Generating B2C Recommendations Using a Fully Decentralized Architecture

Domenico Rosaci and Giuseppe M.L. Sarné

DIMET, Università Mediterranea di Reggio Calabria
Via Graziella, Località Feo di Vito
89122 Reggio Calabria, Italy
{domenico.rosaci,sarne}@unirc.it

Abstract. In the last years, Business-to-Consumer (B2C) E-Commerce is playing a key role in the Web. In this scenario, recommender systems appear as a promising solution for both merchants and customers. However, in this context, the low scalability of the performances and the dependence on a centralized platform are two key problems to face. In this paper, we present a novel recommender system based on a multi-agent architecture, called Trader REcommender Systems (TRES). In TRES, the agents exploit their user's profiles in their interaction, to make the merchants capable to generate effective and efficient recommendations. The architecture we have adopted is fully decentralized, giving to each merchant the capability to generate recommendations without requiring the help of any centralized computational unit. This characteristic, on the one hand, makes the system scalable with respect to the size of the users' community. On the other hand, the privacy of each customer is preserved, since the merchant retrieves information about each customer simply monitoring the customer behaviour in visiting his site.To show the advantages introduced by the proposed approach some experimental results carried out by exploiting a prototype implemented in the JADE framework are presented.

1 Introduction

In these years, E-Commerce (EC) activities are playing a key role in the Web as attested by the increasing number of commercial transactions therein performed. As a consequence, a significant number of powerful and sophisticate tools have been recently developed for supporting traders in all their commercial processes with a high automation level. In particular, a great attention has been reserved to the Business-to-Consumer (B2C) activities, comparable with the retail trade of traditional commerce, both from the customers that can exploit the opportunity offered by the EC for a simple and comfortable access to an open-world market without time and space boundaries, and from the merchants that can offer their products to a wide audience by using a convenient media [1, 2].

1.1 Motivations: Recommender Systems can Effectively Support EC

In the aforementioned scenario, recommender systems [3–7] are among the most meaningful applications both for merchants and for customers. In particular, the merchants can take advantage of the improvement of the performances for their e-Commerce sites; on the other hand, the customers are supported in their decision-making process by means of some useful suggestions about objects, products, or services potentially interesting for them. Different techniques have been proposed in the literature to implement recommender systems in order to generate effective suggestions. Based on the adopted technique, recommender systems are generally partitioned in three main categories [5], namely: (*i*) *Content-based*, that suggest to a user items which appear the most similar to those he has already accessed in the past; (*ii*) *Collaborative Filtering*, that suggest to a user items which have been also considered by similar users; (*iii*) *Hybrid*, that combine both content-based and collaborative filtering techniques to generate recommendations. This latter approach is usually recognized as the most promising solution [5].

To carry out such an activity, recommender systems need to exploit a suitable representation (*profile*) of the customer's interests and preferences. This representation is automatically derived by monitoring and interpreting the large amount of information that each user spreads during his/her Web trading activities. These data can be implicit data as purchase histories, Web logs, cookies, etc. and/or explicit data that a user can provide through his/her beaviour. In particular, in the automatic construction of such a profile the most part of the existing recommender systems mainly consider only the number of accesses to a specific product or to a product category. In the EC field, and specifically in B2C activities, this approach risks to be misleading because a trading activity consists of more different phases and usually the interest degree corresponding to two accesses to a product in two distinct trading phases (e.g., a visit and a purchase) is different.

1.2 Our Contributions: A Behavioural Model to Represent B2C Processes and an Agent-Based Recommender System for EC

In order to realize a faithful representation of the real customer's interests and preferences and consequently to generate more effective suggestions, in this paper we propose a recommender system, called *Trader REcommender System* (TRES). This recommender system exploits users' profiles that, preserving customer's privacy (see Section 3), take into account their interests and preferences in the different phases involved by a Web B2C process. In such a manner, it will be possible to suitably consider in the interest computation each access to a product with respect to its real trading context.

Then, the first contribution of our paper consists of a model to represent business-to-customer activities in the Web context. In the past, other behavioural models have been introduced to capture the different phases carried out by enacting an EC process. Several of such models are extensions of other ones designed for the conventional commerce, such as the Nicosia Model [8] or the Engel and Blackwell Model [9], or explicitly thought for EC, such as the Nissen's Commerce Model [10] or the E-commerce Value Chain Model [11]. Another widely known approach is the Consumer Buying Behaviour (CBB) model [12], structured in six different phases, namely: *(i)* "Need Identification" - a user identifies his/her needs to satisfy; *(ii)* "Product Brokering" - a potential customer searches for products able to satisfy his/her needs; *(iii)* "Merchant Brokering" - when it is known what to purchase a consumer searches a merchant from whom to buy the chosen goods or services; *(iv)* "Negotiation" - transaction terms (i.e. price, quantity, quality of service, etc.) are fixed in this phase; *(v)* "Purchase and Delivery" - a customer fulfils the purchase choosing a payment option and a delivery modality, among those available; *(vi)* "Service and Evaluation" - a customer can estimate his/her satisfaction degree about a purchase. For the CBB model many and different extensions have been proposed, as for example the $E - CBB$ [13] and the $E^2 - CBB$ [14] to consider emerging behaviours as the formation of coalitions and the EC-site visits, respectively.

However, we remark that some modifications can be made to these models in order to more effectively represent EC activities carried out over the Web. In particular, with respect to the original CBB model, we observe that several trading processes allow the first three CBB stages to be unified without any substantial difference. In fact, it is not possible for all the B2C processes to effectively split the customer's behaviour in these three phases (e.g., they could be performed almost entirely at the same time during a B2C process over a Web EC site, where he/she can contextually discovery a need, find an item and find a merchant able to satisfy it). For such a reason, we think that the choice to unify the first three CBB stages can improve the efficiency of a behavioural model without significant losses of precision; thus we consider as a unique activity (i.e. stage) all the actions performed by a user from when he/she understands to have a need to when he/she knows how to satisfy it. Moreover, often the negotiation stage currently involves also the delivery mode, differently from the original CBB model that only considers the price negotiation of the desired product. Consequently, the purchase phase only consists of the purchase order that represents the effective customer's will to perform that purchase. Finally, in the adopted behavioural model the last CBB stage is not considered because it is implicitly considered in the customer's profile; in fact only a satisfied consumer will perform in the future other similar purchases.

We argue that these characteristics allow us to more effectively model the actual traders' activities in only three phases, respectively called "Visit", "Negotiation" and "Purchase", than the others proposals present in the literature. In the following of this paper we refer to this Web Trader Model (WTM) to represent the B2C processes of a customer. A graphical comparison between WCM and the CBB model is depicted in Figure 1

A promising solution to implement recommender systems that have to deal with different Web sites is represented by the agent technology in which *software information agents* [15–22]) autonomously and proactively perform some tasks

