International Journal of Information Security manuscript No.
(will be inserted by the editor)

Esma Aïmeur · Gilles Brassard · José M. Fernandez · Flavien Serge Mani Onana

ALAMBIC: A Privacy-Preserving Recommender System for Electronic Commerce

Abstract Recommender systems enable merchants to assist customers in finding products that best satisfy their needs. Unfortunately, current recommender systems suffer from various privacy-protection vulnerabilities. Customers should be able to keep private their personal information, including their buying preferences, and they should not be tracked against their will. The commercial interests of merchants should also be protected by allowing them to make accurate recommendations without revealing legitimately compiled valuable information to third parties. We introduce a theoretical approach for a system called ALAMBIC, which achieves the above privacy-protection objectives in a hybrid recommender system that combines content-based, demographic and collaborative filtering techniques. Our system splits customer data between the merchant and a semi-trusted third party, so that neither can derive sensitive information from their share alone. Therefore, the system could only be subverted by a coalition between these two parties.

Keywords privacy protection · recommender system · secure two-party computation · semi-trusted third party · Web personalization

1 Introduction

During the industrial revolution, judge Thomas Cooley [26] defined privacy as being "the right to be let alone". Since then, this definition gradually evolved to take into account other changes in the World, in particular those relating to new technologies. For instance, Westin [75] defined privacy as the claim of individuals to determine what information about themselves is known to others, as well as when and how it is used. That definition, which is still current, means that each individual must be the master of his person and of the data that he considers private. In the context of e-commerce, it is important to know how to *classify* data as private or not. Indeed, the concept of privacy is relative in the sense that what is private for one person is not necessarily private for another [30]. In addition, it is important to investigate how to *preserve* privacy about individuals, while taking into account the classification that they made for their data. If classification can be taken care of relatively easily by the customer himself, privacy preservation is far more difficult. In fact, apart from customers, privacy preservation relies on the good faith of individuals and organizations that gather data on these customers in an open electronic environment such as the Internet.

Web personalization is an Internet technique for adapting websites to individuals [64,58,61,53,45,59,38,70,13, 32,72, etc.]. To quote the Preface of [52], "The actions can range from simply making the presentation more pleasing to anticipating the needs of a user and providing customized and relevant information". In the case of electronic commerce, web personalization is also used to match products (goods and services) as well as advertising content to customers. In general, it is based on the *user profile*, which summarizes what the website knows about the individual user. This profile includes the user's demographic information, interests, preferences, purchase records, navigation behaviour, etc. There exist several ways to compile user profiles [72]: First, information can be solicited directly from the user through a questionnaire or an interview. Second, the website can observe what the user is doing online. This can be done, for instance, through the use of *cookies*, *spyware* or *web bugs* [17]. Third, user profiles can be generated through marketing research, using techniques such as data mining. In the case of electronic commerce, profiles can also be built based on what customers have bought in the past (pur-

This work is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC). In addition, G.B. is supported in part by the Canadian Institute for Advanced Research (CIFAR) and the Canada Research Chair Programme.

E. Aïmeur · G. Brassard · F. S. Mani Onana
Département d'informatique et de recherche opérationnelle
Université de Montréal
E-mail: {aimeur—brassard—manionaf}@iro.umontreal.ca

J.M. Fernandez
Département de génie informatique,
École Polytechnique de Montréal,
E-mail: jose.fernandez@polymtl.ca

chase patterns), a technique widely used by `Amazon.com` for instance.

These different approaches should consider "the size and the heterogeneous nature of the data itself, as well as the dynamic nature of user interactions with the Web" [52]. In addition, user modelling [7], which consists in the construction of computer-based models for handling a user's mental activities and behaviour, captures and manages his attitude in the past (if any), as well as what can occur in his mind when he faces the website. Finally, *recommender systems* make inferences from data provided by users on other issues or by analysing similar users [68]. In other words, recommender systems used on the Web are a special case of the general problem of Web personalization.

In e-commerce, recommender systems allow entities providing goods or services to guide the choices made by potential *customers*. The recommendation can be issued by the merchant who sells the goods or services themselves or by intermediary brokers, but we shall henceforth use the generic term *merchant* to refer to the entity operating the recommender system. Similarly, we use the generic term *product* to refer to any item offered by the merchant, whether it be a digital or physical good, or a service to be provided.

Recommender systems have been the subject of much research. In the seventies, the stress was laid on information retrieval in which some terms were searched inside the documents. In the eighties, information filtering was introduced with the comparison of some characteristics versus documents. Then came in the nineties content-based filtering (characteristics versus artefacts) and collaborative filtering (users versus documents). Nowadays, this evolution has led to hybrid recommender systems in which several algorithms are designed to efficiently output product recommendations to the customers. We review some of the basic principles in Section 2.

One major drawback of recommender systems, however, is that in order to obtain accurate recommendations, customers often have to reveal to the merchant information that they may consider private, such as their identity, demographic characteristics, previous buying behaviour, etc. On the other hand, potential privacy-preservation solutions to this problem should not be at the expense of the merchant's interests. In particular, the ability to make good recommendations might provide a competitive advantage that must be protected. Furthermore, the catalogue itself could have in some cases intrinsic commercial value. In both cases, the merchant should be able to protect its interests from competitors, and in particular from those who might be posing as potential customers and users of the recommender system. Thus, the problem of privacy protection in the use of recommender systems applies to both customers and merchants.

The purpose of this paper is to offer techniques to fight on behalf of the customer against profile creation or dossier constitution—and ultimately against the looming threat of Big Brother [22]. Our most important task is to help the customer buy products without compromising the privacy of his profile. In particular, the merchant must be able to recommend products to the customer without knowing the latter's profile and identity. It may seem on first sight that this privacy concern goes against the philosophy of collaborative or demographic filtering (Section 2), in which each customer benefits from the preferences of other customers. Yet, we shall see that this is not necessarily so.

We provide a proof of principle that cryptography can come to the rescue of privacy without sacrificing the genuine benefits that traditional recommender systems would offer. We propose a privacy-preserving hybrid recommender system, consisting of content-based filtering, demographic filtering and collaborative filtering. The main originality of our approach is the introduction of a *semi-trusted third party* that customers trust with certain private information but not with particulars of what they are looking for. In our system, the *Alambic agent* acts on behalf of this third party. In collaboration with the merchant, it provides to the customers accurate recommendations based on their private information. Our system is designed in such a way that neither the merchant nor the semi-trusted third party learn more about the customer than they should. The system can only be subverted by a collusion between the merchant and the semi-trusted third party, a violation of the *division of trust principle*, which we introduce in Section 5.1 and discuss in detail in Section 6.5. In order to be able to compute recommendations, the Alambic agent distils an *essence*, which encodes sensitive information about the customers in a way that cannot be decrypted by any of the concerned parties alone, including the Alambic agent itself. Yet, the essence can be used to produce recommendations by combining standard cryptographic tools (Section 4.2) and secure two-party computation (Section 4.3) between the customer and the merchant.

After this introduction, we review in Section 2 the basic notions of recommender systems. Privacy issues are described in Section 3. We then review in Section 4 privacy protection solutions introduced in other contexts as well as some relevant security concepts. Our detailed solution to protect the privacy of customers as well as merchant sensitive data, namely the ALAMBIC system, is given in Section 5 . This is followed in Section 6 by a discussion of the privacy and security offered by the ALAMBIC system. Finally, we conclude with directions for future work in Section 7.

2 Recommender Systems

Many approaches to recommender systems have been proposed in the literature. According to Burke [12], the best known of these are as follows:

– **Collaborative Filtering**: Accumulates customer product ratings, identifies customers with common ratings and offers recommendations based on inter-customer comparison. In other words, recommendations for a given customer are based on the behaviour and the evaluations of the other customers.

– **Content-based Filtering**: Uses the features of the products and the customer's interest in these features to produce recommendations.
– **Demographic Filtering**: Groups customers according to their demographic information. Products are recommended to a given customer based on the group to which he belongs.
– **Utility-based Filtering**: Uses the features of a product to compute its utility. The customer must input the utility function to apply over the products that describe his preferences. Applying the utility function enables product ranking from which recommended products emerge.
– **Knowledge-based Filtering**: Bases the recommendation of products on inferences about the customer's needs and interests. Here, the knowledge is about the association of a particular product to the need of a particular customer.

In this paper, we introduce our approach to privacy by concentrating on three of these filtering techniques: content-based filtering (CN), demographic filtering (DF) and collaborative filtering (CF). We describe them in more detail in the rest of this section. (Preliminary work dealing exclusively with privacy-preserving demographic filtering, has already been reported [4].) While some are better than others [54], none of the above techniques is perfect. For this reason, *hybrid* recommender systems are often used. This can be done in a variety of ways [12]. Our solution offers a particular kind of *mixed* recommendations, which means that we can simultaneously offer recommendations based on all three filtering techniques that we support. We have chosen to concentrate on the first three of these techniques, as they are currently the most commonly used. We believe that our approach could be equally used for hybrid systems also incorporating utility- and knowledge-based filtering, but this remains to be demonstrated and is the subject of ongoing research.

Example 1 Today, a concerned citizen wanting to keep himself abreast and well informed on world affairs and politics has a myriad of information sources from which to choose. Even if we restrict him to Internet sources, there are hundreds of sources ranging from organized Media, websites of political parties and lobbies, interest groups, *ad hoc* posting in newsgroups, and even blogging and podcasting. Beyond the sheer number of such information sources on any controversial topic, a user might have preferences on the source to be used, such as length and depth, political inclination, objectivity, means of delivery, etc. We will thus consider for our example a hypothetical search-engine on Web items associated with world affairs and politics. The links returned by the initial user query constitute the catalogue of products (or items) matching the minimum requirements of the customer (the concerned citizen). In this case, the objective is to use recommender systems to prioritize this (potentially huge) list of links. Implicit information about user preferences and behaviour can be gathered by the system by keeping track of which URLs were followed from search-engines Web pages, i.e. their *clickstream*.

Because it is reasonable to think that user preferences will be influenced by demographic characteristics, such as age, gender, level of education, wealth, geographical location, etc., we shall presume that this search-engine uses a recommender system based on *demographic filtering* to rank the links returned to a particular user. Such a recommender system could output recommendations of the form "Users with characteristics similar to yours went primarily to these URLs".

At the same time, a *collaborative filtering* recommender system would use clickstream correlations to find previous users with similar clickstreams and make recommendations based on where they went next, for example "Users that went to the URLs that you just visited also went to these ones".

Finally, a *content-based* recommender system would go through the whole list of links and compute similarity values between them and other links that this same user has followed in the past. The similarity values would be based on the value vector of such features as means of delivery (text, RSS, video, audio, etc.), age of the item, language, type of source (commercial, independent, state-operated, individual), length, general topic, etc. The recommendations would be extracted from the new list by selecting those links that are most similar to those selected in the past by this same user. □

2.1 Collaborative Filtering

Collaborative Filtering (CF) was first introduced by the developers of Tapestry [35], a filter-based electronic mail system designed for the intranet at the Xerox Parc Palo Alto Research Center. CF techniques are becoming increasingly important in the context of e-commerce, with the unprecedented growth in the number of customers. The main characteristic of CF is that recommendations for a given customer are based on the behaviour and the evaluations of the other customers [62].

There exist three main categories of CF algorithms [11, 57]: memory-based algorithms, model-based algorithms and hybrid algorithms. Memory-based algorithms use statistical techniques on the entire database of votes to find the neighbourhood of customers having similar tastes. This neighbourhood is then used to output recommendations. Model-based algorithms use a variety of classification techniques, such as data clustering, Bayesian networks, association rules or sequence pattern discovery, to create categories of users and associated generic recommendation patterns (i.e. the "model"), which can be used to issue recommendations without accessing the whole database of votes. Hybrid CF algorithms use both memory-based and model-based algorithms.

For the purpose of discussion, we shall use a popular (memory-based) CF technique, based on the *Nearest Neighbour* classification algorithm [28,27]. However, our general approach to privacy preservation is not tied to this choice. Here, data is represented by a matrix whose entry $v_{u,i}$ represents the rating customer u gave to product i. This entry is set to a null value in case customer u has not rated product i, in which case this entry is not used in the computations. Suppose that the merchant's catalogue, T, contains t products, $p_1, p_2, \ldots, p_t$, and that m customers, $u_1, u_2, \ldots, u_m$ have rated some products from T. Rating predictions for a given customer are produced in two stages, as we now review.

In order to estimate similarity between customers, various metrics have been proposed. One of them, for example, is the *Pearson correlation* [62]. The results obtained range from -1 for negative correlation to $+1$ for perfect positive correlation. Specifically, let

$$corr(c,u) = \frac{\sum_{j \in J}(v_{c,j} - \overline{v_c})(v_{u,j} - \overline{v_u})}{\sqrt{\sum_{j \in J}(v_{c,j} - \overline{v_c})^2 \sum_{j \in J}(v_{u,j} - \overline{v_u})^2}} \tag{1}$$

stand for the correlation between customers c and u, where J is the set of products rated by both customers c and u, $v_{\ell,j}$ is the rating customer ℓ gave to product j and $\overline{v_\ell}$ is the average rating of customer ℓ for the products that belong to J, for $\ell \in \{c,u\}$.

The weighted sum equation below can then be used to predict the rating of a customer c on a product j. The resulting predictions are sorted and those with highest values are considered for recommendation purposes

$$P_{c,j} = \overline{v_c} + \frac{\sum_{u \in U} corr(c,u)(v_{u,j} - \overline{v_u})}{\sum_{u \in U} |corr(c,u)|}, \tag{2}$$

where U is the set of all the customers who have supplied a rating for product j. (In particular, customer c, for whom predictions are being computed, does not belong to U.)

2.2 Content-Based Filtering

When Content-based Filtering (CN) techniques are used, products are compared based on their content. The content of a product can be described by explicit features (attributes or characteristics). In the case of a movie, for instance, the features could be its genre (action, adventure, etc.), year, main actors, etc. The description can also consist of textual documents with their titles, illustrations, tables of contents, etc. Here, we consider only the case of explicit features, but our approach can easily be generalized to textual documents if words are used as features.

CN defines products of interest by their characteristic features. More precisely, a recommender system using CN learns a customer's interests from the features present in products the customer has rated. This enables the system to create customer profiles [55]. As was the case for collaborative filtering, such profiles are long-term customer models. These models can be updated as long as customers rate products and implicitly change their preferences.

CN is also known as "product-to-product correlation" [67]. An algorithm based on product-to-product similarity can therefore be used. For this purpose, products can be represented as vectors in a multidimensional space, where each dimension corresponds to a product feature and each feature value is mapped to a numerical value on that dimension. The similarity between two products i and j can then be computed by using the associated vectors $\mathbf{i}$ and $\mathbf{j}$

$$Sim(i,j) = \frac{\mathbf{i} \cdot \mathbf{j}}{\|\mathbf{i}\| \cdot \|\mathbf{j}\|}. \tag{3}$$

We then use the following equation to compute a prediction for customer c and product j

$$P_{c,j} = \frac{\sum_{i \in S} v_{c,i} Sim(i,j)}{\sum_{i \in S} |Sim(i,j)|}, \tag{4}$$

where S is the set of all products similar to j and the ratings $v_{c,i}$ are the same as in Equation 1.

