



Gruber Thomas, BSc

A personalized lightweight event management environment

Master's Thesis

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Computer Science

submitted to

Graz University of Technology

Supervisor

Univ.-Prof. Dipl.-Ing. Dr.techn. Alexander Felfernig

Graz, May 2018

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

Date

Signature

Acknowledgment

I would first like to thank my thesis supervisor Univ.-Prof. Dipl.-Ing. Dr.techn. Alexander Felfernig for the big support during the time of creation of this thesis. He showed great commitment and always gave me important feedback in order to improve my work continuously. Furthermore, he invested a lot of time to test the application and tried to animate other colleagues and institutions to use EVENTHELPR.

Another thank goes to Dipl.-Ing. Dr.techn. Martin Stettinger, who helped me to deploy the application on the final server. In addition to that, he was always on the spot if I had any technical questions or problems.

I also would like to thank the volunteers for taking part in the final evaluation and giving me interesting feedback on the usage of EVENTHELPR.

Finally, I would like to thank my family and my roommates for providing me with continuous encouragement in the past years and through the process of writing this thesis and developing the application.

Abstract

In today's world, the organization of events has become an important task. Due to the fact that there exist many powerful and complex event management solutions, this thesis introduces a very lightweight and easy-to-use solution. EVENTHELPR is a web-based application which can be used to manage events of all kinds.

The core idea of this new application is to centralize the whole information, which is related to an event, on one single page. Furthermore, events can be accessed by unique links which are shared via e-mails for later participation. EVENTHELPR introduces many different settings for events. These settings enable creators of events to customize events in order to satisfy their needs. This high degree of variability leads to a large area of application, in which EVENTHELPR can be used.

Probably the most important component of events is the agenda. In addition to the standard functionality, the agenda can be extended to be interactive. This feature allows participants of events to manage items and to rate items with likes and comments. For the visualization of group preferences within the interactive agenda, a new approach has been introduced. With the help of this new approach, participants are supported in decision making and in prioritizing of items.

In the design phase of the application, it was necessary to identify which combinations of components or features are allowed in which configuration. Therefore, the thesis discusses an approach for modeling variability, which is based on a basic feature model of the EVENTHELPR environment.

A final evaluation examines the understandability of the event settings and the participation process in general. In this chapter, test persons were asked to perform defined tasks and to give informal feedback afterwards.

Kurzfassung

In der heutigen Welt ist die Organisation von Events eine wichtigen Aufgabe geworden. Aufgrund der Tatsache, dass es viele mächtige und komplexe Event Management Lösungen gibt, stellt diese Arbeit eine einfach zu bedienende Lösung vor. EVENTHELPR ist eine web-basierte Anwendung, welche für die Verwaltung von Events aller Art verwendet werden kann.

Die Kernidee dieser neuen Anwendung besteht darin, alle Informationen zu einem Event auf einer einzelnen Seite zusammen zu fassen. Mit eindeutigen Links, die via E-Mails geteilt werden, kann auf Events zugegriffen werden. EVENTHELPR stellt verschiedene Einstellungen für Events zur Verfügung. Diese erlauben es den Erstellern die Events so anzupassen, dass sie ihren Ansprüchen genügen. Das hohe Maß an Variabilität führt zu einem großen Anwendungsbereich, in dem EVENTHELPR verwendet werden kann.

Die vielleicht wichtigste Komponente von Events ist die Agenda, welche zu einer interaktiven Agenda erweitert werden kann. Dieses Feature erlaubt es Teilnehmern von Events, Einträge zu verwalten sowie Einträge mit Likes und Kommentaren zu bewerten. Für die Anzeige von Gruppenpräferenzen innerhalb der interaktiven Agenda wurde ein neuer Ansatz entwickelt. Mit Hilfe dieses neuen Ansatzes ist es möglich, Teilnehmer bei der Entscheidungsfindung und der Priorisierung von Einträgen zu unterstützen.

In der Design-Phase war es notwendig, herauszufinden welche Kombinationen von Features in bestimmten Konfigurationen erlaubt sind. Dazu wird in dieser Arbeit ein Ansatz zum Modellieren von Variabilität diskutiert, welcher auf einem Feature-Modell von EVENTHELPR basiert.

Eine finale Evaluierung untersucht die Verständlichkeit der Event-Einstellungen sowie des Teilnahmeprozesses. Für die Evaluierung wurden Testpersonen gebeten definierte Aufgaben durchzuführen und danach informelles Feedback zu geben.

Contents

Abstract	iv
Kurzfassung	v
1 Introduction	1
1.1 Motivation	1
1.2 Goals	2
1.3 Structure of the thesis	3
2 Analysis of Existing Solutions	4
2.1 Existing software	4
2.2 Differentiation to EVENTHELPR	6
3 Related Work	7
3.1 Decision Making	7
3.2 Group Recommender Systems	8
3.3 Requirements Engineering	10
4 Design	12
4.1 Scenarios	12
4.2 Requirements	14
4.3 Parameters	18
4.3.1 General	18
4.3.2 Configure Agenda	19
4.3.3 Configure Participation	20
4.4 Use Cases	21
5 Managing Variability	26
5.1 Feature Models	26
5.2 Inconsistencies	29

Contents

5.3	Conclusion	32
6	Technologies	33
6.1	Spring Framework	34
6.1.1	Architecture and Features	34
6.1.2	Spring MVC Model	36
6.1.3	Spring Boot	37
6.2	Thymeleaf	38
7	Implementation	39
7.1	Overview	39
7.2	Project Structure	40
7.2.1	Java Source Code	41
7.2.2	Resources	43
7.3	Model Classes	43
8	User Interface and Functionality	46
8.1	Sign-up and Sign-in	46
8.2	Creation of Events	47
8.3	Administration of Events	51
8.4	Participation in Events	52
8.5	Interaction with Events	53
8.6	Managing Events	60
9	Evaluation	62
9.1	Procedure	62
9.2	Results	66
10	Summary	70
10.1	Conclusion	71
10.2	Future Work	72
	Bibliography	73

List of Tables

4.1	List of basic requirements for EVENTHELPR	15
4.2	Use case for event creation (Anonymous)	22
4.3	Use case for event creation (Personalized)	22
4.4	Use case for event administration (Anonymous & Personalized)	23
4.5	Use case for event participation (Anonymous & Personalized)	24
4.6	Use case for event interactions (Anonymous & Personalized)	25
4.7	Use case for user sign-up and sign-in (Anonymous & Personalized)	25
5.1	Summary of analysis operations for feature models. FM_{nr} denotes a non-redundant feature model.	31

List of Figures

4.1	Basic model of problem domain	14
5.1	Variability model of EVENTHELPR	28
5.2	Example of a faulty feature model	30
6.1	Architecture of Spring Framework	35
6.2	Request handling in Spring MVC	37
7.1	Basic project structure of web application	40
7.2	Class diagram of all model classes	44
7.3	Class diagram of model classes which are related to the agenda item class	45
8.1	Sign-in and sign-up form	47
8.2	Personal user area	48
8.3	Snippets of event creation form	49
8.4	Different types of contact information in the event creation form	49
8.5	Overview page of newly created event	50
8.6	Snippet of event administration page containing the first section	51
8.7	Snippet of event administration page containing the summary input	51
8.8	Event participation page with two different ways to participate	52
8.9	Snippet of event page with header and location information	54
8.10	Fully interactive agenda with a list of items	56
8.11	Detail page of interactive agenda item including discussion	57
8.12	User interface for uploading images to an event	59
8.13	Snippet of event overview page for user	60

1 Introduction

The organization of events is a task that many people have to deal with in the business or private domain. Although there are existing event management tools, this thesis comes up with a new lightweight approach.

1.1 Motivation

In this day and age, people tend to organize their lives with the help of digital media. Since smart phones conquered the market, everyone can access all kind of information everywhere and at any time. The majority uses modern communication tools like instant messaging and share information on social media platforms. Although new technologies raise communication to a new level, e-mails are still a proven medium to spread information. In cases where personal data like complete name or phone numbers are not known, an address is enough to send an e-mail.

If a group of people wants to organize something (for example a meeting), there are many ways to realize that. A possibility would be to send out an initial e-mail and handle the whole information flow with subsequent messages. In this case, it is very difficult to keep an overview about what is going on. Members have to search the e-mail history to look up any information which is needed yet. If the people know each other better, another approach could be to create a group in an instant messaging application. In this group everybody can write messages and share links, locations and files. No matter which variant is used, the main disadvantages remain the same. Imagine there are many participants and even more messages that contain important or unimportant information. It is nearly impossible to manage or find the information which is needed right now.

1 Introduction

In order to make life easier, there exist many event management tools which will be analyzed shortly in Chapter 2. The variety of tools range from standalone applications to web based solutions and of course mobile applications. Most of them are very powerful and provide all kinds of functions which are necessary to fit various use cases. If someone wants to use such a tool, it is mostly required to create an account and pay some fees. For the usage of a standalone application, a license has to be bought. This type is often relevant for companies which do not want to store internal or confidential data in the world wide web.

1.2 Goals

Due to the situation mentioned above, the idea was to create a lightweight event management environment. This new system is a web-based application and is called EVENTHELPR. EVENTHELPR is absolutely free to use and can be accessed on-line: <https://www.eventhelpr.com>.

With EVENTHELPR it is possible to create events such as meetings, workshops, conferences, travels and many more. The main advantage of the system is that an event can be shared via just one e-mail and the participants of the event can view all details on one page and get the information they need. All relevant places regarding the event are displayed on one map. In this map, the detailed location and many other places such as accommodations and restaurants are visible. The application provides a detailed agenda, which could also be extended to an interactive agenda where participants are able to rate items with likes and comments. With these additional features, the application can help participants to make group decisions and to prioritize tasks.

The creator of an event has many options available to customize it for several usage scenarios, which will be explained later in Chapter 4. Another main concept of EVENTHELPR is the anonymous mode. It is possible to use the application without creating an user account or specifying any e-mail address. This behavior is achieved by using auto-generated unique links to access each event. The advantage of using EVENTHELPR with an account

1 Introduction

is that users get an overview of all events which have been created or in which a user is participating.

The goals of the thesis are the following:

- Realize EVENTHELPR as a web-based application
- Introduce several parameters for the application to make it highly customizable
- Create a consistent and easy-to-use user interface for the application
- Evaluate the application with focus on the understandability of the parameters and the participation process

For the realization of EVENTHELPR as a web-based application, some technical requirements regarding used frameworks and technologies had been predefined. To be exact, the usage of the *Spring Framework* combined with the *Thymeleaf* template engine was mandatory. Further, the use of *Bootstrap* for the front-end of the web application was required.

1.3 Structure of the thesis

This thesis consists of 10 chapters. The first chapter (Chapter 1) gives a brief insight into the emergence of the project and the basic idea behind it. After that, some existing software in the context of event management tools is listed and analyzed shortly in Chapter 2. In Chapter 3, the focus is on the discussion of related topics which are *Decision Making*, *Group Recommender Systems* and *Requirements Engineering*. The design phase of the project including use cases and requirements is described in Chapter 4. Since managing variability of software is an important aspect in the Requirements Engineering (RE) context, an approach to model variability based on the EVENTHELPR environment is discussed in Chapter 5. The two subsequent chapters (Chapter 6 and Chapter 7) deal with the used technologies and give an overview about the implementation of the web application. Then, the user interface and the functionality of the application is explained in detail on the basis of screen shots in Chapter 8. At the end, the results of the evaluation are presented in Chapter 9 before a short summary and possible future work in Chapter 10 conclude the thesis.

2 Analysis of Existing Solutions

Before the project could be started, it was necessary to analyze existing tools and platforms that are dealing with event management topics. In order to get a general overview of supported features, application areas and requirements, existing web-based and standalone solutions have been examined. Since nearly all solutions are not free to use, the main focus of the analysis was placed on the advertised features of the respective solution and the accessibility for private users.

2.1 Existing software

In the following section, a couple of existing software solutions are listed. Every solution is characterized from an objective point of view by just stating different aspects which have to be considered.

