Abstract

This project aims to study and implement a prototype of a Web-Based Collaborative Event Planning service. In the proposed model, each user can suggest plans for collaboration such as blog posts in addition to browsing other users' ideas. We have assumed that each plan seeks a target, e.g. the number of agreements by a certain time. In this work, a framework for making recommendation is presented and a simple prototype is launched and tested. Furthermore, a simple scheme for simulating a collaborative event planning system is suggested.

Table of Contents

1 INTRODUCTION

1.1 Motivations

Web-based solutions for augmenting human collaborations and socialising have had an immense impact on today's lifestyle. However, the journey towards a truly socially coordinated world has only just begun. To some extent, the majority of advances in the context of web-based collaborative solutions are related to social awareness and information sharing and it is very likely that not all of the potential outgrowth of this increasing global awareness has been explored. In this respect, one motivation for researching this area could be to develop more effective web-based services, specifically designed for helping collectives of subjectively likeminded people—who may or may not already know each other—with achieving agreement and coordination in order to accomplish favourable collaborations. For now, the term "Collaborative Event Planning" (CEP) will be generally applied to these types of services; more specific and formal definition will be given later on.

Predictably, there are numerous services and products operating in this field, but the question that stimulates us is: "*How could we build more effective, more intelligent and more applicable Collaborative Event Planning systems*?" We believe that there is a lot more to explore in the field of Web-based Collaborative Planning (WBCP), and investigating both theoretical and applicational improvements of WBCP backs up the primary motivation of our project.

The following case studies are intended to give an understanding of "Web-based Collaborative Planning" in this project.

WBCP and Cinema Scheduling

The majority movies are made to be watched on cinemas screens and, as not everyone has a private cinema at home, people end up watching films on DVD or only whatever is offered in public cinemas. In fact, public cinemas are the final link of a movie distribution chain. Traditionally, distribution companies are those that deal with filmmaking studios and they decide on how many copies (prints) of which movies to buy. They then show their purchased movies to prospective (screening) buyers representing the theatres and, finally, those buyers negotiate with the distribution company on which movies they wish to lease and the terms of the lease agreement. The rest of the story is clear: for a specified number of weeks (the engagement time) movies will be shown on screen. Finally, at the end of the engagement time, the theatre sends the print back to the distribution company and makes payment on the lease agreement (1). The most interesting part of this multibillion-dollar chain market is that end users are not involved in any direct way in the subject and timing of show schedules. Not too infrequently, the order of this chain is the only reason for the high rate of ticket sales for many movies, as many viewers leave the theatre disappointed by the film they have seen. The aim of WBCP is to let viewers' preferences influence cinemas' schedules more directly, with the aim of reducing the risk of customer dissatisfaction. Web-based Collaborative Planning can present a solution to this issue by providing a website that presents collections of movies trailers, critics' reviews and feedback from viewers. Users can create their own profiles detailing preferred choices of cinemas, times and types of movies. They can also access other viewers' reviews and follow the trends (highly preferred plans), as well as commenting on the plans, movies and recommendations of other viewers. The Collaborative Event Planning system designs movie schedules for cinemas, while the estimated coverage of the schedule (i.e. the number of people who are likely to turn up to see a movie) is optimal and of course advantageous to all companies involved with the distribution of the movie. The system could also include some weighting discipline or a user crediting system specifying the significance of users' promises, i.e. their intended movie choices. Note that it is highly important to make the system for browsing other people's ideas and preferences as intelligent and fast as possible for the user. Finally, an evolution of this idea should address two questions: 1) assuming the presence of such a movie scheduling system, whether the number of tickets sold has increased (benefiting the suppliers), and 2) whether there has been a reasonable increase in viewers' satisfaction (to the benefit of customers).

Table 1 gives a brief illustration of collaborative cinema scheduling. The table contains a toy set of information gathered from a few users, detailing their preferred cinema viewing times and preferred movies. Intuitionally, we may deduce that showing the movie *Godfather I* at 19.00 is likely to attract a relatively high number of users, assuming that people who like to watch *Godfather II* would also be likely to choose to watch *Godfather II*. Note that the idea is to convey how we can design intelligent plans—certainly cleverer than any human might be capable of—by employing this sort of reasoning on a large scale.

User name	**Preferred movies**	**Preferred show time**
Joe	La Strada; Godfather II	any time
Jake	Amarcord; Paris Texas	only 14.00 or 17.00
Alice	Annie Hall; Monty Python; Godfather II	approx 18.00
Bob	Godfather I; Fight Club	only 19.00 or 14.00
Steve	Casablanca; Amarcord; Godfather I	19.00
John	Matrix; Fight Club; Godfather I	any time

Table 1: A toy dataset of users' preferences.

WBCP and Collaborative Purchasing

As a rule, suppliers offer better deals on high-volume orders. A brief description of the aims of collaborative purchasing is to coordinate users (in terms of what and when they buy), with the aim of increasing their purchasing power. This could work as follows: A website offers a free space to all suppliers, where they can represent their goods with up-to-date pricing. Customers can search for anything they require and make different lists with different priorities, based on what and when they may consider buying, and make a request for quote. Moreover, people can post reviews and make recommendations to other users on different shopping items and plans. A collaborative event planning system designs optimum shopping lists with the aim of reducing costs for the buyers. The evaluation of such a system should in the long term seek to benefit the suppliers and buyers involved in the collaborative purchasing process.

Collaborative Money Transferring [1]

Typically, money transferring agencies and brokers offer better services for better customers (2), for instance lower transaction fees for handling higher volumes of money. Roughly speaking, if customers could by some means coordinate their transactions, hypothetically they could together become entitled to a preferable deal. A collaborative planning service can take all the orders—along with their priorities—and at the output design bigger requests for quotes. Obviously, without having a detailed picture of the business, it is impossible to pose a realistic problem in the context of computer science; yet again the idea sounds motivating enough to support research on the web-based collaborative event planning systems.

Other Scenarios

By looking at the pattern of given scenarios it is possible to come up with additional applications for collaborative planning. To some degree, all of these applications share similar characteristics: Chiefly, they all consist of tasks that are more efficient or less costly when they are coordinated by larger collectives. Examples of such collaborations could include:

- Collaborative planning for online live gatherings, e.g. webinars; live lectures with highly sought-after tutors; auctioning; and live forums of people with common interests.
- Physical gatherings of people of same intention, e.g. collaboratively planned concerts; leisure and sporting aims; charity functions; or planning for demonstrations.

Consideration of all above points should give us a fundamental understanding of web-based collaborative event planning and some of its potential applications. Due to the scope of this work, no

[1] Many thanks to Dr. Julian Gough, who originally pointed to the potentials of applying Collaborative Event Planning in Transferring Money.

specific application has been analysed. Instead, the intention has been to focus on the theoretical basis of WBCP.

1.2 Aims, Objectives and Organisation of the Presentation

The aim of this work is to research and implement a simple prototype model of a web-based collaborative event planning service that is designed to help collectives of contextually likeminded people to achieve agreement and coordination on favourable collaborations.

To this aim, the following objectives are proposed:

1- Setting a formal framework for collaborative event planning and posing a prediction problem for improving the information retrieval process, i.e. an automated plan recommender

2- Implementing a simple prototype of the collaborative event planning website

The first objective is cleared in Chapter 2, and the third chapter covers the requirement analysis and design specification of a general-purpose prototype of a collaborative event planning website. Conclusions, evaluations and future works are explained in the final chapter.

The remainder of the current chapter reviews the literature and some existing solutions in the context of collaborative planning.

1.3 Background

This section attends to a collection of technological background. After that, a brief review of existing similar business and working models of collaborative event planning is given.

1.3.1 Context

Collaborative event planning is contextually associated with the following topics:

User-Generated Content

A user-generated content (UGC) media tends to openly—or almost without centralized authorization—involve the end users of the system in the process of generating content. Despite the considerable issues of such systems, they have significantly influenced most businesses in relation to gathering content, and in fact the list of today’s applications and services based on user-generated content platforms is extensive. In general, this revolutionary structure of providing content has a number of drawbacks, particularly with legal issues (e.g. copyright dilemma, i.e.who ultimately owns the generated content); liability (of content providers as well as of websites that allow UGC); privacy; and quality. There is a range of organizations and schemes available to help UGC systems with providing contents with higher degrees of qualities and standards, e.g. full or partial monitoring of contents by administrators; user rating and user abuse reporting systems; and user crediting or rewarding systems. Examples of UGC web-based systems are: discussion boards, blogs, wikis, social networking sites, customer review sites, memory boards, image and video-sharing websites, etc (3).

Noticeably, in terms of content, a collaborative planning website is fed by its users and, therefore, ultimately it should deal with issues that may threaten its functionality, e.g. users that tend to post insignificant or disruptive plans, or possibilities of abusing the system. This is covered in more detail in the next chapter.

Collective Intelligence

Although collective intelligence is a fairly old notion[2], the very exemplary working models of such systems are among the latest achievements of mankind, e.g. Google, Wikipedia. By observing Google one could see the emergence of a kind of exceptional and unprecedented higher intelligence..Google as a system—that is, not only the its technologies, but all the indexed web pages, links produced by humans and all the users' activities—can perform searches more intelligently than any other existing machine or individual. In fact, the Google system is an epitome of a collective intelligence (CI)-based system. CI is a whole new class of intelligence and can be defined as "*groups of individuals and machines doing things collectively that seems intelligent*" (4).

The idea of Collaborative Event planning (CEP) involves applying the intelligence of users along AI techniques to design optimally fitting plans for groups of users. Hence it is important to set a platform for CEP that not only allows users to act intelligently, but also to learn from the collection of these intelligent actions. Soon after, this document attends to a field—namely collaborative filtering—that is intended to improve information retrieval through the use of Collective Intelligence.

- **Electronic Calendaring**

In practice, event planning deals with the time and the location of an event, as well as identifying the content of the event. Usually, the most challenging part of event planning is event calendaring. Naturally, people who can easily achieve agreement over the content of an event may find it difficult to agree on a fixed time to hold the event, with the exception of events that are occur on specific dates, such as anniversaries. For this reason, we briefly review web-based calendaring context.

An evolutionary idea in the field of calendaring was enabling users to share their calendars with a specific level of visibility. This feature of electronic calendars has become important and arguably[*] useful, since the introduction of groupware software. Groupware software is aimed to enhance group

[2] "A precursor of the concept is found in entomologist William Morton Wheeler's observation that seemingly independent individuals can cooperate so closely as to become indistinguishable from a single organism (1911)"

working experiences, to aid people to communicate and work together in collaborative environments. Perhaps the most dominant of today's electronic calendars are web-based electronic calendars, with the core feature that the personal schedule, event lists and daily memo can be stored online. Even while most mobile phones today include calendaring capabilities, their quite common ability is to synchronize with an online storage facility (5).

Here is a brief list of the features that are provided by today's major online calendaring services: To do list; file attachment (a feature that allows users to attach a file to an appointment or event); automatic reminder of upcoming events; conflict reporters; defining multiple calendars (e.g. work calendar, children's school calendar) and multiple viewing options (i.e. daily, weekly, monthly or yearly); automated alternative time suggesting; printing and appearance customization functionalities; web-based interface which enables users to access their calendars from any computer or mobile device (this feature is important for desktop calendaring applications such as Outlook, Mozilla Sunbird). But perhaps the most favourable functionalities are regarding *availability sharing* functionalities of calendars, such as:

- Making some part or the whole of a calendar visible only to certain people (usually chosen from the user's email contact list)
- Making some part or the whole of a calendar editable for certain people
- Publishing an only visible calendar, by providing the link address
- Enabling the subscription (importing and exporting calendars in certain file formats)
- Collaborative calendaring (the capability of proposing meeting times to all of the participants)
- Group Calendar – a calendar showing dates of groups in addition to individual calendars
- Email – electronic mail communication system. This can be tied into the appointment calendar to send reminders and notify the participants of issues arising with scheduled meetings

- **Market pull**

To some extent, our given application scenarios of collaborative event planning characterize a modern approach of supplying goods and services, namely market pulling*;* in contrast to market pushing. Market pulling is a business term that describes the movement of services, products or information when they originate from clients rather than providers (6). Currently, some large companies tend to distribute their products based on the market pull supply strategy. The following remark is a part of a recent inquiry, made by Ford Car Company, which concludes that it is best to approach the market of environmentally friendly fuel-adapted cars via the market pull strategy:

"*The achievement of the industry's growth potential will be particularly dependent on the building and maintenance of consumer demand. Ford Australia believes a biofuels industry* *based largely on market pull will be an inherently sounder industry than one based largely on production push.*" (7)

Note that the events in our mentioned example applications (transferring money, online gathering, and collaborative purchasing) are originally generated by users and in fact the users' demands directly generate the events. In contrast to market pull, there is the usual strategy of marketing, namely market push. For example, the usual cinema movie scheduling method dedicates the majority of resources to recently advertised movies. In market push strategy clients are pushed to choose from a fixed set of already provided options. Generally a push-based market supply chain involves three sections, respectively research and development, production and marketing (8).

Figure 1 contrasts these two market supply strategies. Due to the indirect relation between individual clients and producers, market push strategy usually reacts slowly to changes in users' demands. For example, in the movies market, despite all research and the competitive nature of the market, occasionally we notice that an uninteresting movie happens to be highly advertised, while a movie that has never been appreciated at cinemas gradually attracts a lot of attention through DVD distribution.

Collaborative event planning can be considered as an example of this cutting-edge marketing discipline. Later in this chapter we look into an active company in the collaborative purchasing business.

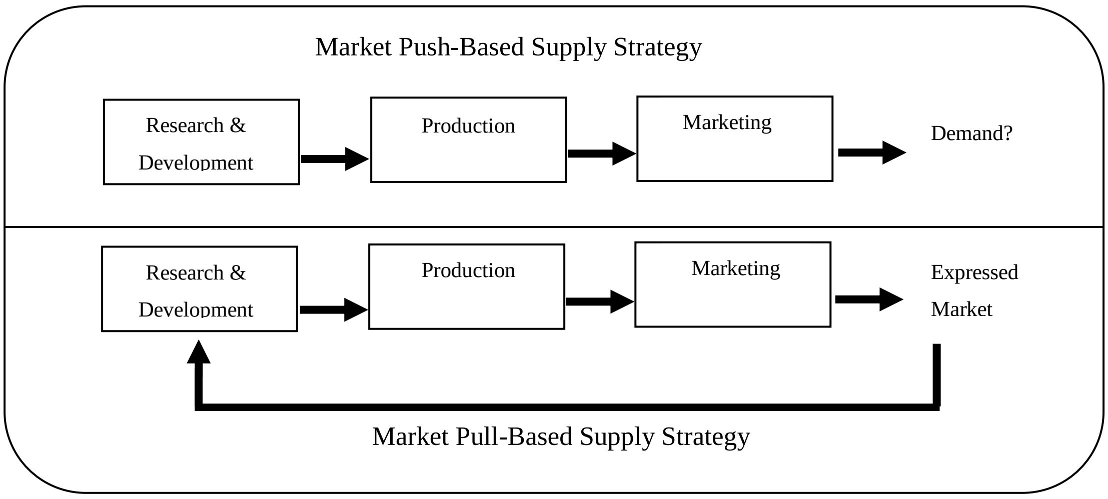


Figure 1: Market Pull Strategy versus Market Push Strategy (8)

1.3.2 Comparable Working Systems

In about the two past decades there has been an emerging amount of work towards improving the traditional discipline of event organizing, through employing web-based solutions. Early types of these works were used to serve event planners' service, for example by providing contact list or to do list managing tools. Today's online event managing systems tend to be more socially aware and combined with some kind of social networking platform. Table 1 shows a brief list of websites that are all to some degree involved in the event planning business. Regardless of the presentation and service model, most of them share the same characteristics and functionalities. For example, they allow users to create events, browse and propagate events, and attend or follow them (just become informed of the updates). Therefore, in short, we review only a few of these services.

Google Calendar	PurpleTrail.com	ilike.com
Sched.org	Phonevite.com	Localist.com
Socialzr.com	Ping.com	Flogs.com
Eventbrite.com	Anyvite.com	Myspace.com
Whichdateworks.com	Doodle.com	last.fm
Twtvite.com	Facebook.com	Evite.com
Upcoming.org	Meetup.com	Eventplanner.com
Zvents.com	Cvent.com	Eventful.com

Table 2: A list of websites involved in collaborative Event planning

- **SchoolQuote.co.uk and Collaborative Purchasing**

There are currently a number of websites that offer collaborative purchasing services in some specific domains. One of these domains is purchasing for schools. Apparently, every school in the UK has a fixed annual budget for providing certain items, from pencils and printers to energy supplies. Since schools are interested in increasing their purchasing power, they keep revisiting catalogues of large supplier consortiums, such as YPO[3] and KCS[4], and request for quote (9).

One of the active websites in the collaborative purchasing for schools is SchoolQuote.co.uk that collects updated quotes from different suppliers. An interesting feature of this website is called *storm*, which basically encompasses big purchasing events. As many schools might participate in these events, suppliers will compete with their best prices and schools can take the advantage of driven down prices (10).

- **Google Calendar and Collaborative Calendaring**

A dominant web-based calendaring service of the world is Google Calendar. It uses the iCalendar format for its calendars, which is a largely accepted calendaring format (11). iCalendar allows users to

[3] Yorkshire Purchasing Organisation
[4] Kent County Supplies

send meeting requests and tasks to other users via email or sharing files. An advantage of iCalendar is that recipients of the iCalendar data file can respond easily to the sender, or counter-propose another meeting date/time. iCalendar is independent of the transport protocols and a web server can distribute iCalendar data and publish busy times of individuals simply by using the HTTP protocol (12).

Since Google Calendar is well coupled with other free web services of Google such as Gmail and GoogleGroups, it is actually a remarkable and precise tool for collaborative planning.

▪ **Yahoo Upcoming**

A classic website that provides collaborative event calendaring is Yahoo's Upcoming.org. The main idea of Upcoming is to ease the process of organizing gatherings and of general events, for friends and acquaintances. For this reason, Upcoming intended primarily to preserve the functionalities of a social networking website: once set up on the site, the user can send invitations and add friends. In addition, users can make a list of favourite venues, as well as attending to events or watching them or simply indicating that they are "interested". Most importantly, every user is authorized to propose events. In order to do so, the user is only required to publish the event information, i.e. description, time, venue, just like a blog post. Once a user publishes an event, the people within his network will receive notification. In addition to this, the events are searchable and browsable. Upcoming also offers users reminders via email or SMS when a particular event is about to occur. More advanced features of Upcoming are:

- Content syndication features: Upcoming uses iCalendar, RSS, and GeoRSS for content
- Open API for searching or submitting event data
- A Java script powered API, which help users to post badges on their own websites, to see all their upcoming events

▪ **Eventful.com**

Some of the websites listed in Table 1 are focused on providing an online database of events rather than the social networking aspect of the process. The type of the event they may index may vary, but they usually have a conventional mechanism for publishing users' data. For example, Eventful.com is an example of such a system that indexes a fairly extensive database of events. Anybody can search for any kind of events, and registered users can also post an event. Other websites with a similar approach include Localist.com and Zvent.com. In short, Localist.com tends to skilfully take a location-oriented approach, and Zvent.com tends to be a professional and well connected place for advertising events. The following claim can be found on the Zvents website, at the time of writing this document:

"Zvents helps more than 8 million monthly users find things to do in their local area and helps thousands of event advertisers increase attendance at their events. Our network of locally branded entertainment guides reach the largest audience of people actively looking for things to do. More than 15,000 advertisers work with Zvents every month to promote a wide range of local events and activities." (13)

- **Facebook and Collaborative Planning with Social Networking Websites**

A large proportion of web-based event planning, in terms of the number of users, is nowadays achieved via social networking websites. Some of these websites are dedicatedly focused on event planning, while others only provide event planning as a feature alongside all other social networking features, e.g. Facebook.com. Facebook offers a free basic solution for collaborative event planning through the Facebook publisher. The Facebook event planning system aims to enable people within a network—such as colleagues, friends, or fans of a common subject—to organise events between themselves. Facebook's event planning system is as easy as updating the user's status, only enhanced with a calendar and some additional detail. For example, if a user wants to arrange a get-together, he can quickly publish his plan as an event, with some modifiable level of visibility, and then see who is able to attend. While Facebook's event planning solution's simplicity has resulted in its use by millions of people on a daily basis, it has also forced some consideration to its business model. In particular, in Facebook and typically any social networking website, messages including event notifications can only be passed through a pre-existing network. Note that Facebook is focused on structuring *explicit* and permanent networks of people. By having the confirmation of users on building these permanent networks, this approach serves the strategy of keeping users hanging around numerous ad-enriched inner web pages. An alternative approach could be relying on a pub/sub[5] platform, with implicit and temporary networks of people.

- **Music Industry and Event Planning**

In the past few years many music sites have branched out into the event planning business, particularly the concert industry; e.g.Last.fm and iLike.com. Their main working model is to build a knowledge base of their users' tastes (top playlists).

