program parts that will be affected by changes in a given scenario. By comparing these effects to those deduced before the change, we can reason about the orthogonality of scenarios.

4.4 Related Research

As narrative descriptions of use, scenario-based methods have become prevalent in various applications, including requirements analysis [11, 29, 62] and user interface design [9]. The design of software systems from scenarios (as opposed to purely functional, solution-first specifications) is the cornerstone of our approach. Most relevant to our presentation is Potts's distinction between inducing scenarios (from interaction) and deducing scenarios (from specifications) [46]. The operational definition of goal-achieving actions emphasized in [46] is similar to our explanation structure. For instance, Potts employs a goal hierarchy where leaves are associated with user actions and which serve to operationalize goals. In addition, we are able to mechanically transform an information space using attributes of such an explanation structure (by partially evaluating the programmatic representation of the operationalized explanation). Other applications in automated software engineering [23] and information pattern extraction [28], while supporting explanation-based views of scenario analysis, have not been connected to the partial evaluation aspect, which is critical for the PIPE methodology of personalization.

The PIPE approach is also related to the use of task models in software design. Traditionally, such integration has been achieved by symbolic modeling techniques, motivated by object oriented (OO) design [53] and languages such as UML [31]. More recent efforts in personalization applications can be found in [13, 21, 36, 52]. In [21], the derivation of models of interaction from use cases is presented. Kramer et al. [36] emphasize the importance of task analysis and advocate end-user analysis of algorithms and tools employed in personalization systems. In [52], the authors emphasize an OO modeling of an information system, where personalization is introduced as a function from the conceptual design stage. PIPE's support for personalization, on the other hand, is built into the programmatic model of the information space and doesn't require any special handling. It also emphasizes properties such as the closure of personalization operators and the factorizability of information spaces, that help relate design decisions to needs identified through scenario analysis. While these same issues are pertinent in [21, 52], the approach there is more reminiscent of design patterns (and integrating requirements via OO analysis), whereas our idea is to use explanations to identify opportunities for providing personalization facilities. PIPE's approach also makes more effective use of domain-specific knowledge, both embodied in the scenarios and assumed in the modeling of the information seeking activity.

As Russell and Norvig point out, empirical analysis of efficiency is central to EBG [56]. They emphasize that 'the efficiency of an [information system factorization] is actually its average-case complexity on a population of [scenarios that are likely to be encountered].' Being too specific when operationalizing explanations will lead to making more distinctions than losing them, contributing to lesser orthogonality (salience, as used in [46]) among scenarios. Defining operationality [33] carefully in the personalization context is an area for future research.

5 Discussion

This research makes contributions to the state-of-the-art in both personalization systems and scenario-based design. For personalization, we have clarified the aspects of requirements specification and high-level elucidation of goals, showing how explanations from usage scenarios can provide models for PIPE. In particular, the problem of designing a representation has been formalized as a search through a space of PIPE models, driven by the operationality criterion. The techniques presented in this paper can also be used to analyze existing personalization facilities, by determining if they address the requirements and opportunities of observed usage scenarios.

Our methodology of developing explanations of scenarios also adds value to the overall effort of creating scenario-based descriptions of software and systems and is a further argument for adopting scenario-based design (SBD) methods. By adopting the EBG view of operationalization, and for applications such as personalization, we can use a strong domain theory to reason about how scenarios can be effectively supported. In particular, our methodology helps to propositionalize information system designs.

From the SBD viewpoint, PIPE emphasizes programmatic approaches of transforming between representations. We can extend our approach to investigate other opportunities for partial evaluation and also to include other program transformation techniques. This will support the provision of views on scenarios, performing tree-manipulation operations, and propagating effects of changes through a representation. For instance, the paradigm of 'training wheels in a user interface' [10] which relies on masking functionality can be expressed using such methods.

It should be remarked that both PIPE and the explanation-based view of operationalization are two *specific* choices that we have made in understanding the early stages in the lifecycle of personalization systems. Programs and partial evaluation serve the role of a modeling methodology and transformation technique for information spaces; explanations supply the mechanism that connects needs and requirements identified from SBD to modeling choices in PIPE. While this is admittedly only one (and to our knowledge, the first) approach, it provides a glimpse into how other lifecycle models can be organized and how they will differ from models for general software systems.