2.3 Demographic Filtering

Recommender systems based on demographic filtering (DF) aim at categorizing customers based on their demographic information and recommend products accordingly. More precisely, demographic information is used to identify the types of customers that like similar products. DF can be used by any merchant who sells products by using data on individual customers. The key element of DF is that it creates categories of customers having similar demographic characteristics, and tracks the aggregate buying behaviour or preferences of customers within these categories. Recommendations for a new customer are issued by finding to which category he belongs in order to apply the aggregate buying preferences of previous customers in that category.

Even though many categorization techniques have been proposed, we shall concentrate on *Data Clustering* to illustrate demographic filtering as well as our privacy-protection solutions, which are introduced in Section 5. However our approach is independent of the specific categorization technique used in a DF-based recommender system. Clustering has been the subject of much research, particularly in statistics, machine learning and databases (see for example [41]). It aims at forming clusters of similar objects. Objects in a given cluster are similar to each other, but different from objects in another cluster. A clustering algorithm thus needs a metric to compute the distance between two objects. Therefore, objects are typically represented as vectors in a multi-dimensional space and similarities are calculated using the Euclidean distance (or another instance of the Minkowski distance) considering all or a subset of the dimensions. Each cluster is represented by a centroid or a medoid[1], and sometimes radius and density information.

In demographic filtering, clustering is used to create the customer categories mentioned above by considering the set of all previous customers. The objects are the customers, and each dimension of the space represents one of their relevant demographic characteristics. For a given cluster C, its density represents the number of customers in it and its radius

[1] The centroid is a virtual point corresponding to the average of all the points in the cluster, while the medoid is the median point of the cluster.

is a measure of how demographically dissimilar they are. Then, the historical data on buying behaviour or preferences of each customer in C is used to associate to the cluster C an *aggregate* buying behaviour. In its simplest form, this aggregate consists of the list of products $\{p_1, p_2, \ldots, p_c\}$ that were purchased or for which positive feedback was given by customers in C. When a new customer requires a recommendation, the recommender system computes the cluster to which he is closest, say C, and then produces as a recommendation the corresponding list of products.

Of course, for such clusters to be used effectively in recommender systems, the aggregate buying preferences within a cluster must also show sufficient similarity. In particular, the corresponding list must be sufficiently small. Achieving this objective can be done in a variety of ways, such as merging or separating existing clusters, considering similarity distances on the combined space of demographic characteristics and buying preferences/behaviour, or by attempting to apply clustering techniques on the space of products. See [3] for instance. However, the specific way in which clusters are formed is immaterial to our privacy issues.

Example 2 Our concerned citizen happens to be a young male engineering student living in a Northern European country, of Arab origin, who likes soccer, drives a yellow car, and has indicated business and computer games as interests; these characteristics (among others) constitute his demographic profile. The DF-recommender has identified him to be in a cluster of well-to-do and educated young people in developed countries. Historically, people in this cluster have mostly clicked on links going to specialized in-depth print magazines, a certain subset of prolific bloggers, and multimedia clips from state-operated Western European broadcasters. When he queries the search engine for items about the war in Iraq in the last week, he will be provided (in order) with an in-depth article in The Economist, two posts by bloggers on the latest bombing and an audio clip from Deutsche Welle. Of course, there will be other links on that topic, but they will appear after in the ordered list of matches.

On the other hand, the collaborative filtering system would have identified that his clickstream always follows a certain pattern (e.g. sessions normally include two or three out clicks to media clips on the websites of Al-Jazeera, CNN, Deutsche Welle in English, and the Lebanese LBC). As a consequence, and given that many people going to such sites also do so, the recommender system will recommend media clips from the arabic BBC site on the same topics.

Finally, by using the feature extraction methods based on keywords in the text or on semantic information found on XML or RSS-enabled Web pages, the content-based recommender finds items very similar to those that he has clicked on in the past, for example those containing keywords "soccer", the format tag "video", etc. Recommendations would take the form "Since you liked such items as *'Egyptian Movie-Star runs away with Woody Allen'* and *'Beckham scores three goals in UEFA final'*, you will definitely like *'Former Spice Girl Victoria Beckham stars in new film: The Mummy IV: Revenge of the Mummy'*". □

3 Privacy Issues

Recommender systems are very useful, but they typically come at a price: in order to operate, it would seem that they need to collect information on those who use them. We now discuss privacy concerns according to the three techniques presented above.

In the case of *collaborative* and *content-based* filtering, the customer typically provides the merchant with information on his buying behaviour, especially the products he has bought in the past, as well as his ratings on these products. This is done because the computation of predictions and similarity scores is based on the $v_{u,i}$ matrix that stores the rating customer u has given to product i. In the case of *demographic* filtering, customers provide the merchant with demographic information in order for the clusters to be constructed and for new customers to be associated with the appropriate cluster.

The potential abuses from unscrupulous merchants are obvious. They can use the data for many purposes that are illegitimate if not intended by the customer, such as data mining and the collection of statistics. Merchants could pool their information with other merchants and/or governments. They could also sell it. Information coming from diverse sources could be linked, resulting in the constitution of a formidable dossier on the customer [22]. All this could result in a serious erosion of the customer's privacy. Privacy violations are prohibited in many countries, but there is a lack of effective methods to enforce the law. This problem is exacerbated when information is used about individuals without their knowledge. Should the customer have the proof that his privacy has been violated by the merchant, he could complain to the proper authorities, so that justice might be served. However, no amount of "justice" can suffice to restore his privacy.

On the other hand, the privacy of data legitimately accumulated by the merchant must also be protected. In a competitive market, pricing information contained in the catalogue might be sensitive and the merchant might not want to reveal it to competitors. Furthermore, depending on the type of products, the catalogue itself might have value, such as in the case of information brokers. While the merchant might be willing to reveal a few products and prices to a *bona fide* customer, he must take precautions so that the information revealed cannot be used by an unscrupulous competitor who is masquerading as a customer. In fact, the "customer" may be interested in what the merchant is selling, and at what price, for the sole purpose of undercutting him! On the question of whether hiding catalogues and prices would be beneficial or detrimental to customers and merchants, a recent study by Zhu [78] "challenge[s] the 'information transparency hypothesis' (i.e., open sharing of information in electronic markets is beneficial to all participating firms)" and concludes that "in contrast to the popular belief, [...] information transparency could be a double-edged sword". Most importantly, once the merchant has operated long enough that the *essence*—see end of Section 1 or, for more detail, Section 5.2—he has distilled is sufficiently good for use in a recommender system, the essence itself becomes of the highest value. Hence, it should be protected with utmost vehemence from potential competitors.

It is hard to grasp all the implications of the concept of privacy because it depends on so many components (Section 3.2). It is therefore important to design a *privacy framework* to take such components into account. But first, we present some studies that have been conducted with the purpose of classifying customers according to their concerns about privacy.

3.1 Studies About Privacy Concerns

Several surveys have been conducted in order to categorize customers with respect to their privacy concerns. According to Ackerman, Cranor and Reagle [1] there are three main types of customers:

– **Privacy fundamentalists** are against any use of their personal information. They generally refuse to provide websites with such information, even if privacy protection measures have been deployed.
– The **pragmatic majority** is made of users who are also against the use of their personal information. However, a compromise such as the presence of privacy policies on websites may be sufficient for them to purchase online.
– **Marginally concerned** people may often express a general concern about privacy, but they easily provide their personal information to websites.

This study found that 17% of respondents are privacy fundamentalists, 56% are pragmatists and 27% are marginally concerned. In the same study, concerns about persistent identifiers (such as *cookies*) were also considered. Such identifiers, which can be used to track customer activities over time, are used for customized services and advertising across websites.

A subsequent study concentrated on privacy preferences versus customer behaviour [69]. The authors used the questionnaire developed in the previous survey to investigate privacy preferences. Their study showed a significant increase in the number of privacy fundamentalists (30%) and a small decrease in the number of marginally concerned (24%). Furthermore, they split the pragmatic majority into two meaningful groups: *identity concerned* (20%) and *profiling averse* (26%) users. Interestingly, they investigated whether or not customers behave in a way consistent with their stated privacy preferences; it turns out that customers easily forget their privacy concerns once they are inside the Web!

In a more recent study, Teltzrow and Kobsa [71] used 30 customer surveys on Internet privacy to analyse the impact that personalized systems have on the privacy preferences of customers. Amongst the many interesting results obtained from this compilation of surveys, it has been found that customers are more willing to share personal information with merchants when they are allowed to view, edit and delete their data. For instance, 69% of customers wanted to have control on their data [40] and around 90% were concerned about the sharing of their data by merchants for a purpose different from the original's [73,63]. Also, results concerning customer tracking [31,39] showed that around 60% of customers were concerned about being tracked on the Internet. More significantly, 91% of customers felt uncomfortable being tracked across multiple websites.

Furthermore, Flinn and Lumsden [30] described an online survey they have conducted to explore typical Internet user awareness and knowledge of technologies related to their privacy. This survey showed that the meaning given by different people to the notion of privacy can vary significantly from one person to the next.

The studies presented above highlight the importance of grouping customers according to their attitude towards privacy. These studies also show the great interest that customers demonstrate in retaining control over their personal data even when they are in the merchant's platform. Furthermore, customers have significant concerns about tracking. To take into account these privacy problems, we introduce below a privacy framework whose objectives are to identify the dimensions involved in customer data privacy and to design adequately our privacy-preserving recommender system.

3.2 Privacy Framework

Our task is to provide an architecture for recommender systems in which privacy issues can be addressed, taking into account the studies presented in Section 3.1. With this in mind, we consider the following components of the customer's data that are of interest from a privacy point of view.

– The **identity** refers to information that makes it possible to determine *physically* who the customer is (or at least to seriously circumscribe the possibilities). This includes data such as his name, address, taxpayer number or biometric data used for access control.
– The **demographic profile** refers to demographic characteristics of the customer, such as age, gender, weight, race, ethnic origin, language, level of education, etc.
– The **purchase history** is about what the customer has bought in the past. It concerns also the identity of the merchants from whom the customer has bought.
– The **rating history** refers to the votes that the customer has provided on products he has bought or on those for which he has received a recommendation in the past.
– The **browsing behaviour** describes the customer's navigation on the Internet. One of its principal components is the *clickstream*, which is the sequence of URLs that the customer has followed.
– The **request** is a description of what is of interest to the customer. Usually, it consists of keywords, which are at-

tributed to product features. An example of request is "news Italy", meaning that the customer is interested in news concerning Italy.

– The **recommendation** consists in the list of products that have been recommended to the customer.
– The **purchase** is the set of products that the customer finally decides to buy (if any) in the current transaction. These products are usually chosen among those that have been recommended.

The above elements constitute the personal information on which we base our privacy framework for recommender systems. They lead us to identify four levels of privacy, which take account of the fact that different customers may have different privacy concerns, as explained in Section 3.1.

1. ***No*** **privacy**: the customer does not care about the privacy of his personal information. He does not mind the compilation of a dossier about him that consists of his demographic information as well as his buying and browsing behaviour.
2. ***Soft*** **privacy**: the customer wants to keep his identity, demographic profile, purchase history and rating history secret from the merchant, but he does not mind if the merchant knows in which product(s) he is interested.
3. ***Hard*** **privacy**: the customer wants to keep his identity, demographic profile, purchase history and rating history secret from the merchant. Furthermore, he does not want the merchant to know in which product(s) he is interested nor anything about the recommendations he receives.
4. ***Full*** **privacy**: the customer wants to keep secret every component of his personal data. In particular, this includes his browsing behaviour as well as his actual purchase.

Recall from the end of Section 1 that our solution involves a novel entity called the Alambic agent. Sometimes, information that should be kept private from the merchant (except in the no-privacy case), such as purchase history and rating history, can be revealed without fear to the Alambic agent provided it cannot associate such information with specific products in the merchant's catalogue. We shall come back in detail on this subtle issue when we describe our solution in Section 5. The identity, of course, should remain secret from all parties considered, except perhaps in the no-privacy setting.

Another dimension in privacy, which is independent of the personal data listed above, concerns the *tracking* of customers by the entities involved. Even under soft, hard or full-privacy constraints, some users might not want the merchant to know that they are returning as the same (anonymous) person who shopped on a given previous occasion. Indeed, studies have shown that customers do not like being tracked when buying online, even if anonymously [71]. With this in mind, we introduce the following terminology to account for the level of tracking that different customers might accept.

1. ***Strong*** **tracking**. The customer does not mind being tracked, even by the merchant. Combined with soft or hard-privacy constraints, this level is typically achieved through the use of an anonymous pseudonym used by the customer each time he connects to the merchant's system. Tracking is made possible if the customer keeps the same pseudonym when he connects to the same merchant, although he can use different pseudonyms with different merchants.
2. ***Weak*** **tracking**. The customer does not want to be tracked by the merchant, but he does not mind being tracked by some third party such as the Alambic agent. This is particularly interesting if that third party does not know what the customer is buying or even what appears in his request.
3. ***No*** **tracking**. The customer does not accept to be tracked under any circumstances or by any parties. This constraint can be obeyed only if no entity, merchant or otherwise, is ever able to know whether a given user is in fact a previous user of the system. In particular, this means that the customer must provide anew whatever information is required to make recommendations each and every time he uses the system.

Within our privacy framework, the privacy requirements of the various types of users can and must be expressed as a choice on both of these two independent components: privacy of personal data *per se* and tracking of users. For example, the marginally concerned can be described as having no problem with websites operating under the no-privacy and strong-tracking constraints. At the other end of the spectrum, the privacy fundamentalists would probably only accept to interact with sites enforcing at least soft privacy. In circumstances where it makes sense, they might even require hard privacy. The theoretical possibility of full privacy might also appeal to them, but since this is largely incompatible with current Web technology and common practice, it is doubtful that they would demand it today. On the tracking side, the fundamentalists would most definitely require that the no-tracking constraint be observed by all sites with which they would interact.

As for the pragmatic majority, we postulate that compliance with the soft-privacy and weak-tracking constraints would be satisfactory for their privacy needs. We base this assumption on the fact that we expect most people in this group to abide by the *division of trust principle*, which we shall introduce and discuss in Section 5.1. The ALAMBIC system described in this paper meets these requirements, and thus meets what we believe are the wishes of the pragmatic majority. Nonetheless, we shall discuss in Sections 5 and 6 how the more stringent constraints of hard privacy and no tracking can be met with variations of the ALAMBIC system.