[5] "Publish/subscribe (or pub/sub) is a messaging pattern where senders (publishers) of messages are not programmed to send their messages to specific receivers (subscribers). Rather, published messages are characterized into classes, without knowledge of what, if any, subscribers there may be. Subscribers express interest in one or more classes, and only receive messages that are of interest, without knowledge of what, if any, publishers there are. This decoupling of publishers and subscribers can allow for greater scalability and a more dynamic network topology." (15)

- **Cvent.com**

One of the largest companies in the business of event planning is Cvent.com, which offers a variety of web-based software solutions regarding event management and web surveys. Basically, Cvent, has defined three levels of users: *Attenders*, *planners* and *suppliers*. The attenders of an event are the actual the reason for holding the event. They are all those people who have intended to be present at a same time, at a same place and for a same purpose; which also implies that they have an idea of what they are going to be presented with. Events are offered (or they are invited to) by the planners of the event. Usually an event is not completely fulfilled by the effort of attenders, in the sense that the event is demanded to be supplied with some services or products. In view of that, suppliers of an event are another major party that needs to be considered. Presumably, suppliers can manage to propose the availability and details of their services or products within a reasonable time, whenever they are requested. Nevertheless for various reasons they cannot have a very firm and predictable connection with the attenders. For example, as a supplier, the management of a hotel can constantly advertise their services and responses to any Request for Quote (RFQ); but the people that are concurrently staying in a hotel are not really forming an event, as they may not have a same purpose for being there. In fact, in the context of event management, it is necessary to see planners of events as a third party with a different view, especially when an event demands a combination of suppliers, or if the event is too occasional. In short, the goal of the planner of an event is to increase the attendance and decrease the costs. A planner is not essentially concerned about the content or substance of the event, as long as the attenders are happy with it; instead his concerns are mostly the quality of issues related to the formation and presentation of the events. Unlike suppliers, planners seek to establish and preserve relations and networks (among the suppliers and attenders). They should be able to sense the needs and priorities of groups and societies (generally potential attenders) regarding the time and other attributes of the event that they intend to plan for.

Cvent.com provides the event suppliers (such as hotels, airlines and catering services) with enhanced marketing to event planners. For event planners Cvent.com is an enhanced environment for efficiently managing their plans, choosing suppliers and informing potential attenders of their plans.

1.4 Summary

The emergence of web 2.0 technologies has had an immense impact on the infrastructure of most applications and services, by taking the increased social awareness of users into account. For example, the ability of users to share their personal calendars has reformed the electronic calendaring industry. In fact, at the top of the highly socially aware infrastructures, a whole new class of intelligent systems has been approached, e.g. the Google search engine, empowered by collective intelligence. The current project is inspired by this trend and aims to investigate the novel ways of doing things collaboratively, using collective intelligence. Accordingly, the title of the project is “Web-Based Collaborative Event Planning” , since a real-world collaboration needs to be based on precise plans. In particular, the idea of web-based CEP is to employ a collective intelligence-based planning system, while all the preferences and ideas of the users are taken into account. Subsequently, the introduction of this document discusses the idea and gives a view of some possible applications of CEP systems. Additionally, some of the existing similar systems and services are reviewed and a background for constructing the project is provided.

2 THEORETICAL FOUNDATION

T his chapter aims to set a formal definition for the main problems with collaborative event planning, as well as introducing some useful machine learning techniques for solving those problems.

2.1 Intuitions

2.1.1 Events

Event is a primitive concept with a good perceptive basis. Intuitionally it has an emphasis on instantaneous occurrence of something, by a collective of contextually similar people. Note that here the underling meaning of an event is considerably more straightforward than most significant areas of computer science, such as event-based architecture, Automata languages and Petrinets; those usually appraise events as an abstract notion that causes an alternation of some type of state and results in discharge of some response. However, here event refers simply to some human collaboration in the short term—for instance a live performance, a ceremony, a convention or a conference—by a certain group of people and possibly with some outcome. Yet it is not necessary to focus on any particular types of events, especially as this might cause some potential applications of what we eventually design to be disregarded.

We are most commonly concerned about two aspects regarding any event: the event's *attributes* and the event's *attendance*. Every event involves the attendance of some people and, typically, a higher number of attendees is a desirable factor. However, an event planner cannot be totally assured of the attendance until the event has taken place; before that, the level of attendance is only approximated based on previous patterns of attendance, invitations etc. For the sake of brevity, we shall use the term *plan* instead of *plan for an event*.

In short, an event planner aims to achieve higher attendance by carefully assigning attributes to his plan. Typically, there are two categories of attributes for events: Content-describing attributes and coordination-describing attributes.

<u>Content-describing attributes</u> deals with the content or the semantics of the event. For example, the following questions could clarify some of the content-associated attributes of the event: What is the subject of the event? What category is the event considered to fall in? What is expected to happen or be performed?, etc. Normally we deal with the content of an event through tags, category labels, title, and content-related description, probably given in text, image, video and/or external links.

<u>Coordination-describing attributes</u>**:** These attributes are mainly about the time (event calendaring) and the venue (event locationing). In other words, the coordination is supposed to address 'When?' and 'Where?' types of questions regarding the event. As we can imagine, two very similar events can happen in different times or different locations, and disagreement with the coordination results in no

attendance of either, even if the event content is desirable. For this reason we exclude the coordination-associated range of information, from the substantial information of the event. Depending on the case and the level of view, the coordination-associated information can be seen or stored in different forms. For example, one may want to deal with all times and locations, respectively by GMT and latitude/ longitude of the venues.

2.1.2 Collaborative Event Planning

Consider the familiar experience of the students of a same class who try to choose the most appropriate time for holding an extra study session. In this scenario, the teacher usually chooses a number of time options and exhaustively for each option asks students to raise their hands to indicate their availability. In practice, the elegance of this approach is the collaboration of students, or in fact the potential for negotiation, since human decisions are indeterminate in their nature. Nevertheless, this manual process fails to perform well in the case of large numbers of people, time options and events. Accordingly, one inspiration could be to automate the mentioned process of participants of a given event in order to find an appropriate time for holding that event. Note that the most appropriate time for holding an event should cover as many as possible of the participants' time preferences, and consequently it is expected to result in the highest number of attendees. Finding the maximally voted attribute of the event forms the main problem of collaborative event planning. However, the chances are that some certain attribute—between the ideal attributes of disagreed users—works for everybody. The discovery of such potentials is another interesting problem.

2.1.3 Formation of Events

As we previously indicated, *plans* are similar entities to events, except that a plan is not definite about attendance. We can usually predict the level of attendance for our plans at least to some extent. Similar to events, the attributes of a plan are associated with some content-describing attributes and some certain coordination.

Figure 2 is a brief representation of formation of events. Note that by placing ideas/interests at the base of the pyramid, we implicitly consider the variety of interests and ideas to be greater. The arrows show the direction of formation steps. Accordingly, plans are supposed to be based on ideas and interests. Predictably, we may value plans as highly as ideas and interests. This could be due to the low popularity of an idea or interest, or the fact that a particular segment of ideas/interests is not yet discovered. Events are based on the plans and are at the top of the pyramid, indicating that not every plan will actually occur. In reality, we may think of many plans, while at the end very few of them are realised. Finally, from the outcome of events we learn how to plan better. Similarly, people's interests/ideas are also influenced by events as well as plans; for example by advertising and marketing a particular plan.

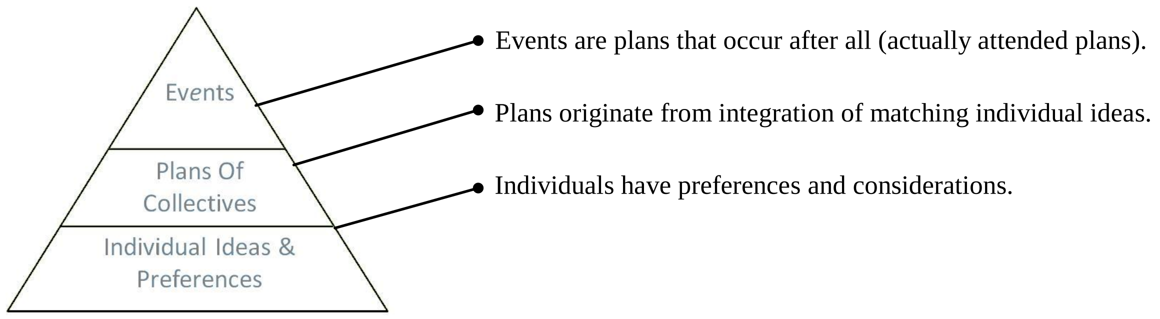

Events
Plans Of Collectives
Individual Ideas & Preferences
Events are plans that occur after all (actually attended plans).
Plans originate from integration of matching individual ideas.
Individuals have preferences and considerations.

Figure 2: The idea of forming events from ideas

2.1.4 Independent Attendance Assumption

Here we make a fundamental assumption that is given if a pair of an event-user, the user decision—regarding whether he should attend the event or not—depends only on his preferences and the attributes of the event (which means he does not need any other information except the attribute vector of the event). In other words, we hold the independent attendance assumption for users. To some degree, independent attendance of users seems rational, as events are supposed to serve the ideas and interests of people, including their preferred coordination and preferred content. Nevertheless, one may argue this hypothesis according to the fact that, in many cases, people do care about "Who else is coming?", as well as about the attributes of events itself. Yet, for the sake of simplicity, we hold to this assumption. According to this assumption, to design more attractive plans, it is only required to assign the most popular attributes to our plan.

2.2 Formal Problem Setting

This section starts by giving an insight into perhaps the most favourable problems of collaborative event planning, after which it discusses each of these problems individually.

2.2.1 Two Main Problems

As we previously mentioned, in our context a plan—or more precisely "a plan for a group"—is supposed to assign coordination to specific tasks. In particular, the plans discussed are statements with a template such as:

{Doing *x* at coordination *t*}.

For the sake of simplicity, we restrict the coordination to *time* that is simultaneous—or acceptably simultaneous—to be coordinated. Yet we could expand this definition by adding some other considerations to the above template, e.g. at venue *v*, with attendance preconditions *y*, with the costs of *c*. Yet, for the sake of simplicity, we stick to the given template.

Besides plans there are individual preferences, which following more closely the template below:

{User *j* would like to attend doing *x*, at time *t*}

As dealing with uncertainty is inevitable in this context, we also assign a probability to user preferences and develop the following template for statements of preferences:

{User *j* would like to attend doing *x*, at time *t*, with probability *p*}

Assuming that we have a data set of individual preferences, we would like to draw out plans with the highest expected level of attendance. This involves finding specific tasks like x^* and specific time like t^*, while the expected value for the number of users who would like to attend doing x^* at t^* is maximum. We shall refer to this problem as *Maximal Coverage Planning*.

Once we have a certain plan, we might be interested in finding users who are likely to attend our plan. Reasons might include advertising with smart targeting systems, or just presenting better browsing and information retrieval systems. In any case, we are led to another significant problem, which we call "attendance prediction problem", and state it as "for any given pair of plan and user, compute the probability of the user attending to the plan".

2.2.2 Maximal Coverage Planning

Let U, E and $\hat{E}$ respectively be the set of users, possible plans and observed events. By definition, $\hat{E}$ is the subset of plans that have actually taken place or, more technically, where observation regarding their received attendance is available. Besides, we denote the set of all users who have attended to event $\hat{e}$ by $A(\hat{e})$. Moreover, we denote the probability of user *u* attending the event based on plan *e*, by $\Pr(\bar{e} \cong e | u \in U)$, or in short: $\Pr(e|u)$. We relax this notation for events as well and set $\Pr(\bar{e}|u) = 0$ for all users, except those who have been present at the event $\bar{e}$, where $\Pr(\bar{e}|u)$ is set to be equal to 1. Finally, $\|A(\hat{e})\|$ and $\|A(e)\|$, respectively, signify the total number of users who have attended to event $\hat{e}$ and the popularity of plan e. We define the popularity of a plan as the expected value of the total number of users who would attend the event based on that plan. By definition, we have:

$$\|A(e)\| = \sum_{u \in U} \Pr(e|u)$$

The problem of maximal coverage planning or, in other words, designing the most popular plan can be defined as below:

- *For a given set of users U and their preferences, find a plan like e, in which* $\forall\, e' \in E$*:*

$$\|A(e)\| \geq \|A(e')\|.$$

Expectedly, the challenge is to compute $\Pr(e|u)$s, as typically they are numerous and not explicitly given. Occasionally, users have preference profiles, which can help us with computing the probabilities. Assuming that the interest profiles of users are presentable by a real valued vector like $profile(u_i)$, we define a profile-plan distance function like $d^{pp}: U \times E \rightarrow [0,1]$. For example, $d^{pp}(u_i, e)$=1 and $d^{pp}(u_i, e)$=0, respectively, means that user i is fully disappointed and completely unhappy with the substance and coordination of the event e. In case of profile-based maximal coverage planning, our task is to design plans like e, while $\sum_{u_i \in U} d^{pp}(u_i, e)$ is minimized.

On the other hand, providing that observation regarding the outcome of previous plans is available, it is possible to take a learning approach and compute the popularity of a plan, based on the estimated liking of users. We name such cases History Based Maximal Event Planning, where users have not explicitly indicated their interests. History Based Maximal Coverage Event Planning can be denoted formally as below:

- *Let* $\Omega = \{\langle \bar{e}, u\rangle | u \; has\; attended\; to\; \bar{e}\}$ *be a set of observations (history). Estimate the* $Pr(e|u)$ *for any given pair of plan and user.*

Traditionally, such problems involve defining a similarity function—or, alternatively, a distance function—in the item's feature space. In particular, we could define a mapping as $\sigma: E \times E \rightarrow R$, which assigns a real value to every given pair of plans in order to represent their similarities. Depending on the context and the exact definition of plans, this function can be constructed. For example, one idea is to assign a weight value to each feature dimension of the plan and compute the inner product of two weighted feature vectors of plans. Once we are able to compare plans precisely, we can measure the popularity of an idea or plan. Subsequently, a naive estimation of $\Pr(e|u)$ can be the weighted average of available observations over the probability of attendance:

$$\Pr(e|u) \cong \frac{\sum_{(e',u)\in\Omega} \sigma(e', e).\Pr(e'|u)}{\sum_{(e',u)\in\Omega} \Pr(e'|u)}$$

The whole of the next section attends more extensively to this problem . For now, we introduce a slightly more realistic version of the problem. In this variant, we assume that event planners are concerned about some restrictions, e.g. budget. Here is a simple setting for a family of these problems: Let us imagine that for any given plan we are able to compute the cost of the event that might be based on that plan. More clearly, we suppose to have a mapping like $c: E \rightarrow [0, +\infty[$, from the set of all possible plans to the range of all possible costs (here assumed to be non-negative real numbers). Now, assuming that we cannot spend more than β for holding events, we would like to find a set of consistent plans such as $(e_1, e_2, .. e_k)$, where $\sum_{i=1}^{k} A(e_i)$ is maximized, while:

$$\sum_{i=1}^{k} c(e_i) \leq \beta.$$

Note that the consistency of a set of plans implies that they all can be held, without affecting the attendance of each other. We name this problem Budget-concerned Maximal Coverage Planning. Although this document does not aim to focus on this problem, there are specific domains of applications that might involve this version of CEP, e.g. in the case of collaborative money transferring, planners would be concerned about different fees and some limited overall budget.

2.3 Useful Techniques

This section focuses on few possible approaches for solving the problem of maximal coverage planning. These approaches are considered to be in the field of Information Retrieval; the keyword for such techniques is “filtering”. There are three main types of filtering: Content-based filtering, collaborative-based filtering and content-collaborative-based filtering. Computing the probability of a given user attending a given plan based on content-based filtering, techniques would work as follows: initially users indicate their preferences explicitly via a personalized *content filter*—or what we refer to as the preference profile—and subsequently recommendations can be brought to users via a matching process. This method is pretty straightforward and well understood, however it brings with it a tricky issue: In practice, users hardly ever tend to explicitly make any complete and precise preference profile. In fact, usually the only chance is to estimate as cleverly as possible the fittingness of a given plan for a collective of users. For this reason, this study focuses on prediction methods and learning techniques, as opposed to pure content-based filtering. Nevertheless, combining content-based filtering and collaborative filtering will be discussed further in Section 2.3.2.

2.3.1 Collaborative Filtering

Collaborative Filtering (CF) is technology complementary to content-based filtering, yet it is not based on the content of items or users' profiles. Suppose that we have a set of users and items in which users might have rated or selected some of the items. Collaborative filtering aims to predict the behaviour of users in the future, by looking at the previous user-item transactions. For instance, a CF solution could involve discovering the similarities of users and uncovering the patterns of their behaviour and applying these patterns in case of unobserved transactions. Quintessentially, CF helps us with making recommendations over those items that are expected to be chosen or highly rated. CF is a promising and well-liked technique in many areas of Computer Science. Particularly, where we want to extract knowledge from large user-item types of databases and at the same time we do not fully understand the content of items or the preferences of users, e.g. online booksellers, large retailers. Today, there are many customizable packages of CF freely available online, e.g. Fast Maximum Margin Factorization in Matlab (14) and C/Matlab Toolkit for Collaborative Filtering (15).

2.3.1.1 Formal Problem Setting of Collaborative filtering

Imagine U and X respectively to be the sets of users and items and that observations are available for some user-item pairs (u, x), where $u \in U$ and $x \in X$. Also, with observing (u, x) we assume that user u has implicitly indicated her interest to item x. These observations could be generated by shopping or clicking of users. In these cases CF tends to assign a probability value to any given unseen pair of user-item. In fact we would like to learn $\Pr(u, x)$ or $\Pr(x|u)$ if there is a particular user that we are interested in. Recall that we set a similar notation ($\Pr(u, x)$) show the probability of user u attending plan e. In fact, this setting of collaborative filtering is naturally fitted to our collaborative planning problem. It is only required to see the plans as items and the history of attendance as the observed transactions.

Sometimes CF deals with explicitly rated items. In such cases observations are in form of (u, x, ψ), where $\psi \in \Psi$ are ordinal values (namely ratings) that a user can assign to an item, explicitly indicating her interest. Predictably, the goal is to forecast the ratings. More precisely we would like to learn the mapping $\mathrm{g}: \mathrm{U} \times \mathrm{Y} \rightarrow \Psi$, by computing the conditional probability of $\Pr(\psi|u, x)$, i.e. how likely it is that the user u, rates the item x, with ψ. Similar to the previous setting, if there is a particular user (active user) who we tend to make recommendation for, we need to learn the probability $\Pr(\psi, \mathrm{y}|\mathrm{u})$. Note that $\Pr(\psi, x|u) = \Pr(\psi|x, u).\Pr(x|u)$. Needless to say, making good recommendations over plans could result in increase in attendance.

2.3.1.2 Approaches of Collaborative Filtering

There are three main approaches to perform CF: memory-based (or neighbourhood) approach, model-based approach, and hybrid approach, which involves combining memory-based and model-based techniques. Here we briefly exemplify the approaches.

Memory-Based Collaborative Filtering

The earliest attempt of CF [16] is considered to be memory-based collaborative filtering, otherwise know as Neighbourhood Collaborative Filtering (NCF). NCF is the traditional way of dealing with the problem, and is still successfully employed by a wide range of applications, perhaps most famously by Amazon.com[17]. Amazon usually tends to categorise the set of users by a distance measure, based on dissimilarity of users' ratings, and consequently perform a nearest neighbour search to predict the unobserved ratings value. Alternatively, an NCF-based approach might define a distance measure for the set of items rather than for the users and, accordingly, make a recommendation to a given user, by searching for a *near* item to the set of highly rated items of that user. While different studies build different distance measures, they usually share the same foundations. An example of neighbourhood approach is a family of algorithms called *slope one algorithms*. In the original proposed paper (18), slope one uses a single parameter regression ($f(x) = x + b$) from one item's ratings to another item's ratings, while this single free parameter is the average ratings difference between the two items' ratings. Despite the simplicity of this method, slope one is usually regarded as a potentially promising choice, which provides reasonably good results.

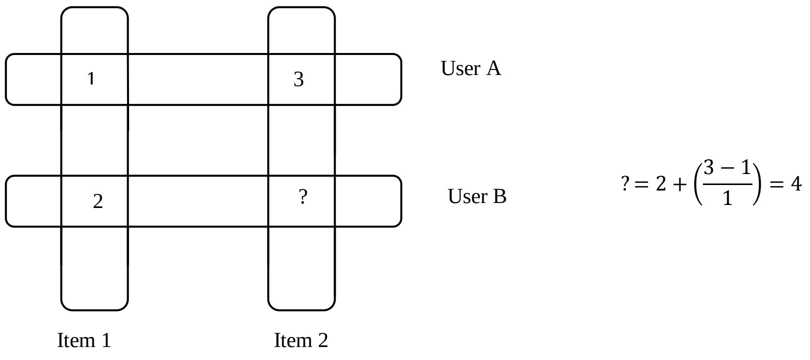


Figure 3: The main idea of the slope one algorithm. The missed rating is the average rating difference between the two items (18)

To describe the slope one algorithm more precisely, we use the following notations: indices are over items. The ratings from a given user are called an evaluation and χ denotes the set of all evaluations (training set). r_i is an incomplete array of ratings of user u_i and the subset of all items that are rated in r_i is $S(r_i)$.The number of all elements in a set S is $card(S)$ .The average of ratings of user u_i is denoted by $\overline{r}_i$, and $S_i(\chi)$ is the set of all evaluations r_i, in such that they contain item i ($i \in S(r_i)$).