This investigation suggests many interesting avenues for future research. It is good EBG tradition to identify novel ways in which explanations can be generalized and personalization is fertile with opportunities. For instance, PIPE models allow for out-of-turn interactions (by partial evaluation). This helps overcome the mismatch between the user's mental model and the facilities available for describing the information seeking goal. In a non-PIPE implementation, the user has to manually reconcile this mismatch and perform an exploratory mode of seeking before being able to use the system to satisfy the goal. We would like to generalize such scenarios to recognize when 'exploratory steps have been used because out-of-turn interaction was not possible.' For instance, this would allow us to explain that 'Nancy clicked on all links at the top-level page, not because this is what she wanted but because she was exploring to see which one of them led to her choice of link at the second-level.' Another form of generalization pertains to the constructive induction of intermediate subgoals in explanation structures. Recall that we employed think-aloud protocols to augment our domain theory to support certain explanations. Automating the induction of subgoals for recurring patterns of interaction (such as manual information integration) is a possible direction for future work.

The concept of operationality can be explored more carefully in the context of personalization. Our study has exploited only two dimensions of operationality, namely the number of explanation trees and the operationality boundaries in each. Once again EBG research [33] suggests other important dimensions — such as variability, granularity, and certainty — which can be used to define an 'operationality assessment procedure.' Studying these concepts for information systems will allow the characterization of personalization applications in terms of operationality dimensions. For instance, differences between news-feed customization services (e.g., `myCNN.com`, PointCast) can be expressed in terms of operationality. Such characterizations will also aid in clarifying the concept of utility (and usability) of personalization systems, since operationality is primarily concerned with empirical efficiency of models.

The notion of a personalization lifecycle can be usefully extended, to support iterative improvement of PIPE models and to include stages like verification and validation. Support for iterative refinement is important in extending and composing existing personalization systems. It requires a tighter integration between the explanation construction procedure and the way in which scenarios are selected for explanation. Iterative improvement can also benefit from existing approaches to scenario repair and related EBG techniques such as 'learning by failing to explain [22].' Methodologies for verification and validation can be incorporated in our framework, in the form of analytic and empirical frameworks for utility analysis [17]. Such frameworks can take advantage of the characterization of the space of PIPE models produced by explaining scenarios and prior knowledge of the distribution of problem scenarios.

An emerging frontier involves modeling *context* in information systems. Consider:

> **Person:** Remember the hotel where we hosted the annual convention?
> **Secretary:** Yes.
> **Person:** Reserve it for next Friday's event.

Creating a personalization system that exploits context amounts to storing and retrieving smaller (or partial) explanations for use in constructing larger-scope explanations. Scenarios that do not permit complete explanations can also be interpreted as activities for building and organizing context, for use in later situations. The explanation-based view of scenarios allows the decomposition of structures to aid in such reasoning.

Our systems-oriented view of personalization will find greater acceptance if tools and software are available for automating various aspects of the methodology. For instance, scenario management tools and explanation engines can be prototyped for targeted information spaces. Specific techniques for web mining and modeling user interactions can be incorporated as reusable sub-explanations. This will allow us to design personalization systems around existing system infrastructure. To aid in the maintainability of PIPE models, specific scenario libraries (called 'chunking [37]' in AI) and 'frequently used explanations' can also be designed.

The central role played by the domain theory in our methodology signifies a back-to-basics approach in personalization system design. For a given information seeking activity, a domain theory is characterized by its 'explanatory power' and how effectively it allows us to define the parameters of a personalization space. This suggests that we should aim for a more fundamental understanding of how domain theories characterize information spaces and the situations to which they can be usefully applied. Our methodology is the first in which such questions can be directly expressed. Extending work in these directions will help us to architect an information resource for personalization and to provide rigorous metrics for evaluating the applicability of PIPE in a new situation. Together, they will take important steps in establishing a lifecycle of personalization system design.

Acknowledgements

Saverio Perugini helped implement the study presented in the Appendix. Marcos Gonçalves identified several pertinent references. Robert Capra helped make connections from our work to mixed-initiative interaction and contextual abstractions. All three colleagues read drafts of this paper and provided important comments.

References

[1] G. Adomavicius and A. Tuzhilin. Using Data Mining Methods to Build Customer Profiles. *IEEE Computer*, Vol. 34(2):pages 74–82, February 2001.

[2] C.C. Aggarwal, J.L. Wolf, K.-L. Wu, and P.S. Yu. Horting Hatches an Egg: A New Graph-Theoretic Approach to Collaborative Filtering. In *Proceedings of the Fifth ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'99)*, pages 201–212. San Diego, CA, 1999.