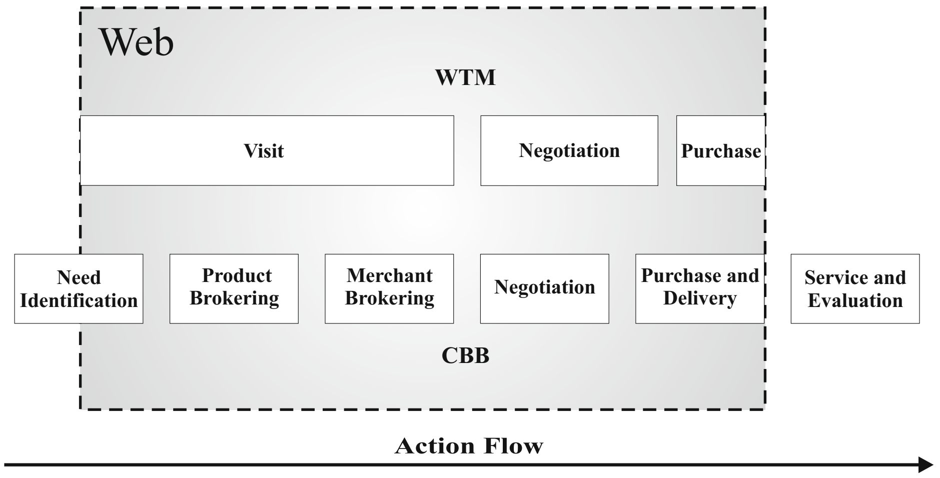

Fig. 1. Comparison between the Web trader model (WTM) and the Consumer Buying Behaviour model (CBB)

on the behalf of its human user. Just as a second contribution of our paper, we propose an approach based on a multi-agent architecture to recommend EC activities. Specifically, in the proposed context each trader is associated with a software agent that constructs and stores an internal profile to take into account his/her whole Web trading history. Furthermore, an agent should adopt suitable techniques to generate an initial user's profile for beginning to work. Note that to allow the updating of the traders' profile, with respect to all the visited EC sites, the profiles can not be stored in the Web sites while the traders navigate on different sites. The agents exploit their user's profiles in the interaction with the EC sites in order to generate effective recommendations based on both content-based and collaborative filtering techniques for adapting the site presentation with useful suggestions. The purposes of our system are (*i*) to produce better suggestions than the traditional approaches, due to the adoption of a suitable customers' behaviours representation and (*ii*) to show high efficiency in the construction of users' profiles, due to the distributed multi-agent architecture.

1.3 Evaluation of the Proposal

To evaluate the effectiveness of TRES for generating suitable suggestions, we have implemented a message-based agent platform to support customers and merchants during all the phases of a B2C process, arranged accordingly to the WTM. In the TRES platform each trader is associated with a personal XML-based agent that monitors his/her B2C activities and specifically *(i)* a customer's agent automatically builds a profile of its user by monitoring his/her interests and preferences during all his/her EC site visits, while *(ii)* a merchant's agent, associated with an EC site (i.e., a merchant), automatically builds a profile by collecting interests and preferences shown by the visitors of its site. TRES agents exploit their profiles to take care of the traders in a personalized and homogeneous way by means of reciprocal message-based interactions. As a result, customers will be provided during their EC-site visits with personalized Web presentations built on-fly exploiting the suggestions generated by using both a content-based and a collaborative filtering approach. The other advantages of the TRES platform can be summarized as follows:

– The usage of the Extensible Markup Language (XML) [23] allows: *(i)* to unify the representation of products belonging to various categories and catalogues overcoming several heterogeneity problems (platforms, languages, applications and communication modalities); *(ii)* to manage the formalization of agent messages and profiles in a light and easy way; *(iii)* to formalize the agent communications in the versatile Agent Communication Markup Language (ACML) [24], an XML coding of the FIPA-ACL [25, 26], that offers some significant advantages as the usage of existing XML message parsers and the opportunity to enrich messages with a large variety of features (i.e., SSL, links, etc.)
– A customer, in order to preserve his/her privacy, can decide which information to send to a merchant agent for building on-the-fly a personalized Web site presentation.
– EC actors can exploit all the TRES features described above in an easy way by means of their usual Web-browsers.

We have carried out some experiments, over the TRES platform, to evaluate the performances of TRES in supporting B2C traders. The experiments have confirmed our expectations and the performances shown by TRES show significant improvements if compared to other similar profile-based recommender systems [27, 28].

The paper is organized as follows: in Section 2 we present the TRES framework and the activities performed by the agents in order to monitor customers' behaviours. Section 3 describes the TRES algorithm while Section 4 presents the Related Work. In Section 5 some experiments are presented and TRES performances are discussed. Finally, in Section 6, we draw some final conclusions.

2 An Overview of the TRES Framework

In this section, the TRES framework will be described in detail. In such a framework (see Figure 2) each customer C (resp. merchant M) is associated to a personal agent c (resp. m) logged into the TRES Agency (Ag). Below we will describe the knowledge representation model used by the agents (Section 2.1), the structure of the agents (Section 2.2) and the agency (Section 2.3), as well as the trading support (Section 2.4) provided by the agents.

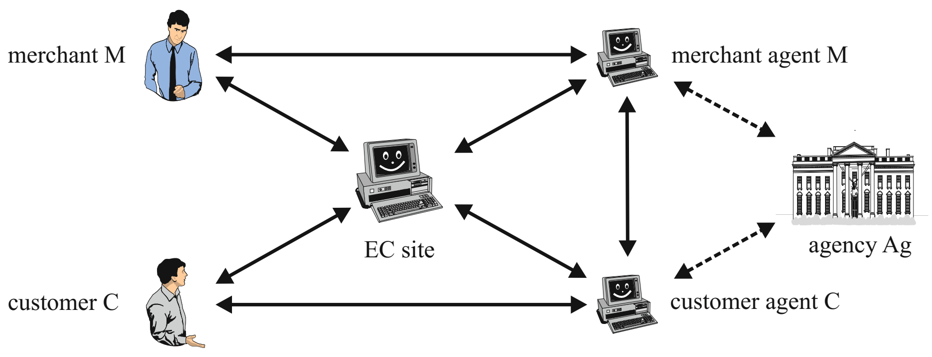

Fig. 2. The TRES framework

2.1 Representation of Objects and Categories of Interest

TRES agents support traders in their commercial tasks for products that meet their interests. To this aim, all the TRES agents share the same *Catalogue* $\mathcal{C}$, representing the common agent knowledge. In this catalogue each *product category* (*pc*) of interest is described with a pair $(code, td)$, where *code* is the identifier of the product category and *td* is a *textual description*.

Usually, each Web page includes more product instances belonging to different product categories. It is reasonable to assume that each time that a user clicks on a hyperlink pointing to a product instance *pi* (and its associated product category *pc*), this means he/she shows an *interest* for *pi* (resp., *pc*). These interests are measurable by taking into account frequency and context of their access. The TRES agents build their users' profiles by monitoring all the product instances and product categories accessed by the customers in order to measure the associated interests. Moreover, to determine collaborative filtering recommendations, each merchant agent also computes the similarity among its visitors by exploiting the values of interest shown in the offered product instances (see below).

Currently, the TRES catalogue adopts the six digit edition of the *NAICS* coding [29], an official, public, hierarchical classification used in North America to classify businesses in categories. The catalogue is implemented by means of a simple XML-Schema [30] where a product category is described by using the notion of *element* and the product instances are represented by XML *element instances*.