Example 3 Our young student might not want it known that he speaks Arabic (part of his demographic profile). However, that might be relevant for the recommender to suggest items such as Al-Jazeera (original Arabic version). At the same time, his past clickstream might have demonstrated interest in a specific medication... This, he definitely wants to keep private. Furthermore, under the hard-privacy scenario, he might not even want the merchant, or anyone, to know that today he is looking for information on recent events in Iraq. Lastly, while he had been using an anonymous Yahoo! account to identify himself to the merchant's recommender system, he has recently been worried that the merchant might be able to know from one time to the next that he is the same person. He thus prefers to re-enter his preferences anonymously every time he uses the system rather than have the merchant know that he is that same guy that used the system and was clicking till 3am five days ago. With this, he hopes to enforce the no-tracking constraint.

On the other hand, consider that the merchant's business model is to charge a certain amount for the number of links returned, e.g. for 1 euro, you can purchase 25 queries, which will return up to 3 links each. Additional links are available at additional cost. To maintain customer satisfaction, it is clear that the merchant must use a recommender system to tailor which 3 links to return to the user. Furthermore, he does not want to let the user browse all of them, because that would result in a loss of profit (operating a good search engine does cost money, after all). Finally, knowing which links to provide to which users (the essence) is the information on which the whole business model is based. Losing that information to another search-engine provider would mean losing a significant competitive advantage. □

4 Related Work and Preliminaries on Security

In this section, we review previous work related to the privacy-preservation paradigm. Although we assume that Public-Key Cryptosystems, Secret-Key Cryptosystems and Public Key Infrastructures are well known, we briefly review the corresponding notation. We also present the concepts of Secure Two-Party Computation (STPC) as well as Surrogate Computing, the latter being a set of techniques to allow one party to perform a computation securely inside another party's platform.

4.1 Privacy Preservation

Much research has been done to address the problem of privacy preservation in data mining. In particular, Verykios, Bertino, Fovino, Provenza, Saygin and Theodoridis [74] classify several approaches using five dimensions. *First*, they consider the fact that data could be centralized or distributed. The *second* dimension is about the modification that can be applied on data, such as perturbation, aggregation, swapping, sampling, etc. The *third* dimension is concerned with algorithms that are used in data mining (for instance decision tree, association rule, clustering, rough sets, Bayesian networks). The *fourth* dimension considers whether it is the raw data or the aggregate data that must be hidden. Finally, in the *fifth* dimension, it is important to highlight how the privacy-preservation techniques are used for the modification of data. Here, techniques are based on heuristics or cryptography.

There have also been several approaches developed for privacy-preserving *clustering*. For example, the approach of Meregu and Gosh [50] uses transformations to perturb the dataset before the clustering algorithm is applied. The approach of Jha, Kruger and McDaniel [42] consists in designing algorithms that use secure multi-party computation. In our paper, when it comes to demographic filtering, the clustering process is performed by the Alambic agent through the feedback it receives from customers. Detail is given in Section 5.

The problem of privacy-preserving recommender systems based on Collaborative Filtering (CF) has been addressed before. The purpose of such schemes is to allow customers to take part in the recommendation process without compromising their privacy. This can be achieved by using *trusted identity proxies* such as *anonymizers* [10,33]. According to this approach, the customer discloses his profile to the trusted proxy, which then participates in the collaborative filtering process without revealing the customer's identity. One problem with identity proxies is that the merchant cannot trust the feedback he gets from the customers because unscrupulous competitors could completely disturb the votes by a process known as *shilling*.

More precisely, the use of trusted proxies (or any other form of anonymity) means by definition that recommender systems must remain open to raters without any authentication procedure. Unfortunately, this could encourage malicious users to cheat the system with false ratings. Many motives can be behind this kind of behaviour, among them are *fun and profit* [47]. Indeed, some well-known e-commerce entities have already admitted that their recommender systems were subject to shilling. This practice brings out a dangerous aspect for the online store since customers may lose their trust in commercial websites. In particular, incidents caused by shilling may seriously affect the reputation of the online store and reduce the number of purchases.

An even more serious drawback in the use of trusted identity proxies is that they must indeed be trusted: the customer has to put his complete trust in that single entity. To get around this problem, a peer-to-peer privacy-preserving collaborative filtering scheme has been proposed by Canny [16], which does not depend on the need to trust a proxy. In this approach, a community of customers can compute a public *aggregate* of their private data without revealing them, through the use of "totalers". Each customer takes part in the construction of the aggregate and gets personalized recommendations by using local computation. The problem with such schemes is that they are *on-line*, meaning that several customers need to participate simultaneously in the computation and subsequent updates of the aggregate, a necessary step for issuing recommendations.

Also based on peer-to-peer networks, Miller, Konstan and Riedl [51] introduced the idea of *personal recommenders* for delivering recommendations on palmtop computers. Here, the authors argue that customer privacy can be protected by storing personal data locally (on palmtop computers or on personal computers at home), or by sharing them in encrypted form. For this last point, they combined ideas of Canny [15,16], Crammer, Gennaro and Schoenmakers [29] and Pedersen [56] to design a peer-to-peer privacy-preserving collaborative filtering algorithm. This algorithm is based on a secure source of random bits and the notion of a *secure blackboard*, which is a peer-to-peer network making use of a write-one, read-many blackboard.

As discussed by Polat and Du [60], such peer-to-peer solutions are suitable for community-driven collaborative filtering efforts but probably less "appropriate for systems that provide on-line collaborative filtering services, such as Amazon and Yahoo". For this reason, they proposed a scheme in which the customer disturbs his private data before sending them to the merchant, using *Randomized Perturbation* techniques reminiscent of those mentioned above for privacy-preserving clustering. In this way, the merchant cannot compile truthful private information about the customer. Although the data are disturbed, Polat and Du argue that their scheme should still allow for successful collaborative filtering.

In contrast to the approaches outlined above, we do not require several customers to be online simultaneously on a peer-to-peer network, nor do we introduce random perturbations into the data that might alter the precision of recommendations. Instead, our filtering process is achieved by the Alambic agent through the ratings it receives from customers. Note that we do not require the Alambic agent to be fully trusted because it cannot obtain any sensitive information by itself. Nevertheless, we have to assume that the Alambic agent will not collude with the merchant. Once again, detail is given in Section 5.

4.2 Secret- and Public-Key Cryptography

Consider the problem of two parties A and B wanting to secretly exchange information between each other. As his customary in cryptography, let us refer to these two parties as *Alice* and *Bob*.

The traditional method to address the above problem is the use of *secret-key* cryptosystems, in which Alice and Bob share a common secret key k. If Alice wants to send a message m to Bob, she computes the encrypted message $c = S_k(m)$ from m and k by using the agreed upon encryption function S. Once Bob has received the encrypted message c, he retrieves the original plaintext message m by applying the corresponding decryption function S^{-1}, i.e. $m = S_k^{-1}(c)$.

In *public-key cryptography*, however, different keys must be used for communications from Alice to Bob and from Bob to Alice. If Alice wants to send a message to Bob, she must first know Bob's *public* key b. From a plaintext message m she constructs the encrypted message $c = E_b(m)$ using the agreed upon encryption function E. To decrypt the message, Bob must use his *private* key b', known only to him, and the decryption function D, to retrieve m, i.e. $m = D_{b'}(c)$. For messages from Bob to Alice, Alice's public and private keys, a and a' respectively, must be used in a similar manner.

It is good practice, known as *Kerckhoffs' principle*, to assume that there are no secrets about the encryption and decryption functions S, S^{-1}, E and D, and that efficient algorithms to compute them are in the public domain. All security must reside in the secret and private keys only.

4.3 Secure Two-Party Computation

In its simplest form, Secure Two-Party Computation (STPC) is concerned with the problem of evaluating a function $f(x,y)$ for which the first party, Alice, provides secret input x and the second party, Bob, provides secret input y, such that the output becomes known to both parties while the inputs x and y remain secret from Bob and Alice, respectively, except for what can be logically inferred from one's private input and the joint output. More generally, there could be two *distinct* functions f and g so that Alice learns the value of $f(x,y)$ and Bob that of $g(x,y)$. This includes the case in which one of the two functions returns a standard fixed answer (such as 0), so that the corresponding party does not learn anything from the interaction, yet his/her participation is necessary for the other party to obtain her/his legitimate output.

The first general STPC protocol was given by Yao [76]. By assuming the intractability of factoring, Yao showed that every two-party interactive computational problem has a private protocol. Other solutions followed later (see for example [77,44]). Despite recent efforts to implement generic protocols for STCP [49], it remains most likely that efficient implementations will rely on simplifications based on the particular structure of the function to evaluate. In this paper, the merchant and the customer, assisted by the Alambic agent, are invited to execute a STPC if, in addition to having no information about the customer's profile, the merchant is not allowed to know the products of interest to the customer, i.e. in the hard-privacy case (Section 3).

4.4 Surrogate Computing

We use this generic term to refer to the possibility for Alice to securely execute a computational task within a platform under Bob's control. This situation arises, for example, when Alice wants to perform a certain task with Bob, but she will be unavailable to do so when the time comes. The solution involves a *surrogate* that will act on Alice's behalf, but will be called upon and be controlled by Bob. The surrogate can be viewed as a programme, whose code and internal state is not known to Bob, but whose execution and input/output is controlled by Bob. The conditions that Alice require of the surrogate in order to protect her data and the correctness of the outcome are as follows:

Inviolability of data. Whatever data the surrogate contains, it must not be possible for Bob to examine its contents, at any moment.

No reverse engineering. It must not be possible for Bob to obtain any more information about the logic or code of the surrogate, other than what can be inferred by observing the sequence of input and output values.

Tamper protection. It must not be possible for Bob to unrestrictedly change the internal state or the logic of the surrogate, other than by changing its input values.

In practice, there are essentially three ways in which such objectives can be met:

Trusted computing. A trusted computing module is nothing more than a hardware-based surrogate (such as a smart card connected to a PC via a USB card reader or a trusted microprocessor embedded within a PC's circuitry) that runs within a larger computing platform owned and operated by Bob. The Trusted Computing Group (TCG) is a consortium of several industrial partners aiming to define the standards for, and eventually develop, hardware solutions that safely implement surrogates, which are referred to as a *Trusted Platform Module*. Several competing technologies have been proposed that try to achieve the above objectives. For example, smart cards protect their memory (containing code and data) relatively well, but some models were found to be susceptible to timing and power reverse-engineering attacks. Improved smart cards and other newer technologies will likely make it possible to implement safe surrogates in the near future.

Code encryption and obfuscation. The technique of *Code encryption* is used to protect mobile code that is executed on remote and possibly untrusted computers. The security problems related to mobile code in general, and mobile software agents in particular, have been studied by Sander and Tschudin [66]. For example, a mobile agent must be able to protect the integrity and the execution of its code, compute with secrets in public, and preserve the privacy of the code and internal data. Sander and Tschudin also introduced the notion of *computing with encrypted functions*. Collberg, Thomborson and Low introduced the concept of *code obfuscation*, which is a process that transforms a program so that it becomes more difficult to understand and more resistant to reverse engineering. The resulting code is called *obfuscated code*. In practice, obfuscated code can be the result of one or more code transformations. Even though it differs significantly from the original code, the obfuscated code must produce the same result as the original despite a possible slow down.

If all of these techniques can be safely implemented, then they can in principle be combined to create a purely software-based surrogate meeting the above conditions. Unfortunately, even though several proposals have been put forth for code obfuscation, its possibility has been seriously challenged by Barak, Goldreich, Impagliazzo, Rudich, Sahai, Vadhan and Yang [8]. Nevertheless, these authors admit that their theoretical results "do not mean that there is no method of making programs 'unintelligible' in some meaningful and precise sense", and indeed one of the authors of that 2001 study subsequently offered hope [48].

Sealed computing. A simpler hardware-based solution involves Alice placing a "sealed computer" under Bob's care. This computer is sitting inside a safe or other relatively hard to open container. While Alice may not have a full guarantee that Bob cannot open the container and hence examine or change the contents (data or code) of the computer, she does have the assurance that Bob cannot do so without it being detected at a later date. There are several techniques that are commonly used in the physical security world that would make tampering by Bob self-evident. Thus, as long as the surrogate's code is secure and cannot be exploited from without the box (by providing malicious inputs), Alice, upon verifying the container, can be assured that the contents are still safe. In the presence of suitable procedures and a credible deterrent (i.e. punishment for Bob's tampering), this might prove a cost-effective and sufficiently secure solution.

In the rest of this paper, we consider that the Alambic agent is indeed a surrogate for a semi-trusted third party (a notion discussed in detail in Section 5.1), which has been suitably and safely implemented by using one of the above techniques or a combination thereof. This will prevent the merchant from being able to alter the behaviour of the Alambic agent in a way that would reveal sensitive information. To be as general as possible, we shall assume henceforth that the surrogate does not enjoy a large amount of secure memory, so that voluminous tables must be kept in an insecure memory that belongs to the merchant. This will force the Alambic agent to encrypt the information it needs to store in that memory, of course, but it must also be able to make sequences of memory accesses that will not reveal sensitive information. Should a large amount of secure memory be available to the surrogate, various simplifications would become possible, making our scheme not only simpler but also more efficient.

An entirely different perspective, not requiring surrogate computing, would consist in keeping the Alambic agent outside of the merchant platform. The benefits and disadvantages of this *three-tier* architecture are discussed in Section 6.4 but we do not elaborate on this option until then.

5 The ALAMBIC system

In this section, we present the ALAMBIC system, our solution for designing a privacy-preserving generic recommender system. We first introduce the main concepts and general principles behind this system, before describing its global architecture and components. A detailed description of the demographic, content-based and collaborative filtering processes follows. Then, we describe the advantages of the ALAMBIC system compared with other privacy-preserving filtering solutions. Finally, we discuss the performance overhead incurred to preserve privacy. The version of the ALAMBIC system presented here is greatly improved compared to our original architecture, which supported demographic filtering only [4].

5.1 Main Concepts and Components

ALAMBIC is a generic architecture and protocol for building generic recommender systems employing simultaneously demographic, collaborative and content-based filtering, while achieving the privacy objectives described in Section 3.2.

The system achieves these objectives by combining the well-known security primitives reviewed in Section 4 with the following principle:

"Trust no one, but you may trust two..."

The basis behind this *division of trust principle* is that while customers might trust merchants with information about what they *want*, they do not trust them with information about who or what they *are* or what they *did*. Conversely, they might trust *someone else* with information about who and what they are, but not with information about what it is that they are looking for or what they previously bought. In recommender systems, this amounts to saying that customers will trust the merchant with the query on the database of available products, albeit a generic one, but with nothing else. On the other hand, they will trust another party with their demographic information, but they will not want it to know anything about what they bought or are now interested in, i.e. in what kind of transactions they have with merchants. Hence we refer to this additional actor as the *semi-trusted third party*. We discuss in more detail in Section 6.5 the legitimacy of this principle.