Facebook (<https://www.facebook.com>)

- Simple definition and publishing of events
- Participants statistics
- Availability of the Facebook community
- Not suitable for business context
- Sign in to Facebook necessary

Evensi (<https://www.evensi.com>)

- Free platform for events of all kinds in the whole world
- Strongly connected to Facebook
- Essentially designated for public events where many people should participate

2 Analysis of Existing Solutions

Converia (<https://www.converia.de>)

- Very complex and powerful software with the focus on conferences
- Supports registration of participants, submission of articles, reviews, cash management
- No communication between participants
- No location information
- No flexible adaptation of agenda items
- Not free to use for customers

Evention (<https://www.evention.eu>)

- Extensive solution for event management
- Not free to use
- Not suitable for private use

Eventleaf

- Focused on event administration
- Interactive agenda and visualization of location and surrounding
- Rather not qualified for administration of events in private scenarios

Meeteor (<https://www.meeteor.com>)

- Mainly designed for meetings
- Agenda including notes and actions
- Not free to use
- Registration needed

Solid (<https://getsolid.io>)

- Meeting agenda including notes and actions
- All information available on one page
- Integration of calendars
- Not free to use

Cisco Spark (<https://notes.ciscospark.com>)

- Meeting agenda including notes and actions
- All information available on one page

2 Analysis of Existing Solutions

- Integration of calendars
- Access to events handled by sending links via e-mail
- Registration required, but free to use

Less Meeting (<https://lessmeeting.com>)

- Meeting agenda including actions
- Integration of calendars and e-mail possible
- Not free to use

Meetin.gs (<https://www.meetin.gs>)

- Planning of meetings
- Integration of calendars and contacts possible
- Live communication and attachments are supported
- Not free to use

Meeting King (<https://meetingking.com>)

- Meeting agenda including actions
- Attachments are supported
- Not free to use and registration required

2.2 Differentiation to EventHelp

The main difference between most of the listed solutions and EVENTHELPR is that the usage of EVENTHELPR does not necessarily require any account and is absolutely free for everyone. Furthermore this new application combines useful features and allows the creation and administration of events in a simple way. EVENTHELPR also supports joint decisions related to the arrangement of an agenda. Another important point is that EVENTHELPR does not only focus on event or meeting management but also on interaction between participants in form of postings, images or attachments.

3 Related Work

In the upcoming chapter, some related topics are surveyed. As mentioned in Section 1.2, EVENTHELPR introduces an interactive agenda with the possibility to rate items with likes and comments. This feature enables the application to assist participants of events in the decision process. Since agenda items may represent various types of information, there are many different applications where the interactive agenda can be applied. Especially, the interactive agenda refers to *Decision Making* in general, *Group Recommender Systems* and *Requirements Engineering*.

3.1 Decision Making

Decision making is regarded as a process in which we go through a set of thinking and other processes to identify various options at first before one of them is chosen [8]. There exist very simple decisions and more complex ones. In the decision process, the decision maker consciously or subconsciously goes through a set of thinking to obtain a final decision. Decisions can be made individually or as part of some group. Furthermore, there are different types of decisions and ways how a decision is made. In a rational decision making process a recognized series of steps is systematically followed to choose the final option. There exist several approaches to deal with rational decisions. All of these approaches rely on a common theme. In this theme, information or facts about a situation are gathered and assessed in order to make a rational choice. In the real world, most of the decisions are very complex and can not be characterized by artificially rational steps. Many factors like fuzziness, interaction between machines and people involved, problems, technologies and social norms influence decisions. The outcome of a decision process in this case can be described as a mixture of these

3 Related Work

factors at a point in time and does not refer to a rational process any more. In addition to the factors mentioned above, there are a lot more things to consider when making complex decisions. The decisions strongly depend on the relative power basis and the personal values of the decision makers. Further, many other things like time, affected people, law, decision environment, thinking traps and rational approaches have to be considered [8].

In group decisions, the outcome is a choice which has been made collectively by all contributing individuals of the group. Compared with individual decisions, the decision process has additional dependencies such as social influence, group norms and the distribution of information among the group members. As information could be either shared among all group members or given to selected group members only (unshared information), the behavior of groups in decision making is varying. Some biased sampling studies suggest that unshared information is not typically pooled in discussions within the group. Due to this phenomenon, group decisions are based on the evaluation of information which is available. Another interesting aspect in group decisions are group norms. Depending on whether the group follows consensus norms or critical norms, shared and unshared information is valued differently. As a consequence, the decision quality is affected by the differential evaluation of information [33].

The items in the interactive agenda of EVENTHELPR could represent alternatives in a group decision process. The provided information should be available to all members, but it is possible to pool unshared information via postings. These postings can either be positive, negative or neutral and could initiate a discussion. Finally, both the shared information as well as the unshared information contribute to the final decision. The outcome may depend on the group norm and on the amount of unshared information that members provided before.

3.2 Group Recommender Systems

The following section is based on the book [17] and gives a short overview of how some aspects of group recommender systems are suitable for the

3 Related Work

interactive agenda of EVENTHELPR.

A recommender system in general is a decision support system that helps users to identify options or solutions which match their preferences best [14]. Recommender systems can support users in decision making processes in different ways. They can either reduce a large number of alternatives to a so-called consideration set or help the user to finally select one item of the consideration set by providing appropriate representations and explanations of items. In group recommender systems, the preferences of all engaged users have to be taken into account. In contrast to a single-user recommendation scenario, the recommendation task becomes more challenging and complex.

There exist many different basic recommendation approaches for individual users which can be integrated into corresponding group recommendation scenarios. A *Collaborative filtering* recommender uses the preferences of determined nearest neighbors to predict how the actual user will like an item which has not been considered up to now [10, 28]. The *Content-based filtering* approach is based on the similarity of the content of already consumed items and new items [21]. These two approaches rely on item ratings respectively textual item descriptions. *Constraint-based recommendation* is based on deep knowledge about the items and relies on user requirements (preferences) which are internally represented as constraints [11, 16]. In *Critiquing-based* recommender systems, the user is able to navigate through the item space by specifying critiques starting from a recommended reference item [25]. Further it is possible to combine these approaches in different ways to develop hybrid recommender systems.

The pros and cons analysis of items in the interactive agenda could be seen as a simple multi attribute utility theory (MAUT) analysis [19]. MAUT is applied on the basis of a defined utility scheme to evaluate and rank items. In the case of agenda items, the number of positive and negative postings represent two interest dimensions. The interest dimensions are related to the given set of items in order to determine a user-specific utility of each individual item.

A very important part of recommendations are explanations and visualizations related to recommended items. Explanations should help users to make better decisions and to get a better understanding of the recommended

3 Related Work

items [7]. Furthermore, explanations are used to increase a user's trust in and overall satisfaction with the recommender system. In group recommender systems, explanations try to show how the interests of each group member are taken into account in order to create recommendations. Additionally, visualizations may boost the impact of explanations and improve the user's interpretation of the results [39].

For the visualization of group preferences in the context of the interactive agenda in *EVENTHELPR*, a new approach has been introduced. This approach uses a horizontal bar on a per item basis where the number of positive, negative and neutral postings is shown. Additionally, the calculated utility of each item is displayed. A detailed description on this new approach is given in Section 8.5.

3.3 Requirements Engineering

Requirements engineering (RE) is an important phase in the development process of software projects and includes the definition, documentation and maintenance of requirements [32]. Requirements can be functional or non-functional and describe needs concerning the functionality and quality of a software. Persons who are involved in a project have to deal with different types of decisions in RE processes. These are decisions which refer to a single requirement (quality, classification and properties) or deal with many requirements such as preference decisions. Since decisions in context of RE processes become more and more difficult with increasing size and complexity, there has been developed an intelligent recommendation & decision support system [20].

The *INTELLIREQ* environment exploits recommendation technologies to figure out and prioritize high-level requirements in software projects. In a specification book, which is generated by the application, all necessary information units like requirements, dependencies and a release plan are summarized [31].

OPENREQ is a currently ongoing project that focuses on the development of intelligent recommendation and decision technologies for different phases

3 Related Work

of RE. The project uses AI-based techniques to pro-actively support involved users within the scope of RE [18].

In the context of EVENTHELPR, agenda items could represent software requirements which consist of a detailed description and an assigned user. Since users are able to provide positive and negative postings within items, the pros and cons analysis of items leads to a group-based prioritization of the requirements.

4 Design

The following chapter elaborates the basic ideas and concepts for the application from Section 1.2 and provides a rough scaffold for later implementation. At first, a wide range of usage scenarios for the application is listed. In order to support all of those different scenarios, it is necessary to introduce several parameters which could be set in the creation process of events (see Chapter 5). Based on the general purpose of the application and the description of the scenarios, software requirements in form of used entities, underlying attributes and features have been derived. Furthermore, basic interactions between end users and the system are described in order to get a better understanding of what is about to happen when those interactions are carried out. In the following, such interactions will be denoted as *use cases*.

4.1 Scenarios

The principal purpose of EVENTHELPR is the ability to handle events of all kinds. The most important information related to basic scenarios is where and when the event takes place, the agenda and who is participating. If users should be able to take part in a decision process, it is required that agenda items can be modified and in some cases also rated by participants. Furthermore, it could be necessary to assign several agenda items to one or more participants. This feature allows the monitoring of the completion of defined tasks within given deadlines. Another fundamental part is the communication between participants. In addition to a contact function via e-mail, postings that appear on the main event page could facilitate the exchange of information. To provide a better overview of possible activities outside of the main event, the creator of the event should be able

4 Design

to specify side events such as dinners or give some hints where restaurants or accommodations can be found.

Using the above description as a starting point, the following more specific scenarios have been derived:

- Project meetings with external participation (for example international partners)
- Internal project meetings with definition and prioritizing of to-dos
- Meetings with external companies and partners where conversations and presentations take place
- Workshops and conferences
- Talks and informative events
- Board or shareholders meetings
- Paper writing for conferences

Since the whole data which is required to describe an event could be easily generalized, EVENTHELPR is able to deal with further scenarios as well. Nearly every personal event can be represented if the creator maps the existing information meaningfully into the provided inputs. With the introduction of a configurable participation mask, creators of events are able to gain more user specific data of the participants. This feature extends the possible areas of application for EVENTHELPR as well.

Thus, it appears that EVENTHELPR could also be used for the following purposes:

- Weddings
- All kinds of parties (for example a birthday party)
- Trips and Excursions
- Travels
- Travel summaries
- Organizing orders

In the travel scenario, agenda items usually represent destinations. Especially for round trips, all locations can be used to display a complete map with all relevant places in it. With the possibility of adding images to events and agenda items, an event could also act as a guide for a certain journey where places of interest are illustrated and described.

4.2 Requirements

After analyzing relevant scenarios, the next task was to figure out a concept and to define requirements for the new software. For the derivation of requirements from the scenarios, all different aspects of each individual scenario have to be considered.

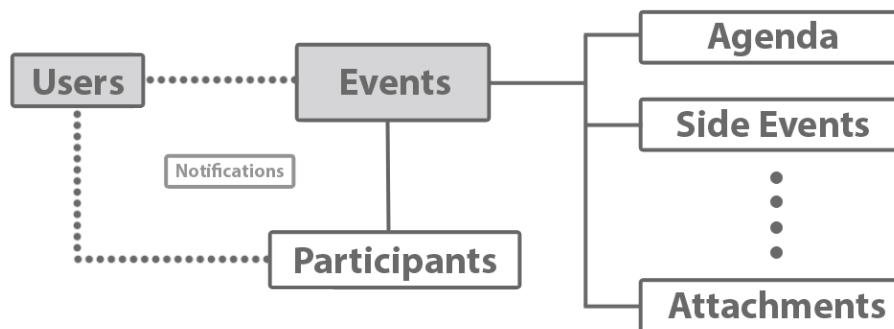


Figure 4.1: Basic model of problem domain. Solid lines represent a connection between entities while dotted lines represent an optional connection.