2.2 The Agents

Each generic user U (i.e. a customer C or a merchant M) is associated to a personal agent a (i.e. a customer agent c or a merchant agent m), which manages the *User Profile* (UP) of U, graphically described in Fig. 3. In particular, this user profile consists of a tuple $\langle UD, WD, BD \rangle$, where:

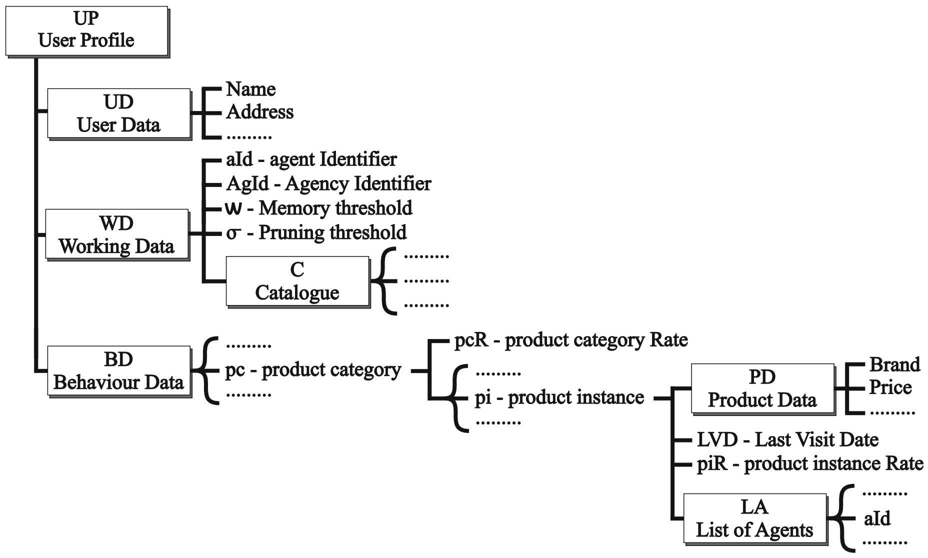

Fig. 3. The User Profile (UP)

- the *User Data* (UD) stores personal U's data as name, address, financial data, etc.
- the *Working Data* (WD) collects the identifiers of the agent (aId) and the agency ($AgId$), the *Memory* parameter (ω) used in the computation of the concept and instance rate, the *Pruning* threshold (σ), described below, and the current product catalogue $\mathcal{C}$.
- the *Behaviour Data* (BD) of a user contains some parameters that describe the past behaviour of the user. Moreover, even though customer and merchant agent share the same structure of profile, some parameters have different meaning for the two types of agent (see Table 1). More in detail, if U is a customer (resp. merchant), BD stores information on each product instance pi and corresponding category pc accessed by the customer (resp. offered by the merchant). Specifically, BD consists in a list of product categories accessed/offered by U and for each category the profile stores its rate pcR (see below) and a list of product instances visited/offered by U and belonging to that category. Furthermore, for each product instance in BD, the profile stores (in PD) some information associated to it (e.g., brand, price and so on), the last visit (LDV) to that instance, its rate piR (see below) and a list of agent (LA) related to that instances.

The behaviour of an agent can be summarized in only two main steps identified as: *(i)* **setup**, that includes some simple semi-automatic procedures to affiliate/delete an agent a to/from the TRES framework; *(ii)* **operational**, that consists of all the tasks to support trading activities (note that the recommendation generation will be described in Section 3). More in detail:

Table 1. Parameters stored in *BD* having a different meaning for a customer and a merchant agent

Name	Customer	Merchant
LVD	Date of his/her last visit to *pi*	Date of the last customer access to *pi*
L	List of the merchants that sell *pi*	List of the customers that accessed to *pi*

setup steps: they are performed *(i)* when the agent a is activated, some basic UP parameters have to be set and sent to the agency Ag that will provide for its affiliation into the agent community and *(ii)* in presence of a deactivation of a that will be performed by Ag for a specific U's request.

operational steps: an agent a is automatically activated when a Web activity starts and deactivated when it ends or for an explicit user's choice. Operatively, the agent a monitors each trading activity, taking into account its context as specified by the WTM model. After each access to a product instance, the agent updates/sets its $BD \in UP$. Furthermore, each agent periodically prunes its UP from negligible data based on both the pcR and σ values.

In other words, when a customer visits an EC site associated with a merchant, both their agents monitor the customer's behaviour in his/her visit (i.e., the accessed product instances and the trading context, accordingly to the WTM). We remark that TRES is a message-based framework in which a message is exchanged between agents due to *(i)* a trader's decision (i.e., a product research) or *(ii)* an action performed by its owner (i.e., a purchase) or *(iii)* an automatic running process (i.e., during an automatic negotiation). In this way, agents easily monitor all the trading activities by means of the exchanged messages, while customers and merchants can access to a message content by using a Web interface. Then, as a consequence of such a monitoring activity, a customer (resp., merchant) agent, with respect to a product instance *pi* belonging to a product category *pc*, could take into account the interest rate *piR* and *pcR* for that *pi* and respective *pc*. More in detail, we propose to compute *piR* and *pcR* by means of the following formulation:

$$piR = (1-\omega)\cdot\frac{piR}{log_{10}(10+\Delta)} + \omega\cdot\frac{\rho_i}{log_{10}(10+N_i)}$$
$$pcR = (1-\omega)\cdot\frac{pcR}{log_{10}(10+\Delta)} + \omega\cdot\frac{\rho_i}{log_{10}(10+N_i)}$$

where $\omega \in [0;1]$, is a system parameter that represents the "memory" of *piR* and *pcR* with respect the past trading activities of the customer. High values for the ω parameter give more relevance to the most recent accesses. Moreover, Δ is the temporal distance expressed in days between the date of the current visit and LVD, and ρ is the weight assigned to the performed WTM stage. We have assigned for the "Visit", "Negotiation" and "Purchase" stages the following values, based on the monitored customers'behaviour (see Section 5): $\rho = 0,01$, $\rho = 0,25$ and $\rho = 1$. Finally, N_i is the number of the same stages performed for the same product in the same day.

Therefore, the parameters piR and pcR, by means of ω, are able to take into account the whole history of accesses to a product (instance and category) suitably weighting both the past and the current accesses. Furthermore, the past customer's interest in a product is updated to the current date based on the time (expressed in days) passed from his/her last access to it. The current WTM stage is considered by exploiting the ρ_i values weighted by $1 + log_{10}(N_i)$, where N_i is the number of times the i-th WTM stage has been performed in the same day. In this way, each time a WTM stage is repeated in the same day, its contribution is considered in a decreasing way because it derives by the same customer's interest.

To illustrate how piR (pcR) works, we present two examples, where we set $\omega = 0.1$. As for the first example, we consider a customer that searches for a new product and carries out the "Visit" WTM phase ($\rho = 0,01$), thus LVD will be set to the current date, while piR (pcR) is equal to 0.001. If the customer decides to immediately performs also the WTM "Negotiation" phase ($\rho = 0,25$), then LVD will be set to the current date and piR (pcR) will be 0.0259. If the customer performs also the purchase of the product, then LVD does not change and the value of piR (pcR) will be equal to $0,1233$. As for the second example, consider the case of a product accessed in the WTM 'Visit" phase twenty days ago. If today the customer accesses three consecutive times for the same WTM phase, then piR (pcR) will assume after each access the values $0,0016$, $0,0022$ and $0,0027$, respectively.

2.3 The Agency

The Agency manages (in terms of insertions, deletions and updating) its Agency Profile (AgP), describes agents affiliations/deletions and provides them with some services.

The AgP (Figure 4) includes: *(i)* the *Agency Identifier* ($AgId$); *(ii)* the *System Data* (SD) that contains the *Agent Pruning* threshold (Σ) exploited to deallocate long-time inactive agents, the *Memory* (ω) and the *Pruning* (σ) thresholds, that have been already described in Section 2.2, and the current catalogue $\mathcal{C}$; *(iii)* the *agent List* (aL), where we store the identifiers of all the affiliated agents. The behaviour of the Agency, except for some activities presented in Section 2.4, can be described as a two-steps process, namely:

affiliate managing: Ag automatically carries out the following operations: *(i)* When Ag receives an *Affiliation Request* it replies by sending an agent (aId), the system parameters ω and σ and the current catalogue $\mathcal{C}$. At this point the agent is logged in and becomes active in the TRES platform; *(ii)* Ag deletes an affiliate following a specific user's request or in an automatic way after an inactivity time greater than the pruning threshold Σ (in such a way the potential growth of inactive affiliates is limited).

service managing: Ag provides the agent community with a yellow page service.