[3] M. Balabanović and Y. Shoham. Fab: Content-Based, Collaborative Recommendation. *Communications of the ACM*, Vol. 40(3):pages 66–72, 1997.

[4] M. Beckley. Comment on online personalization. The Appian Web Personalization Report, 2000.

[5] N.J. Belkin. Helping People Find What They Don't Know. *Communications of the ACM*, Vol. 43(8):pp. 59–61, August 2000.

[6] H. Berghel. Caustic Cookies. *Communications of the ACM*, Vol. 44(5):pages 19–22, May 2001.

[7] D.W. Binkley and K.B. Gallagher. Program Slicing. *Advances in Computers*, Vol. 43:pages 1–50, 1996.

[8] J.M. Carroll. *Making Use: Scenario-Based Design of Human-Computer Interactions*. MIT Press, Cambridge, MA, 2000.

[9] J.M. Carroll. Making Use: Scenarios and Scenario-Based Design. In *Interfacing Reality in the New Millennium: Proceedings of OzCHI 2000*, pages 35–48. Canberra: Ergonomic Society of Australia, December 2000.

[10] J.M. Carroll and C. Carrithers. Training Wheels in a User Interface. *Communications of the ACM*, Vol. 27(8):pages 800–806, August 1984.

[11] J.M. Carroll, M.B. Rosson, G. Chin Jr., and J. Koenemann. Requirements Development in Scenario-Based Design. *IEEE Transactions on Software Engineering*, Vol. 24(12):pages 1156–1170, December 1998.

[12] I. Cingil, A. Dogac, and A. Azgin. A Broader Approach to Personalization. *Communications of the ACM*, Vol. 43(8):pages 136–141, August 2000.

[13] K. Claypool, L. Chen, and E.A. Rudensteiner. Personal Views for Web Catalogs. *IEEE Data Engineering Bulletin*, Vol. 23(1):pages 10–16, March 2000.

[14] R. Cooley, B. Mobasher, and J. Srivastava. Data Preparation for Mining World Wide Web Browsing Patterns. *Knowledge and Information Systems*, Vol. 1(1):pages 5–32, February 1999.

[15] D.R. Cutting, D. Karger, J. Pedersen, and J.W. Tukey. Scatter/Gather: A Cluster-Based Approach to Browsing Large Document Collections. In *Proceedings of the Fifteenth Annual International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 318–329. Copenhagen, Denmark, June 1992.

[16] S.K. Debray, W.S. Evans, R. Muth, and B. De Sutter. Compiler Techniques for Code Compaction. *ACM Transactions on Programming Languages and Systems*, Vol. 22(2):pages 378–415, March 2000.

[17] G. DeJong. Explanation-Based Learning. In A.B. Tucker, editor, *The Computer Science and Engineering Handbook*, pages 499–520. CRC Press, 1997.

[18] G. DeJong and R. Mooney. Explanation-Based Learning: An Alternate View. *Machine Learning*, Vol. 1:pages 145–176, 1986.

[19] S.K. Donoho and L.A. Rendell. Rerepresenting and Restructuring Domain Theories: A Constructive Induction Approach. *Journal of Artificial Intelligence Research*, Vol. 2:pages 411–446, 1995.

[20] N.S. Flann and T.G. Dietterich. A Study of Explanation-Based Learning Methods for Inductive Learning. *Machine Learning*, Vol. 4:pages 187–266, 1989.

[21] N. Guell, D. Schwabe, and P. Villain. Modeling Interactions and Navigation in Web Applications (Extended Version). In *Lecture Notes in Computer Science: Proceedings of the World Wide Web and Conceptual Modeling'00 Workshop, ER'00 Conference*, volume 1921. Springer, Salt Lake City, 2000.

[22] R.J. Hall. Learning by Failing to Explain: Using Partial Explanations to Learn in Incomplete or Intractable Domains. *Machine Learning*, Vol. 3:pages 45–77, 1988.

[23] R.J. Hall. Explanation-Based Scenario Generation for Reactive System Models. *Automated Software Engineering*, Vol. 7:pages 157–177, 2000.

[24] M. Hearst. Next Generation Web Search: Setting Our Sites. *IEEE Data Engineering Bulletin*, Vol. 23(3):pages 38–48, September 2000.

[25] W.C. Hill and J.D. Hollan. History-Enriched Digital Objects. *The Information Society*, Vol. 10:pages 139–145, 1994.