Given two evaluations r_i and r_j, we define the scalar product $\langle r_i, r_j \rangle$ as $\sum_{i\in S(r_i)\cap S(r_j)} r_i \cdot r_j$. Prediction $\mathcal{P}(r_i)$ represents a vector where each component is the prediction corresponding to one item.

Given two evaluation arrays r_i and r_j the philosophy of slope one algorithm is to search for the best predictor of the form $f(y) = y + b$, to predict the ratings of a user according to other users' ratings by minimizing $\sum_i (r_i + b - r_i^{'})^2$. Setting this derivation equal to zero we get $b = \frac{\sum_i r_i - r_i^{'}}{n}$. In other words we propose to set the constant b to be the average difference between the two arrays. Accordingly, in the slope one scheme, given a training set χ and any two items i and j with ratings r_i and r_j respectively in some user evaluation r (annotated as $r \in S_{i,j}(\chi)$), we identify the average deviation of items *i* with respect to items *j* as: (18)

$$dev_{i,j} = \sum_{r\in S_{i,j}(\chi)} \frac{r_i - r_j}{card\left(r \in S_{i,j}(\chi)\right)}.$$

The symmetric matrix defined by $dev_{i,j}$ can be computed once and updated quickly when new data is entered. Given that $dev_{i,j} + r_i$ is a prediction for r_i given r_j , a reasonable predictor might be the average of all such predictions:

$$\mathcal{P}(r)_j = \frac{1}{card(R_j)} \sum_{i\in R_j} dev_{j,i} + r_i$$

Where $R_j = \left\{ i \middle| i \in S(r), i \neq j, card\left(r \in S_{i,j}(\chi)\right) > 0 \right\}$ is the set of all relevant items. For a sufficiently dense data set (almost all pairs of items have ratings or $card(S_{i,j}(\chi)) > 0$) we can approximate the above calculations as most of the time $R_j = S(r)$ for $j \notin S(r)$ and $R_j = S(r) - \{j\}$ when $j \in S(r)$. Since $\bar{r} = \sum_{i\in S(r)} \frac{r_i}{card\,(R_j)} \cong \sum_{i\in R_j} \frac{r_i}{card\,(R_j)}$ for most *j*, we can simplify the prediction formula for the slope one scheme to:

$$\mathcal{P}^{s1}(r)_j = \bar{r} + \frac{1}{card(R_j)} \sum_{i\in R_j} dev_{j,i}.$$

Note that our implementation of Slope One does not rely on how the user rated items individually, but only on the user's average rating and significantly on which items the user has rated (18).

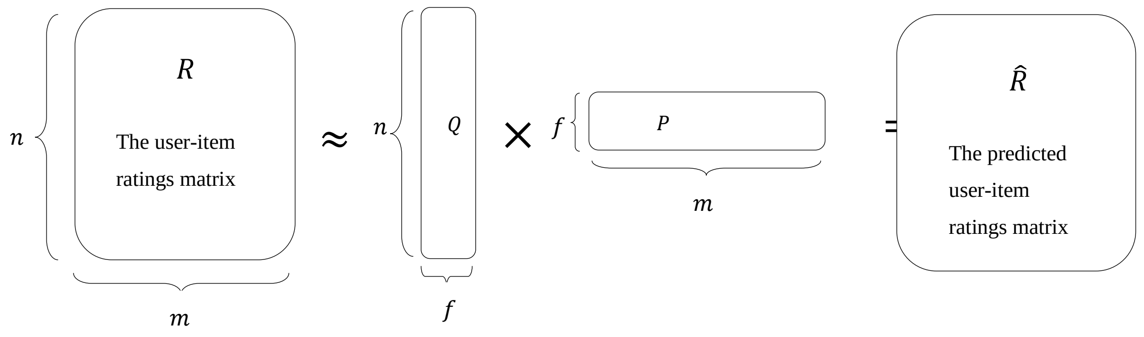
Model-Based Collaborative Filtering

Model-based collaborative filtering is usually a more sophisticated approach to CF that presents item recommendation by developing a model of user ratings. Algorithms in this category take a statistical approach and portray the collaborative filtering process as computing the expected value of a user prediction, given his/her ratings on other items. The model building process can be done by different machine learning algorithms, such as: Bayesian network, Clustering (determining certain classes and estimating the probability that a particular user is in a particular class *C*), rule-based algorithms (to find association between co-liked items and then generate item recommendation based on the strength of the association between items). Here we quickly review a highly attended approach to building a probabilistic user-item rating model that is called matrix factorization (19).

Matrix factorization models map both users and items to a joint latent factor space of dimensionality *f*, such that ratings are modelled as inner products in that space. Accordingly, each user u is associated with a vector $p_u \in \mathbb{R}^f$ and each item is associated with a vector $q_i \in \mathbb{R}^f$. More precisely, a matrix factorization model predicts the ratings by the rule (20):

$$\hat{r}_{ui} = q_i^T p_u$$

Where $\hat{r}_{ui}$ denotes the predicted rating that user u gives to item i , and p_u and q_i^T are the decomposed factors of initial user-item rating matrix, respectively associated with users and items. Figure 4 illustrates matrix factorization model. The beauty of matrix factorization-based CF is that the latent factors are sometimes insightful, in the sense that they can imply a particular attitude or tastes of users (19). In the case of collaborative planning, these factors could be particular subjects or categories of plans. A typical drawback of model-based CFs is that they do not cope well in dynamic cases, where there are high frequencies of newly entered items or users.



n
R
The user-item ratings matrix
m
$\approx$
n
Q
f
$\times$
f
P
m
=
$\hat{R}$
The predicted user-item ratings matrix

Figure 4: The main idea of matrix factorization model-based CF.

Note that $R_{n\times m} = Q^T{}_{n\times f} P_{f\times m}$ and $mn \gg nf + fm$.

2.3.2 Combining Content and Collaborative Filtering

This section introduces collaborative content filtering, which is our final candidate approach for predating the popularity of plans. First we explain the idea of tag-based collaborative filtering which can be useful in CEP if plans are tagged, e.g. collaborative cinema scheduling where movies are tagged. Next, we set up a general framework, which can be applied in any type of collaborative event panning, after customization.

2.3.2.1 Tag-Based Collaborative Filtering

An interesting way of connecting content-based filtering methods with CF is tag-based collaborative filtering (21). Traditionally we assign tags to our data in order to improve the process of content retrieval and categorization. Tagging mechanisms have been highly re-evaluated since the emergence of Web 2 and user-generated content-based systems, particularly through social tagging[6]. To some extent tags could also tell about the users who have assigned them, in the sense that an item's tag could represent *why* the user likes that item. As neighbourhood CF involves a stage of user similarity search, we can study their tagging behaviour in order to improve our user similarity search. The stages of this process are:

- If a user likes an item then he might tag that item with his favourite tags. These tags are supposed to tell what that item means to him.
- The system computes the user's similarities based on their commonly tagged item and the similarity of tags they have chosen.
- For a new given item we calculate the predicted rating based upon the user's similarities computed in last step.
- The system recommends an item to the active user, based upon the predicted ratings.

The similarity of 2 users due to their chosen tags for the same item is computed by:

$$sim_{ccf}(A,B) = \frac{1}{n}\sum_{k=1}^{n}\{sim(T_{A\to K}, T_{B\to K}) + 1\}$$

where *n* is the number of commonly tagged items between the users *A* and *B*. $T_{A\to K}$ are tags that user *A* has assigned to the item *k* and $T_{B\to K}$ is the tag vector that user *B* used for commonly tagged items. $sim(T_{A\to K}, T_{B\to K})$ is the similarity between two sets of tags that users *A* and *B* have assigned to item *K*, and can be calculated through the cosine similarity of the two-tag vector, as below:

$$sim(T_{A\to K}, T_{B\to K}) = \frac{T_{A\to K} \cdot T_{B\to K}}{|T_{A\to K}||T_{B\to K}|}.$$

Once user similarities have been calculated, the predicted score for some user *A* for an unevaluated resource *x* is calculated by the following:

$$score(A,x)_{prediction} = \frac{\sum_{i=1}^{n}\{sim(A,S_k) \times score(S_k,x)\}}{\sum_{k=1}^{n} sim(A,S_k)}$$

[6] Social tagging is basically about the collaboration of users for assigning natural language words as metadata to a resource, along weighting them due to their duplications and significance. (24)

where *n* is the number of other users. Essentially in this scheme all other users' scores, weighted by their similarity, are averaged to predict *A's score for x.* Finally, this predicted score helps us with estimating the popularity of a given plan, in a tag-based framework.

2.3.2.2 A Framework for Joint Kernels

For a number of reasons, the tag-based CF does not sound the most appealing framework for dealing with collaborative event planning. Here we set up a framework that can be adopted in almost any case of collaborative event planning[7]. As mentioned before, $e \in E$ denotes plans and E is the set of all possible plans, or the *plans' space*. In order to design a joint framework—which would accept both content-based and collaborative-based information filtering techniques—we define a mapping like $\Psi: U \times E \longrightarrow \mathbb{R}^D$ to extract *D* features from user-plans pairs. This assumption—that we can uniquely map any user-plan pairs to a real vector space of dimension *D*—might be not very trivial. In fact, finding such a representation is the classic challenge of many machine learning problems. Soon, we propose an advantageous method to build this mapping. Once we have Ψ, we define a family of functions *F* that are linear in the chosen feature mp via:

$$F(u, e; \vec{w}) = \langle \Psi(u, e), \vec{w} \rangle$$

Where $\vec{w} \in \mathbb{R}^D$ are weight vectors that needed to be learned and $\langle , \rangle$ is the inner product. To predict whether a user would attend or not attend at an event (based on the given plan), we compare the result of the above product, with an attendance threshold, denoted by θ_δ, where δ is a satisfactory probability that we set ourselves. Finally, the prediction function works as follows: if $F(u, e; \vec{w}, \theta)$ is greater than or equal to θ_δ, we predict that user *u* would attend the plan *e*. More formally:

$$F(u, e; \vec{w}, \theta) = \langle \Psi(u, e), \vec{w} \rangle \ \geq \ \theta_\delta \ \Leftrightarrow \Pr(e|u) \geq \delta$$

Where $\Pr(e|u) \geq \delta$ is indicating that the probability of *u* attending at plan *e* is more than δ. Alternatively, by setting a fixed attendance threshold, we would have a simpler scheme. Attendance threshold θ and weight vectors $\vec{w}$ can be estimated through a range of Machine Learning techniques, particularly by studying the previously held events.

As promised earlier, we now propose a method for building the Ψ plan representing mapping. For this aim, we assume that a mapping like $\Lambda: E \longrightarrow \mathbb{R}^G$ exists, which can perfectly describe every plan in the

[7] This framework is highly inspired by "A Joint Framework for Collaborative and Content Filtering", a work of J. Basilico and T. Hofmann. (26)

plan space and a mapping like $\Phi: U \rightarrow \mathbb{R}^H$ which can perfectly describe the user's preferences. Note that the perfect description of plan should cover every piece of information that a user might need to know about a plan, e.g. the time of the event; venue (or any geological information); tags (if it is applicable); etc. Also, the perfect description of a user's preferences should include all his priorities, e.g. preferred times; preferred content; etc. In this manner, we have taken the content (user's interests and preference profiles) into account. Now to construct the joint feature maps of users-plans we apply the tensor product of user feature maps ($\Phi: U \longrightarrow \mathbb{R}^H$) and plan feature maps $\Lambda: E \longrightarrow \mathbb{R}^G$. By this we mean a two-stage process of combining every dimension of Λ multiplicatively with every dimension of Φ to get $\Psi(u,e) = \Lambda(u) \otimes \Phi(e) \in \mathbb{R}^D$, where $D = G.H$. This method of constructing Ψ has an apparent computational advantage, since we can compute the inner product as:

$$\langle (u,e), (\acute{u}, \acute{e}) \rangle \equiv \langle \Psi(u,e), \Psi(\acute{u}, \acute{e}) \rangle$$

In fact, the elegance of this discipline—at least in theory—is that we can independently design kernel functions K_e for users and K_u for plans, and combine them multiplicatively to define a joint kernel.[8]

To give an idea of such a trick, we exemplify a few possible kernels:

- Identity kernel: The simplest kernel function is the diagonal kernel matrix of identity features of plans and users (K_u^{id}, K_e^{id})
- Attribute kernel: These kernels can be built in order to give an explicit representation of plans and users. For users these attributes may correspond to their profile information such as age, gender, location, category of interests, and for plans they may encode the time of the event, the venue, and the content of the event (K_u^{at} , K_e^{at})
- Correlation Kernel: This kernel is intended to employ collaborative filtering as well as content filtering. To do so, we can define a kernel matrix C based on a correlation measure between users, e.g. via co-attendance of the users.

Due to the scope of the project, we end this section without getting into detail of any particular example. An implementation of this framework is considered as a further work of the project.

2.3.3 Supervised Planning

One thing that could be an issue for a web-based collaborative event planning is that of measuring the trustworthiness of users' promises. Note that so far we have assumed that once a user indicates his attendance we consider this claim in computing the popularity of the plan. However, it is possible that

[8] A brief review of kernel trick is given in appendix I.

for some reason a user does not really turn up at his (supposedly) attended event. As we previously mentioned, typically events cannot be held unless some certain level of attendance is guaranteed. For this reason, we need to take the trustworthiness of the users into account, and we propose a very simple supervising system. In this scheme, we consider a trustworthiness factor for every user and, each time he attends an event, we update this factor. In particular, we can assign +1 each time the user keeps his promise (regarding attending to the events) and assign -1 each time the user breaks his promise (i.e. when he does not turn up at the events that he was supposed to attend). In this manner we compute the trustworthiness of a user based on the average of all his previous scores:

$$w_i^t = \frac{\sum_{j=1}^{t} s_i^t}{t}$$

$s_i^t = +1$ if user u_i has kept his *t*h promise (regarding attending some events) and $s_i^t = -1$ if u_i has not in fact turned up for the event *t*h. Where w_i^t is the trustworthiness of the user u_i or weight of his opinion.

2.4 Summary

In this chapter, we looked into some theoretical aspects of collaborative event planning as an abstract problem. Accordingly, we introduced the main entities of our project, events and users and discussed the mechanism of formation of events from ideas and opinions. Next we set the problem of maximal attendance coverage, which is simply finding the most—potentially—popular plan. One issue with computation of the most popular plans is that normally not all the users' preferences are available. Respectively, the main body of this chapter corresponded to the possible approaches of estimating the users' preferences and popularity of plans. In particular, we discussed some methods of collaborative filtering and content filtering. Furthermore, we proposed a joint framework for both collaborative and content filtering of plans. An advantage of this framework is the option of employing kernel trick. Finally, we discussed the issue of trustworthiness of users' promises in collaborative planning and we suggested a simple discipline for weighting users' preferences.

3 BUILDING A SIMPLE PROTOTYPE

As the empirical work of the project, a simple prototype model of a web-based collaborative event planning system is implemented and launched at www.remerger.com. This chapter consists of a methodological explanation of how *remerger.com* is designed and built.[9] Accordingly, the first part of this chapter aims to review the idea and expected functionalities, in order to elicit and analyse the requirement of remerger.com. In the second section, we outline a naive but concrete model. Finally, a selected collection of comments on the actual code is given, which is more of a manual documentation.

3.1 Technology

First of all, remerger.com is seen as a web service, in the sense that it is self-contained and there is no immediate need to think of any API or software on the user side, apart from a browser. In terms of the mythology, the object-oriented paradigm is followed. Respectively, the major part of this chapter consists of UML[*] diagrams, i.e. use case diagrams, state diagrams, sequence diagrams and class diagrams [10]. In terms of software used, we have employed Lamp, the free open-source solution stack that is a combination of Linux (operating system), Apache HTTP Server (web server), MySQL (database software), and PHP (Hypertext Preprocessor), as the principal components.

3.1.1 A General Overview and the Scope

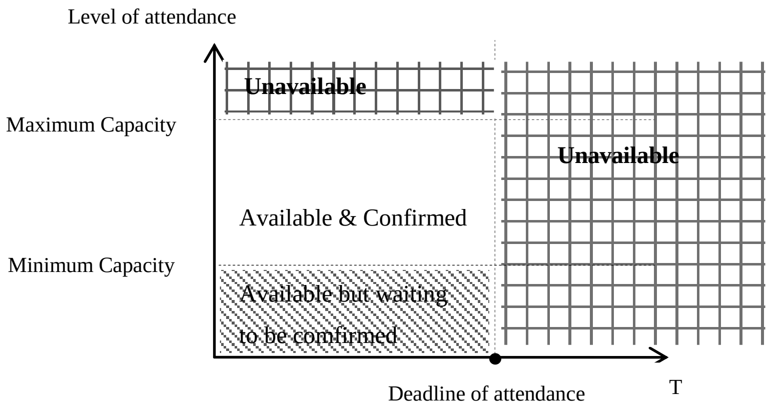
We are going to design a website that allows registered users to post a specific type of objects, so-called *events*. Events—or more precisely plans for events—include some ideas of the user who generates it, regarding collaboration with other people. Primarily, an event has a title, category and description, and a particularly specific time (marked in a calendar). The website promises to help users with collaboration by broadcasting these events' ideas. Mainly, once a user likes the ideas suggested in an event, he can join the event. So far, the whole idea is that of a simple blogging platform, if we see events as blog posts and joining an event as liking a post. Yet it is our intention to consider a novel enhancement in the system, that is where other users can also add their suggestions to the event. This is what we shall refer to as collaborative planning and, particularly for this project, collaborative calendaring. Collaborative calendaring of an event is to find the time for holding the

[9] www.remerger.com consists of roughly 3,500 lines of codes. The codes are submitted to University of Bristol, as an attachment of the project.

[10] UML is a highly popular collection of standard model of diagrams, quintessentially used in object-oriented software engineering. UML is well developed to describe both behavioural (dynamic) and structural (static) aspects of software, including web-based applications (source: www.uml.org) .

event in calendar where the largest number of attenders would be in agreement. Furthermore, we would like to equip every posted event with a graphical status representation. This graphical status representation is supposed to show how likely it is that the event reaches its targeted level of attendance by the time of the deadline. More precisely, a list of our favourable features would be as follows:

Everyone has the ability to register, log in and log out. A user who is logged in has the ability to: Post events, search for events, join/opt out of and comment on events, as well as to complete searches. A registered user can receive recommendations for available events. An unregistered user can only browse the events, comments and categories. As previously mentioned, in addition to title and descriptions an event should point to a specific time/times of a calendar. Note that it may be the case that the originator of the event has not managed to fix a certain time for his idea and wants people to elect their preferred time, from a range of time options. For this reason, an event can offer multiple time options. Moreover, we assume that the originator of the event has some consideration regarding the level of attendance in his mind. In particular, he probably seeks a minimum level of attendance, and at the same time he cannot serve more than a certain number of clients. We call these bounds, minimum attendance and maximum attendance. Once an event reached its targeted number of attenders, then the status of the event would be changed to *confirmed*. Predictably, joining an event is not possible until it has already reached the upper bound of attendance, or the maximum number of attendance. Finally, every posted event has a deadline or expire date. After this expire date, the status of the event will be unavailable and therefore would not be recommended to any user or presented as a search result. There are some other possible statuses for events in addition to 'available', considering the event's deadline and the proportion of the number of joined users of the number of total required level of attendance or target of an event (Figure 5). As previously mentioned, we would like to add a little fancy feature to the system, which is a graphical representation of the statuses for each event.



`

Figure 5: Different Statuses of events

We could envision many of today's typical characteristics of Web2 services, developed within CEP, e.g. user-centric presentation, interpretability and open standards and, in general, ease of information sharing. Nevertheless, we restrict the scope of our project to the given attributes.

3.1.2 Actors and Use Cases

Due to the given discussions, we design three levels of users in the systems, namely *registered user*, *unregistered user* and *administrator (admin)*. The administration level is supposed to be able to use all methods of registered and unregistered users. This is the usual fashion, in most simple websites. Figure 6 summarizes all the available use cases for unregistered users. Obviously, users can register in addition to retrieving events' information. Unregistered users can view the list of all events, the list of all events in a specific category, and the list of all comments on every event. Besides—and somehow unusually—unregistered users can comment on events.