Figure 4.1 shows an overview of entities which are involved in a basic model of the problem domain. As expected, *Events* are the core entities of the application. Every event consists of attributes and additional entities like *Agenda Items*, *Side Events* and *Attachments*. Users (registered or non-registered) are able to create, edit and view events. Furthermore, every user who has access to a given event should be able to participate in the event and interact with the event. Since the creation of a user account is not mandatory in order to use the application, the connection between *Users* and *Events* respectively *Participants* is marked as optional. If the application is used without a user account, it is sufficient to provide an e-mail address to be able to create events or participate in events. The creation of events should be even possible without providing any contact information which conforms to an anonymous usage. Thus, it is necessary to make events accessible by a unique identifier. With the help of those unique identifiers, unique links for every single event can be derived which are then sent via e-mail and enable recipients to access the event. For example, the creator of an event receives an e-mail with one link to view the event and another link

4 Design

to edit the event. After creating an event, it is possible to invite contacts via e-mail. Invited contacts are then able to participate in the event by following the provided link from the e-mail. Additionally, participants of events could receive *Notifications* for the event via e-mail (assuming that an e-mail address has been provided).

In Table 4.1 the basic functional requirements which the application should cover are listed.

Actor	Description
User	Create event
User	Administratre event
User	Delete event
User	Send out invitations for events via e-mail
User	Contact other event participants via e-email
User	Participate in event
User	Create, delete and view agenda items of event
User	View side events of event (including participation)
User	Upload files to event
User	Upload images to event
User	Add postings and replies to event
User	Create user account
User	Sign-in to user account
Registered User	Edit user account
Registered User	Change password for user account
Registered User	Manage events
System	Send notifications regarding events via e-mail

Table 4.1: List of basic requirements for EVENTHELPR

Some of those requirements are refined later in Section 4.4 by combining one or more aspects in several use cases to get a better understanding of what is about to happen.

4 Design

The next step was to define attributes and entities which have to be stored for every event in order to represent events and support the desired functionalities. Entities itself consist of another attributes and can be seen as additional information units within events. In contrast to standard attributes, the relation between events and entities is a *One-to-many* relationship which means that one event could have one or more elements of each entity.

Attributes. The primitive attributes which describe an event are the following:

- Name
- Short description
- Description
- Link for further information
- Title image
- Date and time
- Location
- Summary
- Settings (Parameters)
- Access data
 - Unique identifiers
 - Info about creator
 - View count
 - Other meta-data

Entities. The different entities which can be seen as information units of events are listed here:

- Agenda items
 - Title
 - Description
 - Link for further information
 - Date and time
 - Presenter/Contributed by
 - Status
 - Location
 - Attachments

4 Design

- Likes
 - Postings (Name, title, text and type)
 - Images with caption
- Side events
 - Name
 - Description
 - Link with further information
 - Date and time
 - Location
 - Participants (Name and e-mail)
- Polls (Name and link)
- Accommodations (Name, link and location)
- Restaurants (Name, link and location)
- Attachments (Name and file)
- Sponsors (Name, link and logo)
- Interesting places (Location and link)
- Images (Caption and file)
- Participants (Name, e-mail and individual additional fields)
- Postings (Name, title and text)
- Invitations (E-mail)

The defined attributes and entities just give a general overview of the most important information which is needed to represent events. In case of anonymous usage, this basic structure would be enough to handle all necessary actions within the system. For the purpose of user accounts, it is required to store representative data about the person which would like to sign-up.

User attributes. The attributes which are needed to enable sign-up and sign-in for users within the system are the following ones:

- E-mail address
- Name
- Password
- Profile picture
- Settings
- Contacts

- Notes (event specific)

4.3 Parameters

In order to make events even more customizable, it should be possible to define the behavior and the allowed actions for every event separately. Therefore, a variety of parameters (also called “Settings” in the event context) have to be introduced. Those parameters are represented by boolean values and should be adjustable either in the creation process or afterwards in the administration context. Furthermore, the parameters will be divided into three logical groups later, namely “General”, “Configure agenda” and “Configure participation”. In this section, all different options and their meaning are explained in detail.

4.3.1 General

Allow participants to upload images

Allow/Deny participants to upload and remove images from the event. Usually only the creator of the event is authorized to perform this action.

Allow participants to upload files

Allow/Deny participants to upload and remove attachments from the event. Usually only the creator of the event is authorized to perform this action.

Allow participants to invite friends

Allow/Deny participants to invite friends to the event. Usually only the creator of the event is authorized to perform this action.

Allow participants to contact participants via e-mail and to create postings

Allow/Deny participants to contact participants of the event via e-mail and to write postings or replies.

Notify participants via e-mail if a new posting has been added

4 Design

Enable/Disable e-mail notifications for all participants of the event if someone has added a new posting or a reply.

Notify participants via e-mail if event has been updated

Enable/Disable e-mail notifications for all participants of the event if it has been updated (updated in this context means the change of event information as well as adding new elements like images or postings to the event). Additionally, an interval to check for event updates can be specified.

Notify participants via e-mail before the event starts

Enable/Disable e-mail notifications for all participants of the event before it starts. Additionally, a number of days can be provided to specify how many days before the notification gets triggered.

4.3.2 Configure Agenda

Allow participants to upload images in agenda entries

Allow/Deny participants to upload and remove images from agenda entries of the event.

Allow participants to create and edit agenda entries

Allow/Deny participants to create and edit agenda entries of the event and to write postings within entries. If this option is selected, the agenda becomes more interactive and allows participants to take part in the decision process.

Allow participants the rating of agenda entries with likes and comments

Allow/Deny participants to rate agenda entries by the help of positive or negative postings and likes. As a result it is possible to make group decisions or to prioritize items.

Rename "Agenda" to...

4 Design

This parameter enables creators of events to customize the display name of the agenda. An example usage would be in a travel scenario where agenda items usually represent destinations.

4.3.3 Configure Participation

Participation in event is only possible for logged-in users with invitation

In many cases, it is necessary to regulate the access to the event. Typically, everyone who has the event link could access the event and share this link outside of the application (for example just forward the e-mail). If this parameter is set, only registered users which are logged in and invited can access the event to ensure that no other unwanted users are able to get information about the event.

Show list of event participants on event page

Show/Hide the list of participants on the event page.

Allow participants to remove other participants (except registered participants)

Allow/deny participants to remove other participants from the list of event participants. Registered users can only be removed by themselves.

Show list of event participants on participation page

Show/Hide the list of participants on the participation page. If this option is selected, potential participants can see who is already participating in the event.

Show link to event page after successful participation

Show/Hide the link to the event page after participation has been created successfully.

E-Mail address for participation is...

This parameter enables creators of events to customize the need of an e-mail address for event participation.

Info text for participants

This parameter enables creators of events to specify a custom information text which is displayed in the participation form.

Input fields in addition to standard fields (Name/E-Mail)

This parameter enables creators of events to add additional input fields to the participation form. Every field has a name and is either of type input field or type check box.

Change text for "participate button"...

This parameter enables creators of events to change the text which is displayed on the participate button in the participation form.

4.4 Use Cases

The following use cases show the basic actions which end users of the system could perform. All of those actions are suitable for the different scenarios which were described earlier. The focus is on the basic procedure of the several actions in order to keep track of which operations have to be carried out. Since it is possible to use `EVENTHELPR` without providing any personal data like an e-mail address, some use cases have to distinguish between anonymous and personalized usage.

The Tables 4.2 and 4.3 show the basic use cases for event creation. In the anonymous case it is not required to provide any e-mail address or other personal information whereas in the personalized case the user might just enter an e-mail address or authorize with a user account. The main difference between these two possible ways is that the user is responsible for storing the event access links if there is no e-mail address provided.

In the Table 4.4 it is indicated how a user can manage an event which has been created beforehand. If the event was created by a logged in user, it appears in the personal list of events and can be accessed from there.

Event participation is a more complex use case because it depends on the parameter *Participation in event is only possible for logged-in users with*

4 Design

Use Case: Event creation	Anonymous
Description	
How can a user create a new event without revealing an e-mail address	
<p>User clicks "Create event" on start page or "New" in navigation bar</p> <p>User adds all available information to event and adjusts settings</p> <p style="padding-left: 40px;">User does not provide any e-mail address</p> <p style="padding-left: 40px;">User clicks on submit button</p> <p style="padding-left: 40px;">System creates event and displays overview</p> <p style="padding-left: 40px;">System does not send an e-mail to creator</p> <p style="padding-left: 40px;">User can send invitations on overview page</p> <p>User should copy unique links for event access in the future</p>	

Table 4.2: Use case for event creation (Anonymous)

Use Case: Event creation	Personalized
Description	
How can a user create a new event with an e-mail address	
<p>User clicks "Create event" on start page or "New" in navigation bar</p> <p>User adds all available information to event and adjusts settings</p> <p>User provides e-mail address (automatically set if user is logged in)</p> <p style="padding-left: 40px;">User clicks on submit button</p> <p style="padding-left: 40px;">System creates event and displays overview</p> <p style="padding-left: 40px;">System sends an e-mail with unique links for later event access</p> <p style="padding-left: 40px;">User can send invitations on overview page</p>	

Table 4.3: Use case for event creation (Personalized)

invitation. Table 4.5 shows the procedure of participating for both options. If the parameter is set to *true*, users are only allowed to participate if they are currently logged in to the system and an invitation has been sent to the corresponding e-mail address. This behavior enables event creators to get more control over participants and subsequently over all actions which are

4 Design

Use Case: Event administration	Anonymous & Personalized
Description	
How can a user manage an event in EVENTHELPR	
User navigates to administration link obtained from the e-mail	
OR	
If event has been created by a logged in user, the creator can access the event administration page via "My events" page or directly from the event page	
User can edit all information, change settings and manage event	
User clicks on save button	
System saves updated event	

Table 4.4: Use case for event administration (Anonymous & Personalized)

performed inside an event. If the parameter is set to *false*, users are able to participate by providing a name and an e-mail address (depending on the settings). Regardless of this parameter, users could additionally provide further information if extended participation fields are defined by the creator of the event. The general benefit of participating as a logged in user is that the event appears in the list of personal events and can be found easily.

Table 4.6 show how a user can interact with an event. Depending on whether a user is signed-in to an existing user account or not, there are two different ways of viewing an event. If the event could be associated with an account, the user is able to access it from the list of personal events. Otherwise, the only possibility is to follow the event link obtained from the confirmation e-mail or copied from elsewhere (in the anonymous case the link has to be stored manually). On the event page, the user is able to perform several actions regarding the current event like for example to upload images. The available actions strongly depend on the actual settings of the event.

The user registration and login procedure is kept simple and results in a straight forward use case which could be found in many other software solutions. The most interesting part in Table 4.7 is mainly the verification of a newly created user account.

4 Design

Use Case: Event participation	Anonymous & Personalized
Description	
How can a user participate in an event	
User navigates to participation link obtained from e-mail or perhaps other media	
System displays participation page	
Case 1: Participation is only possible for logged-in users with invitation	
User login is required to continue	
After successful login, system checks for existing invitation	
User provides additional information for participation if specified by creator of event	
If all preconditions are fulfilled, system adds new participant (with name and e-mail from account) to event	
System unlocks view link and user can display event	
Additionally, this link is sent in a confirmation e-mail	
Case 2: Participation is possible for everyone with link	
User provides name (mandatory) and email (mandatory or optional)	
User provides additional information for participation if specified by creator of event	
System unlocks view link and user can display event	
Additionally, this link is sent in a confirmation e-mail	
OR (E-Mail is not required)	
User provides name (mandatory)	
User provides additional information for participation if specified by creator of event	
System unlocks view link and user can display event	
User should store this link for event access in the future	

Table 4.5: Use case for event participation (Anonymous & Personalized)

4 Design

Use Case: Event interactions	Anonymous & Personalized
Description	
How can a user interact with an event	
User clicks on event link obtained from e-mail or perhaps other media	
OR	
User selects event in overview (only possible with user account)	
System displays event page	
User is able to perform the following actions within the event:	
<ul style="list-style-type: none"> • Remove participation and participate again • View, create or remove agenda items • View side events and participate therein • Upload or remove images • Create or remove postings and replies 	

Table 4.6: Use case for event interactions (Anonymous & Personalized)

Use Case: User sign-up & sign-in	Personalized
Description	
How can a user create an account in EVENTHELPR	
User clicks on "Create account" in navigation bar	
User fills in a valid e-mail address and a password	
User accepts privacy policy	
User clicks on submit button	
System creates user account and sends verification e-mail	
User clicks on activation link in verification e-mail	
System activates user	
User enters credentials and clicks login	
System logs in user and redirects to "My events" page	

Table 4.7: Use case for user sign-up and sign-in (Anonymous & Personalized)

5 Managing Variability

The following chapter is based on the article Consistency Management Techniques for Variability Modeling [15]. My contribution in this article focused on the working example, the EVENTHELPR variability model, and the working example for explaining diagnosis approaches.