[26] H. Hirsh, C. Basu, and B.D. Davison. Learning to Personalize. *Communications of the ACM*, Vol. 43(8):pages 102–106, August 2000.

[27] P. Hsia, J. Samuel, J. Gao, D. Kung, Y. Toyoshima, and C. Chen. Formal Approach to Scenario Analysis. *IEEE Software*, Vol. 11(2):pages 33–41, March 1994.

[28] S.B. Huffman. Learning Information Extraction Patterns from Examples. In *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing*, pages 246–260. Springer Verlag, 1996.

[29] M. Jarke, X.T. Bui, and J.M. Carroll. Scenario Management: An Interdisciplinary Approach. *Requirements Engineering*, Vol. 3:pages 155–173, 1998.

[30] N.D. Jones. An Introduction to Partial Evaluation. *ACM Computing Surveys*, Vol. 28(3):pages 480–503, September 1996.

[31] H. Kaindl and M. Carroll. Symbolic Modeling in Practice: Introduction to the Special Section. *Communications of the ACM*, Vol. 42(1):pages 28–30, 1999.

[32] P.B. Kantor, E. Boros, B. Melamed, V. Menkov, B. Shapira, and D.J. Neu. Capturing Human Intelligence in the Net. *Communications of the ACM*, Vol. 43(8):pages 112–115, August 2000.

[33] R.M. Keller. Defining Operationality for Explanation-Based Learning. *Artificial Intelligence*, Vol. 35:pages 227–241, 1988.

[34] D. Kieras. Task Analysis and the Design of Functionality. In A.B. Tucker, editor, *The Computer Science and Engineering Handbook*, pages 1401–1423. CRC Press, 1997.

[35] J.A. Konstan, B.N. Miller, D. Maltz, J.L. Herlocker, L.R. Gordon, and J. Riedl. GroupLens: Applying Collaborative Filtering to Usenet News. *Communications of the ACM*, Vol. 40(3):pages 77–87, March 1997.

[36] J. Kramer, S. Noronha, and J. Vergo. A User-Centered Design Approach to Personalization. *Communications of the ACM*, Vol. 43(8):pages 45–48, August 2000.

[37] J.E. Laird, A. Newell, and P.S. Rosenbloom. SOAR: An Architecture for General Intelligence. *Artificial Intelligence*, Vol. 33(1):pages 1–64, 1988.

[38] P. Maglio and R. Barrett. Intermediaries Personalize Information Streams. *Communications of the ACM*, Vol. 43(8):pages 96–101, August 2000.

[39] U. Manber, A. Patel, and J. Robison. Experience with Personalization on Yahoo! *Communications of the ACM*, Vol. 43(8):pages 35–39, August 2000.

[40] B. Mobasher, R. Cooley, and J. Srivastava. Automatic Personalization Based on Web Usage Mining. *Communications of the ACM*, Vol. 43(8):pages 142–151, August 2000.

[41] M.D. Mulvenna, S.S. Anand, and A.G. Buchner. Personalization on the Net Using Web Mining. *Communications of the ACM*, Vol. 43(8):pages 123–125, August 2000.

[42] M. Perkowitz and O. Etzioni. Adaptive Web Sites. *Communications of the ACM*, Vol. 42(8):pages 152–158, August 2000.

[43] P. Pirolli. Exploring and Finding Information. In J.M. Carroll, editor, *Toward a Multidisciplinary Science of Human-Computer Interaction*. Morgan Kaufmann, San Francisco, CA, 2001. to appear.

[44] P. Pirolli and S.K. Card. Information Foraging in Information Access Environments. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'95)*, pages 51–58. Denver, CO, 1995.

[45] P. Pirolli, J. Pitkow, and R. Rao. Silk from a Sow's Ear: Extracting Usable Structures from the Web. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'96)*, pages 118–125. Vancouver, Canada, 1996.

[46] C. Potts. Using Schematic Scenarios to Understand User Needs. In *Proceedings of the Symposium on Designing Interactive Systems: Processes, Practices, Methods, and Techniques (DIS'95)*, pages 247–256. Ann Arbor, MI, August 1995.

[47] N. Ramakrishnan. PIPE: Web Personalization by Partial Evaluation. *IEEE Internet Computing*, Vol. 4(6):pages 21–31, Nov-Dec 2000.