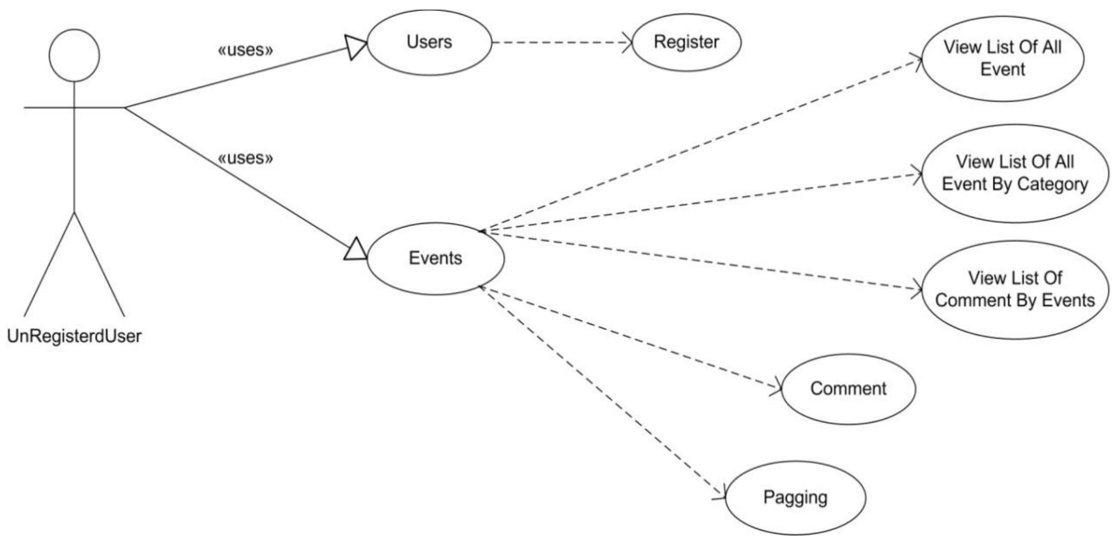


Figure 6: Use cases of unregistered users

Lastly, the paging use case offers a better presentation of the event's information. Many programming languages—including PHP—offer the paging method for a better presentation of the search results. Note that when the number of records in a table is large, then it is not good to display all the records on one page and require visitors to scroll down in order to see them all. This will also have drawbacks regarding the loading process of records. Paging simply breaks all the collected records into different pages with a fixed number of records per page. Using paging, we need to provide navigational links indicating the previous and next pages.

The next level of users (or actors) is registered users. Primarily, they should be able to log in, log out and have the same access methods as unregistered users. Additionally, registered users have the authority to create events, join events and opt out of events that they have already attended. Furthermore, registered users receive personalized recommendations for available events. This would be completed by an automated agent actor of the system. Later on we explain more about our automated recommender agent (Figure 7).

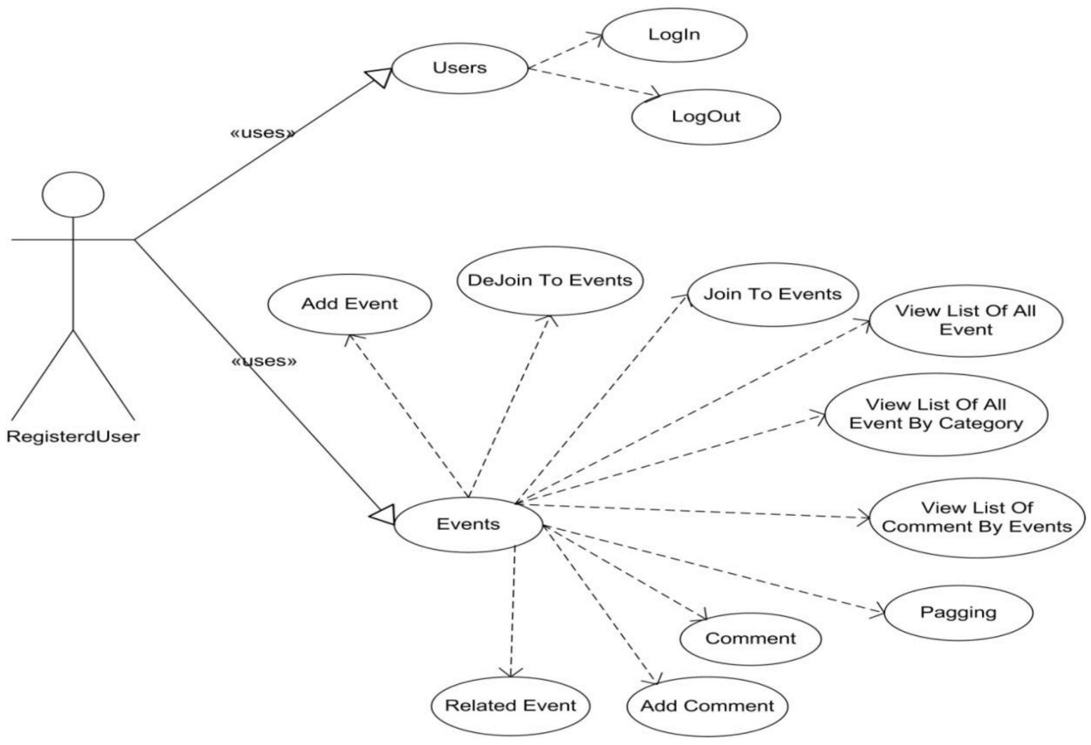


Figure 7: Use cases of registered use

As we mentioned before, once a user logged in with an administration username/password, he can have access to all the methods of user class and event class (Figure 8).

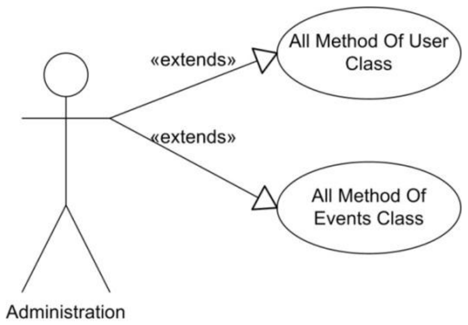


Figure 8: Use cases of admin

Recall that we have previously pointed to users and events as the main entities of our system. In this section we also specify other classes of the system.

3.1.3 Functional Requirements

The following is the list of functionalities that are considered for our initial prototype:

- Get a list of all events (with fixed number of events in each page)
- Get a list of all events of any particular category
- Get events by searching their title/event ID
- Log in, log out and register
- Add events (for logged users)
- Join —and opt out of—events (for logged users)
- Get the list of events that a user has joined by the user-ID (for logged users)
- Add category (once the user is logged in)
- Get recommendations for events or a list of related events (for logged users)
- Get the profile of information of users by user name
- Add comment (for logged-in users)

3.2 Design

Based on our requirement analysis, we now design our prototype. This section tries to explain both behavioural and structural aspects of our design, with the help of UML diagrams. Usually, sequence diagrams and state machine diagrams are two tools that can describe the behaviour of the system, and the structure of the system is explained by class diagrams.

3.2.1 Sequence Diagrams

In order to display interaction of the user and the system, sequence diagrams are used. Sequence diagrams consist of vertical lines of sequences and horizontal arrows with a message written above them. Solid horizontal arrows with full heads are synchronous calls; solid arrows with stick heads are asynchronous calls; and dashed arrows with stick heads are return messages. Activation boxes, or method-call boxes, are opaque rectangles drawn on top of lifelines to represent that processes are being performed in response to the message (ExecutionSpecifications in UML) (22).

According to the previously mentioned list of functions, we now review thirteen sequence diagrams.

- User register sequence (Figure 9)

It is important to make sure that the user's email address does not already exist in the database. Beside this, the input data should be in valid type. Hence, there are two exception messages seen for this sequence.

- User log-in sequence (Figure 10)

A registered user can log in once he correctly inputs his user name and password. Apparently, four sequences can be seen for the log-in process of user log-in, i.e. log in (initial state of the sequence), check username, check password and user set session. Following a successful log-in, the log-in session of the user would be set 'true'.

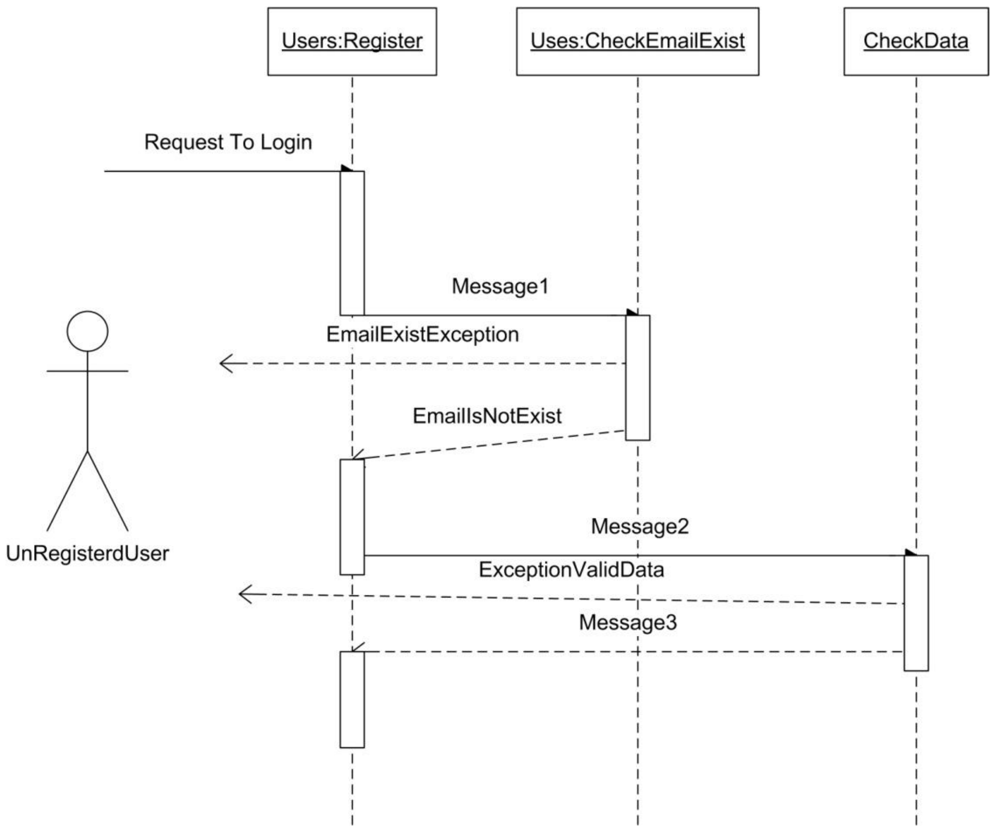


Figure 9: User register sequence

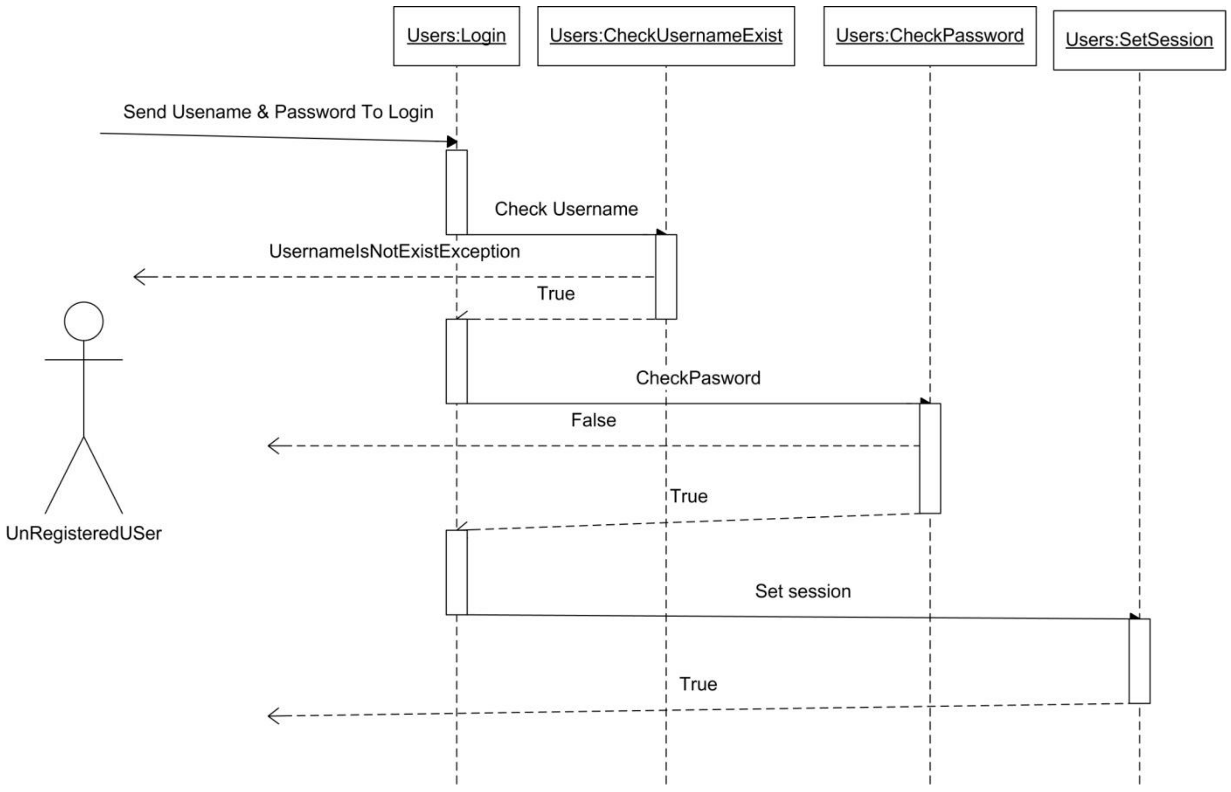


Figure 10: User log-in sequence

- User log-out sequence (Figure 11)

The log-out process includes checking the existence of the username and results in unsettling the user session to 'true'.

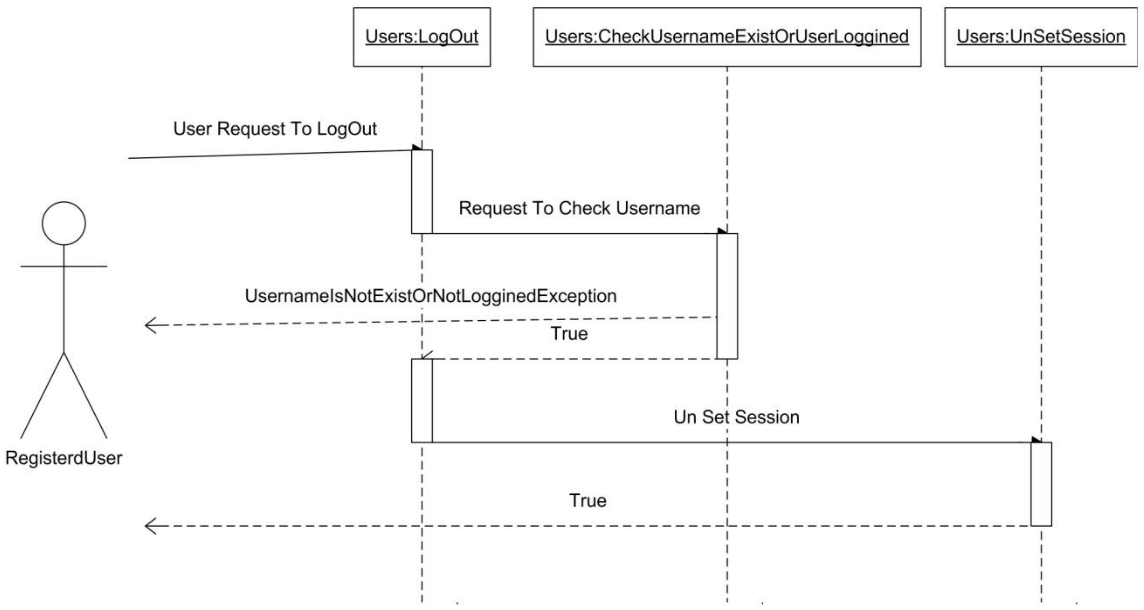

Users:LogOut
Users:CheckUsernameExistOrUserLoggined
Users:UnSetSession
User Request To LogOut
Request To Check Username
UsernameIsNotExistOrNotLogginedException
True
Un Set Session
True
RegisterdUser

Figure 11: User log-out sequence

- User show profile sequence (Figure 12)

User show profile sequence aims to select the registered users' data from the database. Apart from the possible database exception messages, the system would give exception if the username did not exist.

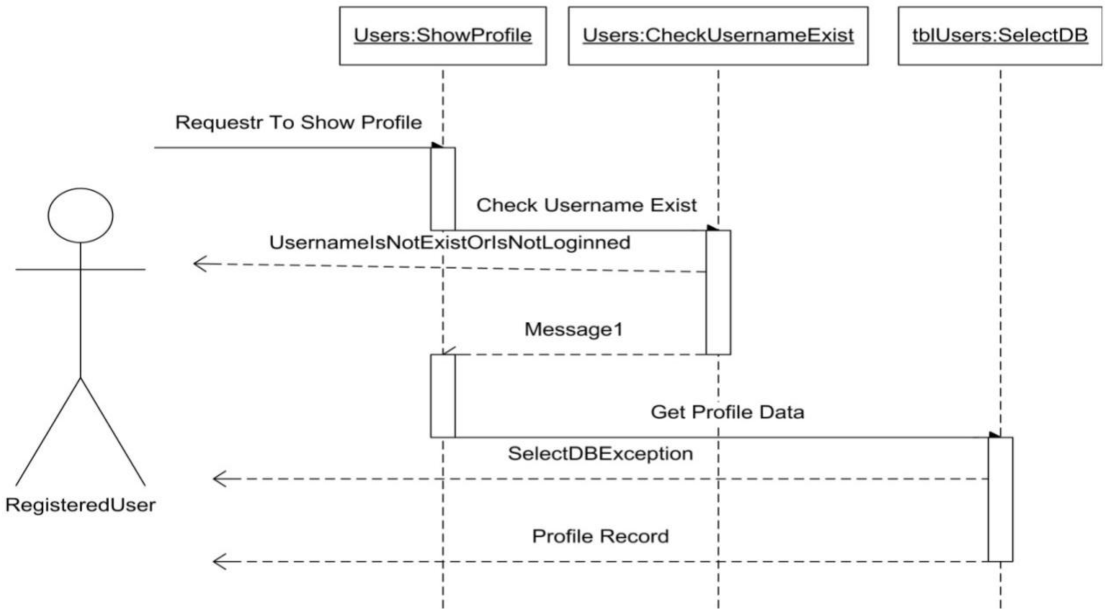


Figure 12: User show profile sequence

- Add event sequence (Figure 13)

As the core function, users can add new events to the system. This process has three steps: 1) user sends the request (we need to make sure that the request is not null); 2) validate the input data and give exception message if something is wrong with the input; and 3) add the new event to the table of events.

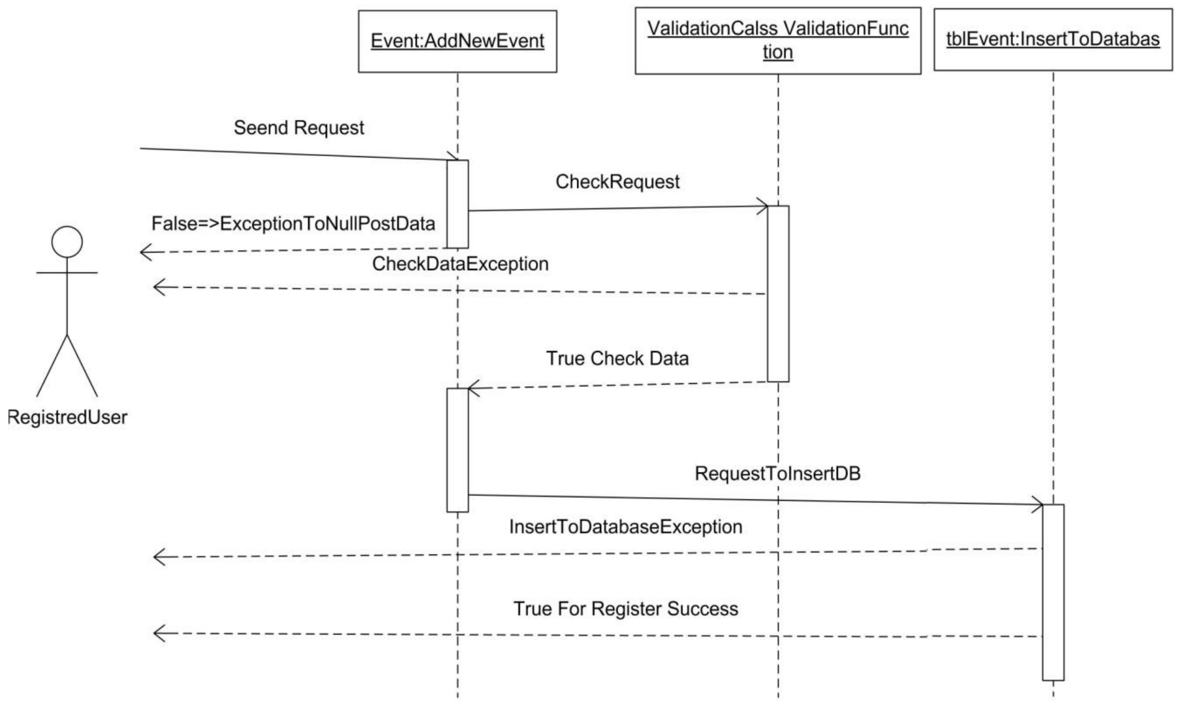


Figure 13: Add new event sequence

- Get the list of all events (Figure 14)