Managing variability of software is an important aspect in the Requirements Engineering context [27, 26, 29, 30]. In the following, an approach for modeling variability which is based on an example of the EVENTHELPR environment is discussed.

The main target of variability modeling is to figure out which combinations of components or features are allowed when delivering a software [5, 24, 36]. With the help of conflict detection [23] and model-based diagnosis [34] in feature models, it is possible to automatically identify inconsistencies [24, 40, 2]. Furthermore, these approaches can be used for automated testing and debugging, redundancy detection, and the improvement of software quality [13].

The approach discussed in the following was used as a basis to define and manage the variability of EVENTHELPR features.

5.1 Feature Models

The design of feature models is an error-prone activity that results from a cognitive overload of engineers engaged in the development and maintenance of such models [6]. Feature models can be built by using modeling concepts which are features, relationships between features (represented as constraints [5]) and cross-hierarchy constraints [24]. Feature models

5 Managing Variability

are categorized in three different types namely basic feature models [24], cardinality-based feature models [9] and extended feature models [3]. Features are the main elements in a feature model and such a model defines combinations of features that can be jointly included in a configuration [13]. The inclusion or exclusion of features into a corresponding configuration is denoted by a boolean value $\{true, false\}$. Additional restrictions on possible combinations of features are specified via constraints which are partially defined by relationships. For the formalization of feature configuration tasks as Constraint Satisfaction Problems (CSPs) [38], the following definitions are necessary:

Definition 1 (Feature Configuration Task). A feature configuration task can be defined by a triple (F, D, C) where F represents a set of features and each feature $f_i \in F$ has an associated domain $dom(f_i) \in D = \{true, false\}$. Finally, $FM \cup CREQ$ represents a set of constraints $c_i \in C$ where FM are domain model constraints contained in the feature model and $CREQ$ is a set of customer requirements used to specify customer-specific requirements with regard to a feature configuration ($FM \cup CREQ = C$).

Definition 2 (Feature Configuration). A feature configuration is a complete set of value assignments ($val(f_i) \in \{true, false\}$) to features $f_i \in F$. A feature configuration is *consistent* if the value assignments are consistent with the constraints in C . Furthermore, it is regarded as *valid* if it is consistent and *complete* (each variable has a corresponding value assignment).

A feature configuration is a solution to a feature configuration task. Different configurations can be determined based on the definition of a feature configuration task.

There are six different types of constraints which are typically used in feature modeling [5]:

- *Mandatory* relationship between f_2 and f_1 - if a feature f_1 is part of the configuration, f_2 must be also part of the configuration (and vice-versa)
- *Optional* relationship between f_2 and f_1 - if a feature f_1 is part of the configuration, f_2 can be part of the configuration. In the other direction, f_1 must be included if f_2 is part of the configuration

5 Managing Variability

- *Alternative* relationship between f_1 and a set $\{f_{11}, f_{12}, \dots, f_{1n}\}$ - exactly one of a given subset of features has to be included in a configuration (“xor” semantic)
- *Or* relationship between f_1 and a set $\{f_{11}, f_{12}, \dots, f_{1n}\}$ - at least one of a given subset of features has to be included in a configuration (“or” semantic)
- *Requires* relationship between f_1 and f_2 - inclusion of f_1 requires the inclusion of f_2 (regarded as one type of cross-tree constraint)
- *Excludes* relationship between f_1 and f_2 - inclusion of f_1 excludes f_2 and vice-versa (regarded as one type of cross-tree constraint)

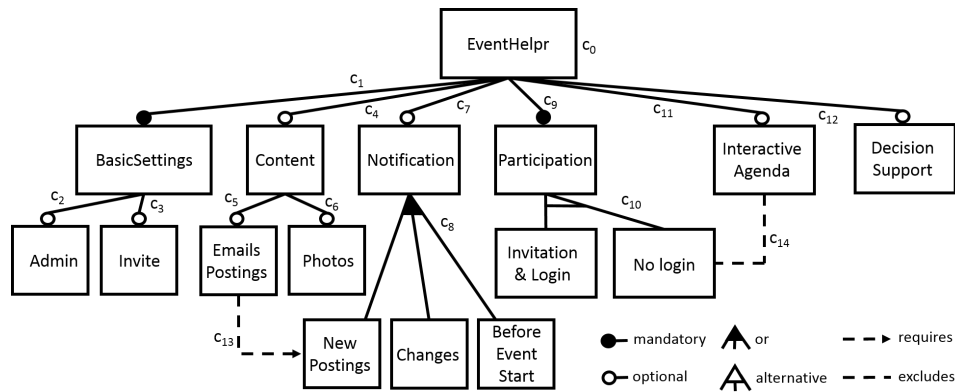


Figure 5.1: Variability model of EVENTHELPR. The different variants describe ways in which the application can be presented to end-users.

A feature model of the EVENTHELPR environment is shown in Figure 5.1. In this model, only some basic features are considered to keep it simple.

From the feature model, a constraint-based representation can be derived [38]. This representation consists of the features F , a domain D and domain model constraints FM . There exist many valid value assignments for features that result in different possible configurations (solutions) for the feature model.

- $F = \{eventhelpr (eh), basicsettings (bas), admin (adm), invite (inv), content (con), emailspostings (email), photos (phot), notification (not), newpostings (newpost), changes (ch), beforeeventstart (bef), participation (part), invitationlogin (invit), nologin (nolog), interactiveagenda (intag), decisionsupport (dec)\}$.

5 Managing Variability

- $D = \bigcup_{f_i \in F} \text{dom}(f_i)$.
- $FM = \{c_0 : eh = true, c_1 : eh \leftrightarrow bas, c_2 : adm \rightarrow bas, c_3 : inv \rightarrow bas, c_4 : con \rightarrow eh, c_5 : email \rightarrow con, c_6 : phot \rightarrow con, c_7 : not \rightarrow eh, c_8 : not \leftrightarrow newpost \vee ch \vee bef, c_9 : part \leftrightarrow eh, c_{10} : part \leftrightarrow (invit \wedge \neg nolog \vee \neg invit \wedge nolog), c_{11} : intag \rightarrow eh, c_{12} : dec \rightarrow eh, c_{13} : email \rightarrow newpost, c_{14} : \neg(intag \wedge nolog)\}$.

5.2 Inconsistencies

Feature models may have undesirable properties which trigger an unintended behavior. Such properties lead to inconsistencies on the logical level. Inconsistent models could include conflicts which are either induced internally by model constraints or externally [23]. With the following definitions, it is possible to explain inconsistencies on the basis of minimal conflict sets:

Definition 3 (Conflict Set). A conflict set $CS \subseteq FM$ is a set of constraints s.t. $\text{inconsistent}(CS)$. CS is *minimal* if $\neg \exists CS' : \text{conflict set } (CS')$.

Definition 4 (Diagnosis). A diagnosis $\Delta \subseteq FM$ is a set of constraints s.t. $\text{consistent}(FM - \Delta)$. Δ is *minimal* if $\neg \exists \Delta' \subset \Delta : \text{diagnosis } (\Delta')$.

Conflicts can be resolved by the deletion of at least one element from the minimal conflict set. A hitting set is defined as a set that contains all elements needed to delete at least one element from each existing conflict. Hitting sets (diagnoses) can be extracted from the individual paths of the tree in *Hitting Set Directed Acyclic Graphs* [34], which are constructed by following a standard diagnosis approach of model-based diagnosis [34]. For the creation of conflict sets and diagnoses, specific parts of the constraints C are analyzed. If the focus is on inconsistencies induced by faulty constraints that are part of the feature model FM , the resulting conflict sets and diagnoses are composed of constraints from FM [12].

There exist several feature model properties which make a model ill-formed (see also Table 5.1). Each property is shortly explained based on an adapted version of the EVENTHELPR feature model (see Figure 5.2).

5 Managing Variability

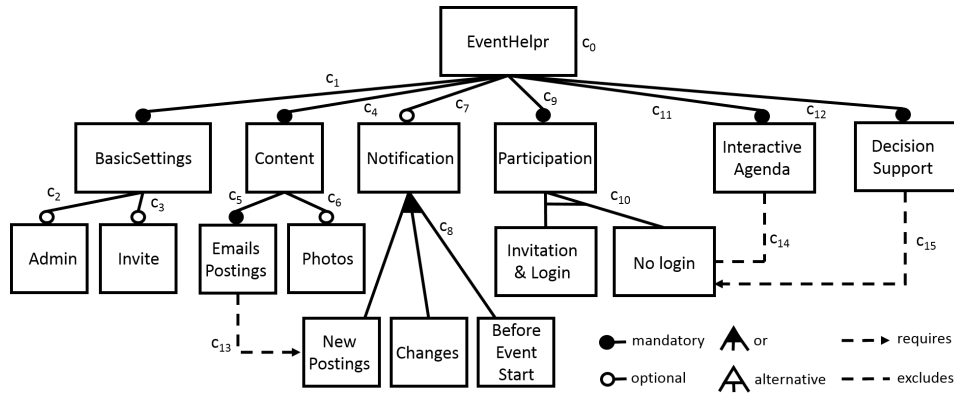


Figure 5.2: Example of a faulty feature model.

From the adapted feature model, a different constraint-based representation [38] than the representation in Section 5.1 can be derived.

- $F = \{eventhelpr (eh), basicsettings (bas), admin (adm), invite (inv), content (con), emailpostings (email), photos (phot), notification (not), newpostings (newpost), changes (ch), beforeeventstart (bef), participation (part), invitationlogin (invit), nologin (nolog), interactiveagenda (intag), decisionsupport (dec)\}$.
- $D = \cup_{f_i \in F} dom(f_i)$.
- $FM = \{c_0 : eh = true, c_1 : eh \leftrightarrow bas, c_2 : adm \rightarrow bas, c_3 : inv \rightarrow bas, c_4 : con \leftrightarrow eh, c_5 : email \leftrightarrow con, c_6 : phot \rightarrow con, c_7 : not \rightarrow eh, c_8 : not \leftrightarrow newpost \vee ch \vee bef, c_9 : part \leftrightarrow eh, c_{10} : part \leftrightarrow (invit \wedge \neg nolog \vee \neg invit \wedge nolog), c_{11} : intag \leftrightarrow eh, c_{12} : dec \leftrightarrow eh, c_{13} : email \rightarrow newpost, c_{14} : \neg(intag \wedge nolog), c_{15} : dec \rightarrow nolog\}$.

The properties which cause inconsistencies in the adapted feature model are listed and explained here:

Void feature model. Such a model is inconsistent per-se, which means that there is no solution for a given feature configuration task. For the feature model in Figure 5.2 there exists no solution because both features, *interactiveagenda* (*intag*) and *decisionsupport* (*dec*) are mandatory, *dec* requires *nolog* and *intag* excludes *nolog*.