[48] N. Ramakrishnan, R. Capra, and M. Pérez-Quiñones. Mixed-Initiative Interaction = Mixed Computation. Technical Report cs.PL/0110022, Computing Research Repository (CoRR), 2001. Available at `http://xxx.lanl.gov/abs/cs.PL/0110022`.

[49] N. Ramakrishnan and S. Perugini. The Partial Evaluation Approach to Information Personalization. *ACM Transactions on Information Systems*, 2001. Communicated for publication. Also available as Technical Report cs.IR/0108003, Computing Research Repository (CoRR), `http://xxx.lanl.gov`.

[50] P. Resnick and H.R. Varian. Recommender Systems. *Communications of the ACM*, Vol. 40(3):pages 56–58, 1997.

[51] D. Riecken. Personalized Views of Personalization. *Communications of the ACM*, Vol. 43(8):pages 26–28, 2000.

[52] G. Rossi, D. Schwabe, and M. Guimarães. Designing Personalized Web Applications. In *Proceedings of the World Wide Web Conference (WWW'10)*. Hong Kong, May 2001.

[53] M.B. Rosson. Integrating Development of Task and Object Models. *Communications of the ACM*, Vol. 42(1):pages 49–56, 1999.

[54] M.B. Rosson and J.M. Carroll. *Usability Engineering: Scenario-Based Development of Human-Computer Interaction*. Morgan Kaufmann, Redwood City, CA, 2001. In press.

[55] D. Rus and D. Subramanian. Customizing Information Capture and Access. *ACM Transactions on Information Systems*, Vol. 15(1):pages 67–101, 1997.

[56] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1995.

[57] G.M. Sacco. Dynamic Taxonomies: A Model for Large Information Bases. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 12(3):pages 468–479, May/June 2000.

[58] I. Somerville. Software Process Models. In A.B. Tucker, editor, *The Computer Science and Engineering Handbook*, pages 2259–2277. CRC Press, 1997.

[59] M. Spiliopoulou. Web Usage Mining for Web Site Evaluation. *Communications of the ACM*, Vol. 43(8):pages 127–134, August 2000.

[60] L. Terveen, W. Hill, B. Amento, D.W. McDonald, and J. Creter. PHOAKS: A System for Sharing Recommendations. *Communications of the ACM*, Vol. 40(3):pages 59–62, March 1997.

[61] F. van Harmelen and A. Bundy. Explanation-Based Generalisation = Partial Evaluation. *Artificial Intelligence*, Vol. 36(3):pages 401–412, 1988.

[62] A. van Lamsweerde and L. Willemet. Inferring Declarative Requirements Specifications from Operational Scenarios. *IEEE Transactions on Software Engineering*, Vol. 24(12):pages 1089–1114, December 1998.

[63] A. Wexelblat and P. Maes. Footprints: History-Rich Tools for Information Foraging. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'99)*, pages 270–277. Pittsburgh, PA, 1999.

[64] M.D. Williams. What makes RABBIT run? *International Journal of Man-Machine Studies*, Vol. 21:pages 333–352, 1984.