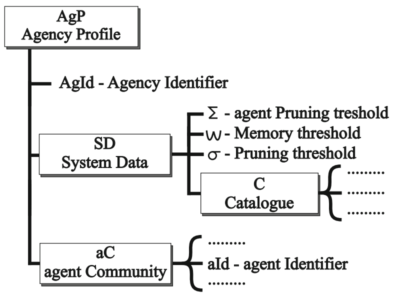

Fig. 4. The Agency Profile (AgP)

2.4 The Trading Support

TRES is a message-based framework designed to transfer, in a consistent and efficient way, business information in order to permit supporting and monitoring the activities carried out in a B2C process, accordingly with the WTM model, by merchant and customer agents. The different activities occurring within a B2C process require the exchange of several message typologies that will be briefly described below together with the different types of agent interactions. Note that in the following notations the first subscript always identifies the sender and the second the receiver, while *data* is an XML document, whose content is context sensitive and it is structured in three sections: *(i) Header*, containing some information like to sender, receiver, WTM stage involved, etc.; *(ii) Products*, that encodes some product data; *(iii) Financial*, that is relative to all the financial information needed to perform a payment. More in detail, the TRES messages are denoted as follows:

- $INF_{x,y}(data)$: it requires/provides commercial information about a product;
- $REQ\_INV_{c,m}(data)$: it requires an invoice for a product offered by M;
- $INV_{m,c}(data)$: it contains the invoice required by $REQ\_INV_{c,m}(data)$;
- $PP_{x,y}(data)$: it is used to negotiate any commercial detail not fixed or specified (i.e., the price for product without fixed price);
- $PO_{c,m}(data)$ (resp., $PO\_A_{m,c}(data)$, $PO\_R_{m,c}(data)$): it is the purchase order relative to $INV_{m,c}(data)$ (resp., the notify if it is accepted or refused);
- $MTO_{c,m}(data)$ (resp.,$MTO\_A_{m,c}(data)$, $MTO\_R_{m,c}(data)$): it notifies that the payment has been performed (resp., accepted or refused) w.r.t. $PO_{c,m}(data)$;

As previously remarked, the interactions occurring between two trader agents (graphically represented in Figure 5) are performed by exchanging messages as it is below syntectically described for each WTM stage. Note that, payments are a relevant issue within any trading event and this is particularly true in an EC scenario where the presence of a network introduces some critical issues [31, 32] absent in traditional payment means. However, in this paper we do not face here this matter because for our aims it is an orthogonal question. In the following, we assume that when a purchase is performed then a payment has to occur without specifying any detail.

More specifically, the agent interactions occurring in TRES in support of customers and merchants can be summarized as:

- **Visit stage (s=1).** During an EC-site visit a customer agent can receive information about a product proposed from a merchant by means of a message $INF_{m,c}$. This message can be explicitly required by the customer to the merchant with a message $INF_{c,m}$ or implicitly when him/her clicks on a hyperlink of the EC-site to a product.
- **Negotiation stage (s=2).** In this stage all the trading details are set, as price, quantity, delivery modality and so on. More in detail, this stage starts with the messages $REQ\_INV_{c,m}$ and $INV_{m,c}$ exchanged between the traders. If the customer accepts the customer's commercial proposal contained in $INV_{m,c}$, then this stage ends. Otherwise, if it is possible, the two traders use in an alternate manner $PP_{x,y}$ messages to negotiate all the trading details; when an agreement has been risen to fix it then the $REQ\_INV_{c,m}$ and $INV_{m,c}$ messages will be again exchanged.
- **Purchase stage (s=3).** Finally, after a "Negotiation" stage, if the customer wants to purchase a products he/she has to sent the message $PO_{c,m}$ to the merchant that can accept/refuse the business by means of $PO\_A_{m,c}(data)$/ $PO\_R_{m,c}(data)$. If the purchase order is accepted, then the customer performs the payment and informs the merchant with an $MTO_{c,m}$ message. He/sheshould receive an $MTO\_A_{m,c}$ or $MTO\_R_{m,c}$ message to confirm the reception.

3 The Recommendation Algorithm

This section presents the recommendation algorithm exploited by TRES to generate suggestions based on both a *content-based* and a *collaborative filtering* approach. A computational complexity analysis of the proposed algorithm ends the section.

The algorithm has been conceived to generate suggestions by exploiting the behavioural information stored into the agent profile of the merchant and of the customer agents. Remember that the information stored into the merchant agent profile are relative to the customers that visited the associated EC site in the past; while the customer agent profile stores only information relative to its owner. In order to produce consistent recommendations, all the *piR* and *pcR* values stored into the two agent profiles are updated before of their use to the current date by means of the coefficient $log_{10}(10 + \Delta)^{-1}$, where Δ has the meaning already explained in Section 2.2. As a result, the product instances supposed the most interesting for the customer, because they match customer's interests and preferences and/or those of other similar customers, are exploited to build on-fly personalized presentation of the visited EC site. In such a way, it is possible to support a customer during its B2C process and, consequently, to improve the commercial performances of the EC site.

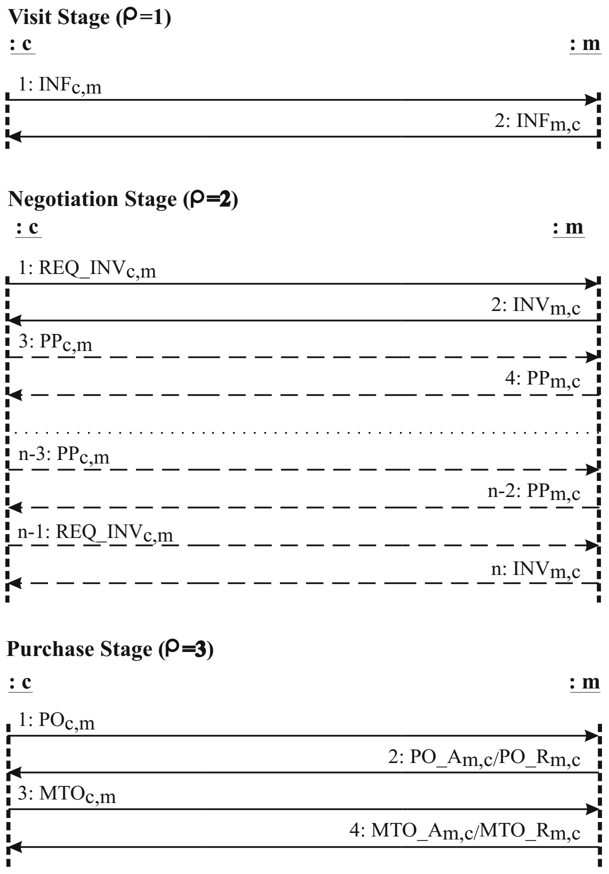

Fig. 5. The TRES support performed for the W-CBB stages

3.1 The Algorithm

The recommendation algorithm of TRES, represented by the function `Recommendations` (Figure 6), is performed by the merchant agent (m) when a customer agent (c) visits the m-monitored EC site. The input of `Recommendations` is a customer agent c and its output are the lists $L3$ and $L4$ of product instances exploited by m to build on-fly personalized Web site presentation for c. Within this function, another function `extract_pc` is called; it receives as input the BD section of the m profile and returns the list $L1$ containing those product categories belonging to $\mathcal{C}$ and currently sold by the merchant. Then the list $L1$ is sent to c by using the function `send`, while the function `receive` waits for the response of c consisting of the list $L2$. This list includes the v product categories that better meet the interests of the customer (where v is a parameter arbitrarily chosen by c). When $L2$ is received, the function `contentbased_pi` is called and returns the list $L3$ containing the first y product instances having the highest rate for each one of the v product categories stored in $L2$ (also y is a parameter arbitrarily set by m).