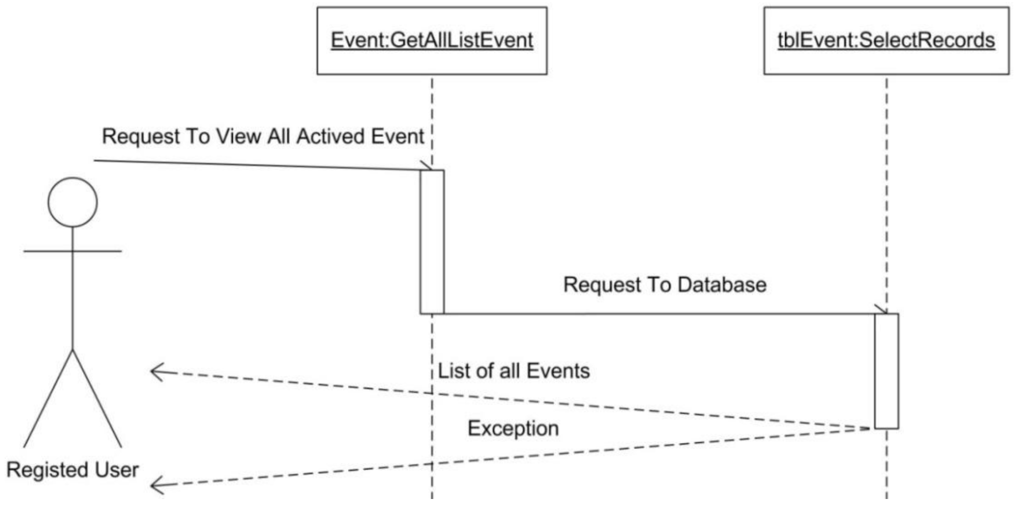


Figure 14: Get the list of all events

- Get event by event ID (Figure 15)

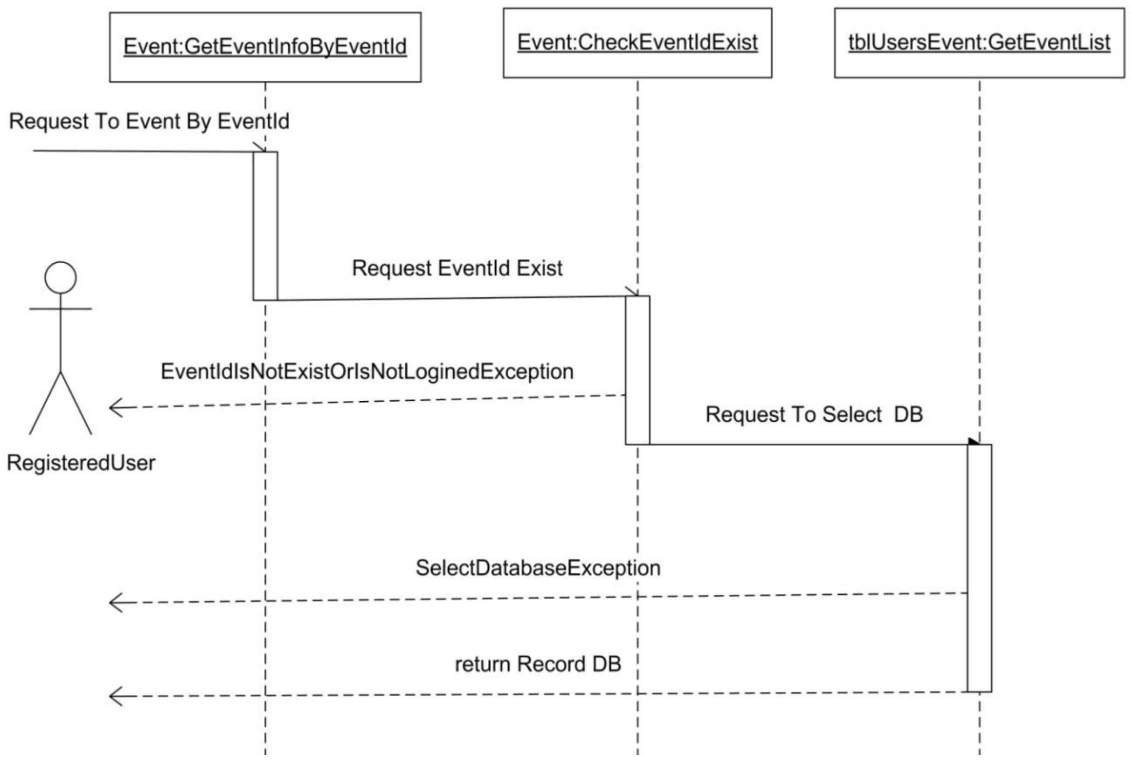

Event:GetEventInfoByEventId
Event:CheckEventIdExist
tblUsersEvent:GetEventList
Request To Event By EventId
Request EventId Exist
EventIdIsNotExistOrIsNotLoginedException
Request To Select DB
RegisteredUser
SelectDatabaseException
return Record DB

Figure 15: Get event by event ID.

- Get event by category ID (Figure 16)

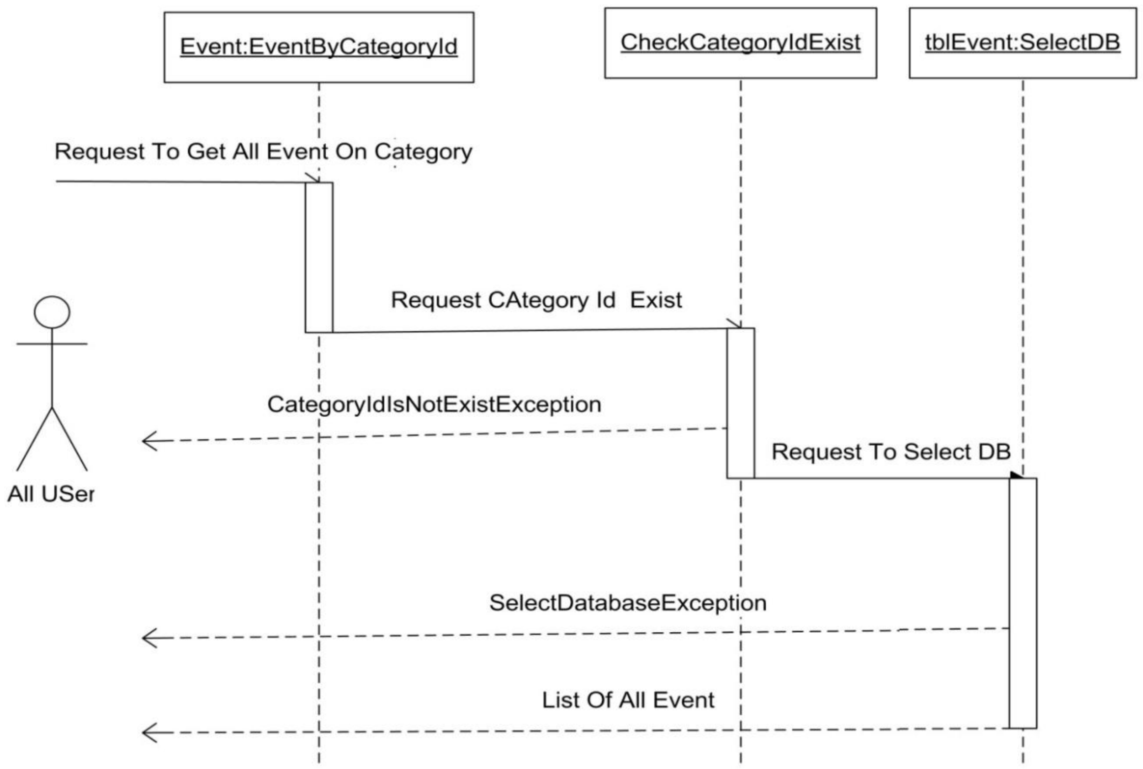


Figure 16: Get event by category ID

- Get event's information by user name ID (Figure 17)

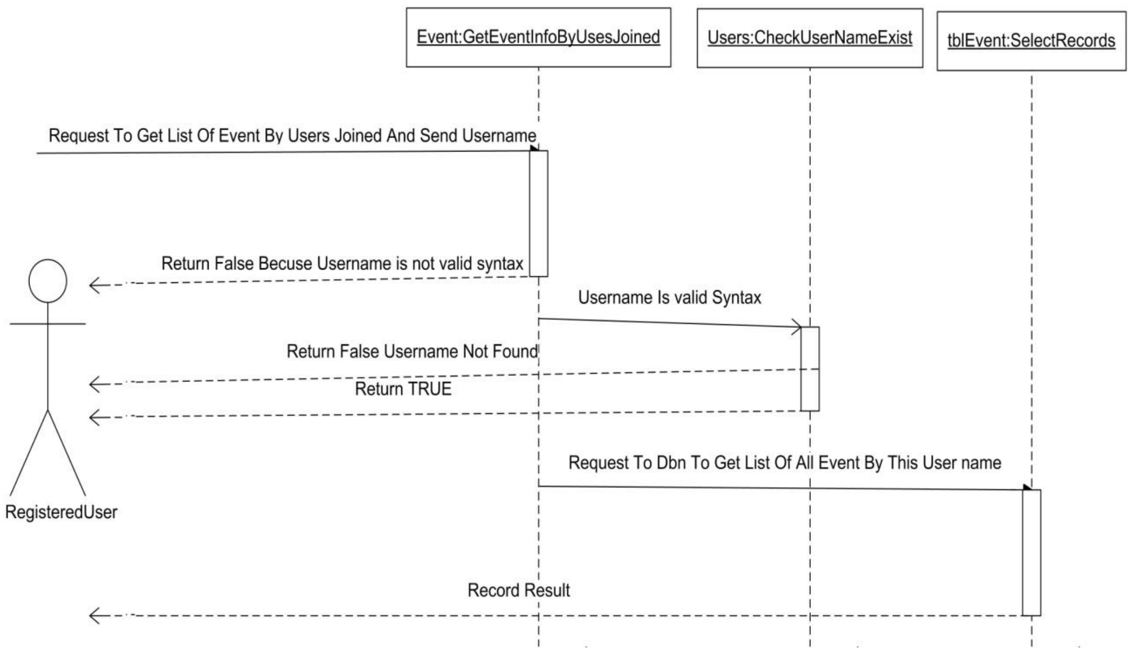

Event:GetEventInfoByUsesJoined
Users:CheckUserNameExist
tblEvent:SelectRecords
Request To Get List Of Event By Users Joined And Send Username
Return False Becuse Username is not valid syntax
Username Is valid Syntax
Return False Username Not Found
Return TRUE
Request To Dbn To Get List Of All Event By This User name
RegisteredUser
Record Result

Figure 17: Get event by the event ID for all users

- Users join event sequence diagram (Figure 18)

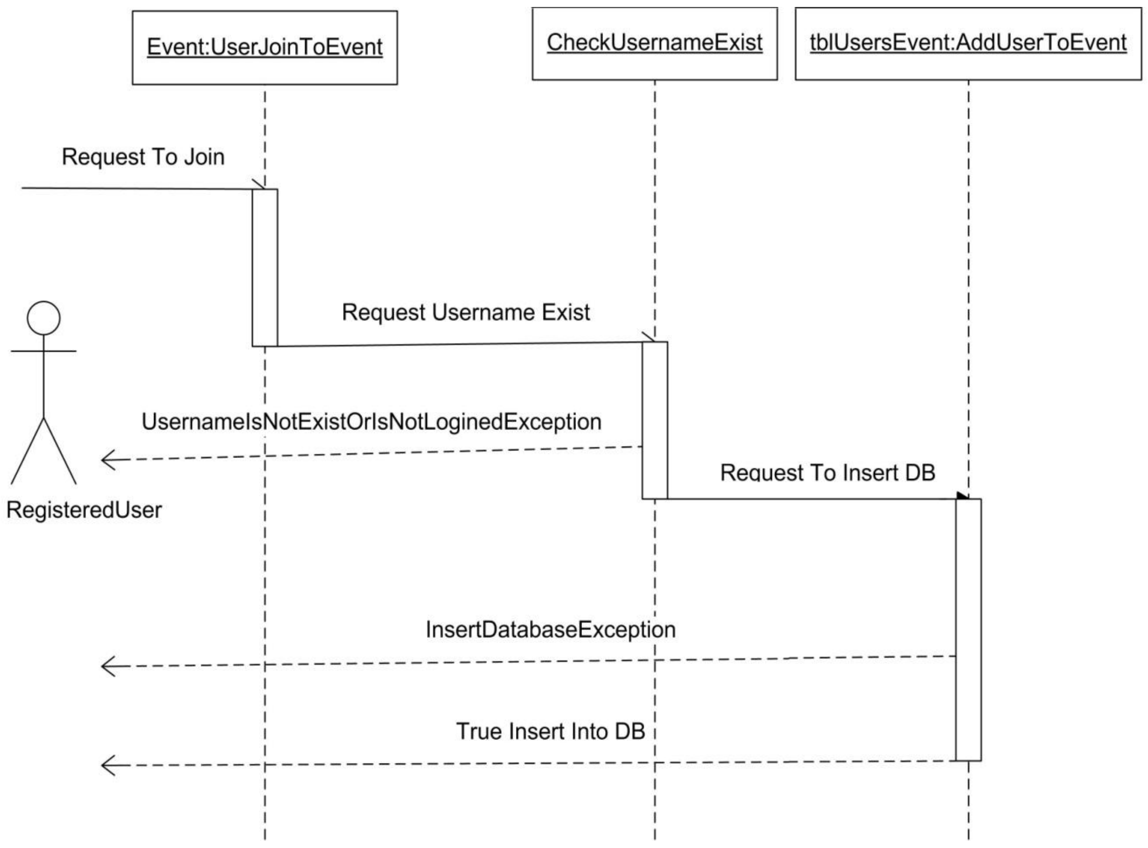


Figure 18: The sequence diagram of registered user joining event

- Paging sequence diagram (Figure 19)

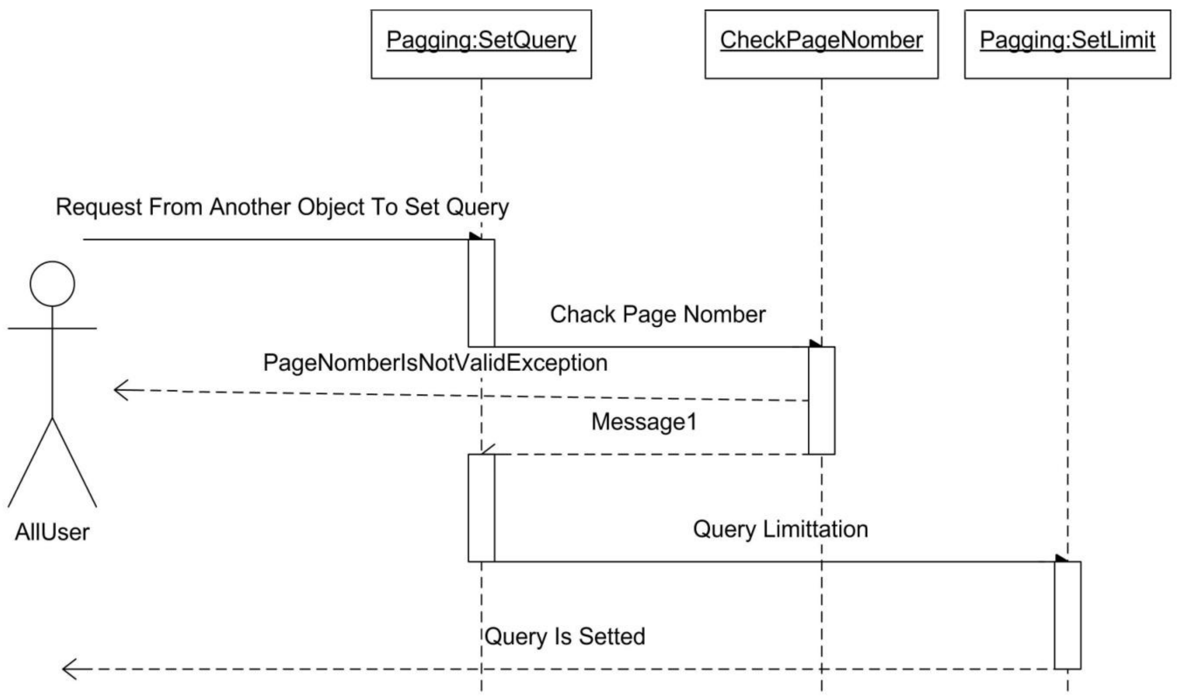


Figure 19: Paging sequence diagram

- Logical delete of events (Figure 20)

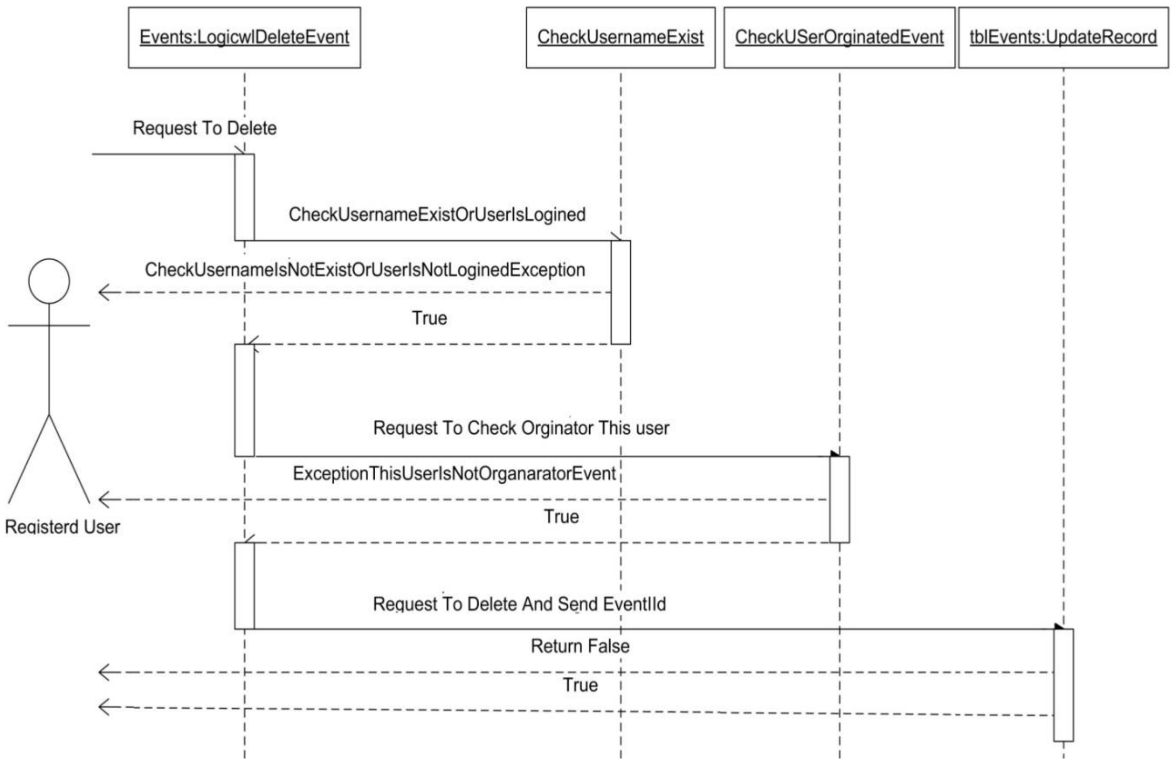

Events:LogicwlDeleteEvent
CheckUsernameExist
CheckUSerOrginatedEvent
tblEvents:UpdateRecord
Request To Delete
CheckUsernameExistOrUserIsLogined
CheckUsernameIsNotExistOrUserIsNotLoginedException
True
Request To Check Orginator This user
ExceptionThisUserIsNotOrganaratorEvent
True
Registerd User
Request To Delete And Send EventIId
Return False
True

Figure 20: The sequence diagram of logical delete of events

- Add comment to events (Figure 21)