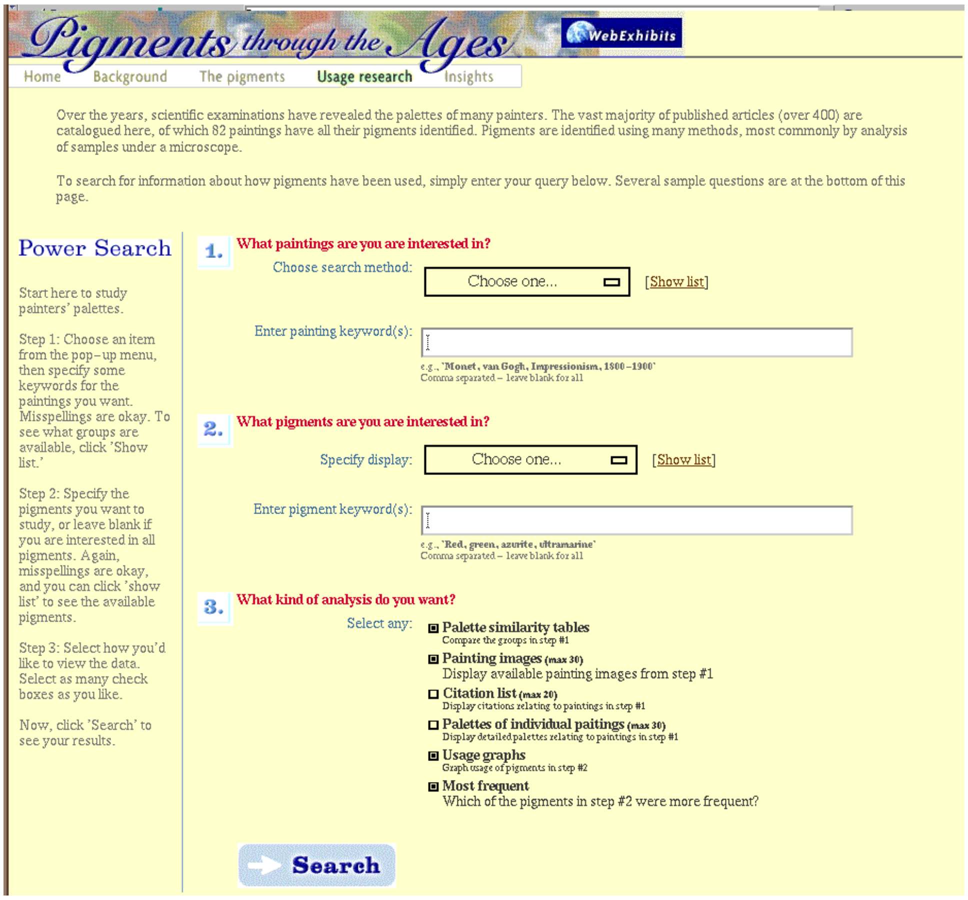

Figure 15: Original layout of the 'Pigments through the Ages' website.

6 Appendix: An Example Application

As a demonstrator of the ideas presented in this paper, we describe the personalization of the 'Pigments through the Ages' website at [`www.webexhibits.org/pigments`], a public service that uses pigment analysis catalogs to identify and reveal the palettes of painters in different eras and genres. As shown in the top-level interface depicted in Fig. 15, a variety of information resources are modeled in this site. Users can search for paintings by artist, style, period, or by membership in a particular pigment group. Notice also that the interface in Fig. 15 provides some 'hardwired' scenarios such as comparing palette similarity tables or analyzing pigment usage in a certain age. However, even a simple query such as 'What is the influence of colors from the baroque era on the neo-classic styles of paintings?' cannot be accommodated without manual information integration because the interaction sequences are hardwired.

Problem Scenario Development

A group of 10 participants were identified and instructed to explore the layout and organization of information at this site. After a period of acquainting themselves with the site, they were asked to identify one specific query (or analysis) and use the facilities at the site to answer their query. The exact interaction sequences (including clicked hyperlinks, manual information integration) was recorded for all the participants. One such scenario is described in Fig. 16.

Jeremy's scenario : Jeremy is attempting to compare how colors from the baroque era were used in the neo-classic paintings. In particular, he is interested in usage graphs for pigments in neo-classic that are most similar to ones used in baroque. Jeremy surveys the existing facilities at the site and chooses 'by Artist, Style, or Period' as the `search method`. Next, he specifies 'neo-classic baroque' in the text box for `painting keywords.` He (correctly) reasons that this specification identifies the paintings to be used for analysis. Next, he chooses 'All pigments' from the `specify display` dropbox and selects 'Usage graphs' as the `analysis kind`. He (incorrectly) assumes that this will compare every painting from baroque with every painting from neo-classic and that the system will present an usage graph for each such comparison. On inspecting the results Jeremy notices, instead, that the usage graphs are for *all* pigments used in the set $\{$neo-classic $\cup$ baroque$\}$, not quite what he had in mind. He wonders for 5 minutes and realizes that the site does not provide any direct interface to specify his form of analysis.

Jeremy pursues an alternative strategy. He is going to first find the common colors across baroque and neo-classic, and then determine their usage patterns in neo-classic. He opens an additional browser window. In the new one, he specifies 'by Artist, Style, or Period' as the `search method` and 'neo-classic baroque' as the `painting keywords.` He clicks on the palette similarity table checkbox and obtains a matrix of values that indicate how colors from one period were utilized in another. As he expected, this time the specialized interface interprets that the two groups he specified in `painting keywords` have to be compared with each other. The results page provides a matrix whose entries denote similarity levels. He picks out the pigments corresponding to the highest similarity levels and shifts control to his old browser window. There, he types in these pigments in the `pigment keywords` textbox and, this time, types only 'neo classic' in the `painting keyword` textbox. All other settings were as he left them (including the 'Usage graphs' request). This time, the output screen provides a histogram of the usage of the baroque pigments in neo-classic, which satisfies Jeremy's information seeking goal.