The next step deals with the construction of the array of lists, called PC, where each array element is a list associated to an agent c monitored by m and contains the v most interesting product categories. To this aim the function `customersInterests` is called and it receives as input the BD data section of the profile of m and the integer v. This is the most expensive function of the recommendation algorithm. Each array element $PC[\,]$ is a list constructed *(i)* by computing for that agent the sum of the piR values of all the product instances for each product category that have met an interest in the past (each sum represents a global measure of the agent interest in a product category) and *(ii)* by ordering such sums in a decreasing order and selecting, for each agent, the v product categories having the highest sum value.

When PC has been computed, the function `collaborativefiltering_pi` is called; it receives as input the list $L2$ (provided by c), the array of lists PC, the BD section of the merchant agent profile and the two integers z and x, arbitrarily set by m. This function exploits PC to compute the similarity degree between c and the agents that have interacted with m in the past; this is a measure used by m to select the z agents most similar to c. Thus, for each product category in $L2$ considered by an agent, all the product instances having the most high piR values are selected and inserted in the list $L4$. $L4$ denotes the function output such that, for each of the z selected agents it contains the first x product instances with the highest rate. More in detail, the similarity is computed as the sum of the contributions provided by each pair of elements common to the two lists. For example, suppose that the product category pc_j belongs both to $L2$ and $PC[k]$, this latter associated with the generic agent k; its contribution to the similarity degree between c and k is given by $(v-|pos(pc_j^{L2})-pos(PC[k]_j)|)*(v-pos(pc_j^{L2}))$, where v is the number of product categories inserted by c in $L2$ and pos is an integer that identifies the ordinal position of the $j-th$ product category pc in a list (i.e., 1 for the first element, 2 for the second element and so on). Note that in the content-based phase, m does not look at the information about c stored in its profile because they are considered less informative than those contained in $L2$.

On the customer agent side, when the list $L1$ coming from the merchant agent m is received, the function `productOfInterest` is executed. In `productOfInterest`, the function `extract_pc` is called (the same used by m) to obtain the list $L5$ containing the product categories of interest for C. After that, the function `intersection_pc` is called and the intersection $L1 \cap L5$ is computed. The function `select_pc` receives as input the list $L6$ and an integer v and then it orders $L6$ in decreasing order based on the pcR value of the agent profile; finally, it returns the first v product categories. The resulting list is the new $L2$, returned as the output of `productOfInterest`.

To preserve customer's privacy, we can note that TRES faces this issue in a simple and effective way. Specifically, it is the customer itself to decide which information has to be sent to a merchant agent. Obviously, the quality of the suggestions generated by TRES (and consequently of the Web presentation built on-the-fly by the merchant agent) is based on these suggestions and related to the precision degree to the customer's interests and preferences representation. Thus, less precise it is this representation and less attractive will be the suggestions (i.e., the Web presentation) proposed to the customer.

```
void Recommendations(customerAgent $c$, ListOfProductInstances$L3$,
                                        ListOfProductInstances$L4$) {
    ListOfProductCategories $L1$=extract_pc($m.UP.BD$);
    send($L1$,$c.Ad$);
    ListOfProductCategories $L2$=receive( );
    ListOfProductInstances $L3$=contentbased_pi($L2$, $m.UP.BD$, $y$);
    ListsOfCustomersInterests $PC$[ ]=customersInterests($m.UP.BD$, $v$);
    ListOfProductInstances $L4$=collaborativefiltering_pi($L2$, $PC$[ ], $m.UP.BD$, $z$, $x$);
    return;
}

ListOfProductCategories productOfInterest(ListofProductCategories $L1$) {
                        ListOfProductCategories $L5$=extract_pc($c.UP.BD$);
                        ListOfProductCategories $L6$=intersection_pc($L1$, $L5$);
                        ListOfProductCategories $L2$=select_pc($L6$,$v$);
                        return $L2$;
}
```

Fig. 6. The recommendation algorithm exploited in the generation of suggestions

3.2 The Computational Complexity of the TRES Recommendation Algorithm

Now we analyze how the time cost and the storage space required by TRES to perform the recommendation task are strictly related to the dimension of the multi-agent system, the dimension of the site catalogue and the dimension of the average number of product instances visited for each product category of the catalogue

In this analysis n denotes the dimension of the multi-agent system, represented by the number of the registered users, p is the dimension of the site catalogue, represented by the number of different concepts present in the catalogue, and q is the average number of product instances for each product category visited by the customer, it is computed by considering all the product categories present into the customer's profile.

More in detail, on the merchant side the computational cost to generate recommendation is due to the cost of performing the function `Recommendations`, described in Figure 6, that, in its turn, depends on the cost of the called functions `extract_pc`, `send`, `receive`, `contentbased_pi`, `customersInterests` and `collaborativefiltering_pi`. Preliminarily, with respect to the functions `send` and `receive` we consider the cost for an agent to send or to receive a message (denoted by C_T) as constant and independent from the particular agent. The first function called is `extract_pc` that computes the product categories of interest for the current customer by comparing the number of items in the site catalogue (p in the worst case) with those present in customer's profile (p in the worst case) obtaining a computational complexity of $\mathcal{O}(p^2)$. The task to compute content based suggestions is carried out by the function `contentbased_pi` that should examine p product instances for each one of the v product categories contained in $L2$. By considering that in the worst case $v \leq p$, the cost of this function is $\mathcal{O}(p \cdot q)$.

Then in `Recommendations` the function `customersInterests` is called that, as previously remarked, is the most expensive function of the TRES recommendation algorithm. This function considers in average q product instances for each product category selected by c from his/her profile. The number of product categories is equal to v in average (p in the worst case) for each registered user). As a result, the computational cost of this function in the worst case is $\mathcal{O}(n \cdot p \cdot q)$.

Finally, the function `collaborativefiltering_pi` selects for a customer c the most x interesting product instances (q in the worst case) for each one of the v product categories (p in the worst case) selected by c and considered also from the z most similar agents (n in the worst case). The agent similarity between the customer c and the other customers (n in the worst case) is computed by comparing the selected product categories (p in the worst case). Thus in the worst case the cost of `collaborativefiltering_pi` is $\mathcal{O}(n \cdot p \cdot q + n \cdot p^2)$.

By considering that the number of product instances associated with a seller is significantly greater than the number of product categories included in the catalogue, but that the product instances visited by a customer are a subset of the whole number of instances, it is reasonable to assume that the value of q is greater or equal to p. In this case, the computational cost of the function `Recommendations` results mainly dependent on the cost of the function `customersInterests` and it is $\mathcal{O}(n \cdot p \cdot q)$. The same considerations can be carried out also for the storage cost that is $\mathcal{O}(n \cdot p \cdot q)$, since it is mainly due to the need of storing all the customers' profiles that in the worst case depends on the number of registered users, the number of product categories in the catalogue and the number of product instances for each product category considered.

On the customer side the computational cost due to `productOfInterest` is $\mathcal{O}(p^2)$. In fact, the functions `extract_pc` and `intersection_pc` have a computational cost, that in the worst case is of $\mathcal{O}(p^2)$, while the cost of the function `select_pc` depends on the cost of the sort algorithm exploited therein that can be assumed as $\mathcal{O}(p \cdot logp)$. Moreover, on the customer side the storage cost depends only on the number of product categories, thus the global cost is $\mathcal{O}(p)$.