5 Managing Variability

property	property check	analysis operation
void feature model	$\text{inconsistent}(\text{FM})$	$\Delta \subseteq \text{FM} :$ $\text{consistent}(\text{FM} - \Delta)$
test-induced void feature model	$\exists t_i \in T :$ $\text{inconsistent}(\text{FM} \cup \{t_i\})$	$\Delta \subseteq \text{FM}, \forall t_i \in T :$ $\text{consistent}(\text{FM} - \Delta \cup \{t_i\})$
dead feature f_i	$\text{inconsistent}(\{f_i = \text{true}\} \cup \text{FM})$	$\Delta \subseteq \text{FM} :$ $\text{consistent}(\text{FM} - \Delta \cup \{f_i = \text{true}\})$
full mandatory feature f_i	$\text{inconsistent}(\{f_i = \text{false}\} \cup \text{FM})$	$\Delta \subseteq \text{FM} :$ $\text{consistent}(\text{FM} - \Delta \cup \{f_i = \text{false}\})$
false optional feature f_i	$\text{inconsistent}(\{f_i = \text{false} \wedge f_{i-1} = \text{true}\} \cup \text{FM})$	$\Delta \subseteq \text{FM} :$ $\text{consistent}(\text{FM} - \Delta \cup \{f_i = \text{false} \wedge f_{i-1} = \text{true}\})$
redundant constraints $c_i \in \text{FM}$	$\exists c_i \in \text{FM} : \text{FM} - \{c_i\} \models c_i$	$\text{FM}_{nr} \subseteq \text{FM} : \forall c_i \in \text{FM}_{nr} : \text{FM}_{nr} - \{c_i\} \not\models c_i$
implicit feature groups $\{f_a, f_b\} \in \text{FM}$	$\exists \{f_a, f_b\} \subseteq F :$ $\text{inconsistent}(\{f_a = \text{true} \wedge f_b = \text{false} \vee f_a = \text{false} \wedge f_b = \text{true}\} \cup \text{FM})$	$\Delta \subseteq \text{FM} :$ $\text{consistent}(\text{FM} - \Delta \cup \{f_a = \text{true} \wedge f_b = \text{false} \vee f_a = \text{false} \wedge f_b = \text{true}\})$

Table 5.1: Summary of analysis operations for feature models. FM_{nr} denotes a non-redundant feature model.

Test-induced void feature model. The original model depicted in Figure 5.1 combined with the test-case $t_1 : \text{intag} \wedge \text{nolog}$, i.e., $\text{inconsistent}(\{t_1\} \cup \text{FM})$. The minimal conflict set is $\text{CS} : \{c_{14}\}$ since $\text{inconsistent}(\{t_1\} \cup \{c_{14}\})$.

Dead features. In the adapted feature model in Figure 5.2, feature *invitationlogin* (*invit*) can be considered as dead since it is not possible to include this feature in a configuration. The reason is that feature *decisionsupport* (*dec*) requires the inclusion of feature *nolog*. Due to the *alternative* relationship between *participation* (*part*), *inv*, and *nolog*, it is not possible to include feature

5 Managing Variability

inv.

False optional feature. Since the inclusion of *emailpostings* (*email*) requires the inclusion of *newpostings* (*newpost*), *notification* (*not*) will be included in every configuration. For this reason, *notification* can be regarded as *false optional*.

Redundant constraint. Constraint c_9 can be regarded as redundant (assuming that constraint c_{14} has been deleted) since feature *participation* is part of every configuration (it is required by feature *decisionsupport*).

There are another properties of ill-formed models which are not taken into account by the example model above. *Fully mandatory features* are features that are included in every possible configuration. *Implicit feature groups* occur, if there does not exist a solution in which the features of this group have different values.

5.3 Conclusion

Feature models are a useful concept to model variability in the Requirements Engineering phase of a software development process. These models consist of features which are used to specify the inclusion or exclusion of functionalities in different configurations. Furthermore, inconsistencies of different types can occur in such models. There are some approaches to automatically detect and resolve inconsistencies on the basis of concepts of conflict detection and model-based diagnosis. A simple feature model of EVENTHELPR was used to show the variability of the environment and to explain parts of feature models.

6 Technologies

For the realization of EVENTHELPR as a web application, a couple of technologies and frameworks have been used. EVENTHELPR was built with the *Spring Framework* and runs on top of the *Java Enterprise Edition (Java EE)* platform. With the help of *Spring Boot* it was possible to rapidly create a solid stand-alone application from scratch without spending much time on configuration and infrastructure. Another main advantage is an integrated *Tomcat Server* which simplifies the deployment process of the software very much. In Section 6.1, a short overview of the used framework is given.

The front-end views are created with *Thymeleaf*, which is a Java template engine. *Thymeleaf* can be easily integrated into Spring and builds usual Hypertext Markup Language (HTML) web pages which make the whole content accessible by the end user. An introduction to this template engine is provided in Section 6.2. The web pages use the following common web standards and frameworks:

- *jQuery*¹ - a Java Script library with many features for manipulating and traversing documents, event handling, animation and Ajax
- *CSS3*² - a standard style sheet language used to set the visual styles of documents written in a markup language
- *Google Maps API*³ - an Application Programming Interface (API) for displaying maps on web pages
- *Bootstrap*⁴ - a web framework for front-end development containing different HTML- and CSS-based design templates and Java Script extensions

¹<https://jquery.com/>

²<https://www.w3schools.com/css/>

³<https://developers.google.com/maps/>

⁴<https://getbootstrap.com/docs/3.3/>

All of these frameworks and standards contribute to the visual appearance of EVENTHELPR and guarantee a responsive design across all web browsers and devices including tablets and smart phones.

6.1 Spring Framework

The *Spring Framework* [35] is a collection of modules and core functions to simplify the creation of Java applications. There are also extensions to use the framework with the Java EE for building modern web applications. Spring is modular and has a layered architecture, that means it is possible to use just any part of the framework in isolation. It enables developers to focus on the business logic without bothering about the internal infrastructure or deployment environments.

6.1.1 Architecture and Features

The *Spring Framework* is lightweight and designed to support simple objects, also called Plain Old Java Objects (POJOs). The most important characteristic of these POJOs is that they are independent from any technology. As shown in figure 6.1, the framework consists of individual modules which are all built on top of the core container. Each module is again divided into sub-modules and can be used individually or in combination with other modules [1].

In the core container, the fundamental parts of the framework are located. Its sub-modules define how beans are handled, accessed and manipulated within the framework. Beans are just Java classes which implement the serializable interface and follow a simple coding convention. The container also provides the important features Inversion of Control (IoC) and dependency injection. In the IoC pattern, the framework is not called by the application, it is the framework which calls the necessary components specified by the application. Dependency injection regulates the dependency of an object at runtime. Dependencies are deposited in a central place and not managed by the objects themselves. The benefit of these features is the possibility

6 Technologies

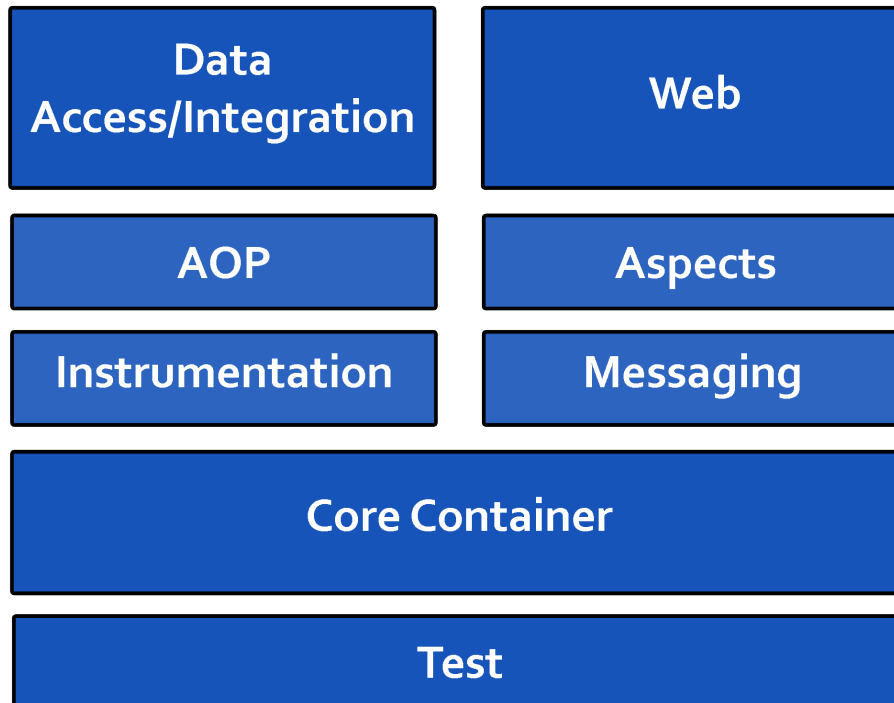


Figure 6.1: Architecture of Spring Framework.

to increase the modularity of the code and to separate configuration and dependencies from the application code [22].

Another interesting feature is Aspect Orientated Programming (AOP) which is provided through the AOP module. With the help of AOP, generic functionalities can be used across several classes to avoid code redundancy and to increase the maintainability.

Furthermore, there are two major modules which are often used in application development. The data access and integration module is responsible for transaction management and the persistence of objects. Interfaces provide methods which can be easily applied to POJOs of the business classes to store them in relational databases. Therefore, several integration layers for object-relational mapping (ORM) APIs such as Hibernate⁵ exist.

⁵<http://hibernate.org/>

6 Technologies

The second major module is the Web module which is best suited for developing web applications and representational state transfer (REST) web services. It consists of basic web orientated features including servlet listeners and a separate application context. In addition, it provides Spring's own Model-View-Controller (MVC) implementation. In the MVC pattern, a web application is divided into different parts. The aim of this pattern is to separate the application logic from the views and the underlying model. In the section 6.1.2, the Spring MVC Model is explained in more detail.

6.1.2 Spring MVC Model

The Spring MVC Model consists of three different parts, which are shortly described here:

- **Model:** The model contains the data which is necessary to describe the problem domain of the application. It is independent of the user interface and manages the data directly
- **View:** The view layer is responsible for representing required data from the model and receiving user input
- **Controller:** The controller handles incoming user requests and interprets the given input in order to process data and afterwards pass this data to the corresponding views for rendering

Figure 6.2 shows how a HTTP Request is handled in Spring MVC. The major component is the *DispatcherServlet*, it is responsible for processing the request and interacts with all other components to create a final response. At first, the *DispatcherServlet* consults the *HandlerMapping* to associate the incoming request to an individual controller. Then the controller calls service methods which execute defined business logic to set model data and return the name of the view. By using the *ViewResolver*, the *DispatcherServlet* takes the correct view for the current request. Finally, the view receives the model data from the *DispatcherServlet* and is rendered on the browser [4].

The advantages of Spring MVC framework are the support of different view types, support of in-build RESTful web services, easy configuration and the possibility to provide multiple views for the same model. Furthermore, it is fully secure by using the Spring Security API.

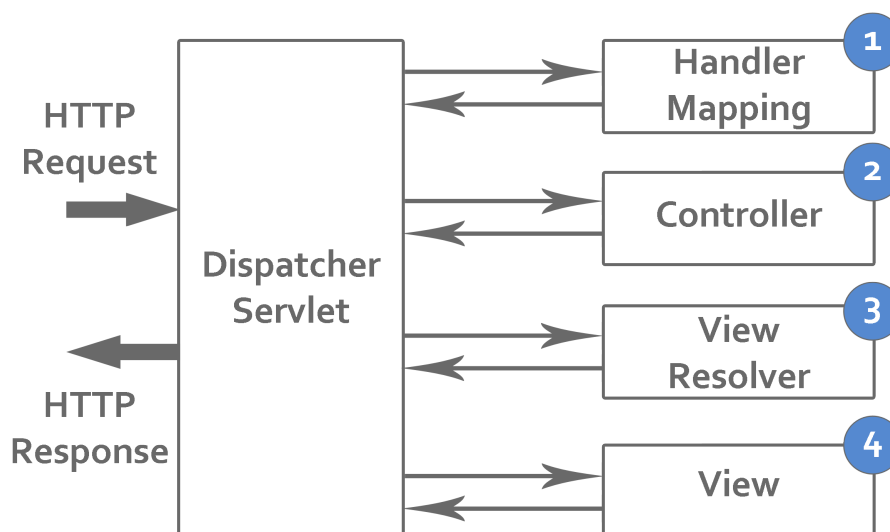


Figure 6.2: Request handling in Spring MVC.

*Spring Security*⁶ is a sub-project of Spring which provides authorization and authentication as well as protection against attacks to Java applications. It can be used across all Spring-based applications and is very powerful and easy to integrate.