The ALAMBIC system relies on this principle and the presence of such semi-trusted third party, called the *still maker*. The system is built in such way that only a collusion between the still maker and the merchant would expose the customer's private data. The still maker uses his platform STILLMAKER to generate the *Alambic agent*, which will be deployed as his surrogate within the merchant platform. This agent will then perform the tasks necessary for the recommendation process by executing on the merchant's platform, without further intervention of the still maker. Alternatively, the Alambic agent could execute on the STILLMAKER platform, a possibility discussed in Section 6.4.

We now describe in more detail how the various parties are involved, as well as the components of the ALAMBIC architecture. These are depicted in Figure 1.

The merchant platform: The merchant's computing platform contains and manages the catalogue of available products. In addition, it also keeps a database of encrypted customer profiles and the encrypted customer rating table. While this information is held by the merchant platform, it cannot be interpreted or otherwise used by the merchant without the collaboration of the Alambic agent. The Controller Unit manages communication between the customer and the merchant platform, including the Alambic agent.

The STILLMAKER: This platform is under the sole control of the still maker. It generates a separate Alambic agent for each merchant platform it supports. Each Alambic agent is issued a separate private/public key pair and a public key certificate that is signed by the still maker by using a Public Key Infrastructure (PKI) with digital signature services that both the merchant and the customer recognize and trust.

The customer: He wishes to receive product recommendations from the merchant. He receives the Alambic agent's public key certificate from the merchant and can verify its authenticity since he has independently obtained the still maker's public signature verification key through the PKI. He can then encrypt his private information (demographic data and product ratings) with the Alambic agent's public key in order to protect its contents from the merchant's prying eyes.

The Alambic agent: Acting as surrogate for the still maker, the Alambic agent serves as an intermediary between the merchant and the customer. It maintains the information necessary to compute recommendations from demographic data and feedback provided by the users through the merchant. In order to protect the integrity of the merchant's platform, the Alambic agent must be able to authenticate itself to the merchant as a creature issued by the still maker. This is done by standard security techniques that depend on the type of surrogate computing technology that is used (Section 4.4).

5.2 The Filtering Process

The specific architecture that we describe in this section concentrates on the case in which we want to achieve *hard privacy* but nevertheless allow *strong tracking* of the customer. In this case, while the merchant will not be able to identify the customer nor know anything about his demographic profile or purchasing/rating behaviour/history, he will be able to associate a given customer with his previous uses of the system. Required modifications of the ALAMBIC architecture to take account of different privacy and tracking requirements are discussed in Section 6.

This architecture simultaneously implements the three kinds of filtering reviewed in Section 2: demographic, collaborative and content-based. In order to do this, the Alambic agent separately keeps and maintains the data required to make recommendations for each filtering method.

In the case of demographic filtering, the information required is the list of user clusters, together with a mapping from these clusters to the aggregated preferences shown by customers in this cluster through previous purchasing behaviour or feedback. We refer to this data as the *demographic essence* or DE. This choice of words reflect the fact that the Alambic agent *distils* this essence from the raw data it obtains throughout its interactions with individual customers. In order to protect the merchant's investment and

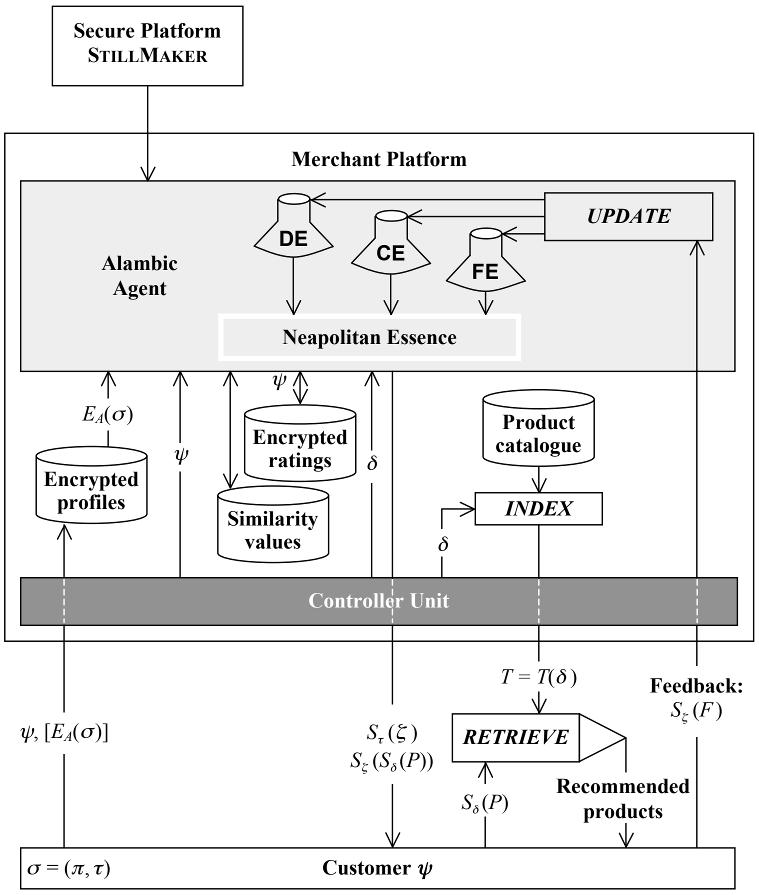

Fig. 1 Architecture of the ALAMBIC system in a strong-tracking scenario. The interactions between the various components are shown from left to right as time goes by, with square brackets to denote optional transmissions.

Table 1 The Alambic agent clustering table (demographic essence), where to each cluster C_j corresponds centroid coordinates and a list of anonymous product indexes purchased/selected by users in C_j.

Cluster	Centroid	Anonymous product indexes
C_1	(x_1, y_1, z_1)	p_1, p_2, p_7
C_2	(x_2, y_2, z_2)	p_5, p_8, p_{20}, p_t
…	…	…
C_n	(x_n, y_n, z_n)	p_{10}

prevent its unauthorized use by the Alambic agent, the list of products rated by the customers in the cluster is masked by using anonymous indexes p_i, which are linked to the actual product indexes s_i through a secret map Σ known only by the merchant. The structure of the DE in the ALAMBIC system is shown in Table 1.

In the case of memory-based collaborative filtering, which we consider here for the sake of simplicity, the Alambic agent maintains the table of product ratings. In that case, we cannot speak of distillation *per se* because there is no statistical aggregation. Furthermore, since this table could become quite big in some systems, we consider the possibility that it be stored in a database under the control of the merchant (depending on the type of surrogate computing being instantiated by the Alambic agent). In this case, it would be encrypted by the Alambic agent to prevent the merchant from being privy to this information. Furthermore, it would be accessed by the Alambic agent through an anonymized index, such as a customer pseudonym ψ. While the use of pseudonyms might make the system more convivial by allowing the storage of personalized preferences and encrypted profiles, it is not strictly required in the ALAMBIC system. In fact, their use allows the Alambic agent and merchants to track customers, which might be undesirable. We shall come back to this issue in Section 6.

Alternatively, model-based CF approaches can also be supported by the ALAMBIC system. For instance, instead of maintaining the complete table of product ratings for each user, the Alambic agent could keep a list of clusters of users with similar voting behaviour (similarly to the clusters of the DE) or a description of the Bayesian network model used to calculate the recommendations. In principle, any model-based approach could be supported since the Alambic agent has all the information required to update the model. The problem of how to initialize these models (the *cold start problem*) is a recognized difficulty of using such recommender systems. Several methods have been proposed to deal with this issue, each with their advantages. All of them can be easily implemented within the ALAMBIC framework, since the Alambic agent would have access to all the data required to generate the models. For hybrid CF algorithms, it would be necessary to default back to keeping the whole table of product ratings. In all cases, we employ the term *collaborative essence*, or CE, to refer to the information kept by the Alambic agent to perform CF, regardless of whether there is actual distillation or not.

To implement content-based filtering, the Alambic agent must keep a table of the product similarities $Sim(i,j)$ described in Equation 3 (Section 2.2). In order to protect the merchant's interests, these similarity values should be computed by the merchant platform and then made available to the Alambic agent. This prevents the Alambic agent from having to know the features of each product. It also allows the merchant to update this essence himself as his product catalogue evolves, and this without violating any privacy constraints since it contains no data pertaining to the customers.

Furthermore, the Alambic agent must have access to the current user's product ratings in order to compute product predictions as per Equation 4. However, note that it does not need access to the whole product rating table (the CE) for this purpose, but only to the current user's entry. Since this data could be obtained directly from the current user, the table of $Sim(i,j)$, for all pairs of products i and j, is all that is needed to constitute the *content-based essence* or *feature essence*, that we represent forthwith as FE.

While several different hybridization techniques for recommender systems exist [12], here we illustrate our architecture by using a particular case of *Mixed Hybridization*. It simply consists of collecting each of the recommendations made by each of the recommender systems and presenting them all to the customer. Of course, one can imagine a variety of ways for merging three lists into one, such as presenting them separately or mixing them one clever way or another. However, there would be no point being more specific about this issue because it is not concerned with privacy preservation. We also introduce the notion of a *combined* essence regrouping the essences DE, CE and FE of each type of recommender system. Since each of these essences contains different types of data ("flavour") and are not mixed, we refer to this combined essence as the *Neapolitan essence* and we speak in general of the *Neapolitan hybridization technique*. We have chosen this approach for the sake of clarity, but other techniques would be possible.

We are now ready to describe in detail the filtering, recommendation, feedback and update processes in the ALAMBIC system. This is shown through the interactions materialized by arrows in our architecture (Figure 1). These interactions are considered from left to right as time goes by. The execution proceeds through the following steps.

Initialization

1. The merchant provides the customer with the public key certificate of the Alambic agent. This allows the customer to learn the Alambic agent's public key A and verify that it is legitimate. (This step is not illustrated in Figure 1.)
2. Let ψ denote the customer's pseudonym. If the customer is new to this merchant, or if he wants to update his demographic profile π, he creates his *combined profile* $\sigma = (\pi, \tau)$, where τ is a secret key that will be used to establish a session key ζ between the customer and the Alambic agent, for the purpose of securing confidential communication through the merchant's Controller Unit. The customer uses the Alambic agent's public key A to encrypt σ as $E_A(\sigma)$. The customer sends his pseudonym, ψ, together with $E_A(\sigma)$, to the merchant. The merchant inserts ψ and $E_A(\sigma)$ in his encrypted profile database. On the other hand, a returning customer simply sends his pseudonym ψ to the merchant, who retrieves the previously transmitted encrypted profile $E_A(\sigma)$ from his database.
   [If, instead of concentrating on strong tracking, we had wanted to implement the more stringent requirement of *weak tracking*, the customer would not send his pseudonym ψ in the clear to the merchant. Instead, he would create the string $\sigma = (\pi, \tau, \psi)$ and send only $E_A(\sigma)$ to the merchant. In the extreme case of *no tracking*, the customer sends $E_A(\sigma)$ to the merchant, with $\sigma = (\pi, \tau)$. In either case, the customer pays the price for reduced tracking by having to send his complete encrypted profile at each new connection to the merchant's website. In order to prevent tracking by the merchant, it is necessary to use a probabilistic public-key encryption algorithm. The steps described below are specifically tailored to the strong-tracking scenario, but we come back to weak and no tracking in Section 6.]
3. The merchant forwards both ψ and $E_A(\sigma)$ to the Alambic agent.
4. The Alambic agent decrypts $E_A(\sigma)$ using its private key to obtain the customer's demographic profile π and the secret key τ.

Demographic Filtering

1. Using profile π, the Alambic agent computes the distance between the customer's demographic profile and the centroid of each cluster, to find the nearest cluster, C_j, for a certain $j \in \{1, \ldots, n\}$, where n is the number of clusters in the demographic essence (Table 1). The Alambic agent thus produces a list $DR = \{p_{d,1}, p_{d,2}, \ldots\}$ of anonymous product indexes that customers in C_j have chosen in the past, where the subscript "d" stands for "demographic". This constitutes the recommendation of the DF part of the recommender system.

Collaborative Filtering

1. Using pseudonym ψ, the Alambic agent can retrieve the ratings of the corresponding customer from the encrypted rating database, containing the ratings of all

users. To prevent the merchant from knowing what products a user has rated, each voting record is encrypted separately.
2. The Alambic agent then computes prediction votes for the products that the customer has never rated. Based on these predictions, the Alambic agent outputs a list $CR = \{p_{c,1}, p_{c,2}, \ldots\}$ of anonymous product indexes that correspond to the predictions with highest values. (See Section 2.1 for examples of functions that compute correlations and predictions.)

Content-Based Filtering

1. Again using ψ, the Alambic agent can obtain the list of products that the customer has rated from the encrypted rating database. It consults the product similarity database, in order to obtain the similarity of each product that the customer has rated in the past with all the products that the customer has never rated. It then computes prediction votes for the products that the customer has never rated. Based on these predictions, the Alambic agent outputs a list $FR = \{p_{f,1}, p_{f,2} \ldots\}$ of anonymous product indexes that correspond to the predictions with highest values. (See Section 2.2 for examples of functions that compute similarities and predictions.)

Hybridization and Recommendation. The Alambic agent doubly blinds the anonymous product indexes in order to protect the privacy of customers and the merchant's catalogue.

1. By mixing the three lists *DR*, *CR* and *FR* that have just been produced, the Alambic agent produces a combined list $P = \{p_1, p_2, \cdots, p_u\}$ of anonymous product indexes. (Again, the specific mixing procedure is immaterial to our discussion.)
2. Two random secret session keys will be needed now in order to blind product indexes. Key δ, shared between the Alambic agent and the merchant, serves to protect the merchant indexation procedure from the competition—in particular his secret map Σ. Key ζ, shared between the Alambic agent and the customer, serves to protect the list of anonymous product indexes that is sent to the customer; this protection is needed because we are dealing with hard privacy. (Key ζ will be used again later.) The choice of δ must be left to the merchant in order to protect him against a potentially dishonest Alambic agent, as discussed in Section 6.3. This key can be communicated directly from the merchant to the Alambic agent but it would need to be encrypted were we using a three-tier architecture (Section 6.4). Key ζ is chosen by the Alambic agent, but it must be transmitted securely to the customer through the merchant's Controller Unit. Therefore, the Alambic agent uses secret key τ it had received from the customer *via* the merchant (Steps 2 and 4 of the initialization) to encrypt ζ as $S_\tau(\zeta)$ before transmission.
3. The Alambic agent applies a double encryption, $S_\zeta(S_\delta(p_i))$, to each anonymous index $p_i, i = 1, \cdots, u$ in list P. It thus obtains the list
$$S_\zeta(S_\delta(P)) := \{S_\zeta(S_\delta(p_1)), \ldots, S_\zeta(S_\delta(p_u))\}.$$
It sends $S_\zeta(S_\delta(P))$ to the customer.
4. The merchant applies procedure *INDEX* to compute $S_\delta(p_k)$ for each anonymous index $p_k, k = 1, \ldots, t$, where t is the total number of anonymous indexes in his catalogue (Table 2). This process results in a table T containing these values as search keys, with the associated product indexes and descriptions. In other words, this table T contains the same information as the initial catalogue but is indexed through anonymous indexes masked by δ.
5. The customer decrypts $S_\tau(\zeta)$ and obtains $\zeta = S_\tau^{-1}(S_\tau(\zeta))$. He then computes the set $S_\delta(P) = S_\zeta^{-1}(S_\zeta(S_\delta(P))) = \{S_\delta(p_1), \ldots, S_\delta(p_u)\}$.
6. The customer and the merchant execute procedure *RETRIEVE* to allow the customer to obtain the descriptions for products in $S_\delta(P)$, but without revealing to the merchant the corresponding anonymous indexes nor the whole catalogue to the customer. This is an asymmetric case of STPC (Section 4.3) between the customer and the merchant from which the latter obtains no information.
[Note that in the case of soft privacy, it would be sufficient for the customer to send the list of masked indexes $S_\delta(P)$ to the merchant, who would then compare this list with his table T and obtain recommended products for the customer. In other words, in this case, the procedure *RETRIEVE* would simply correspond to selecting the products in T that have anynomized and encrypted indexes in $S_\delta(P)$.]