4 Related Work

Recommender systems are generally adopted in several EC sites (e.g., Amazon, CDNOW, GroupLens, MovieLens, etc. [6]) and a large number of models and architectures are based on software agents. In this section, we describe some past approaches that in the generation of the recommendations implement both content-based and collaborative filtering techniques [33–36] and appear as the closest to the proposal presented in this paper. At the end of the current section, differences and similarities with TRES will be pointed out. For a more complete account about this matter the interested reader might refer to the considerable number of surveys that have investigated the state of the art [3–7, 37, 38].

The multi-agent system IMPLICIT [39] uses a search engine (i.e., Google) while personal agents reciprocal interacting in order to generate suitable recommendations by exploiting the notion of *Implicit Culture*. The search engine results are thus complemented with the recommendations produced by the agents. In IMPLICIT each user is assisted by a personal agent during his/her search to find Web links considered relevant and for discovering agents to contact from which obtaining relevant links. The Search behaviour consists of the Google search behaviour and of the Platform search behaviour, which comprises both the Internal and the External search behaviour. In the Google search behaviour the agent processes query to the Google search engine in order to obtain some suggestions for any entered keyword. In the Internal search are generated links, based on the past user actions monitored by his/her agent while in the External search are proposed the most suitable agents to contact, also exploiting a yellow pages service provided by the Directory Facilitator of the platform. When all the agents are contacted and the search behaviour queries suggested by the new agents have been performed, the system shows all the discovered links to the user.

Another multi-agent recommender system is MASHA (Multi Agent System Handling Adaptivity) [40] that, as main feature, considers in its suggestions also the characteristics of the device currently exploited by a user in his/her Web activities. In MASHA each user is associated with a *server agent* that builds a global user profile collecting by the user's *client agent*s, each one associated with a user's device, their profiles, storing information about the user's behaviour when he/she uses that device. The *adapter agent*s, associated with each Web site, exploit global profiles to generate personalized Web site presentations containing suggestions derived from the profiles of the current user and those of other users that have visited the site in the past by exploiting the same device.

Handy Broker [41] adopts an evolutionary ontology-based approach for mobile trading activities. The system assumes users as rational individuals able to evaluate the product relevance uniquely by means of tangible attributes (e.g., the price), while intangible attributes, such as the brand, are not considered. The Handy Broker agent generates its recommendations based both *(i)* on the whole user's history stored in a user's profile in which his/her preferences are witnessed by how many times a product has been selected and *(ii)* an evolutionary mechanism that allows an agent to integrate in its profile those of other users for performing, in such a way, an implicit collaborative filtering recommender stage. In [42] agents support both recommendation and negotiation phases. In particular, the recommendation part is designed to assist a consumer for products rarely purchased. The system builds its knowledge of the products by interfacing with domain experts and uses the acquired knowledge to calculate optimal products that match customers' preferences (provided by means of a questioner). Optimality of the products is computed by using a multi-attribute decision making method based on consumer's needs and features of products.

Moreover, in order to share the experiences of other consumers, a dynamic programming approach is used to exploit social information derived from previous consumer recommendations.

CBCF [27] (Content-Boosted Collaborative Filtering) exploits *(i)* a content-based predictor to process user data (as text documents), rated by each user in six classes of relevance, and *(ii)* a collaborative filtering approach adopting a neighborhood-based algorithm for choosing a subset of users similar to him/her in order to obtain personalized recommendations by exploiting a weighted combination of their ratings.

A Graph can model in an easy way the usage information, as for example in X-Compass [28], SUGGEST [43] and in [44]. More specifically, in X-Compass a XML-based agent is associated with a user U in order to suggest the Web pages potentially interesting for him/her. To this aim the agent monitors the Web pages visited by U and automatically builds and manages a *user profile* modeled with a graph. In this graph each node is associated to a concept of interest for U, its *Attraction Degree* and a *Key Set* representing the semantics of the concept, while each arcs stands for both *is-a* relationships and associative rules. Furthermore, the agent updates two lists containing the visited Web pages, ordered with a temporal access criterion, and their whole visit history, respectively. Likewise, SUGGEST supports user Web navigation by dynamically generating links to unvisited pages so evaluated attractive for him. A complete graph is adopted to model historical user's navigational information (users' sessions are identified by means of cookies stored on the client side) and it is handled by means of an incremental graph partitioning algorithm. In the graph the set of vertices contains the identifiers of the pages hosted on the Web server and the set of edges contains the weight $W_{ij} = N_{ij}/max\{N_i, N_j\}$, where N_{ij} is the number of sessions containing both pages i and j, N_i and N_j are the number of sessions containing the only page i and j, respectively. The graph is then partitioned by using a clustering algorithm (a version of the incremental connected components algorithm) in order to find groups of strongly correlated pages. Subsequently, the cluster having the largest intersection with the page window correspondent to the current session contributes to form the suggestion list.

In [44] a two-layer graph model is exploited, where nodes, intra-layer and inter-layer links of two graphs are respectively used to represent customers and products (each one in a different layer), transactions and similarities. This representation of user-products relationships is exploited to implement content-based, collaborative filtering and hybrid recommenders by using "direct retrieval" (DR), "association mining" (AM) and "High-degree association retrieval" ($HDAR$) methodologies. In the DR methodology: the content-based part is similar to that used in information retrieval where documents similar to the input queries are retried; collaborative filtering is based on users' similarity evaluated by using information extracted from the graph topologies; the hybrid section simply combines the other two techniques. The AM methodology uses transaction histories and associative rules, that obviously change for content-based and for collaborative filtering recommendations, while hybrid approach combines them.

The last method (*HDAR*) deals with data sparsity by transitively exploring layer topologies to search information related to the neighborhoods of each users and to define associative rules for generating content-based and collaborative filtering recommendations. Joining such rules it is possible to obtain a hybrid recommender.

5 Experiments

In this section, we present some experiments devoted to show the effectiveness of TRES to generate useful suggestions for supporting users during their B2C activities. The experiments presented below have been realized by using a TRES prototype, developed in JADE [45, 46]), able to completely implement all its features throughout all the W-CBB activities.

For this experiment session we have *(i)* built a family of 18 XML EC TRES compliant Web sites (Figure 7) by using the NAICS coding as common vocabulary, represented by a unique XML Schema, with 760 product instances belonging to 9 NAICS categories and *(ii)* monitored a set of real users in their B2C activities within the TRES framework. The first 9 sites has been used to obtain an initial profile of the customers' interests and preferences without to exploit any recommendation support and to determine the value of the ρ parameters. Based on such profiles and ρ values, the recommendations have been generated by the merchant agents relatively to the other 9 sites.

We have compared TRES, identified in the following by $TRES - ON$, with a modified version of this system, identified in the following by $TRES - OFF$ where users' interests and preferences are taken into account simply by means of the only number of access to a product instance (i.e., product category) without to consider the trading context, computing *piR* and *pcR*, using the following formulation:

$$\begin{aligned} piR &= \frac{piR}{log_{10}(10+\Delta)} + 1 \\ pcR &= \frac{pcR}{log_{10}(10+\Delta)} + 1 \end{aligned}$$

where Δ has the same meaning explained in Section 2.2.

Moreover, these tests have included the comparison with $CBCF$ and $X - COMPASS$, that are both two content-based and collaborative filtering recommender systems exploiting an user's profile built monitoring the user's behaviour. They are, at the best of our knowledge, two of the most performative recommender systems, as highlighted by the experimental results described in [27, 28] that compare these systems with other well-known recommender systems. Also the $CBCF$ and X-COMPASS agents have been implemented in JADE by following the descriptions of the data structures and recommendation algorithms proposed in [27, 28], respectively.