6.1.3 Spring Boot

*Spring Boot*⁷ is also a sub-project of Spring. It follows the convention over configuration software design paradigm. That makes the work of developers much easier because this paradigm attempts to minimize the effort for configuration and deployment of spring applications. Spring Boot provides a flexible way to create Spring-based applications that can “just run” and need very little configuration. There exist many starter Project Object Models (POMs) which contain standard dependencies for different types of applications. Developers can use this starter POMs and build their own

⁶<https://projects.spring.io/spring-security/>

⁷<https://projects.spring.io/spring-boot/>

application upon them. Further, the deployment of such applications is very simple since an embedded Tomcat server is provided in the framework. After exporting the project into an executable Java Archive (JAR) which contains all necessary data, the application can be started by just executing this JAR.

6.2 Thymeleaf

Thymeleaf[37] is a template engine written in Java. It is able to apply a set of transformations to various kinds of templates (for example HTML5, Extensible Markup Language (XML) or Extensible Hypertext Markup Language (XHTML)) in order to produce documents which display data produced by the application. The engine is based on XML tags and attributes which trigger the execution of predefined logic on the Document Object Model (DOM). For every DOM node, several attributes and syntax features can be used to describe how it will be processed. The *Standard Dialect* defines all of these attributes and syntax features.

The core of *Thymeleaf* is a special high-performance DOM processing engine. This engine is especially designed for building in-memory tree representations of the templates and performing operations on the nodes. Further, a cache for parsed templates is used to ensure faster operation.

Another important aspect of the template engine is the concept of “Natural Templates”. Every template can be correctly displayed by a browser even before it is processed. This is possible because browsers are able to ignore the additional attributes of the nodes.

Thymeleaf also supports the definition and referencing of fragments. This enables the programmer to include the same piece of code in several templates. In addition, fragments can be parameterized to provide a variety of useful applications.

7 Implementation

In this chapter, a detailed insight into the implementation of the web application is given. At first, the basic project setup and the structure is explained. Due to the high amount of files, only the most important ones are stated. Finally, the properties and relations of the model classes are described in more detail. These classes represent the problem domain of EVENTHELPR and are at the same time responsible for the layout of the underlying database.

7.1 Overview

For the implementation of EVENTHELPR, *Eclipse EE*¹ was chosen as Integrated Development Environment (IDE). In combination with *Apache Maven*², the project setup and build process could be simplified very much. *Maven* is a Java-based build management tool which is used to standardize the build process itself and to manage the project dependencies. A Maven project is configured via a Project Object Model (POM) stored in a simple XML file. In the POM, all necessary information about the project is provided. As explained earlier in Section 6.1.3, Spring Boot offers many pre-configured starter POMs. Such a starter POM was used to setup a Spring Web application with basic functionality as a starting point. The main advantage of Maven is, that it needs less configuration and downloads required dependencies dynamically if needed. With the help of Maven, the project can be easily exported into an executable JAR containing all data and an embedded Tomcat server.

¹<https://www.eclipse.org/>

²<https://maven.apache.org/>

7.2 Project Structure

The basic structure of the project is defined and standardized by Maven and is shown in Figure 7.1.

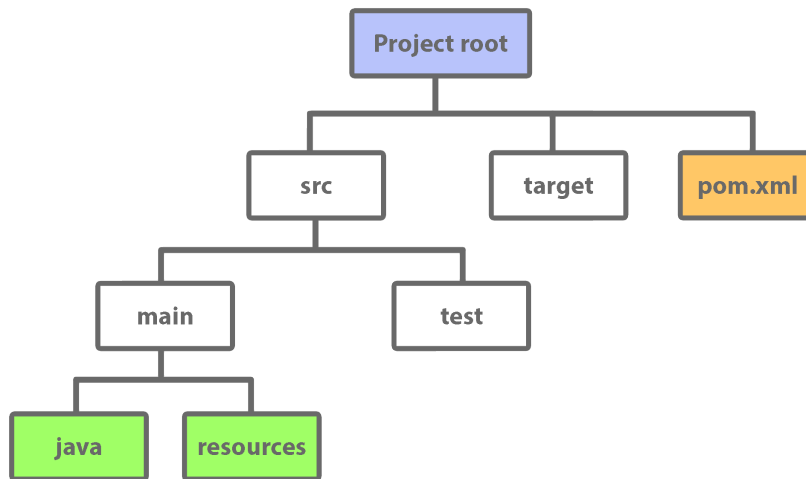


Figure 7.1: Basic project structure of web application.

As previously described, the POM defines the whole project and is located in the root directory. The *target* directory contains all generated files of the Maven build process. This includes the compiled Java classes as well as the final JAR file, which is needed to deploy the application. The input files of the project are located in the *src* directory. The *test* sub-directory usually contains the sources for automated testing, but such tests are not implemented for EVENTHELPR. In the *main* sub-directory, the actual source code and resources of the application are located.

Since the Java source code and the resource files are the most important parts of the implementation, these parts are explained in more detail in the next two subsections.

7 Implementation

7.2.1 Java Source Code

The Java source code is divided into multiple packages. A package is used to group related types and to provide a better overall structure for the Java classes and interfaces. In this section it is explained, what purpose the classes of each package fulfill. If necessary, important classes are picked out and characterized explicitly.

Package - *com.eventhelper*

This package contains the core classes of the Spring Boot application. The *EventHelperApplication* class defines the main entry point and is used to start up the web application with all necessary components. In the *SecurityConfiguration* class, the Spring Security is configured which is responsible for providing authentication and authorization within the whole application. Additionally, a scheduled background task is implemented here which is dealing with notifications.

Package - *com.eventhelper.controllers*

In this package, all controller classes are located which receive incoming requests and pass data from the model to the corresponding views. Usually controllers use service classes to access data from the model. Each controller defines several methods with an appropriate mapping in order to match a given request. The response of a controller depends on the type (MVC controller or RESTful web service controller) and can be either an object represented as Java Script Object Notation (JSON) or a server-side rendered HTML document. For the rendering of HTML documents the template engine is used in combination with the specified HTML view template. A JSON response can be easily created with the help of Data Transfer Objects (DTO).

Package - *com.eventhelper.dao*

This package consists of Data Access Object (DAO) interfaces which provide access to the underlying database. These interfaces extend a basic data repository interface of the Spring Framework and support an easy method creation pattern for all sorts of database operations. A DAO always refers to a model class in this case.

7 Implementation

Package - *com.eventhelper.dto*

The DTO classes defined in this package are simple objects with primitive data types only which just store data and do not have any behavior or logic. DTOs are used to transfer object data from and to the server (especially for RESTful web services) and can be easily converted to JSON. Unlike model classes, a DTO has no connection to the database layer and can bundle data of different model classes in one object. An advantage of using DTOs is that there are less server calls required to transfer an amount of data because the data from different objects could be aggregated in one DTO.

Package - *com.eventhelper.models*

All models which are essential to represent the problem domain of EVENTHELPER are located in this package. Every model class contains some attributes and relations to other models. With the help of special annotations from the Spring Framework, it is possible to define the table layout of every model class in the database. If a model class is annotated as Spring Entity, it is automatically persisted into the database via ORM and is represented by an own table. A complete class diagram containing all model classes and its attributes as well as the relationships between these classes is provided in Section 7.3.

Package - *com.eventhelper.service*

This package contains the classes which refer to the service layer of the application. Service classes hold business logic and use DAOs to interact with the persistence layer. As already mentioned above, the service itself is called by different controllers to access and manipulate model data in a proper way. The purpose of the service layer is to decouple most of the business logic from the controllers to make them more independent of the underlying structure. This aspect becomes important if different controllers have to perform the same actions within the application (for example a traditional MVC controller and a REST controller).

Package - *com.eventhelper.utils*

All classes and methods which can be used across the whole application are bundled in this package.

7.2.2 Resources

The *resources* directory contains all other files which are required to run the application such as properties files and web resources. One of the most important files is the *application.properties* file, which is located in the root of the directory. In this file, the main properties like directories, paths, database connection and mail configuration for the application are defined.

In the *i18n* folder, the properties files needed for internationalization are stored. In those files the texts for the application are provided in the supported languages.

The static web resources used for the HTML pages are located in the *static* folder. The web resources consist of style sheets, images and Java Script files.

The *templates* folder contains the HTML templates which are used by the template engine to create the views.

7.3 Model Classes

Since the model classes represent the problem domain (or business model) of the application, the following two class diagrams show which classes exist and how they are related to each other. Additionally, the multiplicity of every relationship is printed at the end of the corresponding arrow.

Figure 7.2 shows all model classes which are used for the application. Due to high complexity, the sub-classes belonging to *AgendaItem* are hidden in the main diagram.

In Figure 7.3, the relationships of the *AgendaItem* class are shown in detail, whereas other classes with no influence are hidden for a better view.

7 Implementation

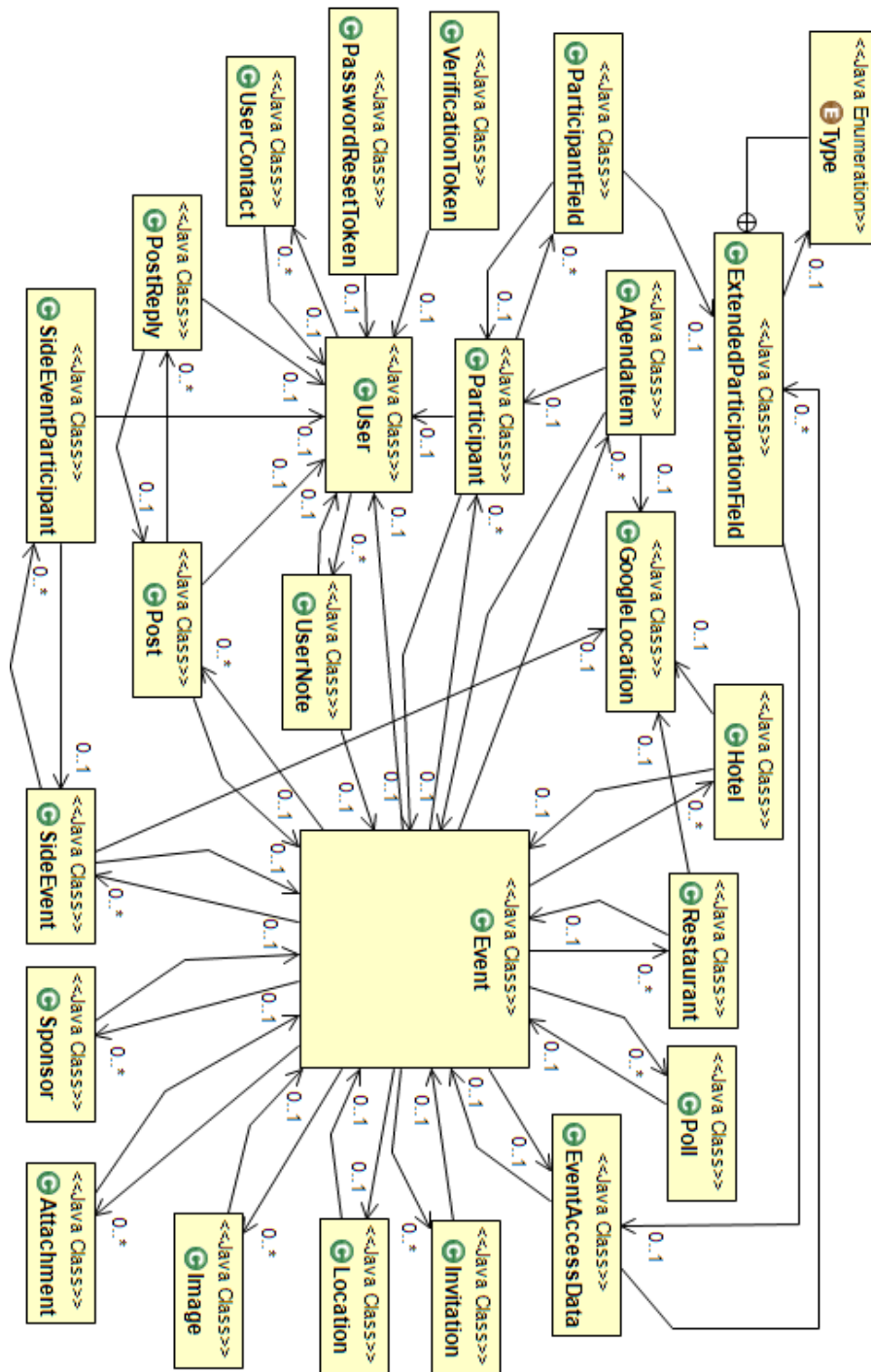


Figure 7.2: Class diagram of all model classes. The child elements of the agenda item class are not shown in this diagram.