Customer Feedback

1. Having associated the recommended products with their attributes, the customer is now able to make decisions about them. Depending on the nature of each product or service recommended, he might decide to purchase or use it, hence providing *implicit feedback*. He might also wish to provide *explicit feedback* by voting or providing numerical ratings on these products.
2. In certain cases, it could be desirable for the customer to provide feedback (implicit or explicit) on items that were not recommended by the Alambic agent, i.e. not in P. These can include impromptu suggestions by the merchant, e.g. items on sale, Top-10, etc. Alternatively, these could be items queried directly, without the use of the recommender system, by using an adequate implementation of the *Blind Electronic Commerce paradigm* [5],

Table 2 Structure of the merchant's catalogue. The anonymous product indexes p_i alone are not sufficient to identify the product. They are related to the publicly known product indexes s_i (e.g. part numbers) by a map Σ known only to the merchant. Associated with these, is the product description and other information relevant to the user in order to make a decision (e.g. price, size, etc.).

Anonymous Index	Product Index	Description	...
p_1	s_1	...	...
p_2	s_2	...	...
...	...	...	...
p_t	s_t	...	...

which allows the customer to query the merchant's catalogue without revealing the query. In either case, clear descriptions of these products would have been provided to the customer. Furthermore, in order to allow him to provide feedback on these additional products, the associated anonymized and encrypted index $S_\delta(p_i)$ would also have been provided.

3. Let P' be the set of products on which the customer is giving implicit or explicit feedback. For each product in P', the customer knows the corresponding anonymized and encrypted index $S_\delta(p_i)$—but not the corresponding anonymous index p_i—and its description. Let v_i represent the customer rating value given to this product.
4. In order to update the essence, the Alambic agent needs to associate the rating v_i with the corresponding anonymous product index p_i. To do so, the customer will need to provide it with the set F of pairs $(S_\delta(p_i), v_i)$. To protect the ratings from the merchant, the customer encrypts them with the session key ζ and sends the set of encrypted pairs $S_\zeta(F)$ to the Alambic agent through the merchant's Controller Unit.

Update
The essence update is based on the set F of pairs that the Alambic agent receives as customer feedback. The necessary modifications might involve some changes for:

– *The demographic essence*: the list of products associated with the cluster C to whom the customer belongs is updated by the Alambic agent. This also changes C itself (changes in the centroid or medoid, density and radius) and might even trigger the merging, separation or creation of new clusters. For our purpose, all that matters is that all of these modifications can be embedded in the code of the Alambic agent and only require as input the feedback pairs in F.
– *The collaborative essence*: for each pair in F, the Alambic agent updates or inserts the rating value expressed by the customer (if any) in the encrypted rating database.
– *The feature essence*: it does not depend on customer feedback and it is therefore not updated by the Alambic agent. It only needs to be updated by the merchant when his catalogue is modified, either because of a product addition/removal or a feature change.

5.3 Advantages of the ALAMBIC Design

In order to better understand the ALAMBIC design and its advantages, let us consider some alternative solutions to the problem of protecting privacy in recommender systems.

One possibility is to give the customer access to the whole catalogue. However, as we have explained at the beginning of Section 3, the merchant might not want to expose his entire catalogue to unchecked scrutiny by customers or, worse, competitors masquerading as customers. In other words, this solution might not satisfy some of the merchant's privacy requirements.

Another solution would involve the use of STPC between the customer and the merchant, where the outcome would be a product meeting the customer's (secret) requirements. However, in both demographic and collaborative filtering recommender systems, it is also necessary to take into account the input (i.e. ratings) of other users of the system. But the information corresponding to those customers also needs to be kept private. Even if we were to employ the more general secure multi-party computation (SMPC) to achieve this goal, such a solution would have the major drawback of requiring all other users to be simultaneously present in the system in order to compute a new recommendation (or at least enough other users to make the recommendation representative). Furthermore, while several theoretical protocols for SMPC have been proposed [24,36,23,9], to this date no practical generic implementation has been constructed.

The ALAMBIC system solves this problem by introducing the Alambic agent, which replaces the other users in an SMPC by providing instead the essence distilled from their inputs. This aggregate essence is sufficient for computing the recommendations. Rather than implementing the recommendation and update processes with a full-fledge three-party SMPC involving Alambic, the customer and the merchant, we have sought to avoid having to implement such complicated protocols altogether. We have achieved this by introducing low-end and easier to implement cryptographic primitives (anonymized indexes, secret-key and public-key encryption). In the end, only a two-party STPC between the customer and the merchant is required in order for the former to obtain his recommendation. Unlike SMPC, there are already some reasonable generic STPC implementations such as Fairplay [49]. Furthermore, in the case of the *RETRIEVE* procedure, specific STPC solutions have been proposed that could be applied, such as Symmetrically Private Information Retrieval (SPIR) [25,46,34,18] or Blind Search (BliS) [5]. These specific solutions would likely have simpler and more efficient implementations than generic STPC, and outperform 3-party SMPC.

Another key element of the ALAMBIC system is the fact that the Alambic agent resides within the merchant platform. We deliberately chose this solution over the three-tier alternative whereby the Alambic agent would reside within the STILLMAKER platform. The advantages and disadvantages of this solution are discussed in Section 6.4.

5.4 Performance Overhead of the ALAMBIC System

Let us consider the performance overhead of the ALAMBIC system that is caused by its privacy-protection features. Let t be the size of the product catalogue, m the number of users that have used the system, having either purchased or rated products (and are hence "present" in the collaborative essence). Let us denote by n the total number of clusters generated by clustering the demographic information (obviously $n \leq m$).

The space requirements for the ALAMBIC system are:

- $O(nt)$ for the cluster information database (demographic essence),
- $O(mt)$ for the table of product ratings (collaborative essence) and
- $O(t^2)$ for the table of product similarity values (content-based essence).

The actual constants will depend on the clustering model and the granularity of rating values. Essentially, space consumption depends linearly in the number of customers and the number of products. This is the case regardless of the privacy requirements we choose to address, including none.

Let us now turn to the time complexity of the ALAMBIC system. We shall begin by analysing the cost of computing each type of recommendation.

Demographic filtering. Without any privacy-preserving features, this process would involve $O(n)$ operations for computing customer-to-centroid distances and finding the nearest cluster, and $O(t)$ to issue the recommendation, since there will be at most t items associated with any given cluster. Because the whole process happens within the Alambic agent, the only overhead of privacy protection is the decryption of the encrypted profile, which happens only once for every session. Since the demographic profile is likely to be reasonably small (a few kilobytes), this will happen within less than a second, regardless of the surrogate computing technology chosen.

Collaborative filtering. Computing product correlations (Equation 1 in Section 2.1) for each pair of users takes $O(t)$ operations. From these values, finding the products with highest recommendation values (Equation 2) can be done in $O(mt)$ operations. Since the product ratings for each customer are kept encrypted on the merchant platform, the overhead imposed by the ALAMBIC system on this process is the decryption of each of these ratings using a secret-key algorithm: this represents at most mt secret-key decryptions.

Content filtering. Computing recommendation predictions for all products and a given customer (Equation 4 in Section 2.2) will require in total $O(t^2)$ operations. In this case, there is no overhead due to privacy-protection features since the product similarity values (Equation 3) contain no private information about the customer and they are stored unencrypted on the merchant's platform.

The indexation process (*INDEX*) requires t encryptions with the secret key δ by the merchant. Protecting the recommendations requires u double-encryptions (with keys δ and ζ) by the Alambic agent and u decryptions by the customer (with key ζ), where $u \leq t$ is the number of recommendations made. The overhead for the *RETRIEVE* procedure, on the other hand, depends on the privacy requirement that we have chosen to implement. In the soft-privacy case, all that is required is for the merchant to perform u decryptions (with key δ) to obtain the product descriptions that have to be sent to the customer.

The cost of the STPC procedure required in the hard-privacy case is unfortunately much more significant than that of all the above secret-key encryptions. For most generic STPC protocols, the number of operations required from the participants is proportional to the number of operations of the abstract function being computed. In our case, it is a row lookup on a table of t entries, and hence it is expected that the total number of operations in the STPC would be $O(t)$, the number of table entries, or at best $O(\log t)$ if a sorted index is used. These bounds are somewhat deceiving as the constants involved are probably quite large. While the same bounds apply to specific protocols such as SPIR or BliS, the constants are likely to be much smaller, which is why they would be more efficient in practice.

Nevertheless, it is possible that, for small catalogues sizes, the ultimate bottleneck might not be the number of operations, but rather the number of rounds of interaction between the customer and the merchant. In fact, round complexity was identified early on as one of the major drawbacks of STPC. Surprising recent theoretical work of Katz and Ostrovsky [43] has shown that generic STPC protocols exist with as few as five rounds. Unfortunately, this was also shown to be optimal. If the customer and the merchant are separated by the Internet, this could lengthen the *RETRIEVE* procedure by several seconds due to transmission delays alone. It is also unclear what the trade-off between rounds and total processing time would be for these newer protocols.

Finally, let us consider the customer feedback and update process. Constructing the feedback vector $S_\zeta(F)$ requires at most t secret-key encryptions by the customer, and corresponding secret-key decryptions by the Alambic agent. The latter will also need to de-mask product indexes sent to the customer, by decrypting the rating pairs in F with the secret key δ, which will require at most t decryptions. To update the demographic essence might require in some cases a readjustment of all clusters, taking as as many as $O(nt)$ operations, but no extra privacy-protection overhead. Updating the collaborative essence simply involves changing the row entry for the current customer in the table of encrypted ratings, and requires at most t additional secret-key encryptions.

In summary, the total overhead of the ALAMBIC system, compared to a similar hybrid recommender system with no privacy-protection features, largely depends on the privacy constraint being addressed. In the hard-privacy case, this overhead is dominated by the time required to complete the *RETRIEVE* procedure. The additional delay might be measurable in seconds depending on Internet latency and the size of the catalogue. If soft privacy is sufficient, however, then the overhead is limited to the time required to perform $O(mt)$ secret-key decryptions (with very small constants). Given that these cryptographic operations can be implemented in the order of tenths of microseconds each, this overhead is comparable to the time needed to access the corresponding records on a standard database in a non-privacy preserving setting.

6 Privacy and Security of the ALAMBIC system

We now address the compliance of the ALAMBIC system with respect to the privacy framework of Section 3.2. First, we discuss how the customer's privacy and tracking requirements are met. We also examine the issue from the merchant's perspective. We then explore the advantages and disadvantages of a two-tier versus a three-tier architecture. Finally, we vindicate the validity of the main principle under which the ALAMBIC system meets these requirements: the division of trust principle.

6.1 Protecting Customer Privacy

According to the division of trust principle, it is fine for the demographic profile to be known by the semi-trusted third party, which we have called the still maker, or by his surrogate, the Alambic agent. (Please note that trusting the still maker and trusting its Alambic agents is one and the same thing in our context.) It is also acceptable for the Alambic agent to know the customer's purchasing and rating history provided that the actual products purchased and/or rated are anonymized by a mapping known only to the merchant. However, except in the trivial no-privacy scenario, none of the these customer data may be known by the merchant. Conversely, while the merchant might be allowed to see the customer's request in the case of soft-privacy, the Alambic agent is not. Under the more stringent hard-privacy constraint, the merchant is not allowed to know the request either. Finally, under the full-privacy constraint even the browsing behaviour of the user (including the click-stream) must be kept private from the merchant. In order to see how these constraints can be met, let us analyse item-by-item how the various components of the customer private data are protected by the ALAMBIC system.

Protecting the identity. In fact, there is no real need for the identity of the customer to be ever revealed to any of the parties (the merchant or the Alambic agent). Indeed, the identity of the customer can be maintained secret, even when a product is actually purchased. While the ALAMBIC system does not address directly this issue, it is important to note that several solutions have been proposed and exist for blind search [5] and anonymous payments [20,21,2].

Anonymous delivery is relatively easily achievable for electronic goods (e.g. music, software) that can be delivered over the Internet. Anonymous delivery by email can be achieved with the pioneering solution proposed by Chaum [19]. Alternatively, *onion routing* [37,14] allows anonymization of IP network traffic. Based on this concept, several anonymizing peer-to-peer network implementations are now freely available (such as Tor, Freenet, Entropy, I2P, etc.), which could be used for anonymous delivery of electronic goods through other applications (FTP, HTTP, peer-to-peer file exchange, etc.).

Even in the case of physical goods, privacy-preserving delivery systems have been proposed by Aïmeur, Brassard, Mani Onana [6] obviate the need for the customer to identify himself to the merchant and protect other potentially private information such as his physical address.

Protecting demographic profile, purchase and rating history from the merchant. There are three possible threats by which the merchant could gain access to this data in the absence of a collusion with the Alambic agent. First, a *crypt-analytic attack* in which the merchant could try to decrypt the encoded information sent by the customer (or stored on the merchant platform). This would require either breaking the public-key cryptosystems or falsifying a public key certificate.

Second, an *inference attack on the encrypted databases* in which the merchant could make strong inferences on the private data of the last customer by comparing consecutive snapshots of these databases. If these databases are external to the Alambic agent (this depends on the choice of surrogate platform), they must be indexed by the customer's pseudonym ψ in order to allow retrieval of the appropriate encrypted information. Analysis of the encrypted profiles $E_A(\sigma)$ will yield no further information, beyond the pseudonym, to the merchant unless he can break the public-key encryption. In the case of the encrypted ratings, however, we cannot simply store the ratings with respect to the anonymized indexes p_i because the mapping between these indexes and the actual products is known to the merchant. This is why they are indexed by the customer's pseudonym ψ and encrypted record-by-record.

Third, an *inference attack on the essence* whereby gaining access to consecutive "clear" snapshots of the Neapolitan essence would allow the merchant to make strong inferences on the demographic profile, as well as on the purchasing and/or rating behaviour, of the last customer. However, since the essence update phase concerns only the Alambic agent, which does not reveal the outcome of that computation, such an analysis is not possible. Thus, in order for the merchant to perform such analysis, he would have to gain access to the essence, something that is against the assumed inviolability property of the surrogate platform on which the Alambic agent is running.