Figure 16: A 'Pigments through the Ages' scenario whose explanation was subsequently operationalized.

Domain Theory

The domain theory for this application was obtained from three sources. The first was an explicit crawl of the site that outlined how interactions result in specification of information seeking attributes. The second was a 'Background' webpage at [`http:// webexhibits.org/pigments/intro/index.html`] that outlined a schema for how the website should be used and how to browse through the various sections. For instance, one mode of operation suggested at the site was to choose the 'Usage Research' category and see which pigments were used in different paintings. Another mode of operation was to jump to a particular pigment page and then browse through categories of information outlining technical details. All of these forms of navigation at the site were modeled as possibilities for satisfying the top-level personalization goal. The third source was from analyzing user interactions that revealed opportunities for information integration across multiple pages of the site. Once again these were modeled as specific possibilities of instantiating the top-level personalization goal. The domain theory was represented in CLIPS but only certain portions of the theory were materialized when conducting explanations. We ensured that all 10 scenarios can be explained by the domain theory.

Constructing and Analyzing Explanations

Explanations of user interactions revealed that starting from either artists, paintings, or eras, the users systematically browsed through subcategories or compared palettes to arrive at the relevant pigments (used by that artist, in the painting, or in that era, respectively). Furthermore, all pigments share common modes of information seeking, such as browsing through their history of use, procedures for preparation, and technical details of their chemical composition.

Operationalization

We hence operationalized the explanation structure(s) as two function invocations in sequence, the first to determine an appropriate pigment category, and the second to browse through the entries in that category by various means. We thus arrived at a single structure in support of all the 10 scenarios. The factorization implied by the structure permits the following analysis:

> For the pigment categories defined by X, provide the details involving Y.

X denotes information such as a genre, a style, a painting, or a particular artist. Y denotes features of pigments such as usage history, chemical composition, and dyeing processes. Each of X and Y could be defined either directly or involving attributes of other entities that relate to them. For instance X could be a painting keyword such as 'Rembrandt' (which means that we are interested in pigments used by Rembrandt) or it could be the result of the palette similarity function applied on two painting styles (which would mean that we are interested in pigments that satisfy some acceptable threshold for similarity). In addition, there are dependencies among the allowed entries in X and Y. Jeremy's scenario satisfies the above template where X denotes the result of applying the palette similarity function to 'neo-classic baroque' and Y denotes 'usage graphs.'

Representation in PIPE

To support the user in defining X and Y, we modeled various information sources such as the catalog contents (which contains paintings from 950 to 1981), the palette similarity table (which is just a function in our program), citations of paintings, and auxiliary information such as images, histories, where the painting is housed, and other legends. Overlaps of painting styles across different periods contribute to sources of semistructure in this site and a corresponding reduction in the composite program size. Our composite program was represented in 2369 lines of C code involving hundreds of program variables that could be turned on or off with user input. Approaches for modeling the various elements in this study are described in [49]. We did not implement the mappings from the (specialized) program back to the information space because we only wanted to evaluate the effectiveness of our modeling (more on this below).

Evaluation

The evaluation of systems designed with PIPE is an interesting issue in itself; we address this topic in greater detail in [49]. While user satisfaction surveys show convincing results (see, for instance [47]), PIPE is more a modeling methodology for personalization, and not a system per se. As such, its effectiveness depends on what is modeled (and how). The research presented in this paper gives us a direct way to assess the modeling capability of PIPE.

We identified a test group of 15 users (different from those who participated in the original scenario analysis) and asked them to experiment with the unpersonalized pigments site. Each of them was then asked to identify and carefully describe 2-3 personalization scenarios. In total, 35 scenarios were identified. An example is the following analysis:

> What are the symbolic connotations of pigments used by artists in the Renaissance era?

(One of the answers to this query is a web page that describes the interpretation of red as invincibility in Jan van Eyck's 1434 classic *Arnolfini Wedding*.) We then evaluated our PIPE representation by the fraction of scenarios that can be described in our modeling (and are hence amenable to personalization by partial evaluation). All scenarios except two passed our test. The two unmodelable scenarios involved the 'Orpiment' pigment which was listed in both the 'Yellow' and 'Orange' categories and was variously referred to by users as belonging to one, but not the other. This ambiguity implies that our modeling did not contain sufficient information to complete the proof (i.e., it could not uniquely distinguish between these two distinct specifications involving X). More contextual information needs to be encoded in our modeling so that this ambiguity is resolved.