7 Implementation

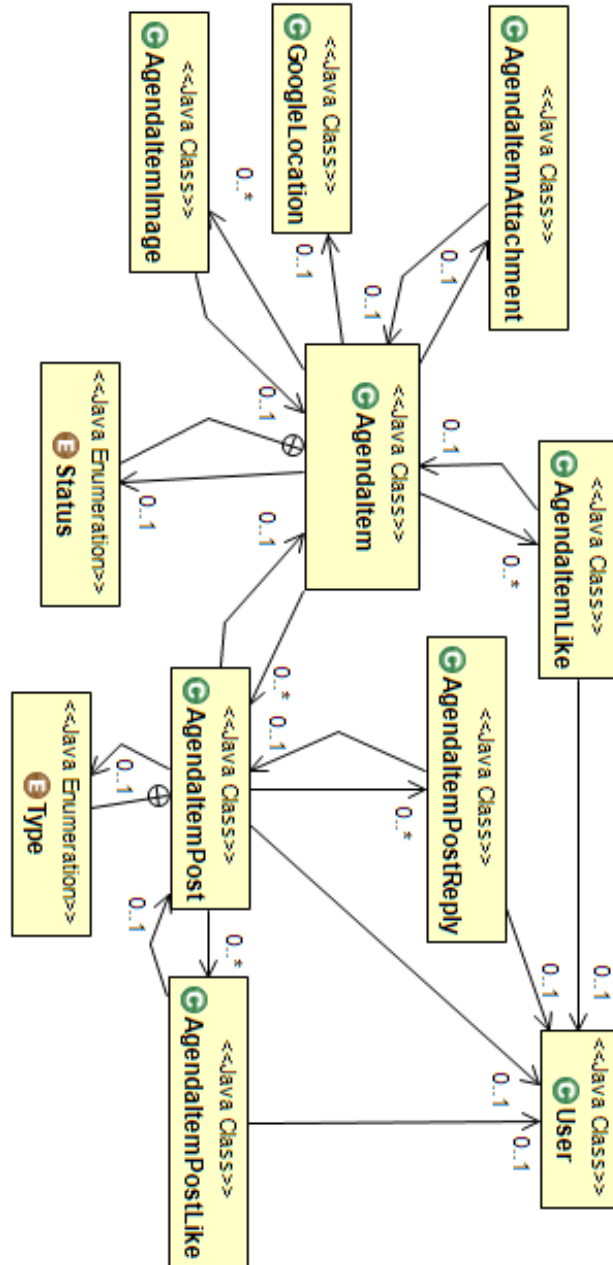


Figure 7.3: Class diagram of model classes which are related to the agenda item class. Other model classes are excluded from this diagram.

8 User Interface and Functionality

The following chapter focuses on the visual appearance of EVENTHELPR and explains the functionality of the application on the basis of user interface screen shots.

8.1 Sign-up and Sign-in

It is not required to create a user account in order to use EVENTHELPR. Nevertheless, registered users enjoy a couple of features which are not applicable in the anonymous mode. A main advantage of using EVENTHELPR with an account is that there is an overview of all events which have been created by the user or in which the user is participating.

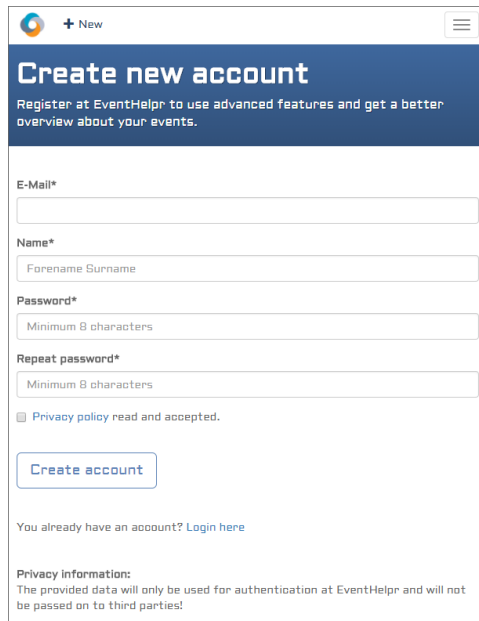
In Figure 8.1a, the sign-up form is depicted. A user is primarily identified by a valid e-mail address that has to be unique within the application. The sign-up process includes an account verification mechanism. After creating an account, an e-mail containing an activation link is sent to the corresponding address. Finally, the sign-in is possible if the account has been activated with the link.

Figure 8.1b depicts the sign-in form for users. If *Remember me* is checked, a cookie is set up to enable auto-login for the user at the next visit. Additionally, a reset password mechanism is provided by the application. The password of a user account can be reset by invoking a reset link which has been sent to the provided e-mail address beforehand.

In the personal area, the user account can be managed. It is possible to edit user data and optionally upload a profile picture. If *Activate e-mail notifications for events* is checked, the user receives e-mail notifications regarding

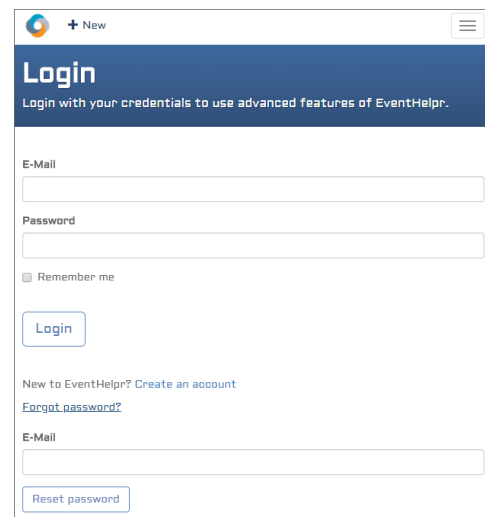
8 User Interface and Functionality

the modification of events (including agenda items) or the availability of new content (agenda items, postings and images) and participants within events. The deletion of a user account removes all references of the deleted user in any context. Events which have been created by the deleted user are not removed but the creator gets cleared.



The screenshot shows a web form titled "Create new account" for EventHelpr. The form includes the following elements: a header with the EventHelpr logo and a "+ New" button; a sub-header with the text "Register at EventHelpr to use advanced features and get a better overview about your events."; input fields for "E-Mail*", "Name*" (with a placeholder "Forename Surname"), "Password*" (with a note "Minimum 8 characters"), and "Repeat password*" (with a note "Minimum 8 characters"); a checkbox for "Privacy policy read and accepted."; a "Create account" button; a link "You already have an account? Login here"; and a "Privacy information" section at the bottom stating: "The provided data will only be used for authentication at EventHelpr and will not be passed on to third parties!"

(a) Sign-up form with corresponding input elements for the creation of user accounts.



The screenshot shows a web form titled "Login" for EventHelpr. The form includes the following elements: a header with the EventHelpr logo and a "+ New" button; a sub-header with the text "Login with your credentials to use advanced features of EventHelpr."; input fields for "E-Mail" and "Password"; a checkbox for "Remember me"; a "Login" button; a link "New to EventHelpr? Create an account"; a link "Forgot password?"; an "E-Mail" input field; and a "Reset password" button.

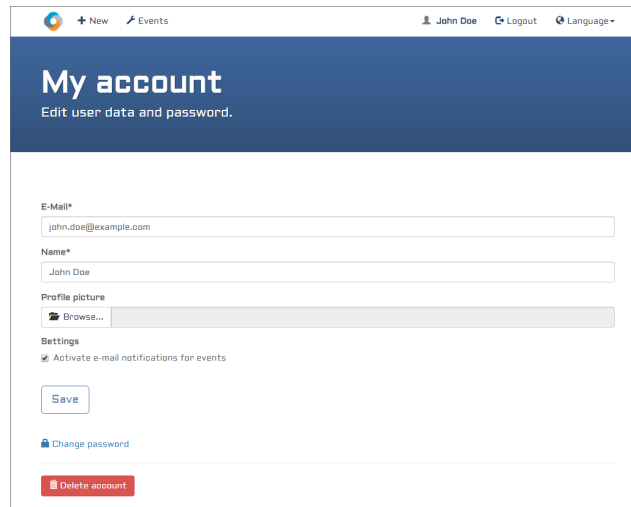
(b) Sign-in form with corresponding input elements for the user sign-in.

Figure 8.1: Sign-in and sign-up form.

8.2 Creation of Events

The event creation process is handled on one single page and uses a simple form to enter data. In Figure 8.3a, the first part of the form including input elements for basic information and the event location is shown. The address input field is connected to a search box from the *Google Maps API* and automatically provides suggestions for places during the user is typing.

8 User Interface and Functionality



The screenshot displays a web application interface for a user's account. At the top, there is a navigation bar with a logo, a '+ New' button, and a 'Events' link. On the right side of the navigation bar, the user's name 'John Doe' is displayed, along with 'Logout' and 'Language' options. Below the navigation bar is a dark blue header with the text 'My account' and a subtitle 'Edit user data and password.' The main content area contains a form with the following fields and options: 'E-Mail*' with the value 'john.doe@example.com', 'Name*' with the value 'John Doe', and 'Profile picture' with a 'Browse...' button. Below these fields is a 'Settings' section with a checked checkbox for 'Activate e-mail notifications for events'. At the bottom of the form, there are three buttons: 'Save', 'Change password', and 'Delete account'.

Figure 8.2: Personal user area with corresponding edit form and additional functions.

If a suggestion is selected, the location gets displayed in the map user interface.

Further, all sorts of items can be added to the different sub-contents of the event. Figure 8.3b depicts an overview of the supported sub-contents. The input fields for a new item are appended to the view immediately (when the corresponding button is clicked), but the items are only saved when the user creates the event by submitting the form.

Since every event of EVENTHELPR is identified via unique links, the creator of an event must be able to remember these links. Figures 8.4a and 8.4b point out two possibilities of providing contact information for a new event in order to keep the links for later event access. If an event is created by a logged-in user, the corresponding user account is stored as a reference in the event. In this case, the user can access the event from within its personal event list later. As mentioned earlier, a user account is not necessarily required for the creation of events. Therefore, the second possibility is to manually provide an e-mail address where the links for event access will be sent. If no e-mail address is given, the event is created anonymous and the links for event access are only listed on the overview page of the newly created event.

8 User Interface and Functionality

Event information

Name of event*

Short description


Description

Link

Event logo

Address

Floor, Room, etc.



(a) Selected part of input elements for basic event information including a map.

Agenda
[+ Add entry](#)

Side events
[+ Add side event](#)

Feedback
[+ Add poll](#)

Accommodations
[+ Add hotel](#)

Restaurants
[+ Add restaurant](#)

Attachments
[+ Add attachment](#)

(b) Additional information units for event.

Figure 8.3: Snippets of event creation form with corresponding input elements and information units.

Contact information

Logged in as:
John Doe (john.doe@example.com)

Please add me to the list of event participants automatically (requires at least a name)

(a) Event creation with signed in user

Contact information

Contact information is optional. If not provided, the creator of the event has to manage the generated event administration and participation links on his/her own.

E-Mail

This e-mail address will only be used for forwarding the generated event and administration links to the creator of the event (you). NO account will be generated on the basis of the provided email address.

Your name

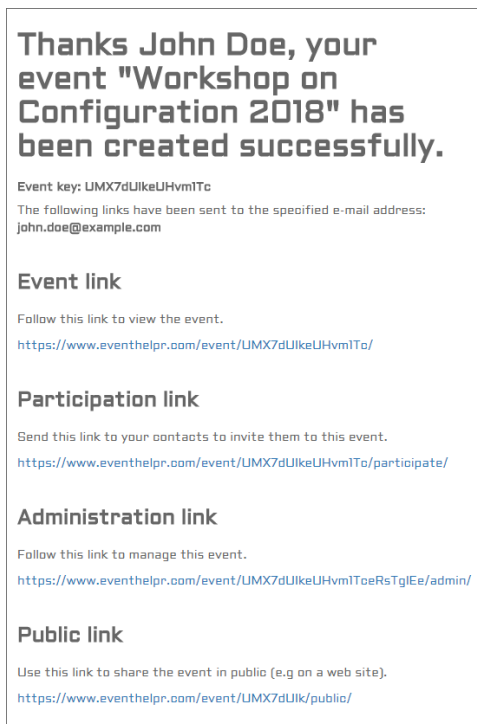
Please add me to the list of event participants automatically (requires at least a name)

(b) Event creation without signed in user.