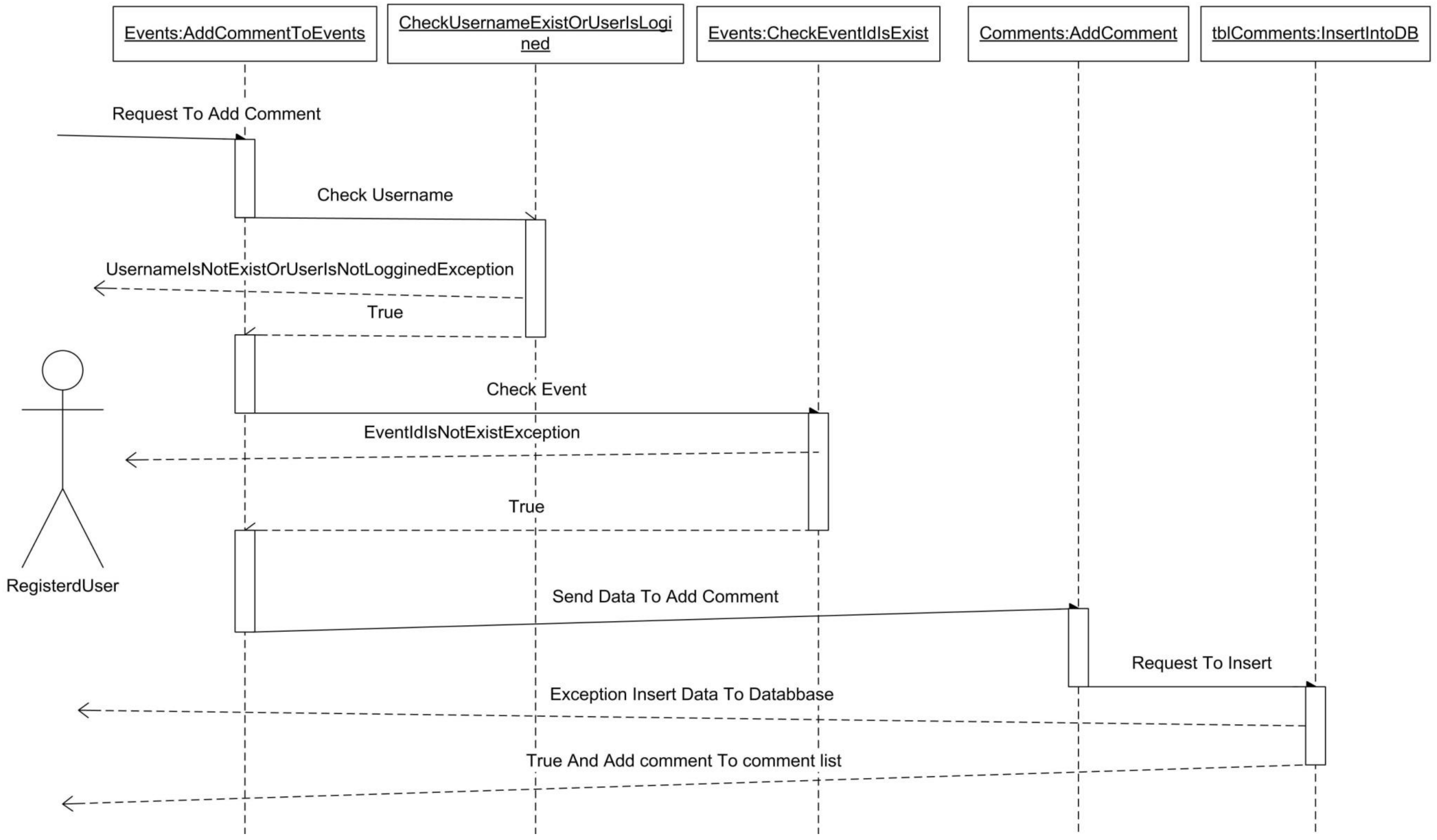

Events:AddCommentToEvents
CheckUsernameExistOrUserIsLogined
Events:CheckEventIdIsExist
Comments:AddComment
tblComments:InsertIntoDB
Request To Add Comment
Check Username
UsernameIsNotExistOrUserIsNotLogginedException
True
Check Event
EventIdIsNotExistException
True
RegisterdUser
Send Data To Add Comment
Request To Insert
Exception Insert Data To Databbase
True And Add comment To comment list

Figure 21: Add new comment

- Get all the comments on an event (Figure 22)

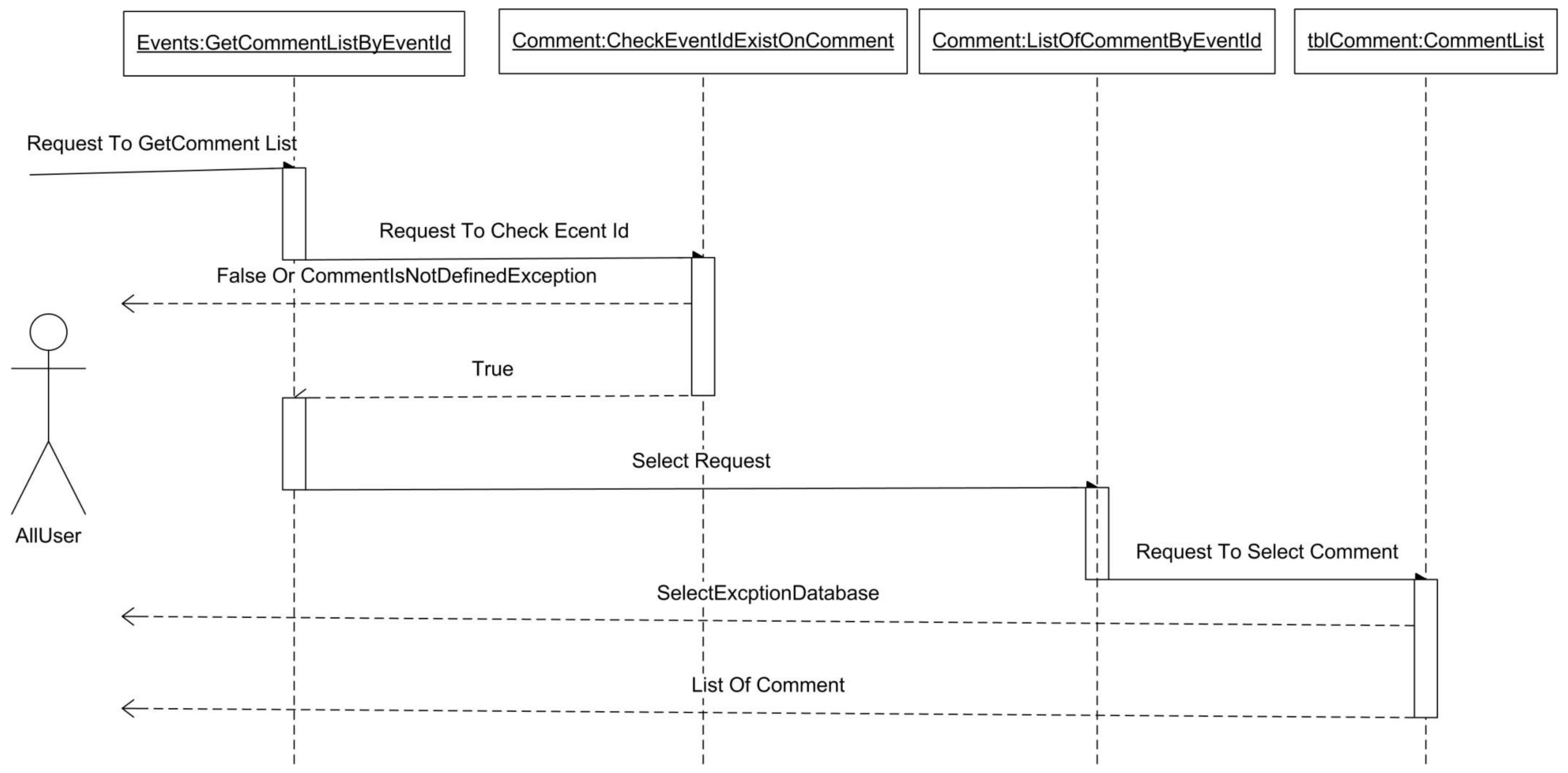

Events:GetCommentListByEventId
Comment:CheckEventIdExistOnComment
Comment:ListOfCommentByEventId
tblComment:CommentList
Request To GetComment List
Request To Check Ecent Id
False Or CommentIsNotDefinedException
True
AllUser
Select Request
Request To Select Comment
SelectExcptionDatabase
List Of Comment

Figure 22: Get all the comments on an event

3.2.2 State Diagrams

Figure 23 consists of five state diagrams which are supposed to complete our document regarding the structure of the system and the discipline that is applied for editing the tables of our database.

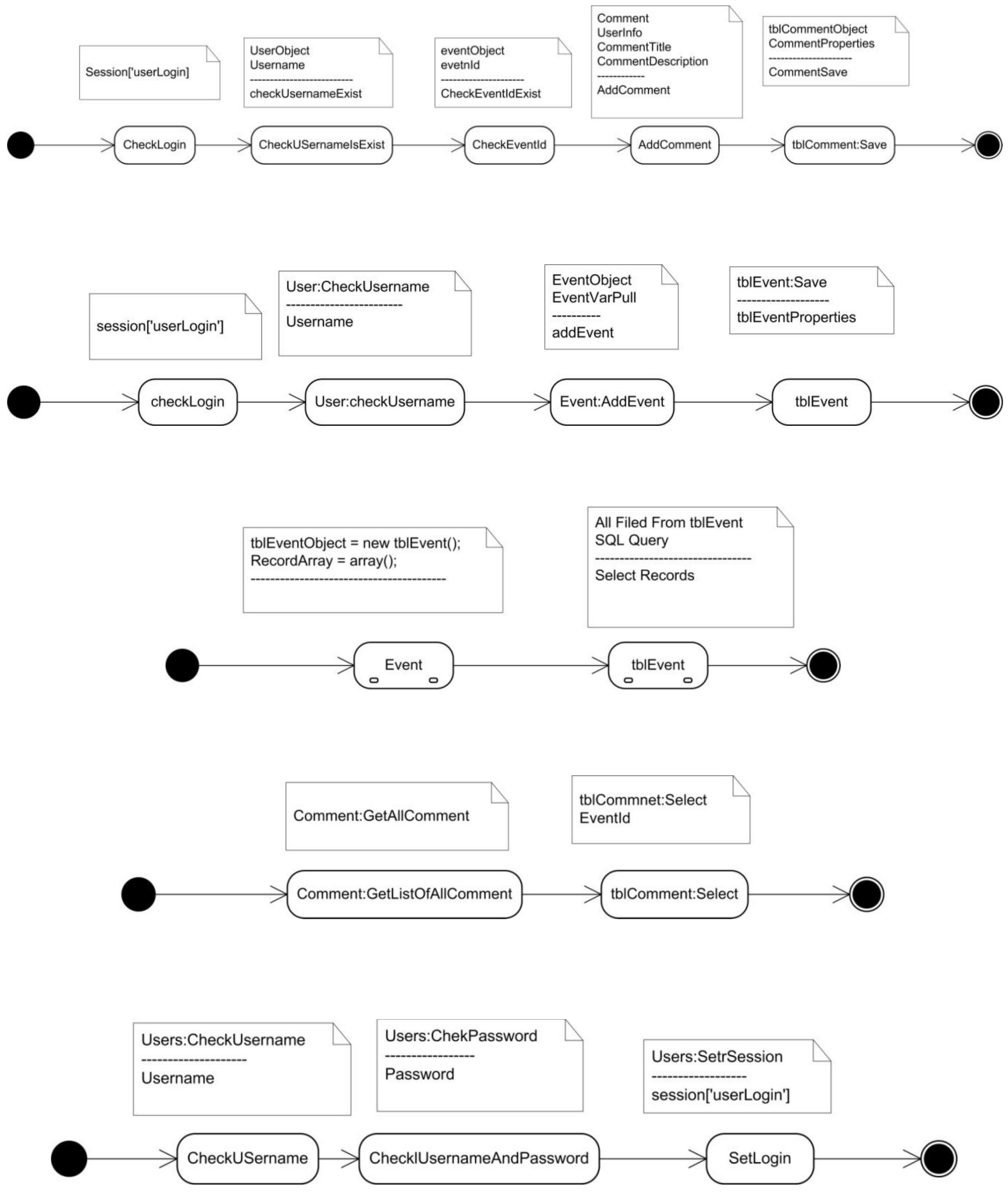


Figure 23: User behavioural state diagrams

3.2.3 Structure

So far a pretty clear picture of the behaviour of our system is achieved. According to the expected behaviours of the system, we draw out the classes and design the database. As previously mentioned, users and events are the main classes of our scheme. Figure 23 is the UML presentation of these classes, as it is implemented in our prototype.

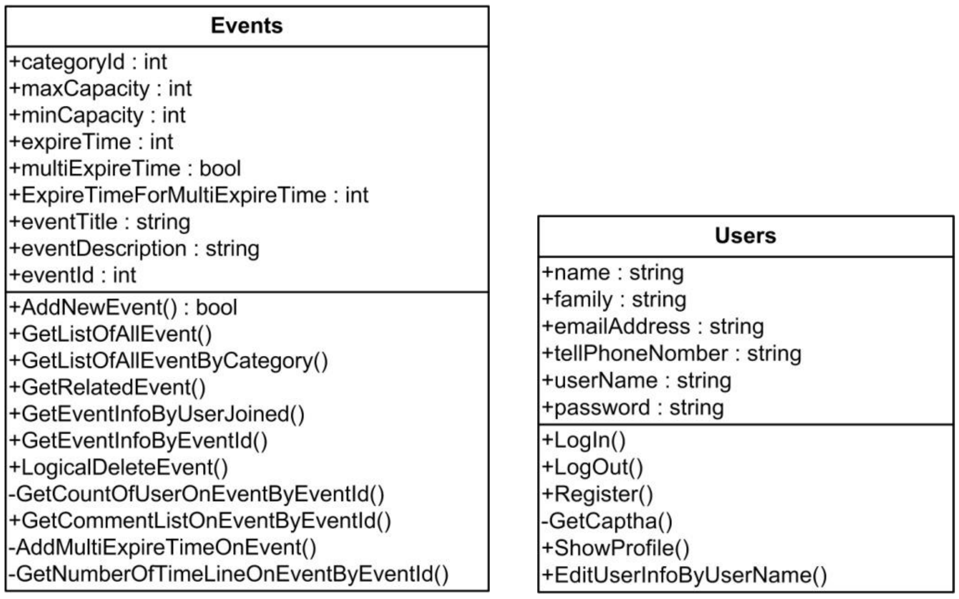


Figure 24: UML representation of Events class and Users class

By now, every method of user class should seem clear, except perhaps the GetCaptha method, that is a response-challenge method, with the aim of ensuring that a human is using the methods. Here are a few remarks regarding the event classes: The expireTime indicates the deadline for joining to the event. As previously mentioned, the creator of event can leave few time options for those who may consider attending. For this reason, the event class has a Boolean valued attribute, namely *multiExpireTime.* An event with multiExpireTime value of 1 implies that the time of the event is left to be elected based on users' votes. Otherwise, the expire time of event is fixed; the rest of the attributes in event classes are clear. Note that minCapacity and maxCapacity refer respectively to the minimum level of targeted attendance and the maximum possible level of attendance.

Additionally, there are eleven methods seen for the event class, many of which relate to the retrieval of the event objects, i.e. get list of all events; get list of all events by category; get related events; get event info by user joined; and get event info by event ID. Note that there is a method for adding multiple time options to the event. Finally, in order to extract the popularity of time options, we considered a method: get number of timelines on event by event ID. The remaining methods are quite straightforward.

In addition to user and event, we have created three other classes, namely Paging, SearchEngine and Comment classes (Figure 24) .

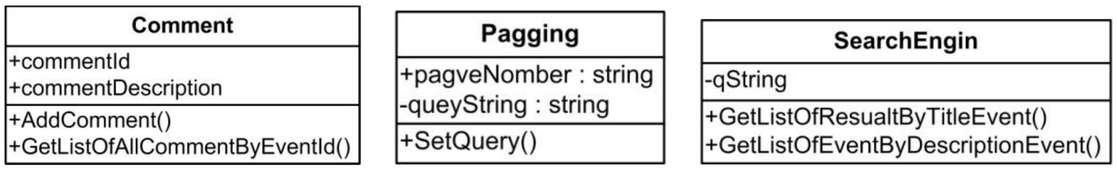


Figure 25: Comment class, Paging class and Searchengine class

These classes with the same design—of attribute and methods—are quite typical in today's web engineering. Finally, to complete the structure of the system, we need to design the database, particularly by showing the relation of different tables. Accordingly, class relation diagrams are designed in order to provide an easy way of understanding the working model of our database. Note that, due to the UML standards, dashed arrows represent the dependency relation or "use" relation.

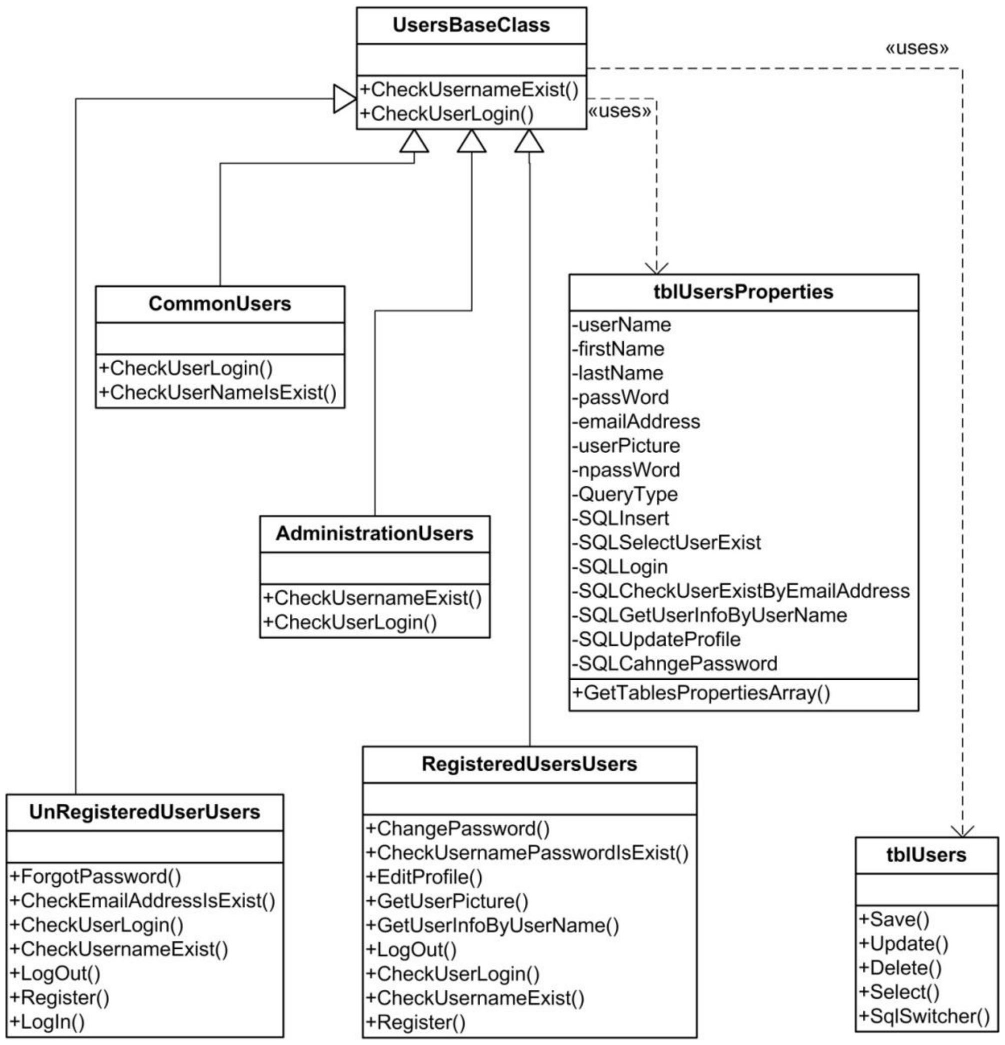


Figure 26: User base class

Figure 26 is the main class diagram of our database. It shows the relation of the parent user class with registered user, unregistered user and admin as well as comments. Note how users' data is stored in two SQL tables, namely *table of users* and *table of users' properties*. Similarly, Figures 27, 28 and 29 respectively represent our database structure regarding class base of events, comments and categories.