Protecting purchase and rating history from the still maker. On this issue, security is partly obtained from the fact that the Alambic agent resides in the merchant's platform. This is in itself an advantage of this architecture with respect to the three-tier alternative. True to the division of trust principle, the merchant would prevent the Alambic agent to send such information to the still maker. However, this is not necessary to guarantee privacy of the customer's data from the still maker. In particular, this goal can also be achieved in a three-tier architecture in which the Alambic agent resides outside the merchant's platform, a possibility that we discuss in further detail in Section 6.4.

What really prevents the still maker from gaining meaningful information on what the user actually buys or prefers is the anonymizing map Σ, which associates anonymous indexes p_i to the actual product indexes s_i (which could, for example, be publicly known part numbers). Without knowledge of this map, neither the Alambic agent nor the still maker (even if he had managed to get a hand on the information accumulated within the Alambic agent) would be able to make any meaningful inference about what the customer has or is buying or on which he is emitting feedback. They could, however, obtain information about how often the customer has purchased or sent feedback (its "quantitative" behaviour), but they would not know on what actual products (the "qualitative" part of the behaviour). Under the weak-tracking constraint, this is acceptable. We discuss in Section 6.2 the issue of avoiding this problem under the no-tracking constraint.

Protecting the customer's request. We have discussed in Section 2 the notion of hard privacy: in addition to protecting his identity, demographic profile and past behaviour (purchases, ratings and browsing history), the user also wants to protect the nature of his request from the merchant, i.e. what he is actually looking for.

This might appear to be nonsense, one might argue, because at the end of the day doesn't the merchant need to know what the customer decides to purchase in order to complete the transaction (payment, delivery, etc.)? However, knowledge of the selected product does not imply knowledge of what the customer initially wanted, i.e. the original request. For example, suppose that we associate the customer's request with a subset of the products that will satisfy his desires. If the customer purchases a single one of these items, even revealing what he is buying will provide less information to the merchant than the request itself. In the ALAMBIC system, the *RETRIEVE* procedure is asymmetric as it allows the customer to retain full control on which of the recommendations made by the Alambic agent he will reveal to the merchant, e.g. only those he intends to purchase.

What about preventing the merchant from knowing what item is being purchased? While this is not one of the objectives of hard privacy as we have defined it, one can imagine scenarios in which this might be possible. The main difficulties are payment and delivery of the product. As we have seen, there are anonymous solutions for both these problems. Moreover, there are scenarios according to which it might not be necessary to have payment for the goods (e.g. flat fee or free services).

Furthermore, one must consider the situation in which the customer is providing feedback on products he has not purchased. This information nourishes the recommender systems and is the raw material from which essences are distilled in the ALAMBIC system. Since there is no payment nor delivery, there would be no need for the merchant to know what the nature of the feedback was, nor about what items it was given, or even what the request was that brought these particular items to be rated. Note that the ALAMBIC system as described allows for this kind of blind feedback while protecting the privacy of the customer. However, hard privacy comes at a heavy performance price, as we have discussed in Section 5.4.

Protecting browsing behaviour. The most significant component of the browsing behaviour, and at the same time probably the most privacy-sensitive, is the clickstream. In principle, and under the assumption that the customer's platform is well protected, it is not possible for the merchant to access the portion of the clickstream referring to other websites. On the other hand, it is slightly harder but definitively achievable to prevent the merchant from gaining information on the clickstream of the same customer on previous sessions at the merchant's website. The "history" data kept by browsers is not typically accessible to Web servers, except through server-accessible and site-specific information such as cookies. By adequately adjusting the privacy settings of modern Web browsers, this issue can be avoided altogether.

It is more insidious to protect oneself against the merchant's Web server keeping track of previous URL requests within the same session. In a typical Web browsing session, the client will keep the same IP address throughout, and hence one request can be linked with another made in that session. The use of anonymizing peer-to-peer networks can also address this problem. In addition to masking his own IP address, a customer could use such tools to make subsequent requests appear to come from different originating IP addresses. For high-volume websites, this could potentially prevent the merchant from associating different requests to the same customer; not so in low-volume websites, where simple timing and traffic analysis techniques can be used to "sessionize" URL requests that are close together. Another problem is the fact many servers will issue and send to the client some kind of session-persistent identification token (either as a local cookie or as URL-based session identifier). However, such mechanisms could easily be thwarted with client-side privacy-protection measures.

In any case, protecting the privacy of browsing behaviour information in itself is not the topic of this paper. For us, what is important is whether the clickstream or other characteristics of browsing behaviour are being used by the recommender system. While the types of recommender systems that the ALAMBIC system can support, described in Section 5.2, do not make use of the clickstream, it is conceivable (as in the three-part example in Sections 2 and 3) that a collaborative-filtering process could take place using not only purchasing and rating histories, but also browsing behaviour. The ALAMBIC system as described could not support such an approach for the simple reason that there does not seem to be, using current Web technology, a simple practicable way to prevent the merchant from knowing the contents of URL requests sent to his own Web server.

In summary, while we can offer no simple workable solution to the problem of supporting the full-privacy constraint, it can be stated that the ALAMBIC system does comply with the hard and soft-privacy constraints on customer data under the following assumptions:

1. The underlying public-key and secret-key cryptosystems are secure and the former is used within a properly implemented Public Key Infrastructure.
2. The Alambic agent's code and internal data cannot be read or tampered with by the merchant, at any phase of the protocol; i.e. all the security properties of surrogate computing technologies are met.
3. (*Hard privacy only*) The STPC protocols are secure, properly used and well implemented.
4. The still maker and the merchant can be trusted not to collude with each other (the division of trust principle is valid).

6.2 Preventing Customer Tracking

Another component of customer privacy discussed in Section 3.2 is the ability of the other parties to track him in his use of the system. Even if the above privacy requirements on the customer data are met, it may be possible for the merchant or the still maker to track the customer, which is what we have referred to as the strong and weak-tracking conditions, respectively.

The ALAMBIC system as we described it in Section 5 allows strong tracking by the merchant. Even though this might not meet the tracking privacy requirements of the privacy fundamentalists, this option has the significant advantage that it allows the use of pseudonyms by the customers. Although customers can and should use different pseudonyms with different merchants, it is easier (in particular for recommendation purposes) if they keep the same pseudonym each time they visit the same merchant. In particular, it provides the convenience of not having to resend their demographic profile or purchasing/rating histories to the recommender system every time. Even though this pseudonym does not identify the customer, it certainly allows the merchant to track him, i.e. to know that the same customer is returning.

Alternatively, this same convenience to the end user could be achieved by having appropriate client software handling this process in a fashion transparent to him. However, in addition to the fact that this raises client-side privacy-protection issues of its own, it would still require more bandwidth and potentially have worse performance. It is arguable, thus, that the more performance-oriented portion of the pragmatic majority might accept this compromise. Nonetheless, the main reason we chose to describe the ALAMBIC system under this constraint was for the sake of clarity. We now review in more detail the suitability of the other implementation alternatives hinted at in Section 5.2 to achieve the more stringent tracking privacy requirements.

Weak tracking. The approach mentioned in Section 5.2 to implement a weak-tracking solution involves the customer sending his encrypted profile to the Alambic agent every time he connects to the system. Since all communication between the customer and the Alambic agent goes through the merchant, it is necessary that what is sent to the Alambic agent be different every time the user connects to the system. This can be achieved safely provided the customer uses probabilistic public-key encryption, in which the same plaintext can have several different associated cryptograms, or simply by using enough random padding within the plaintext block prior to encryption by standard public-key algorithms. In this way, the encrypted version of the profile would be different each time he uses the system, thus preventing merchant tracking. It follows that the encrypted profile database could be abandoned at the cost of asking customers to keep their profiles locally.

Unfortunately, while this solution is satisfactory in the case of recommender systems using only demographic filtering [4], content-based filtering or both, it it is not sufficient to achieve weak tracking in recommender systems using collaborative filtering. The problem lies with the encrypted rating database (the collaborative essence). If privacy is a concern, the product ratings for users other than the current one should not be kept locally by the user. Depending on the surrogate platform that is used to host the Alambic agent, storage limitations might impose that they be kept within the merchant platform. Despite the fact that the ratings are encrypted, the Alambic agent would need to retrieve the ratings of the current user. In the ALAMBIC system, the pseudonym ψ is used as a rating table index, which thus allows tracking. A seemingly more private alternative would be to use a new masked pseudonym ϕ, for example, to index and access the table of product ratings. However, by reverse-engineering the row access pattern, the merchant could deduce which entry ϕ corresponds to the current user, and would therefore be able to track him.

Nonetheless, there are a number of ways of adequately addressing this problem and achieving the weak-tracking condition:

1. *Table embedding.* Embed the entire rating table within the Alambic agent. As discussed in Section 5.4, its size is linearly dependent on the number of previous customers and the size of the catalogue. This might be possible in many cases, for example for merchants having a few hundred items in their catalogue and only a few tens of thousands of customers. For very large catalogues or customer populations, however, this solution might be impractical for hardware and software-based surrogate platforms in the two-tier architecture. Even in this case, this would not represent a problem in a sealed platform or in the three-tier architecture that we discuss in Section 6.4.
2. *Masked table access.* Rather than using random row access to table, the Alambic agent could simply read all rows in the table, in sequence, and use the data from the appropriate row(s) only to compute the recommendation. In fact, it turns out that this can be done with no extra overhead. As can be seen from Equations 1

and 2 (Section 2.1), computing recommendations already involves examining all rows in the rating table.

3. *Model-based collaborative filtering.* Instead of using a memory-based CF algorithm, which keeps all product ratings ever made by all users, a model-based algorithm could be used, which instead would construct an aggregate model of how users behave. One might expect that the size of the model would be much smaller than for the memory-based approach, and that therefore it becomes practical to resort to table embedding in all cases.

In practice, we would expect that weak tracking would be achieved with a judicious combination of the above techniques.

No tracking. Finally, let us consider the possibility of implementing a no-tracking system in which not even the Alambic agent can track customers. First of all, this means that no pseudonyms can be used anymore. Second, no demographic profiles should be stored on the merchant's platform or within the agent. Again, this might be achieved with relatively little loss of convenience by storing the encrypted profile locally on the customer's machine. Computing the recommendations can still be done once the customer sends his profile and previous ratings.

In order to track customers, the Alambic agent would have to associate the current customer c with "another" former customer c' who was perhaps the same person. Note that the content-based portion of the recommendation does not pose a threat, since the feature essence contains no customer-related data. Thus, and with the above precautions put in place, the only indicators left to the Alambic agent to associate c with c' would be:

1. c and c' are in the same demographic cluster C. How likely it is that $c = c'$ depends on the sizes of the clusters in the demographic essence. Since in a useful classification we do not expect that there will be many clusters, the average number of customers per cluster should be high, and hence cluster collisions will not provide reliable customer tracking information.
2. c and c' have the same or very similar rating behaviour. In the case of memory-based collaborative filtering, the situation is worse. Given the large number of products and possible ratings, it is highly improbable that two different users would have had the same ratings on the same items.

There is no obvious simple solution to the second problem. As in the weak-tracking case, the use of a model-based algorithm might alleviate the problem. This is because inferences and information leakage would be significantly reduced, becoming similar to those obtained from demographic essence cluster collisions.

6.3 Protecting Merchant Data

An often overlooked requirement, the merchant's private data must also be protected. There are two things that the merchant must seek to protect:

1. The *Neapolitan essence* in which he has invested expense and effort, and on whose both secrecy and accuracy his competitive advantage might depend
2. In some contexts (such as information brokering), the *catalogue* itself, its contents, and possibly its sensitive attributes such as price, availability, etc.

In the solution proposed in Section 5.2, the merchant's sensitive catalogue is protected by the use of anonymous indexes p_i. They are meaningless to the Alambic agent because they do not identify the particular product or product type that a category of customers might prefer or might have chosen to buy. The map Σ between these indexes and the real product indexes s_i is only known to the merchant, and allows him to reveal the contents of recommendations in terms of product attributes understandable by the customer. Thus, without knowledge of this map Σ, the anonymous indexes p_i in themselves provide no useful information to the Alambic agent about the catalogue.

The essence kept by the Alambic agent is also constructed in terms of these same indexes p_i. Hence, the Alambic agent has only generic non-product specific knowledge about the market trends and other useful information it contains. For instance, knowledge of the cluster structure in the DE might allow it to infer that people in particular age groups are buying the same very popular product. However, knowing only its anonymous index p_i, it will have no indication of what this product is or what its characteristics are. While it might be the case that this information in itself might have some residual value to competitors, we argue that the essence can probably not be interpreted and used in a profitable fashion without knowledge of what products it refers to, i.e. without knowledge of the map Σ.

How could then a malicious party reconstruct, totally or partially, this map Σ? Every time the merchant reveals the product characteristics of a recommendation, i.e. when he "opens" the recommendation in the *RETRIEVE* procedure, he is leaking information to the customer about Σ. Even though, he might not be revealing the corresponding real index s_i, he might be revealing enough product characteristics to uniquely identify it. However, this is of limited consequence, since the customer does not know what is the corresponding anonymous index p_i—he only knows $S_\delta(p_i)$. In other words, in order to gain that partial information about Σ, he needs to know the key δ, which is not revealed to him. Thus, the map Σ is well protected from inference attacks that the competition could mount by repeatedly posing as users of the system, because δ is chosen at random and should be different for every customer session.

But what if the competition and the semi-trusted third party were in collusion? If would then suffice for the Alambic agent to leak the key δ. The fact that the Controller Unit monitors and controls all communications of the

Alambic agent with the outside makes it difficult, but not impossible, for the Alambic agent to voluntarily leak information about the essence to an outside party (e.g. one posing as a *bona fide* customer). Information about the key would have to be leaked using a pre-arranged scheme and using the established protocol and associated messages as a covert channel. While this is not impossible—the use of secret-key encryption does allow the Alambic agent to hide some amount of information—the efficiency of such an attack is questionable given the potential size of the catalogue, and the limited bandwidth that the encrypted messages within the covert protocol could provide. In particular, the impact of such an attack could easily be reduced by the merchant, by simply limiting the number of recommendations opened in the *RETRIEVE* procedures for each key δ used, i.e for each session. Note that unlike δ, the map Σ cannot easily be replaced at every session, because this would require a recalculation of the essence by the Alambic agent and the merchant. There is no simple way to do this, except maybe by STPC, without the essence being revealed to the merchant, something that we want to avoid in order to protect the customer data from an inference attack by the merchant, as discussed in Section 6.1.