To evaluate the results of the experiments, we have inserted in a list, called A, the product instances suggested by the merchant agent and in a list, called B, the corresponding customer's choices. The associated pairs in the two lists

Table 2. Performances of different recommendation algorithms (global/content-based/collaborative filtering)

	$TRES-ON$	$TRES-OFF$	$CBCF$	$X-Compass$
Pre	0.597/0.508/0.412	0.461/0.372/0.325	0.503/0.418/0.374	0.421/0.376/0.287
Rec	0.564/0.482/0.392	0.434/0.336/0.289	0.467/0.392/0.341	0.407/0.354/0.244
F	0.580/0.495/0.402	0.442/0.353/0.306	0.484/0.405/0.357	0.367/0.365/0.264

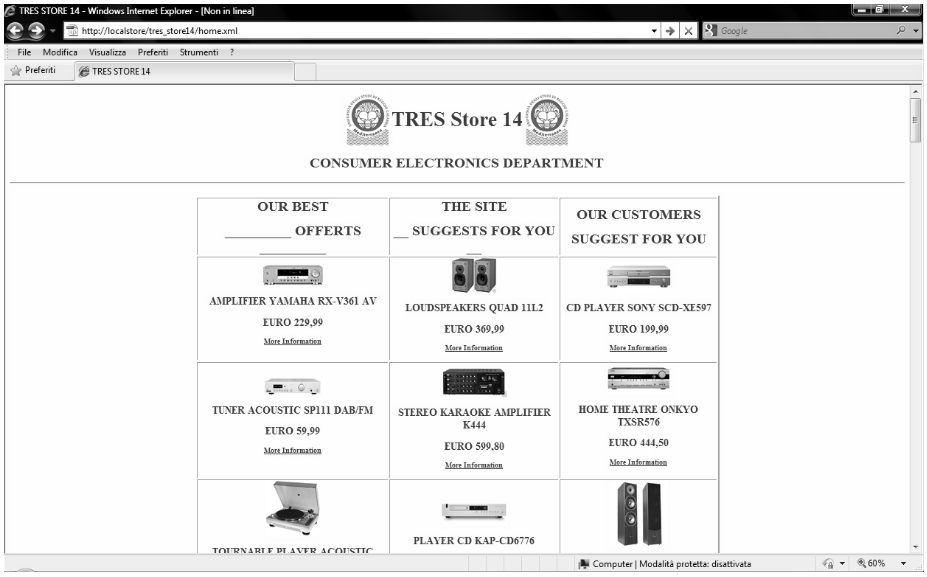
TRES STORE 14 - Windows Internet Explorer - [Non in linea]
http://localstore/tres_store14/home.xml
Google
File Modifica Visualizza Preferiti Strumenti ?
Preferiti
TRES STORE 14
TRES Store 14
CONSUMER ELECTRONICS DEPARTMENT
OUR BEST OFFERTS
THE SITE SUGGESTS FOR YOU
OUR CUSTOMERS SUGGEST FOR YOU
AMPLIFIER YAMAHA RX-V361 AV
EURO 229,99
More Information
LOUDSPEAKERS QUAD 11L2
EURO 369,99
More Information
CD PLAYER SONY SCD-XE597
EURO 199,99
More Information
TUNER ACOUSTIC SP111 DAB/FM
EURO 59,99
More Information
STEREO KARAOKE AMPLIFIER K444
EURO 599,80
More Information
HOME THEATRE ONKYO TXSR576
EURO 444,50
More Information
TOURNABLE PLAYER ACOUSTIC
PLAYER CD KAP-CD6776
Computer | Modalità protetta: disattivata
60%

Fig. 7. The personalization of the home page of the site N.14

have been compared in order to measure the effectiveness of the generated suggestions. We have adopted standard performance metrics that are, precision, recall and F-measure [47] to compare the tested recommender approaches. More in detail, precision is defined as the share of the concepts actually visited by the user among those recommended by the system. Recall is the share of the concepts suggested by the system among those chosen by the user. F-Measure represents the harmonic mean between Precision and Recall. The three measures are definable as follows:

$$Pre(A(x)) = \frac{|A(x) \bigcap B(x)|}{|A(x)|} \quad , \quad Rec(A(x)) = \frac{|A(x) \bigcap B(x)|}{|B(x)|}$$

$$F(A(x)) = \frac{2 \times Rec(A(x)) \times Pre(A(x))}{Rec(A(x)) + Pre(A(x))}$$

The performance of the content-based and collaborative filtering components have been considered both in an integrated way and separately for all the tested recommenders. The parameters v, y, z and x of TRES have been set to 2, 3, 2 and 3, respectively (see Section 3). In terms of results (see Table 2) TRES has outperformed the other approaches chosen for the comparison. TRES shows with respect to CBCF, the best competitor as global, content-based and collaborative filtering performances, an improvement of precision of about 19, 22 and 10 percent and an improvement of recall of about a 21, a 23 and a 15 percent. Furthermore, from the results of Table 2 we can argue that the good performance of TRES, with respect to the other considered approaches, is due to the fact that TRES considers, in determining its suggestions, customers' interests and preferences more exactly than other methods mainly thanks to consider the different phases enacted by a B2C process. To confirm our hypothesis, there is the evidence that the same TRES algorithm implemented without to consider its evolute characteristics produces good results but only comparable with the other tested recommenders.

6 Conclusion

This paper illustrates a recommender system, called TRES, that is able to generate both content-based and collaborative filtering suggestions, to support traders in their B2C activities. Suggestions take into account customers' interests and preferences in B2C activities arranged as in the Web trading model and, as a consequence, contributing to increase the performances of the EC sites.

Some interesting results have been obtained by the experimental simulations carried out using a JADE-based prototypal implementation over an agent platform, appositely conceived to this aim. The effectiveness of TRES in generating suggestions is resulted higher than that of the same TRES algorithm implemented without considering the different trading phases enacted by a B2C process and also higher than that of other two competitor recommenders.

In the next future, further developments are expected from the introduction of different behavioural models acting in the B2C area that might contribute to build a more detailed customer's profile in order to generate more precise suggestions.

References

1. Kauffman, R.J., Walden, E.A.: Economics and Electronic Commerce: Survey and Directions for Research. International Journal of Electronic Commerce 5(4), 5–116 (2001)
2. Zwass, V.: Electronic Commerce and Organizational Innovation: Aspects and Opportunities. International Journal of Electronic Commerce 7(3), 7–37 (2003)
3. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Analysis of Recommendation Algorithms for E-Commerce. In: Proceedings of the 2nd ACM Conference on Electronic Commerce (EC 2000), pp. 158–167. ACM, New York (2000)
4. Schafer, J.B., Konstan, J.A., Riedl, J.: E-Commerce Recommendation Applications. Data Mining Knowledge Discovery 5(1-2), 115–153 (2001)