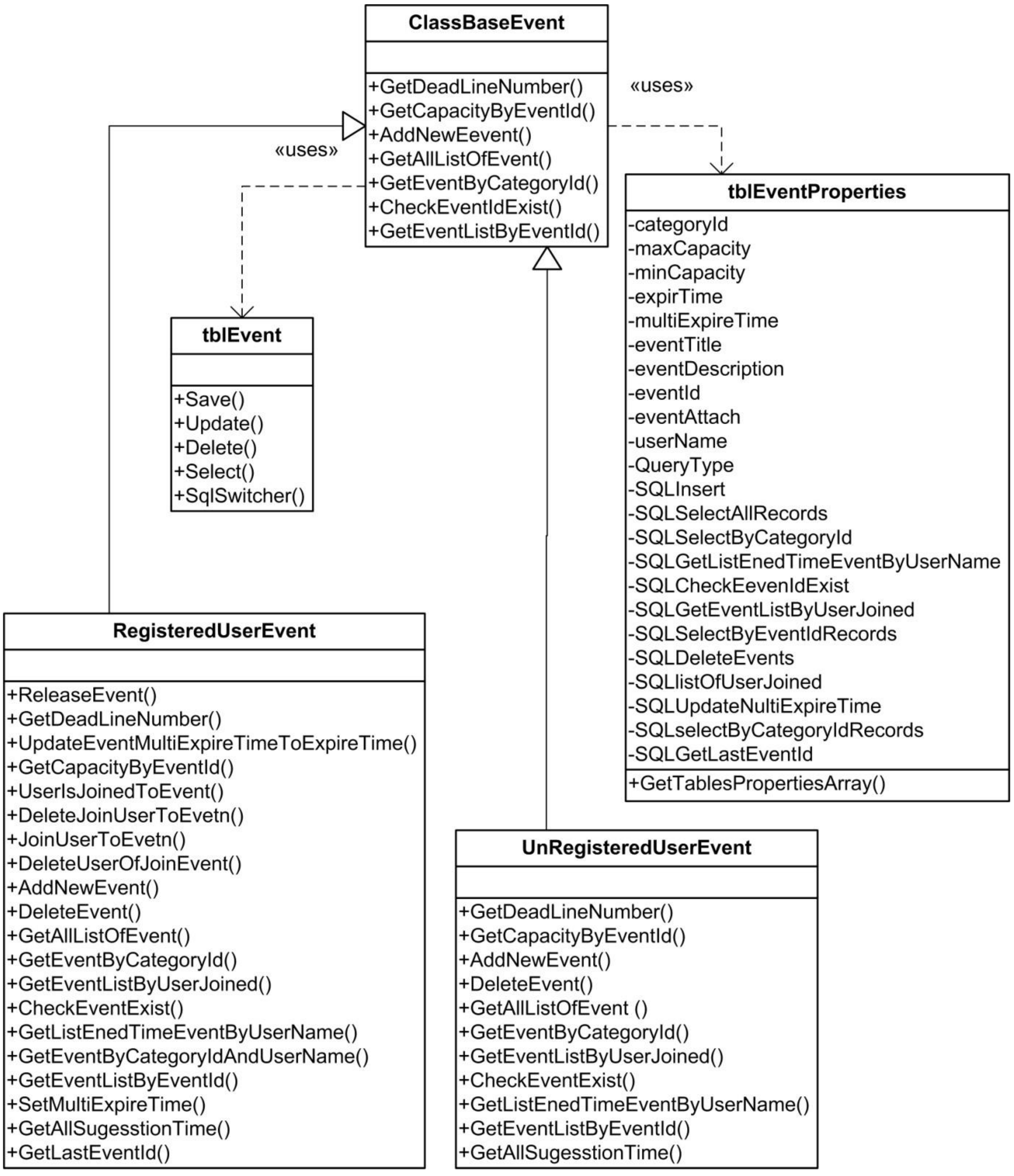


Figure 27: Events class base

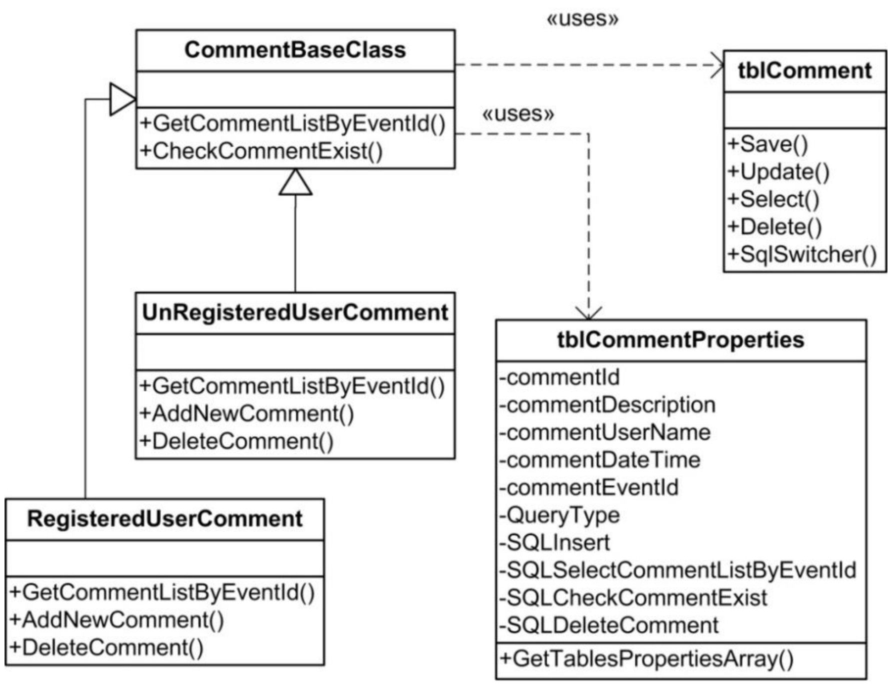


Figure 28: Comments class base

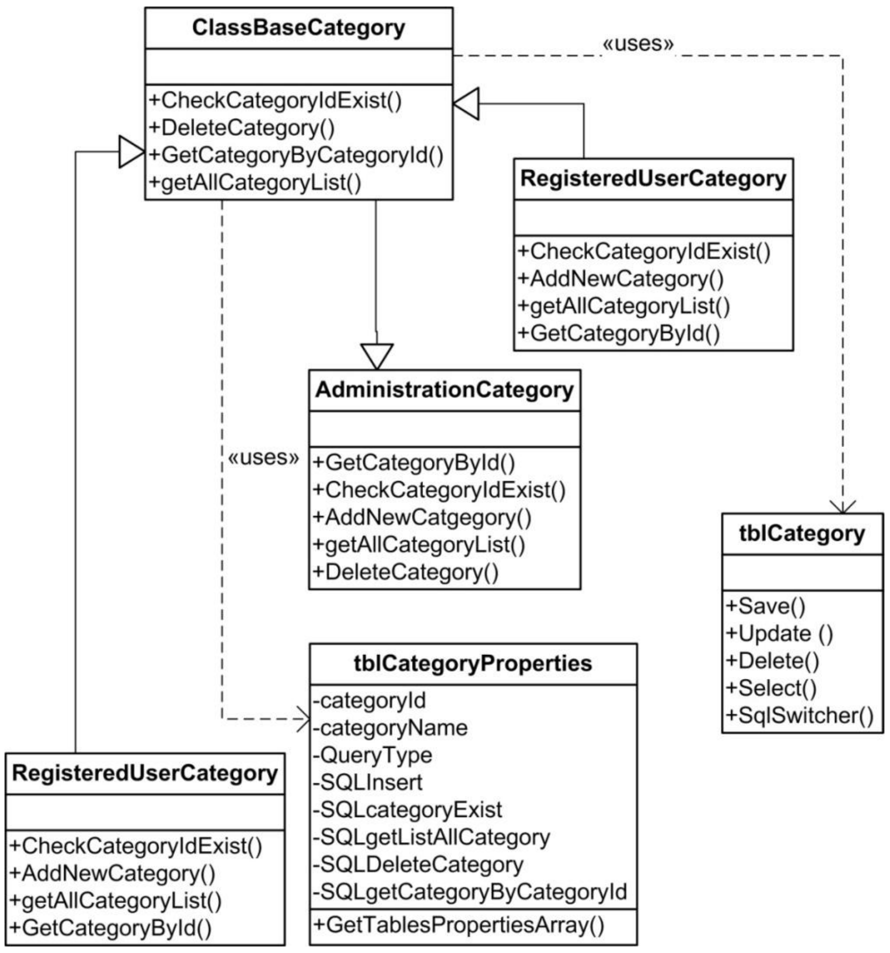


Figure 29: Category class base

3.3 Data Access Objects

This section consists of a number of data access object tables, as the concluding documentation regarding the design of the remerger.com. Note that data access objects aim to explain the abstract interface of the database and its mechanism, without exposing the detail of the database. Table 3 gives the name of core tables of the database, as well as the manner of modifying the database, through two files, namely DBconfig and Query execution. Furthermore, Tables 4, 5 and 6 respectively consist of the list of all components for users, categories and comments.

DAO (Database Access Object) (System Core)	
DBConfig	This file contains all the required information for connecting to the SQL database
QueryExecution	Any modification of the database would be completed by this file. Different queries from the classes of TBL.xxx would be sent to this file and, after connecting to the database, the required editing will be completed
tblUsers	This file would be used whenever it is needed to modify the records of user properties
tblUserJoinEvent	This file would be used in order to modify the database regarding the information of user attendance (or joining)
tblCategory	This file is used for modifying the tables/records of categories in the database
tblEvent	This file is used for modifying the tables/records of events in the database
tblExpireTimeEvent	This file is used for modifying the records of the expiry dates (deadline) of the events.
tblComment	This file is used for modifying the tables/records of comments in the database

Table 3:The core classes of system

Users Component	
Class.Administration.Users	The class of admin user
Class.Base.Users	The main user classes. All other user subclasses inherent from this class
Class.RegisteredUser.Users	The class of registered users
Class.UnRegisteredUser.Users	The class of unregistered users

Table 4: User component

Category Component	
Class.Administration.Category	The class that is used for managing the categories, accessible for admin user
Class.Base.Category	The main class of category. All the category classes inherent from this class
Class.RegisteredUser.Category	A class for showing the categories to registered users
Class.UnRegisteredUser.Category	A class for showing the categories to unregistered users

Table 5: Category component

Comment Component	
Class.Administration.Comment	The class for managing the admin comments
Class.Base.Comment	The main class of comments. All other comment subclasses inherent from this class
Class.RegisteredUser.Comment	A class for managing the comments for registered users
Class.UnRegisteredUser.Comment	A class for managing the comments for unregistered users

Table 6: Comment component

3.4 Event Recommender System

One thing that has not yet been clarified is the body of our event recommender algorithm. This section explains the recommender system that is implemented in our prototype. This recommender system has two moods: 1) random recommendation mood, and 2) co-attendance-based recommendation mood.

The use of random recommendation mood is limited to the very beginning of the event's creation. Note that we know very little about a newly added event, yet the system promises to inform some users from the onset of this event. Therefore, once an event is added, l randomly selected users will receive recommendations over this event, providing that it is not already recommended to them or they have not already attended this event. We refer to this algorithm as l-random recommender.

Predictably, once users start to respond to an event, we can gain more insight regarding the content of the event and make more clever recommendations. Therefore, we build the substantial mood of our recommender system, based on the idea of the users' co-attendance graph. We define this graph as follows: Assuming U (the set of users) are the nodes of our graph, there is an edge between two users, if only there is an event, which both users have attended. We also let the co-attendance graph be a multi-edge graph, as there might be many events that two users have commonly attended. Note that to some extent an updated co-attendance graph can represent the similarities of users. In fact, the idea is to build a distance function based on this graph. In the simplest case, we define the similarity or closeness of two users as the number of the edges that they share in the co-attendance graph. More precisely:

$$similarity(u_1, u_2) = sizeOf\{e \in E | u_1 \in A(e) \; and \; u_2 \in A(e)\}$$

While $A(e)$ is the set of users who have attended event e. Figure 30 is an example of a co-attendance graph, with 9 users and 5 events, while different events are shown with different colours.

Events	*Green*	*Black*	*Yellow*	*Red*	*Blue*
Attendance	u_1, u_2, u_3, u_4, u_9	u_4, u_5, u_7	u_3, u_6, u_7, u_5	u_2, u_4, u_5, u_7, u_8	u_1, u_6, u_8, u_9

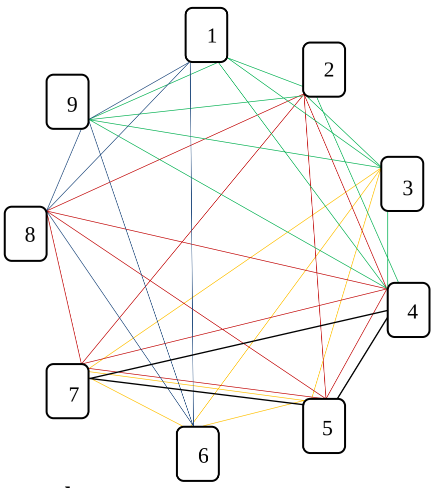


Figure 30: An example of a co-attendance graph

We may also define the closeness in respect to each category, in the sense that the number of commonly attended events between two users is significant only if they are in same category. In fact, this version—category-sensitive co-attendance based recommender—is what has been implemented in our prototype. In conclusion, the recommender system as it is implemented in our prototype works as follows: Once a new event is added to the system, the algorithm makes a fixed number of recommendations to a set of randomly selected users. After that, whenever a user joins an event, the event will be recommended to *k* of users with the highest number of co-attendance with the user who has just joined, providing that they have not joined the event nor that the same event has previously been recommended. Note that the parameter *k* of this method is the number of recommendations that are made. In our scheme, the value of this parameter is calculated by subtracting the current number of joined users from the maximum capacity of the event. The following is a pseudocode of this algorithm (pseudocode 1):

```
//co-attendance-Based Recommender

CoAttendaceRecommender (e,u) // recommends e to top co-participants of u

        k= e.maxCapacity – TotalNumberOfUsersJoiedSofar (e);

        Templist = listOfClosestUsers (k, u); /* listOfClosestUsers (k, u) returns the
        k closest usesr to the user u, while the closeness is in respect to the number
        of co-attendace u */

        For (all members of the list TempList like u)

         {

                If (u ∉ A(e) && e is not already recommended to u)

                then recommend (e, u)} // recommend e to u
```

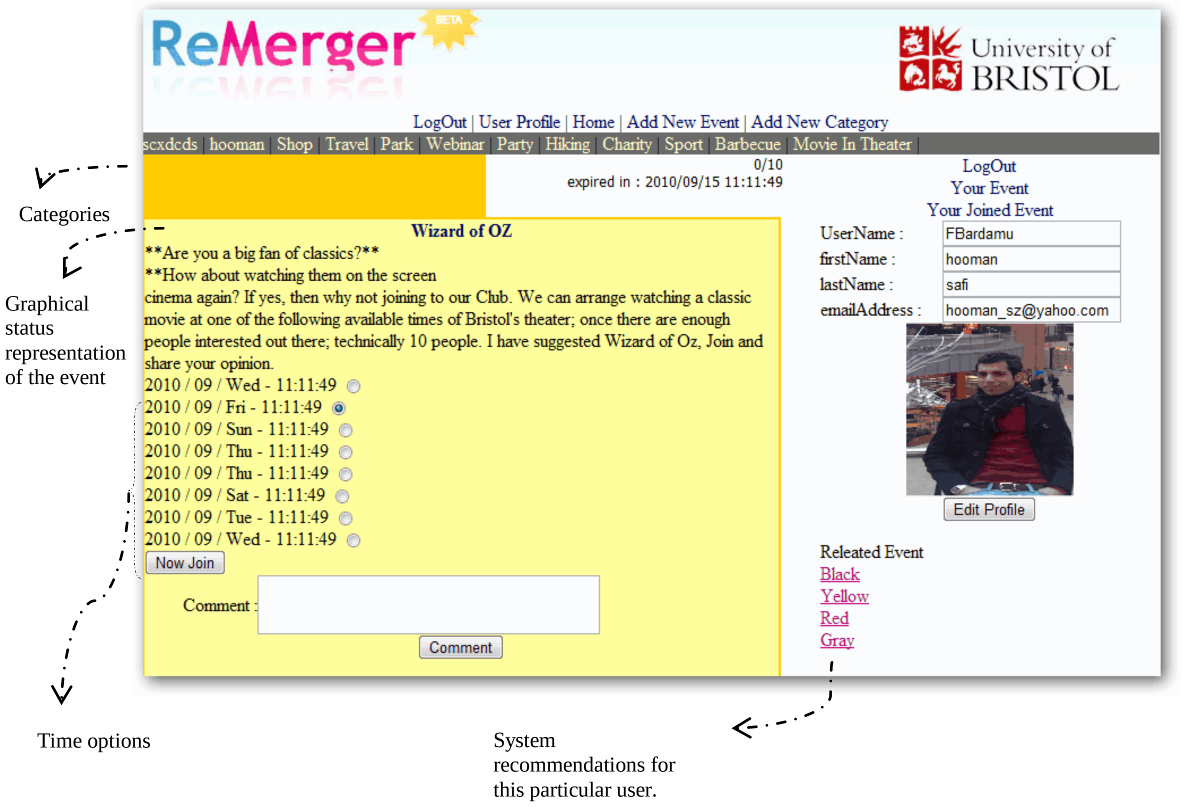
Pseudocode 1: Co-attendance-based recommender

3.5 Summary

This chapter points to the main body of the project, that is, building a prototype version of a Collaborative Event planning System. We have followed an object-oriented methodology and LAMP software technology has been employed, that is, Linux (operating system), Apache HTTP Server (web server), MySQL (database software) and PHP (Hypertext Preprocessor) as the principal components. In order to explain the structural and behavioural detail of the system, we have used UML diagrams: class diagrams, sequence diagrams and state diagrams.

There are three levels of user seen for the prototype: registered users, unregistered users and admin. The main functionalities that are provided in the systems are: signing up, logging in, logging out, browsing events by categories, browsing the events by page index, commenting on the events, creating new events (only for registered users), and receiving recommendations. The recommender system is approached by combining two methods: randomly selecting users, and selecting users with the highest number of previous co-attendance (that is, the number of events that both have attended). We also introduced a fancy feature for the system, that is, a graphical status representation of events. The status of events is supposed to show how much the plan has progressed since it was created. There are two parameters involved in working out an event status: 1) the proportion of users that are already joined to the system to the minimum required number of joining, and 2) the time that is left until the deadline for joining (or expire date of the plan).

The image below is a snapshot of the homepage of our prototype system. This prototype is named "ReMerger" and it is launched online on www.remerger.com by the time of submitting this document.


ReMerger
BETA
University of BRISTOL
Categories
Graphical status representation of the event
Time options
System recommendations for this particular user.

LogOut | User Profile | Home | Add New Event | Add New Category

scxdcds | hooman | Shop | Travel | Park | Webinar | Party | Hiking | Charity | Sport | Barbecue | Movie In Theater |

0/10
expired in : 2010/09/15 11:11:49

Wizard of OZ

Are you a big fan of classics?
**How about watching them on the screen
cinema again? If yes, then why not joining to our Club. We can arrange watching a classic movie at one of the following available times of Bristol's theater; once there are enough people interested out there; technically 10 people. I have suggested Wizard of Oz, Join and share your opinion.

2010 / 09 / Wed - 11:11:49
[✓ 2010 / 09 / Fri - 11:11:49]
2010 / 09 / Sun - 11:11:49
2010 / 09 / Thu - 11:11:49
2010 / 09 / Thu - 11:11:49
2010 / 09 / Sat - 11:11:49
2010 / 09 / Tue - 11:11:49
2010 / 09 / Wed - 11:11:49

Now Join

Comment :

Comment

LogOut
Your Event
Your Joined Event

UserName :	FBardamu
firstName :	hooman
lastName :	safi
emailAddress :	hooman_sz@yahoo.com

Edit Profile

Releated Event
Black
Yellow
Red
Gray

Figure 31: A snapshot of the homepage of www.remerger.com

4 EVALUATION, CONCLUSIONS AND FUTURE WORKS

The first section of this chapter discuses the possibilities of measuring the success of our work and, particularly, investigating the effectiveness of our recommendation algorithm. The second section quickly reviews our work and sums up the results, and the final section is a list of further possible works.

4.1 Evaluation

In order to run a functionality test on the prototype, a set of 1,000 users and 50 events is automatically generated. Furthermore, some pairs of events-users have been randomly selected for adding new attendance. According to this test, all the features of the system are working properly, i.e. commenting, register, log in, log out, add/delete event, join/opt out, browse by category/index, comment, and automated event recommender.

Roughly speaking, the Collaborative Event Planning scheme is supposed to provide an effective environment to make people informed of each other's plans. In fact, our collaborative event planning scheme is a simple blogging platform: Users can post a plan, while we have naively assumed that every plan has some deadline, minimum required number of attendance and maximum number of attendance, together with all other essential details, such as the venue and content of the plan. Yet there are two ideas proposed in our work, hoping to increase the total number of successful plans (plans that actually take place) and ultimately the collaboration of users:

1. Users can post unfixed plans, with the aim of attracting more opinions. This idea is represented in our prototype, by enabling users to leave up to ten time options for the event that they post.
2. The service is enhanced by an event recommender system, which works based on the co-attendance graph of the users.

For this reason, in order to evaluate the effectiveness of collaborative event planning, we can focus on the two enhancements identified above. Arguably, enabling users to post unfixed plans (or events with many options, for the sake of achieving more potential agreement) cannot really weaken the usefulness of the system, as long as it is only an option after all. Therefore, it is probably most important to compare the successfulness[11] of our recommender system with other possible variations of such algorithm. The rest of this section attends to a theoretical work for modelling the system, in particular testing the recommendation algorithm.

[11] Note that a recommendation is successful if it attracts the user's attention and results in his attendance.