Beyond the confidentiality of the essence, the merchant must protect its integrity. Because the Alambic agent resides within its platform, incremental backups of its state can provide some level of protection of the essence against fortuitous or non-malicious erroneous alteration of the essence. Data kept by the Alambic agent within the merchant platform can be backed up as often as required by the latter without intervention or collaboration by the former. Backing up data inside the Alambic agent's memory is a bit more tricky and depends on the surrogate computing technology being used. In the case of code obfuscation, it can be done by the merchant by backing up the state of the Alambic agent. In trusted or sealed computing solutions, it would be possible for the merchant to request backups of the essence from the Alambic agent. This can be done without loss of customer privacy because such backups would be provided in an encrypted form only.

Nonetheless, nothing in the ALAMBIC system protects the essence against malicious customers (real or impersonated by competitors) that would wilfully provide inaccurate feedback (shilling) in order to reduce the quality of the essence or otherwise gain an economic advantage on the merchant. Two such scenarios immediately come to mind. First, one in which malicious manufacturers would shill the Neapolitan essence by posing as customers in order to unfairly push the sales of their products. In a similar fashion, competitors of the merchant could mangle the essence so the quality of recommendations decreases and therefore increase customer dissatisfaction.

In summary, both the merchant's catalogue and the information in the essence are kept private from customers and competitors, provided the semi-trusted party does not collude with them. Even if this is not the case, this information is reasonably well protected, as long as the merchant is diligent and applies a few simple rules: a) carefully choosing private key δ, and b) limiting to a reasonable number the recommendations that are revealed to the customer. Finally, while the merchant can partially protect himself from fortuitous modifications by the Alambic agent, nothing in the ALAMBIC system prevents shilling of the essence by malevolent customers, or malicious alterations of the essence by the Alambic agent.

6.4 Two- vs. Three-Tier ALAMBIC Architecture

Even though, from a trust point of view, the Alambic agent is a surrogate for a semi-trusted third party (the still maker), placing it within the merchant's platform (as we have done throughout) makes sense from an engineering and business perspective. There are several advantages to this two-tier architecture when compared with one in which the Alambic agent would reside in a platform controlled by the still maker (possibly within the STILLMAKER platform itself):

1. Lower communication overhead and latency, resulting in an improved performance of the recommender system.
2. The merchant can retain physical control of the essence, which should be his property by right.

On the other hand, the use of a three-tier architecture would yield two important and immediate advantages:

1. There would be virtually no memory or storage limitations on the Alambic agent, which would allow it to maintain the profile and rating databases within itself, eliminating the need for encryption.
2. There would be no need to rely on surrogate computing technologies.

This last point might represent an advantage because these technologies (software or hardware) are less mature than other security-providing solutions. On the other hand, while the sealed computing solution is simple and elegant, it does depend on assumptions about physical security and strength, as well as on the effectiveness of procedures and deterrents, which might not have been subject to the test of time in these kinds of applications. Motivated by these advantages, it is worth analysing in more detail what further consequences a three-tier architecture would have in terms of merchant and customer data protection.

One of the important changes that a three-tier architecture brings is the possibility that the still maker might host on the same platform several Alambic agents for different merchants. A malicious still maker could then, without impediment, correlate message streams from different merchants in order to facilitate his tracking of customers (even if they use different pseudonyms) or to make stronger inferences about their interests (requests), purchases and preferences (ratings), thus potentially weakening customer privacy. From the point of view of these merchants, this it not so good either, since it might allow the still maker to construct an "essence of essences" indicating global market trends that he would be alone to possess and control.

Even if the StillMaker platform hosts a single Alambic agent containing the essence of a single merchant, there are further consequences to merchant and customer privacy. Recall that in the two-tier solution proposed in Section 5.2, the merchant's catalogue is protected by the anonymous indexes p_i, and that the mapping Σ between these indexes and the real product indexes s_i is only known to the merchant. More importantly, while the Neapolitan essence itself is in the hands of the Alambic agent, it has little intrinsic value to anyone unless it can be interpreted in terms of real products, i.e. unless the map Σ can be totally or partially reconstructed.

In the three-tier scenario, the fact that communication in and out of the Alambic agent is no longer controlled by the merchant might allow for the possibility of collusion between the Alambic agent and customers, real or impersonated by the still maker, for the purpose of inference analysis. At the end of the recommendation process, the customers do find out what is the real nature of the recommendations. If the Alambic agent colludes with the customer and reveals the secret key δ to him, then the customer can invert the S_δ mapping and obtain a partial view of the Σ map for those products that were recommended. It gets worse. If in the *Customer Feedback* phase of the protocol (as described Section 5.2), the merchant allows the customer to provide feedback on new, not-recommended items, then he is essentially providing the means to mount a *chosen-plaintext attack* on the map Σ to the Alambic/customer collusion. As was discussed in Section 6.3, this can be somewhat mitigated if the merchant limits the number of recommendations that are revealed within the same session, i.e. that use the same masking map S_δ.

Nonetheless, the fact that the Σ map is more exposed in a three-tier architecture also has consequences on the privacy of customer data. In a three-tier architecture a malicious still maker could more easily reconstruct the Σ map by the method described above. If he succeeds in doing so, he can ultimately reconstruct the rating behaviour of users (stored in the CE in memory-based CF or obtained during the *Customer Feedback* phase) in terms of real products and hence violate customer privacy.

On the other hand, as it was already pointed out in Section 6.2, the use of a three-tier architecture greatly simplifies the implementation of a weak-tracking system. Since the essence fully resides within the still maker's platform, access to the corresponding tables by the Alambic agent is not visible by the merchant, and hence we can achieve the weak-tracking privacy objective with no further modifications in a three-tier architecture.

In summary, since the confidentiality and the integrity of the essence is our prime consideration, a two-tier architecture offers a better level of protection. Furthermore, this scenario better protects the customer's data from the still maker. Nonetheless, it is important to point out that the Alambic paradigm and architecture can be adapted to a three-tier setting, if it is deemed more suitable.

6.5 Validity of the Division of Trust Principle

The assumption that the still maker and the merchant will not collude with each other is a non-technical one and no mathematical proof of its soundness can be given. Since the privacy protection features of our system, and hence its viability, depend on it, it is paramount to discuss whether it makes sense. We put forth the thesis that it is indeed a sound assumption, as long as the parties have clear vested interests, and that those motivate them to perform what is expected of them in an honest fashion.

This is true for the merchant, because it is in his best interest to provide as accurate and relevant a recommendation to his customers as he can. This will result in a higher customer satisfaction, better return business and eventually higher revenues. Thus, he is willing and motivated to implement a recommender system and furthermore to commit the resources necessary to comply with constraints imposed by law or by the customers, for example privacy protection. For this reason, the essence should remain his property, even though he cannot use it or interpret it by himself. Furthermore, the ability for merchants to maintain and accurately update an as-inclusive-as-possible essence will provide them with a significant competitive advantage.

On the other hand, we see the still maker as a merchant himself, whose business survival depends on his ability and diligence in protecting the privacy of the public. He provides this service to the merchant for a price, but is accountable to the public at large, and thus to any potential customer of the merchant's recommender system, to protect their privacy. This is not a far-fetched notion: it is already the case for those bound by professional secret and other privacy laws. This includes, for example, the figure of a notary public or equivalently the attorney-at-law (or solicitor), but also, to a certain extent, governments and banks. The other obvious example, in this era of Internet and e-commerce, are the commercial Certification Authorities (such as *Entrust*, *Verisign*, etc.) whose business largely depends on the public's confidence in their honesty and professionalism. While there might be an incentive for them to collude with the merchant, modern society has already "dealt" with this problem and provides enough mechanisms (legal framework, checks and balances, and other means of deterrence) to prevent these situations, which are deemed sufficient by most (law-abiding) members of the society. We believe that the use of the Alambic system (or a system similar to it) would make it possible and practical to have such parties intervene in the e-commerce process to protect the privacy of the public.

Last but not least, another fundamental premise is that the *customers themselves* also have a vested interest in the accuracy and relevance of recommendations made by the merchant. Without this condition, they will not be willing to put in the extra effort, albeit small, to provide accurate input into such a system. More importantly, they will not be willing to assume the small residual risk associated with providing their private information into the Alambic system. Finally, there might be particular application domains (e.g. recommendations of medical or pharmacological products) in which privacy is of such high importance, that users would be willing to share the additional cost of constructing and operating such a system with the merchant, for example by paying fees to the still maker.

7 Conclusions

Recommender systems are a special case of Web personalization tools, that use either user-to-user, user-to-item or item-to-item correlations to find items that might be suitable for the current user. In order to make good recommendations, these systems typically require access to potentially private user data, such as demographic profiles as well as purchase and product rating histories. This raises important user privacy concerns in the use of these systems. In this paper, we have introduced and discussed a framework that might be applicable to e-commerce or other Web applications. In particular we have addressed both user data privacy and user tracking.

The main contribution of this paper is the introduction of the ALAMBIC system, a theoretical solution that can help attain the various privacy objectives of the customers, while protecting the merchant's commercial interests. The ALAMBIC system relies on the division of trust principle and the correct use of well-known cryptographic primitives. Each of the following objectives can be achieved by selecting the appropriate parameters.

Soft privacy. All user data is kept private from the merchant, except for the customer's request, which is kept private from a semi-trusted party that we called the still maker and its surrogate the Alambic agent. The overhead for this case is negligible for reasonable user and product database sizes

Hard privacy. Even the customer's request is kept private from everyone. This is achievable through the use of higher-level cryptographic primitive such as Symmetrically Private Information Retrieval (SPIR) or Blind Search (BliS), or even the more general secure two-party computation (STPC). The drawback is that little is known at the moment about how to implement these theoretical constructs in a robust and efficient manner. In any case, it is likely that even for small databases, the overall time overhead might be intolerable to the end user.

Strong tracking. This is the default mode of the ALAMBIC system, in which the merchant does not know who are the users, yet it can tell when a former user returns. This option has many advantages from the usability of the system point of view, but it is arguably less private than weak tracking.

Weak Tracking. It is also possible to prevent the merchant from tracking customers, with relatively simple modifications to the default architecture, at the cost of a reasonable performance overhead.

No Tracking. Completely eliminating the possibility of tracking by the Alambic agent is not possible because of information leakage on the demographic clusters or rating behaviour, which it must know in order to compute recommendations. Nevertheless, it is possible to minimize this leakage, and hence its ability to track customers, by using model-based instead of memory-based collaborative-filtering algorithms.

Merchant data privacy. The ALAMBIC system protects well the confidentiality of both the catalogue and the market trend data (the essence). In particular, even a collusion of the still maker with corrupted users would not disclose significant amounts of information, as long as the merchant applies simple precautions, such as limiting the number of recommendations issued to the same customer in any given session.

The ALAMBIC system can be implemented in both a two-tier architecture, involving only customer and merchant platforms, or a three-tier architecture involving also a semi-trusted third-party platform. The former has the advantage of providing better protection of the merchant's interest (including the confidentiality of his data) and, in the end, better privacy protection of customer data with respect to the semi-trusted third party. However, it relies on relatively new and not yet field-tested hardware or software technologies to securely implement the concept of surrogate computing. Until the widespread acceptance of these technologies, three-tier architectures might provide an interesting intermediary compromise. Ultimately, we suspect that the choice between these two alternatives will depend on the business models and market forces that will eventually drive the introduction of such privacy-preserving features in recommender systems.

While other privacy-protection solutions have been described in the context of recommender systems, the ALAMBIC system shows several significant advantages over them:

1. Customers providing new ratings or updating them need not be simultaneously on-line in order to protect the privacy of their ratings, when updating the aggregate required for issuing future recommendations.
2. It can simultaneously support three different kinds of recommendation techniques: demographic, collaborative and content-based filtering. Furthermore, it can be used with any implementation of these techniques, whether it be additive (e.g. customer vote-based) or not.
3. Unlike random perturbation techniques, the recommendations will be as precise as if they had been issued directly by the merchant in a world without privacy concerns.
4. It does not require each customer to keep data about the others.
5. It does not require high-level and slow cryptographic primitives, except in the hard-privacy setting. Furthermore, even in that case, simple SPIR or BliS protocols suffice instead of the more complicated two-party

(STPC) or multi-party (SMPC) secure computation protocols.

We have made a strong case for the viability of ALAMBIC-based recommender systems in the e-commerce setting, by showing how it behoves all parties to collaborate. In particular, we have argued that the main beneficiary for providing accurate recommendation is the merchant. A key element in achieving this aim is the upkeep of an accurate aggregate market picture, which we have designated as the "essence". The inherent and potentially enormous value of this essence immediately suggests issues that future research should address:

Privacy-preserving essence disclosure. Beyond serving as a tool for providing recommendations, the essence has clear business value to the merchant as a market analysis and decision making tool. It would do great good to the viability and potential interest of such a system to develop privacy-preserving mechanisms by which the Alambic agent could disclose the essence without compromising the privacy of previous customers. More specifically, we can foresee two important advantages to being able to do so. First, from the privacy point of view, it would allow the merchant to change the product anonymizing map Σ from time to time, which would protect his interest and, most importantly, the privacy of customer data with respect to a potentially dishonest semi-trusted third party. Secondly, this process will most probably prove to be an essential tool in assuring quality control of the essence, whether it is to adjust model parameters in model-based CF or DF, or to detect and prevent shilling.

Essence Availability. How can the merchant be protected against a malicious Alambic agent and still maker that refuse to continue to collaborate and hold the essence hostage?

Essence Integrity. How can the merchant be sure that the essence has not been tampered with by a malicious Alambic agent? How can he check this, once the essence is disclosed? How can he protect the essence against shilling, by manufacturers, competitors or other malicious users. While solving this problem becomes even harder in the privacy-preserving world (no deterrence!), we are investigating existing cryptographic techniques that could be used to adequately address this issue.

Full Privacy. We have only briefly discussed the possibility of protecting the clickstream in recommender systems that use it. We have not offered reasonable solutions to this problem. This needs to be explored. Ultimately, the problem does not concern only recommender systems, but Web surfing at large. It is most likely that the solution will involve a new-generation of Web applications or even new WWW protocols.

Other filtering techniques. In this paper we have covered the use of demographic, collaborative and content-based filtering techniques for making recommendations. However, we have not addressed the use of recommender systems using knowledge or utility-based filtering techniques. We believe that the ALAMBIC architecture can relatively easily be modified to provide privacy protection in these cases too.

Other hybridization techniques. We have only shown how to implement one kind of recommender system *hybridization*, the Neapolitan technique, which is a particular case of mixed hybridization. Again, the Neapolitan essence maintained by the Alambic agent should be general enough to support most of the other types of hybridization techniques[2] described by Burke [12]. Verifying that these assumptions about the generality of the ALAMBIC system with respect to filtering and hybridization techniques are correct is also the subject of current research.

Collusion detection and deterrence. Finally, we have based our approach on the presumption of validity for the division of trust principle. However, there are no embedded indicators in the ALAMBIC system that would allow customers to detect that there had been a collusion between the still maker and the merchants, for the purpose of violating their privacy. Hence, there would not seem to be any inherent deterrent for them to do so. One could argue that this is in fact no different than the situations in which we find ourselves today in the non-electronic privacy-preserving world: what guarantees or indicators do we have that trusted parties such as doctors, insurance companies, priests, etc., will not collude and exchange amongst themselves the (partial) private information that we have entrusted to each of them? Nonetheless, this is a somewhat disappointing situation and further research should be directed to address this problem in our context.