A full listing of the scenarios used in evaluation follows. Except for scenarios 5 and 8, all others can be supported. Scenarios 1, 8, 13, 20, and 22 indicate preferences for presentation which can be addressed when we recreate the personalized pages from the specialized program. Scenario 25 states preferences for interactions at many levels. This amounts to repeated partial evaluations of the information space, in the order of attributes stated by that user. Scenarios such as 27 imply a desire to use complete evaluation with the designed PIPE model, not partial evaluation.

1. What are the symbolic connotations of pigments used by artists in the Renaissance era? Arrange the results on a single page, in alphabetical order of pigments.

2. (similar to Jeremy's scenario) I would like to see how colors used in 1800-1900 have influenced paintings in the early part of the 20th century. Show usage graphs for pigments that are similar across these eras.

3. I would like to specify a pigment choice, not based on a property of painting, style, or era. Rather, I want all pigments for which descriptions of chemical composition are available. So, if a pigment does not have this information, it should not be listed.

4. I would like to browse through pigment details at the root page, not go through information about paintings or painters.

5. I am interested to see how usage of pigments of a subcategory compares with that of pigments in the parent category. For instance, does Orpiment usage correlate with usage of Yellow in 1800-1900 paintings?

6. The site facility lists only at most 10 palettes at a time. If I need more, I have to carefully pose multiple subqueries so that each of them does not involve more than 10 paintings. Can you fix this problem so that I can see all palettes?

7. My period specifications don't seem to work at the site. When I manually browse the site, I see annotations such as '1900-2000' and '1650-1750.' But when I pose my range as '1875-1925,' the site doesn't seem to understand. Should I have to break my range up into these prespecified ranges? That seems cumbersome.

8. Can I get a listing of pigments used by the Impressionists along with their parent categories, side-by-side on a single page?

9. I would like the pigments arranged by history (e.g., middle ages), followed by an organization along countries.

10. I would like to be able to choose a pigment according to ease of preparation.

11. I would like to search for pigments using a combination of two criteria (such as geographical use and time period), but the site allows only one at a time.

12. I would like the interaction to proceed as follows: At the first level, I will make a choice of history of usage, after which I will browse the site in the traditional manner.

13. I would like pigments of the Blue category to be identified in alphabetical order. Then I would like to see the swatches of paints from the top five to be placed alongside swatches of paint from the bottom five. This will show me the range of intensities of Blue available.

14. I would like a listing of pigments by their chemical name, not their colloquial names.

15. I find myself repeatedly browsing pigments' pages to study the fascinating stories of how these pigments originate. Some of these pages don't seem to have any stories. Can you provide me a listing of only those pigments that have stories?

16. Which are the pigments that have German names or equivalents?

17. I would like to see the descriptions of pigments that have pictures in them. I am not interested in purely textual descriptions.

18. I would like to directly select a specific pigment from a list, on the first page.

19. Which pigments have been used by Alchemists?

20. Can you cascade the brief descriptions of pigments and remove all the other information pertaining to preparation and technical details?

21. Which pigments have citations to them? I would like a listing of only those.

22. I am interested in the Green earth pigment. I would like a page that has pictures of paintings and along with each, a picture of the swatch of paint from Green earth. This is just so that it is visually easy to see how much the painting emphasizes Green earth.

23. I would like pigments arranged by the year in which they were first introduced.

24. I am interested in making pigments. Can you please instruct me how to make every pigment in the purple category?

25. I am interested in the citation lists for green pigments. However, I would like to browse them by first making a selection of artist. Then I will select a period. And finally I will select among titles, if there are choices. For the green pigments used in these titles, I would like to see the citations.

26. How is the name for the Azurite pigment derived? What is the word origin?

27. Produce the 3D model for Titanium dioxide.

28. I know that Kandinsky has suggested that black indicates an inner harmony of silence. Can I see which forms of blacks were used in his paintings?

29. Can you give information about how pigments are used for body art, tattoos and other non-conventional forms of paintings?

30. What are the time periods when Lemon Yellow was used?

31. For paintings by Monet, can you display the top five most frequent pigments?

32. What forms of white pigments have been used in paintings? Which ages were they introduced in?

33. Give a histogram of how chrome yellow has been used over the times.

34. Arrange histograms of all purple pigments used after the 17th century.

35. How do pigments used by Picasso compare in usage with pigments used in the 1920s, in general?