4.1.1 Modelling Users' Behaviour

Recall that in a collaborative event planning system users might receive an invitation from other users or find an event via searching. However, the only way of becoming informed of available events that we are going to consider in our model is the event recommending system. Therefore, only two user behaviours are considered in our model: Creating event, and deciding whether or not to join a recommended event. This model is inspired by the collaborative-content filtering framework that we presented in 2.3.2.2. According to that framework, first a joint map like $\Psi\colon \mathrm{U} \times \mathrm{E} \longrightarrow \mathbb{R}^{\mathrm{D}}$, extracts *D*, features from every user-event pair. Then we define a family of functions F that are linear in the chosen feature map via:

$$F(u,e;w) = \Psi(u,e).\vec{w}$$

Where <. > is the inner product and $\vec{w}$ is a weight vector of size *D*. Finally, to predict the decisions of users regarding a recommended event, we operate a comparison between the value of the above inner product and an adaptive attendance threshold θ. The estimation of *w* and θ with the proposed algorithm depends only on inner products between feature vectors for user-event pairs with an observed decision. We can apply this approach to our model, with a few simplifications. In particular we will assume that:

1. All users and events are described by vectors of the same size. Note that in our original framework we weren't restricted to this assumption; instead, we suggested a two-stage process of first defining $\Lambda\colon U \to \mathbb{R}^G$ and $\Phi\colon E \to \mathbb{R}^H$ and then combining every dimension of Λ multiplicatively with every dimension of Φ to get $\Psi(u,e) = \Lambda(u) \otimes \Phi(e) \in \mathbb{R}^D$, where $D = G.H$. By assuming users and vectors described by vectors of same size, we eliminate the need for performing tensor products and achieve faster computations, since the joint mapping (Ψ) can simply be defined as the absolute value of subtracting event attributes from users' interests.
2. A same-weight vector can be applied in case of every user-event pair. More precisely, we compute the inner product of $\Psi(u,e)$ and a weight vector of same size, with all the components set to 1.
3. A same attendance threshold can be set for all users. More clearly, the result of the above product will always be compared with a same threshold (user would join to the event, if the inner product is not bigger than the threshold). More precisely, for any given u_i and e_j, we compare the attendance threshold with $f(u_i, e_j; w)$, that is:

$$f\left(\langle u_i^1, u_i^2, .. u_i^D\rangle, \langle e_j^1, e_j^2, .. e_j^D\rangle; \overbrace{\langle 1,1,\ldots 1\rangle}^{\mathrm{D}}\right) = \sum_{k=1}^{D} \left|u_i^k - e_j^k\right|$$

4. In addition to making a decision—regarding joining or not joining the recommended events—we assume that, in every given opportunity, users may or may not add a new event to the system, with a fixed particular probability. We randomly assign this probability to users, at the step of user generation.
5. We do not choose any particular distributions, other than randomly drawn values for users' interest vectors and event attribute vectors. This allows us to perform the whole test considerably faster. There are however many cleverer applicable distributions for modelling users' behaviour. For example Latent Dirichlet Allocation, which can be used to explain the similarity of some parts of the users' interests and events attribute (23). Alternatively, it might be more realistic to assume that users are more likely to generate events with close distance to their own interests. Yet, for the sake of simplicity, we stick to the simple random distribution of users' interest vectors and event attribute vectors. Also, non-parametric statistical[12] might be more advantageous, especially when we intend to model a particular domain of the application and at the same time some dataset over previous observations is available.

4.1.2 A Theoretical Scheme for Simulating Collaborative Event Planning

In this section, we propose an idea for simulating a collaborative event planning environment, based on our given model of users' behaviour, though putting this simulation into practice is seen as a future work of the project. This simulation scheme consists of two procedures: The first procedure generates and saves a set of test users, constructed based on the model that we have just discussed. The second procedure reads a set of generated users, as an array of objects, along the code of the recommendation algorithm that we intended to test—providing that the algorithm is built in the system. Next we run the system for some certain time and report the output as three matrixes, namely matrix of created events $\left([c_{ij}]_{m\times n}\right)$, recommended events $\left([r_{ij}]_{m\times n}\right)$ and the matrix of successful recommendations $\left([s_{ij}]_{m\times n}\right)$, where:

- *m* is the number of test periods (number of time loops) and *n* is the number of users.
- c_{ij} is the created events in time *i* by the user *j*. ($c_{ij} = null$ if u_j creates no event in time *i*). Note that users cannot create more than one event at a time.
- r_{ij} is the recommendation that is made—by the algorithm that is being tested—to user *j* in time *i*. ($r_{ij} = null$ if u_j creates no event in time *i*.)

[12] Non-parametric statistical techniques are methods which do not rely on data belonging to any particular distribution; e.g. distribution-free methods, statistical inference. Occasionally, non-parametric covers techniques that do not assume that the structure of a model is fixed, e.g. non-parametric regression.

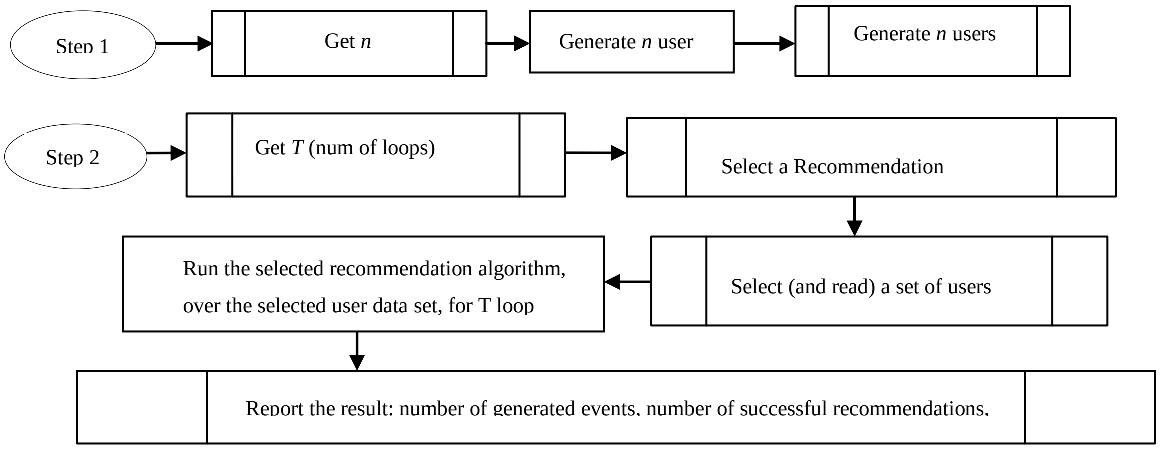
- s_{ij} is the successful recommendation that is made for user *j* in time *i*. Recall that a recommendation is considered to be successful if eventually the targeted user would attend the recommended event. More precisely:

$$r_{ij} = s_{ij} \Leftrightarrow f(u_j, r_{ij}; w) \leq \theta$$

Finally, we would score the efficiency of the tested event recommender $\mathcal{R}$ (after running for *t* time), based on the following formula:

$$Efficiency_{\mathcal{R}}(t) = \frac{1}{t}\sum_{i=1}^{t} \frac{Numbr\ of\ successfull\ recommendations\ in\ time\ i+1}{Number\ of\ possible\ recommendations\ in\ time\ i+1}$$

This scheme assumes that users can join the same event only once. Also, we have assumed that in each turn the recommender cannot make more than one recommendation. Figure 32 is a high-level view of our suggested scheme for testing the event recommender system.



Step 1
Get *n*
Generate *n* user
Generate *n* users
Step 2
Get *T* (num of loops)
Select a Recommendation
Select (and read) a set of users
Run the selected recommendation algorithm, over the selected user data set, for T loop
Report the result: number of generated events, number of successful recommendations,

Figure 32: The zero-level diagram of the proposed procedure of testing event recommenders

Table 7 is an example of the output of such a simulation system. According to this table, the simulation is run for 5 time periods and 4 events are generated by users in total. Note that the red *e*'s represent newly generated events. For instance, user u_2 has generated e_1 at t_1. The remaining components of the table represent the recommended events. For instance, event e_3 has been recommended to user u_5 at t_3 , and e_1 has been recommended to user u_3 at t_2. Finally, green events are successful recommendations and gray events are unsuccessful recommendations. According to our given formula, we can compute the efficieny of the imaginary recommendation algorithm that is used in Table 7:

$$Efficiency_{\mathcal{R}}(5) = \frac{1}{5}\left(1 + \frac{3}{9} + \frac{1}{25} + \frac{3}{17} + \frac{1}{5}\right) \cong 0.349$$

Time	u_1	u_2	u_3	u_4	u_5	Total Possible Recommendations	Successful Recommendations
t_1	–	e_1	–	–	e_2	0	0
t_2	e_1, e_3	e_1	e_1	e_2	e_1	8	2
t_3	e_2	e_2	e_3	e_1	e_3, e_4	24	0
t_4	e_3	e_3	e_2	e_4	e_2	16	2
t_5	e_4	e_4	e_4	e_3	e_4	4	0

Table 7: A schematical output table of the simulation system

4.2 Conclusions

The remarkable emergence of collective-intelligence-based systems of web 2.0 in the past few years indicates a high potential for scientific and commercial works, especially in the field of social translucency. Following this trend, a whole new class of the human collaboration can be expected in future. In view of that, this work was intended to be a combination of some theoretical and experimental study in the context of web-based collaborative event planning systems. According to our definitions, web-based collaborative event planning is intended to support a web environment which is enhanced by specific tools, aiming to ease the process of forming events and browsing events. The main philosophy of the presented work is authorizing the users to freely express their concerns and ideas regarding their preferred events. Therefore, the proposed scheme is a customized blogging platform, which is equipped by tools such as shared calendars, recommender systems and graphical event status representation. According to this scheme, an event—or more precisely a plan for a potential event—will be labelled as confirmed and considered to be doable if there is sufficient integration of preferences around it. This is due to the fact that events of the real world cannot be accomplished, unless there is an adequately high number of participants. Based on this philosophy, we investigated some frameworks that take the users' preferences and event attributes into account. This investigation has formed the body of our theoretical work and mainly covers some techniques in collaborative filtering. Additionally, some possible application domains are discussed as examples.

In terms of experimental works, we have implanted a simple prototype website for collaborative event planning, named Remerger. The main improvement of this prototype, in compression with other similar systems, is the option of posting unfixed plans for users. In particular, users of the system do not need to specify an exact time for their suggested plans and they can leave up to ten time options.

This feature is intended to increase the chance of users achieving agreement over the detail of events. Furthermore, we proposed a scheme for simulating the collaborative event planning systems, based on a simple model of the behaviour of users.

4.3 Future Works

Expanding the Platform: as a matter of fact, web-based collaborative event planning is inherently involved in individual relations, particularly in social networking. To exemplify this characteristic, consider that it is beneficial to keep the connection among the groups of categorically likeminded people or friends, thereby increasing the chance of forming events and collaboration. For this reason, developing a more advanced platform for the system seems essential. For the most part, a future work of this project should cover a new database structure, which could handle social networking functionalities. This may result in a whole new design of the system, in view of the fact that SQL cannot properly support the navigation in even moderately large sets of nodes and links. One candidate database solution, which might be compliant for our future versions, is the Hadoop technology. Apache Hadoop is a software framework that supports data-intensive distributed applications under a free license (24).

Web Syndication: Almost all the reviewed working web-based planning or calendaring systems enable their users to import or export plans. This feature is highly important for most users, since they might already be using a form of personal web-based calendaring. In order to approach this functionality, we should understand the specification of the existing popular calendar formats and adopt a new data storing structure.

Conditional attendance: Enabling users to express their conditional attendance is a novel idea, considered as a further development of our collaborative event planning system. To make this feature obvious, think of a user who has indicated his interest for attending many plans. As this user is not aware of which event will be eventually confirmed, some of his attended plans might be in conflict. In this instance the user cannot attend all his events and, consequently, not keeping the attendance promises may affect the availability of plans. Now, the idea of conditional attendance is to enable users to indicate their attendance priorities. In other words, the system can ask the user to express his conditions once a conflict is observed between two of the attended plans of the users. Clearly, by taking this condition into account we would be treating different events as dependent entities, since the confirmation of one plan might influence other events.

Modelling the system

As a major necessary work of the future, the suggested simulating scheme will be implemented and different recommendation algorithms will be compared. Figure 33 aims to give an idea of one possible structure for our simulation program. According to these classes, first a set of user objects will be created. Next, we run the main body of the program, which is basically a loop. In each loop, every user may add a new event to the system, based on the value of his ProbabilityOfCreatingEvent. Alongside this, each loop gives an opportunity to a recommendation algorithm to recommend a single event—chosen from the set of generated events—to each user. The result of this recommendation — regarding whether the user would attend the event or not—can be computed instantly, by taking the attribute vector of the event and interest vector of the user into account, for instance by comparing the inner product of these vectors with some attendance threshold.

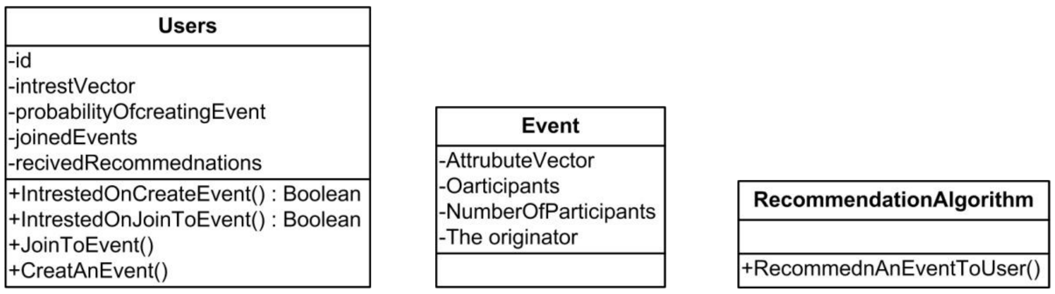


Figure 33: Classes of a simple CEP simulation program

By considering more significant factors, especially specific to the domain of the application, we can develop more sophisticated simulation of the collaborative event planning systems. For example, we can give a better model for a collaborative purchasing system, by understanding the influential factors of the market and the behaviour of the users. Note that an evaluation of a collaborative purchasing system should address two questions, since we would like to know how useful the collaborative purchasing system is for both the suppliers and customers.

User credit system: employing a user credit system, could be another possible further modification for our proposed scheme. As previously mentioned in 1.3.1, user-generated content-based systems may also suffer from abuse of the system. By assigning a credit to each user and restricting his activity to his remained credit of the day, we can prevent such issues.

5 Bibliography

1. **Tyson, Jeff.** How Movie Distribution Works. *How stuff Works?* [Online] http://www.howstuffworks.com/movie-distribution.htm.

2. compare money transfer [Online] http://www.comparemoneytransfer.co.uk/international_money_transfer.html.

3. **Development, Organisation for Economic Co-operation and [WORD MISSING HERE].** *PARTICIPATIVE WEB: USER-CREATED CONTENT.* s.l. : Organisation for Economic Co-operation and Development, 2007. JT03225396.

4. **Thomas W. Malone, Alex Pentland, Karim R. Lakhani.** *A Lecture on Collecive Intelligence.* [Video] s.l. : MIT center for Collective Intelligence, 2007.

5. *Social Translucence: An Approach to Designing Systems that Support Social Processes.* **Thomas Erickson, Wendy A. Kellogg.** s.l. : IBM T.J. Watson Research Center, 2000.

6. **Harrison, Terry P. and Neale, Hau l.Lee and John J.** *The Practice of Supply Chain Management.* s.l. : Springer, 2003. ISBN 0387240993.

7. **Limited), Russell Scoular (Government Affairs Manager at Ford Motor Company of Australia.** *Inquiry into Mandatory Ethanol & biofuels Target in Victoria Submitted to Economic Development & Infrastructure Committee of Parliament House.* 2007.

8. **Martin, Michael j.C.** *Managing and Entrepreneurship in Technology-based Firms.* . s.l. : Wiley-IEEE. , 1994. P.44. ISBN 0471572195.

9. [Online Video] Save Money by Collaborative Purchasing. *teachers.tv.* [Online] June 2010. http://www.teachers.tv/videos/collaborative-purchasing.

10. "What is School Quote?". *SchoolQuote.* [Online] http://www.schoolquote.co.uk/What.aspx.

11. **Sjogreen, Carl.** The Official Google Blog. *It's about time.* [Online] 4/13/2006. http://googleblog.blogspot.com/2006/04/its-about-time.html.

12. I*nternet Calendaring and Scheduling Core Object Specification.* **B. Desruisseaux, Ed. Oracle.** s.l. : Internet Engineering Task Force Organisation, 2009.

13. **do", "Discover things to.** Zevent.com about page . *Zevent* . [Online] http://corporate.zvents.com/.

14. Fast Maximum Margin Factorization. [Online] http://people.csail.mit.edu/jrennie/matlab.

15. C/Matlab Toolkit for Collaborative Filtering. [Online] http://www-2.cs.cmu.edu/~lebanon/IR-lab.htm.

16. *Using collaborative filtering to weave an information tapestry.* **Goldberg, David and David Nichols, Brain M. Oki, Douglas Terry.** s.l. : Communications of the ACM , 1992. ISSN 0001-0782.

17. *Recommendations for Amazon.com: Item to Item Collaborative Filtering.* **G.Linden, B. Smith, and J. York,.** s.l. : IEEE, 2003, Vols. Vol.7,no.1,pp.76;.

18. *Slope one predictors for online rating based collaborative filtering.* **Daniel lemire, Anna Maclachlan.** s.l. : SIAM data mining, 2005.

19. **Franklin, Joel N.** *Matrix Theory.* s.l. : Dover Publications, (1968). ISBN 0-486-41179-6.

20. *Improving Regularized Singular Value Decomposition for Collaborative Filtering.* **Paterek, A.** s.l. : Proc. KDD Cup and Workshop;, 2007.

21. *Tag-Based Contextual Collaborative Filtering.* **R. Nakamoto, S. Nakajima, J. Miyazaki, S. Uemura.** s.l. : IAENG International Journal of Computer Sceince, 2007.

22. OMG Unified Modeling Language (OMG UML),Superstructure, V2.1.2- Page 504. *OMG.* [Online] http://www.omg.org/spec/UML/2.1.2/Superstructure/PDF.

23. *Latent Dirichlet Allocation.* **David M. Blei, Andrew Y. Ng, Michael I. Jordan.** s.l. : Journal of Machine Learning Research 3, 2003. 993-1022.

24. Hadoop Wiki Project Description. *Wiki overviews of Apache.* [Online] 2010. http://wiki.apache.org/hadoop/ProjectDescription.

25. Collective Intellegence. *Wikipedia.* [Online] http://en.wikipedia.org/wiki/Collective_intelligence.

26. **Wikipedia.** Entry for "Publish/subscribe". [Online] http://en.wikipedia.org/wiki/Publish/subscribe.

27. *A Semantic Imitation Model of Social Tagging".* **Fu, Wai-Tat.** s.l. : conference on social computing; proceedings of the IEEE, 2009.

28. *A Joint Framework for Collaborative and Content Filtering.* **J. Basilico, T. Hofmann.** ", a work of J. Basilico and T. Hofmann. : Annual ACM Conference on Research and Development in Information ., 2004.

29. *Annual ACM Conference on Research and Development in Information* . **Justin Basilico, Thomas Hofmann.** s.l. : Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, 2004. SBN:1-58113-881-4.

Appendix

• Kernel and Kernel trick[13]

Kernel is a widely used term in mathematics, but in statistical learning theory, it refers to a weighting function *K* satisfying the following two requirements:

- $$\int_{-\infty}^{+\infty} K(u)\,du = 1$$
- $K(-u) = K(u)$ for all values of u.

Kernels are applied by many non-parametric estimation techniques. Given a set of independent and identically distributed sample of random variables $\{x_1, x_2, \ldots x_n\}$, a classical example of application of kernel functions is non-parametric inference of the distribution *f* using:

$$\hat{f}_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

Where *K* is some kernel and *h* is a smoothing parameter called the bandwidth. Quite often *K* is taken to be a standard Gaussian function inference is *Gaussian function* with mean zero and variance 1, defined as: $K\left(\frac{x-x_i}{h}\right) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(x-x_i)^2}{2h^2}}$, Where *h* is intended to control the variance indirectly.

The idea of kernel trick is to use a linear classifier algorithm for solving a non-linear problem by mapping the original non-linear observations into a higher dimensional space, where the linear classifier is subsequently applied. Figure 34 is supposed to give an idea of the kernel trick.

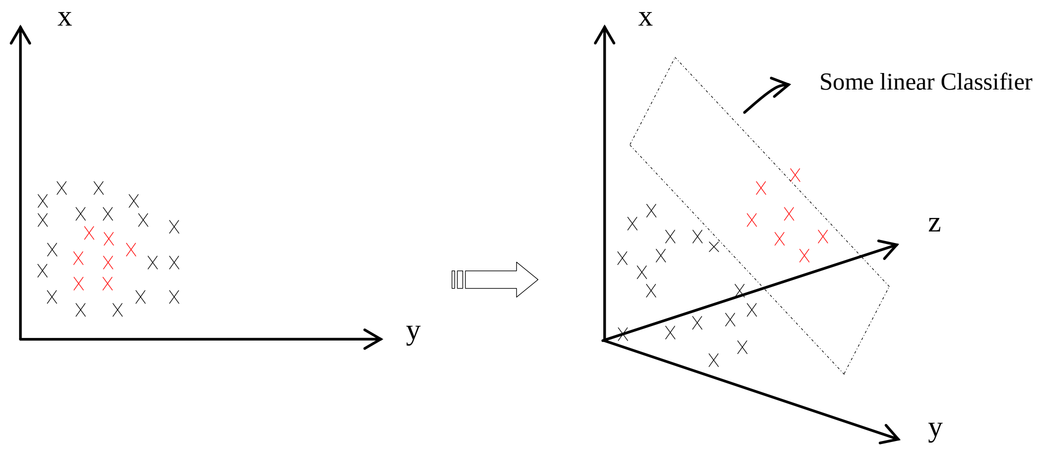


Figure 34: The idea of kernel trick

[13] Source : Wikipedia : http:// wikipedia.org/wiki/Kernel_trick

Kernel trick uses the Mercer's theorem, which states that any continuous, symmetric, positive semi-definite kernel function $K(x, y)$ can be expressed as a dot product in a high-dimensional space. More specifically, if the arguments to the kernel are in a measurable space X, and if the kernel is positive semi-definite , i.e.

$$\sum_{i,j} K(x_i, x_j)c_i c_j \geq 0$$

for any finite subset $\{x_1, ..., x_n\}$ of X and any real numbers $\{c_1, ..., c_n\}$ — then there exists a function $\varphi(x)$ whose range is in an inner product space of possibly high dimension, such that:

$$K(x, y) = \varphi(x).\varphi(y).$$