Acknowledgements We are most grateful to five anonymous referees and Sarabjot Singh Anand for suggestions that allowed us to greatly improve this paper. This gratitude covers also the referees for our earlier conference paper on privacy-preserving demographic filtering [4].

References

1. Ackerman, M.S., Cranor, L.F., Reagle, J.: Privacy in e-commerce: Examining user scenarios and privacy preferences. In: Proceedings of 1st ACM Conference on Electronic Commerce (EC'99), pp. 1–8. New York, NY (1999)
2. Aiello, B., Ishai, Y., Reingold, O.: Priced oblivious transfer: How to sell digital goods. In: Advances in Cryptology: Proceedings of Eurocrypt 2001, pp. 119–135 (2001)
3. Aïmeur, E., Brassard, G., Dufort, H., Gambs, S.: CLARISSE: A machine learning tool to initialize student models. In: Proceedings of 6th International Conference on Intelligent Tutoring Systems (ITS '02), pp. 718–728. Biarritz, France (2002)
4. Aïmeur, E., Brassard, G., Fernandez, J.M., Mani Onana, F.S.: Privacy-preserving demographic filtering. In: Proceedings of 21st ACM Symposium on Applied Computing (SAC), pp. 872–878. Dijon, France (2006)

[2] That is after all one of the advantages of Neapolitan ice-cream: those who want to eat each flavour separately can slice it vertically, those who want to mix them can slice it horizontally, or diagonally or put it in the blender...

5. Aïmeur, E., Brassard, G., Mani Onana, F.S.: Blind electronic commerce. Journal of Computer Security (2006). To appear
6. Aïmeur, E., Brassard, G., Mani Onana, F.S.: Secure anonymous physical delivery. IADIS International Journal on WWW/Internet **4**(1), 55–69 (2006)
7. Ardissono, L., Brna, P., Mitrovic, A. (eds.): Proceedings of 10th International Conference on User Modeling 2005. Edinburgh, Scotland (2005)
8. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. In: Proceedings of Crypto 2001, pp. 1–18. Santa Barbara, CA (2001)
9. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: Proceedings of 20th Annual ACM Symposium on the Theory of Computing (STOC), pp. 11–19 (1988)
10. Boyan, J.: The Anonymizer: Protecting user privacy on the Web. Computer-Mediated Communication Magazine **4**(9) (1997)
11. Breese, J., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of 14th Conference on Uncertainty in Artificial Intelligence (UAI-98), pp. 43–52. Morgan Kaufman, Madison, WI (1998)
12. Burke, R.: Hybrid recommender systems: Survey and experiments. Customer Modeling and Customer-Adapted Interaction **4**(12), 331–370 (2002)
13. Burke, R., Mobasher, B., Bhaumik, R.: Limited knowledge shilling attacks in collaborative filtering systems. In: Proceedings of 3rd International Workshop on Intelligent Techniques for Web Personalization (ITWP 2005), 19th International Joint Conference on Artificial Intelligence (IJCAI 2005), pp. 17–24. Edinburgh, Scotland (2005)
14. Camenisch, J., Lysyanskaya, A.: A formal treatment of onion routing. In: Proceedings of Crypto 2005, pp. 169–187. Santa Barbara, CA (2005)
15. Canny, J.: Collaborative filtering with privacy. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 45–57. Oakland, CA (2002)
16. Canny, J.: Collaborative filtering with privacy via factor analysis. In: Proceedings of 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 238–245. Tampere, Finland (2002)
17. Chang, S.: Strategic Management of e-Business, 2nd ed. chichester edn. John Wiley & Sons (2005)
18. Chang, Y.C.: Single database private information retrieval with logarithmic communication. `eprint.iacr.org/2004/036/`, accessed 1 November 2005 (2004)
19. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM **24**(2), 84–90 (1981)
20. Chaum, D.: Blind signatures for untraceable payments. In: Proceedings of Crypto 82, pp. 199–203. Plemum Press, Santa Barbara, CA (1982)
21. Chaum, D.: Blind signatures system. In: Proceedings of Crypto 83, p. 153. Plemum Press, Santa Barbara, CA (1983)
22. Chaum, D.: Security without identification: Transaction systems to make big brother obsolete. Communications of the ACM **28**(10), 1030–1044 (1985)
23. Chaum, D., Crépeau, C., Damgård, I.: Multiparty unconditionally secure protocols. In: Proceedings of 20th Annual ACM Symposium on the Theory of Computing (STOC), pp. 1–10 (1988)
24. Chaum, D., Damgård, I., van de Graaf, J.: Multiparty computations ensuring privacy of each party's input and correctness of the result. In: Proceedings of Crypto 85, pp. 477–488. Santa Barbara, CA (1985)
25. Chor, B., Goldreich, O., Kushilevitz, E., Sudan, M.: Private information retrieval. In: Proceedings of 36th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 41–51 (1995)
26. Cooley, T.: A Treatise on the Constitutional Limitations Which Rest Upon the Legislative Power of States of the American Union, 2nd edition. Callaghan & Co., Chicago (1888)
27. Cover, T.M.: Rates of convergence for nearest neighbor procedures. In: Proceedings of Hawaii International Conference on System Science, pp. 413–415 (1968)
28. Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. IEEE Transaction on Information Theory **IT-13**, 21–27 (1967)
29. Cramer, R., Gennaro, R., Schoenmakers, B.: A secure and optimally efficient multi-authority election scheme. In: Proceedings of Eurocrypt'97, pp. 103–108 (1997)
30. Flinn, S., Lumsden, J.: User perceptions of privacy and security on the Web. In: Proceedings of 3rd Annual Conference on Privacy, Security and Trust, pp. 15–26. St. Andrews, New Brunswick (2005)
31. Fox, S., Rainie, L.: Trust and privacy online: Why Americans want to rewrite the rules. Pew Internet & American Life Project, Washington DC. `www.pewinternet.org/reports/toc.asp?Report=19`, accessed 29 April 2006 (2001)
32. Freyne, J., Smyth, S.: Communities, collaboration and cooperation in personalized web search. In: Proceedings of 3rd International Workshop on Intelligent Techniques for Web Personalization (ITWP 2005), 19th International Joint Conference on Artificial Intelligence (IJCAI 2005), pp. 73–80. Edinburgh, Scotland (2005)
33. Gabber, E., Gibbons, P.B., Kristol, D.M., Matias, Y., Mayer, A.J.: Consistent, yet anonymous, Web access with LPWA. Communications of the ACM **42**(2), 42–47 (1999)
34. Gertner, Y., Ishai, Y., Kushilevitz, E., Malkin, T.: Protecting data privacy in private information retrieval schemes. In: Proceedings of 30th ACM Symposium on the Theory of Computing (STOC), pp. 151–160 (1998)
35. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using collaborative filtering to weave an information tapestry. Communications of the ACM **35**(12), 61–70 (1992)
36. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game, or A completeness theorem for protocols with honest majority. In: Proceedings of 19th Annual ACM Symposium on the Theory of Computing (STOC), pp. 218–229 (1987)
37. Goldschlag, D.M., Reed, M.G., Syverson, P.F.: Onion routing for anonymous and private internet connections. Communications of the ACM **42**(2), 84–88 (1999)
38. Greenspan, R.: Surfers Prefer Personalization. ClickzStats (2004)
39. Harris Interactive: A survey of consumer privacy attitudes and behaviors (2000)
40. Harris Interactive: Most people are privacy pragmatists (2003)
41. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: A review. ACM Computing Surveys **31**(3), 264–323 (1999)
42. Jha, S., Kruger, L., McDaniel, P.: Privacy preserving clustering. In: Proceedings of 10th European Symposium on Research in Computer Security (ESORICS '05). Milan, Italy (2005)
43. Katz, J., Ostrovsky, R.: Round-optimal secure two-party computation. In: Proceedings of Crypto 83, pp. 335–354. Santa Barbara, CA (2004)
44. Kilian, J.: Founding cryptography on oblivious transfer. In: Proceedings of 20th Annual Symposium on Theory of Computing (STOC), pp. 20–31 (1988)
45. Kobsa, A., Koenemann, J., Pohl, W.: Personalized hypermedia presentation techniques for improving online customer relationships. The Knowledge Engineering Review **16**(2), 111–155 (2001)
46. Kushilevitz, E., Ostrovsky, R.: Replication is not needed: Single database, computationally-private information retrieval. In: Proceedings of 38th Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 364–373 (1997)
47. Lam, S.K., Riedl, J.: Shilling recommender systems for fun and profit. In: Proceedings of 13th International Conference on World Wide Web (WWW '04), pp. 393–402. New York, NY (2004)
48. Lynn, B., Prabhakaran, M., Sahai, A.: Positive results and techniques for obfuscation. In: Proceedings of Eurocrypt 2004, pp. 20–39 (2004)
49. Malkhi, D., Nisan, N., Pinkas, B., Sella, Y.: Fairplay – A secure two-party computation system. In: Proceedings of Usenix Security Symposium, pp. 9–13 (2004)

50. Meregu, S., Ghosh, J.: Privacy-preserving distributed clustering using generative models. In: Proceedings of 3rd IEEE International Conference on Data Mining (ICDM'03), pp. 211–218. Melbourne, Florida (2003)
51. Miller, B.N., Konstan, J.A., Riedl, J.: Pocketlens: Toward a personal recommender system. ACM Transactions on Information Systems **22**(3), 437–476 (2004)
52. Mobasher, B., Anand, S.S. (eds.): Proceedings of 3rd International Workshop on Intelligent Techniques for Web Personalization (ITWP 2005), 19th International Joint Conference on Artificial Intelligence (IJCAI 2005). Edinburgh, Scotland (2005)
53. Mobasher, B., Cooley, R., Srivastava, J.: Automatic personalization based on Web usage mining. Communications of the ACM **43**(8), 142–151 (2000)
54. Pazzani, M.: A framework for collaborative, content-based and demographic filtering. Artificial Intelligence Review **13**(5-6), 393–408 (1999)
55. Pazzani, M., Billsus, D.: Learning and revising user profiles: The identification of interesting Web sites. Machine Learning **27**(5-6), 313–331 (1997)
56. Pedersen, T.: A threshold cryptosystem without a trusted party. In: Proceedings of Eurocrypt'91, pp. 522–526 (1991)
57. Pennock, D., Horvitz, E., Lawrence, S., Giles, C.L.: Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In: Proceedings of 16th Conference on Uncertainty in Artificial Intelligence (UAI 2000), pp. 473–480. Stanford, CA (2000)
58. Perkowitz, M., Etzioni, O.: Adaptive Web sites: Automatically synthesizing Web pages. In: Proceedings of 15th National Conference on Artificial Intelligence and 10th Innovative Applications of Artificial Intelligence Conference AAAI/IAAI, pp. 727–732 (1998)
59. Pierrakos, D., Paliouras, G., Papatheodorou, C., Spyropoulos, C.D.: Web usage mining as a tool for personalization: A survey. User Modeling User-Adapted Interaction **13**(4), 311–372 (2003)
60. Polat, H., Du, W.: Privacy-preserving collaborative filtering. International Journal of Electronic Commerce **9**(4), 9–35 (2005)
61. Pretschner, A., Gauch, S.: Personalization on the Web. Tech. Rep. FY2000-TR-13591-01, ITTC, University of Kansas (1999)
62. Resnick, P., Iacovou, N., Sushak, M., Bergstrom, P., Riedl, J.: Grouplens: An open architecture for collaborative filtering of netnews. In: Proceedings of Computer Supported Collaborative Work Conference (CSCW), pp. 175–186. Chapel Hill, North Carolina (1994)
63. Roy Morgan Research: Privacy and the community. Prepared for the Office of the Federal Privacy Commissioner, Sydney. `www.privacy.gov.au/publications/rcommunity.html`, accessed 29 April 2006 (2001)
64. Rucker, J., Polanco, M.: SiteSeer: Personalized navigation for the Web. Communications of the ACM **40**(3), 73–75 (1997)
65. Salinger, J.: The Catcher in the Rye. Little, Brown and Company (1951)
66. Sander, T., Tschudin, C.F.: Towards mobile cryptography. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 162–167. Oakland, CA (1998)
67. Schafer, J.B., Konstan, J.A., Riedl, J.: Recommender systems in e-commerce. In: Proceedings of 1st ACM Conference on Electronic Commerce (EC'99), pp. 158–166. Denver, CO (1999)
68. Schafer, J.B., Konstan, J.A., Riedl, J.: E-commerce recommendation applications. Data Mining and Knowledge Discovery **5**(1/2), 115–152 (2001)
69. Spiekermann, S., Großklags, J., Berendt, B.: E-privacy in 2nd generation E-commerce: Privacy preferences versus actual behavior. In: Proceedings of 3rd ACM Conference on Electronic Commerce (EC'01), pp. 38–47 (2001)
70. Suryavanshi, B., Shiri, N., Mudur, S.: A fuzzy hybrid collaborative filtering technique for web personalization. In: Proceedings of 3rd International Workshop on Intelligent Techniques for Web Personalization (ITWP 2005), 19th International Joint Conference on Artificial Intelligence (IJCAI 2005), pp. 1–8. Edinburgh, Scotland (2005)
71. Teltzrow, M., Kobsa, A.: Impacts of user privacy preferences on personalized systems—A comparative study. In: C.M. Karat, J. Blom, J. Karat (eds.) Designing Personalized User Experiences for eCommerce, pp. 315–332. Dordrecht, Netherlands, Kluwer Academic Publishers (2004)
72. Turban, E., King, D., Viehland, D., Lee, J.: Electronic Commerce: A Managerial Perspective. Prentice Hall (2006)
73. UMR: Privacy concerns loom large. Conducted for the Privacy Commissioner of New Zealand. Survey summary, Auckland: PC of New Zealand. `www.privacy.org.nz/privword/42pr.html`, accessed 29 April 2006 (2001)
74. Verykios, V.S., Bertino, E., Fovino, I.N., Provenza, L.P., Saygin, Y., Theodoridis, Y.: State-of-the-art in privacy preserving data mining. ACM SIGMOD Record **33**(1), 50–57 (2004)
75. Westin, A.: Privacy and Freedom. Atheneum, New York (1967)
76. Yao, A.C.C.: Protocols for secure computation. In: Proceedings of 23rd IEEE Symposium on Foundations of Computer Science (FOCS), pp. 160–164 (1982)
77. Yao, A.C.C.: How to generate and exchange secrets. In: Proceedings of 27th IEEE Symposium Foundations of Computer Science (FOCS), pp. 162–167 (1986)
78. Zhu, K.: Information transparency of business-to-business electronic markets: A game-theoretic analysis. Management Science **50**(5), 670–685 (2004)