5. Burke, R.D.: Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-Adapted Interaction 12(4), 331–370 (2002)
6. Montaner, M., Lopez, B., de la Rosa, J.L.: A Taxonomy of Recommender Agents on the Internet. Journal of Web Semantics (JWS) 19(4), 285–330 (2004)
7. Wei, K., Huang, J., Fu, S.: A Survey of E-Commerce Recommender Systems. In: Proceedings of the 13th International Conference on Service Systems and Service Management, pp. 1–5. IEEE Computer Society, Washington, DC (2007)
8. Nicosia, F.: Consumer Decision Processes: Marketing and Advertising Implications. Prentice Hall, New York (1966)
9. Engel, J.F., Blackwell, R.D., Miniard, P.W.: Consumer Behaviour. International ed. The Dryden Press, London, UK (1995)
10. Nissen, M.E.: The Commerce Model for Electronic Redesign. J. of Internet Purchasing 1(2) (1997), `http://www.arraydev.com/commerce/JIP/9702--01.htm`
11. Feldman, S.: The Objects of the E-Commerce, Keynote speech at ACM 1999 Conference on OOPLSA, Denver (1999), `http://www.ibm.com/iac/oopsla99-sifkeynote.pdf`
12. Guttman, R.H., Moukas, A., Maes, P.: Agents as Mediators in Electronic Commerce. Electronic Markets 8(1) (1998)
13. He, M., Jennings, N.R., Leung, H.: On Agent-Mediated Electronic Commerce. IEEE Transaction Knowledge Data Engineering 15(4), 985–1003 (2003)
14. Palopoli, L., Rosaci, D., Ursino, D.: Agents' Roles in B2C e-Commerce. AI Communication 19(2), 95–126 (2006)
15. Maes, P.: Agents that Reduce Work and Information Overload. Communication of ACM 37(7), 30–40 (1994)
16. Hayes-Roth, B.: An Architecture for Adaptive Intelligent Systems. Artificial Intelligence 72(1-2), 329–365 (1995)
17. Wooldridge, M., Jennings, N.R.: Agent Theories, Architectures, and Languages: A Survey. In: Wooldridge, M.J., Jennings, N.R. (eds.) ECAI 1994 and ATAL 1994. LNCS, vol. 890, pp. 1–39. Springer, Heidelberg (1995)
18. Franklin, S., Graesser, A.C.: Is it an Agent, or Just a Program?: A Taxonomy for Autonomous Agents. In: Jennings, N.R., Wooldridge, M.J., Müller, J.P. (eds.) ECAI-WS 1996 and ATAL 1996. LNCS, vol. 1193, pp. 21–35. Springer, Heidelberg (1997)
19. Gilbert, D., Aparicio, M., Atkinson, B., Brady, S., Ciocarino, J., Grosof, B., O'Connor, P., Osisek, D., Pritko, S., Spagna, R., Wilson, L.: White Paper on Intelligent Agents. IBM Report, Zurich, Switzerland (1996)
20. Nwana, H.S.: Software Agents: An Overview. Knowoledge Engineering Review 11(3), 11–40 (1996)
21. Russell, S.J.: Rationality and Intelligence. Artificial Intelligence 94(1-2), 57–77 (1997)
22. Iglesias, C.A., Garijo, M., González, J.C.: A Survey of Agent-Oriented Methodologies. In: Papadimitriou, C., Singh, M.P., Müller, J.P. (eds.) ATAL 1998. LNCS (LNAI), vol. 1555, pp. 317–330. Springer, Heidelberg (1999)
23. Extensible Markup Language (XML) v.e. 1.1 (2010), `http://www.w3.org/TR/2004/REC-xml11-20040204`
24. Grosof, B.N., Labrou, Y.: An Approach to Using XML and a Rule-Based Content Language with an Agent Communication Language. In: Dignum, F., Greaves, M. (eds.) Agent Communication. LNCS (LNAI), vol. 1916, pp. 96–117. Springer, Heidelberg (2000)

25. O'Brien, P.D., Nicol, R.C.: FIPA Towards a Standard for Software Agents. BT Technology Journal 16(3), 51–59 (1998)
26. Foundation for Intelligent Physical Agents (FIPA) - ACL URL. FIPA ACL Message Structure Specif. (2010), `http://www.fipa.org/specs/fipa00061/`
27. Melville, P., Mooney, R.J., Nagarajan, R.: Content-boosted Collaborative Filtering for Improved Recommendations. In: Proceedings of the 18th National Conference on Artificial Intelligence, Edmonton, Canada, pp. 187–192. AAAI/IAAI (2002)
28. Garruzzo, S., Modafferi, S., Rosaci, D., Ursino, D.: X-Compass: An XML Agent for Supporting User Navigation on the Web. In: Andreasen, T., Motro, A., Christiansen, H., Larsen, H.L. (eds.) FQAS 2002. LNCS (LNAI), vol. 2522, pp. 197–211. Springer, Heidelberg (2002)
29. North America Industry Classifications (NAICS) (2010), `http://www.census.gov/naics/2007/index.html`
30. Extensible Markup Language (XML) Schema (2010), `http://www.w3.org/XML/Schema`
31. Asokan, N., Janson, P.A., Steiner, M., Waidner, M.: The State of the Art in Electronic Payment Systems. IEEE Computer 30(9), 28–35 (1997)
32. O'Mahony, D., Pierce, M., Tewari, H.: Electronic Payment Systems for E-Commerce, 2nd edn. Artech House, Norwood (2001)
33. Balabanovic, M., Shoham, Y.: Content-Based, Collaborative Recommendation. Communication of ACM 40(3), 66–72 (1997)
34. Aggarwal, C.C., Yu, P.S.: Data Mining Techniques for Personalization. IEEE Data Engineering Bulletin 23(1), 4–9 (2000)
35. Lawrence, R.D., Almasi, G.S., Kotlyar, V., Viveros, M.S., Duri, S.: Personalization of Supermarket Product Recommendations. Data Mining Knowledge Discovery 5(1/2), 11–32 (2001)
36. Tso, K.H.L., Schmidt-Thieme, L.: Evaluation of Attribute-Aware Recommender System Algorithms on Data with Varying Characteristics. In: Ng, W.-K., Kitsuregawa, M., Li, J., Chang, K. (eds.) PAKDD 2006. LNCS (LNAI), vol. 3918, pp. 831–840. Springer, Heidelberg (2006)
37. Wei, C.P., Shaw, M.J., Easley, R.F.: A Survey of Recommendation Systems in Electonic Commerce. In: E-Service: New directions in Theory and Practice. ME Sharpe, Armonk (2002)
38. Manouselis, N., Costopoulou, C.: Analysis and Classification of Multi-Criteria Recommender Systems. World Wide Web 10(4), 415–441 (2007)
39. Birukov, A., Blanzieri, E., Giorgini, P.: Implicit: A Recommender System that Uses Implicit Knowledge to Produce Suggestions. In: Workshop on Multi-Agent Information Retrieval and Recommender Systems at the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005), Edinburgh, Scotland (2005)
40. Rosaci, D., Sarnè, G.M.L.: MASHA: A Multi-Agent System Handling User and Device Adaptivity of Web Sites. User Modelling User-Adaptive Interaction 16(5), 435–462 (2006)
41. Guan, S., Ngoo, C.S., Zhu, F.: Handy broker: an Intelligent Product-Brokering Agent for m-Commerce Applications with User Preference Tracking. Electronic Commerce Research and Application 1(3-4), 314–330 (2002)
42. Lee, W.P.: Towards Agent-based Decision Making in the Electronic Marketplace: Interactive Recommendation and Automated Negotiation. Expert Systems with Applications 27(4), 665–679 (2004)

43. Silvestri, F., Baraglia, R., Palmerini, P., Serranó, M.: On-line Generation of Suggestions for Web Users. In: ITCC (1), pp. 392–397. IEEE Computer Society, Washington, DC (2004)
44. Huang, Z., Chung, W., Chen, H.: A graph model for e-commerce recommender systems. Journal of American Society Information Science Technology 55(3), 259–274 (2004)
45. Bellifemine, F., Poggi, A., Rimassa, G.: Developing Multi-Agent Systems with a FIPA-compliant Agent Framework. User Modelling User-Adaptive Interaction 12(4), 331–370 (2002)
46. Java Agent DEvelop. framew. (JADE) (2010), http://jade.tilab.com/
47. van Rijsbergen, C.J.: Information Retrieval. Butterworth (1979)