Figure 8.4: Different types of contact information in the event creation form.

8 User Interface and Functionality

After the new event has been created by submitting the form, the user gets redirected to an overview page as indicated in Figure 8.5a. On this page, all links for later event access are listed. These links are additionally sent to the e-mail address of the event creator (either taken from referenced user account or from manual input). The unique strings in the links are incrementally built upon each other in the following order: public link - event link - administration link. This pattern allows the derivation of the other links starting from the administration link but not vice versa.



Thanks John Doe, your event "Workshop on Configuration 2018" has been created successfully.

Event key: UMX7dUIkeUHvmlTo
The following links have been sent to the specified e-mail address: john.doe@example.com

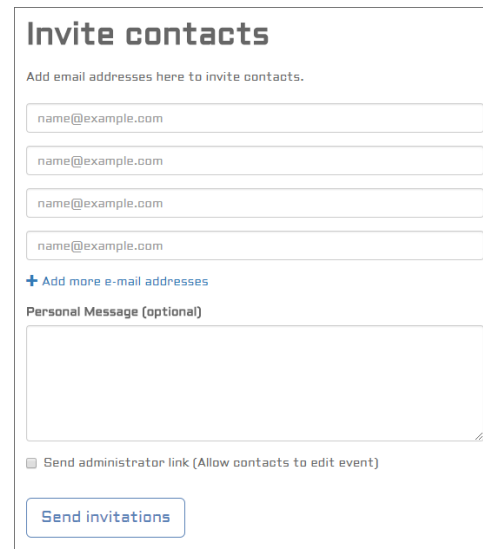
Event link
Follow this link to view the event.
<https://www.eventhelpr.com/event/UMX7dUIkeUHvmlTo/>

Participation link
Send this link to your contacts to invite them to this event.
<https://www.eventhelpr.com/event/UMX7dUIkeUHvmlTo/participate/>

Administration link
Follow this link to manage this event.
<https://www.eventhelpr.com/event/UMX7dUIkeUHvmlToeRsTgEe/admin/>

Public link
Use this link to share the event in public (e.g on a web site).
<https://www.eventhelpr.com/event/UMX7dUIk/public/>

(a) Overview of created event including unique event links.



Invite contacts

Add email addresses here to invite contacts.

name@example.com
name@example.com
name@example.com
name@example.com

+ Add more e-mail addresses

Personal Message (optional)

Send administrator link (Allow contacts to edit event)

Send invitations

(b) User interface fragment for sending event invitations to contacts.

Figure 8.5: Overview page of newly created event.

In addition to the event links, the overview page offers the possibility to invite contacts to the event. Figure 8.5b depicts the user interface fragment for sending invitations via e-mail which is re-used on some other pages. An invitation e-mail containing the participation link for the event is first

8 User Interface and Functionality

generated automatically and then sent to the given addresses. Contacts can also be invited later on the main event page by the event administrator or other participants (only if parameter set).

8.3 Administration of Events

Everyone who owns the event administration link is able to edit the event data. The event administration page basically uses the same form as explained in Section 8.2 with some additions. As illustrated in Figure 8.6, the event links are listed again and there are a couple of advanced operations related to the event.

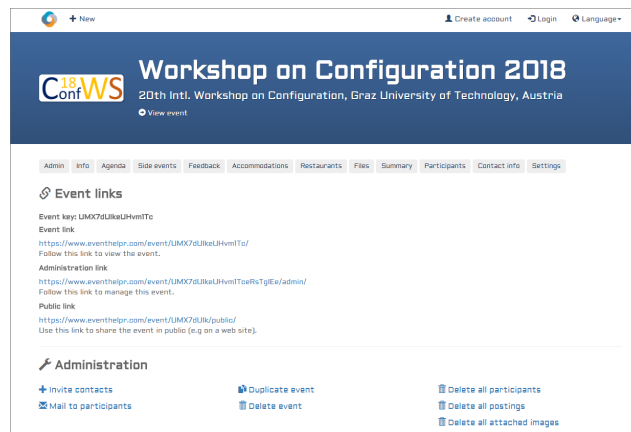


Figure 8.6: Snippet of event administration page containing the first section.

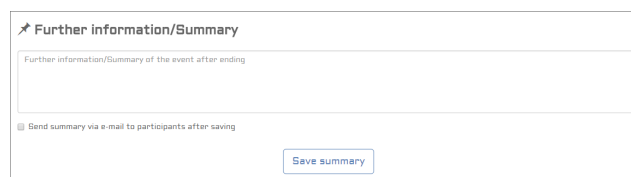


Figure 8.7: Snippet of event administration page containing the summary input.

Figure 8.7 depicts an additional feature of the administration page. A summary of the event can be provided and afterwards be sent to all participants of the event.

8.4 Participation in Events

The participation in an event allows users to access all available information on the event page and to create content like for example postings or images (if respective parameters of event are set). Further, all participants are shown in the event's list of participants and receive notifications regarding the event via e-mail. These notifications contain a short digest from all changes which were made, including new items and content changes.

Participate

What does participation in this event mean?

You can access all available information and you can add images or postings and receive updates/additional information related to the event (via e-mail).

Please provide the following information

Your name*

[+ Participate](#)

ⓘ If you want to complete the participation with a user account, please sign in first - [Sign in](#)

Overview of participants

Thomas Gruber John Doe

Figure 8.8: Event participation page with two different ways to participate.

Figure 8.8 shows the event participation page which can be reached by following the participation link in the invitation e-mail. There are two different ways on how the participation can be performed:

Participation without user account. In this case, only a name is required to participate in the event. The need for an e-mail address is depending on the event settings. This type of participation is only supported if the parameter of the event *Participation in event is only possible for logged-in users with invitation* is not set.

Participation as logged-in user. If the participation is processed as a logged-in user, the user account is stored as a reference in the event participation. In this way it is possible to keep track of all events in which the user is

participating and in further consequence to show these events in the event list of the user.

In both cases, the participant is able to provide further information if the creator of the event has defined additional input fields.

The link to the event page is added dynamically after the participation was successful (if enabled in the event settings). Additionally, a confirmation e-mail containing the event link is sent if an address is available. At the bottom of the page, a list of existing participants is displayed if the corresponding parameter is set in the event settings.

8.5 Interaction with Events

The most important task in `EVENTHELPR` is the interaction with created events. In this section, the focus is on the available information which is displayed on the event page and on the possible actions which users can perform on this page. The event page can be reached using the event link which has been received via e-mail after a successful participation or from the event list in a user account (if existing).

The event page consists of a header and several sub-sections where the whole content of the event is presented to the user. Every sub-section displays some information and allows users to interact with the event. The available actions strongly depend on the settings of the event. The displayed information as well as the possible user actions of every sub-section will be explained in more detail now.

Header and location information

Figure 8.9 shows the upper part of the event page containing the event header and the event map.

In the event header, some basic information about the event like title, description or date is displayed. It is possible to quickly scroll to the list of participants or to the postings section by clicking the links in the status line. The orange edit icon to the right of the last modification date indicates that this event has been modified since the last login of the current user.

8 User Interface and Functionality



Figure 8.9: Snippet of event page with header and location information.

The event map is split into two areas in the default map view. On the left side the detailed location is displayed on a standard map and on the right side the location is displayed as street view. If no street view data is found for the given location, the standard map extends to the available space. The following view modes enable the user to get more location information beside the event location.

Agenda. Marks all agenda items in the map for which a location information exists. This could be very helpful in a travel scenario where agenda items represent destinations.

Side events. Marks all side events in the map for which a location information exists.

Interesting places. Marks all interesting places in the map which have been added to the event.

8 User Interface and Functionality

Restaurants & Accommodations. Marks all restaurants and accommodations in the map which have been added to the event. Additionally, nearby items of those two types which are selected automatically with the help of Google Maps are shown.

Stations nearby. Marks nearby stations for public transport including bus, train, subway and tram in the map.

Airports nearby. Marks nearby airports in the map.

Car rentals nearby. Marks nearby car rentals in the map.

Below the different map view modes, the date of the event is displayed. For every event for which a date is specified, a file in the *iCalendar* format is created. This standardized file format is used for the exchange of calendar data. The *iCalendar* file for the event can be downloaded for later import into any calendar application.

Agenda

The *Agenda* is probably the most interesting sub-section within an event. Creators of events are able to customize the agenda for the purpose of the event. As broached in Chapter 4, there are several combinations of advanced settings regarding the agenda in order to cover different scenarios. Depending on the purpose of the event, agenda items may represent travel destinations, to-dos, requirements, decisions, appointments or anything else. If the agenda is configured as interactive, participants of the event are able to modify items or take part in the discussion and rating of individual items.

Figure 8.10 depicts a fully interactive agenda of an event containing a couple of items. The orange icons right to the item title indicate whether an item has been modified or is new. The horizontal bar visualizes a simple pro/con analysis on a per item basis. It shows the distribution of postings by means of its type (green = positive, gray = neutral, red = negative) proportionally. In addition to the bar, the utility value of the item is displayed above. This value could help participants in decision making or to prioritize items.

The utility of an item is depending on the number of positive and negative postings and is calculated by using the following formula:

8 User Interface and Functionality

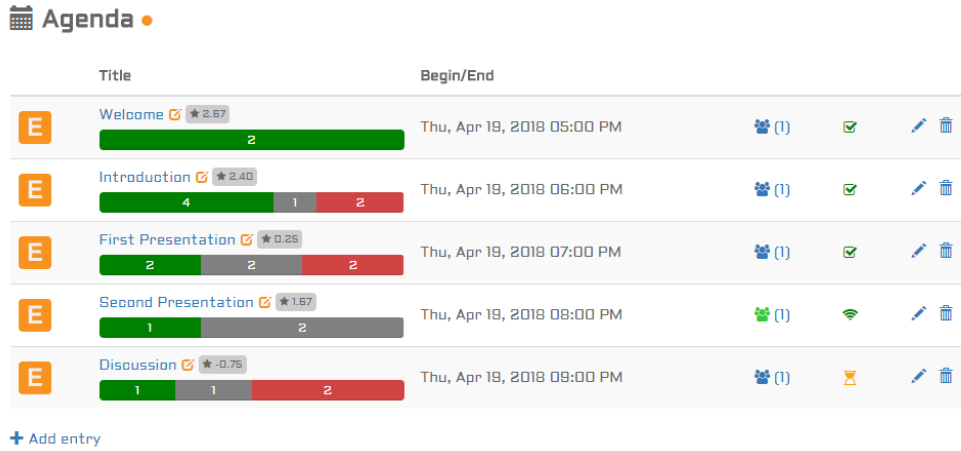


Figure 8.10: Fully interactive agenda with a list of items including a visualization of the pro/con analysis of each item.

$$utility = \sum_{pos \in postings} support(pos) - \sum_{neg \in postings} support(neg) \quad (8.1)$$

$$support(x) = 1 + \frac{S_x}{1 + S_{total}} \quad (8.2)$$

Since positive and negative postings can be supported by users, the number of supports contribute to the utility. S_x denotes the number of supports for the current posting whereas S_{total} denotes the total number of supports added up from all postings.

The contributors and the status of each item are also shown in the list. Due to the fact that there is a lot more content to display for every item, the agenda items can be viewed in a separate detail page.

An example of how such a detail page of an agenda item could look like is given in Figure 8.11. For a better user experience, it is possible to navigate through all agenda items using the arrows at the top. In contrast to the list view of items on the event page, more content is displayed on the detail page. The displayed content is depending on how the agenda is configured and which information is provided. In this example, no location information

8 User Interface and Functionality

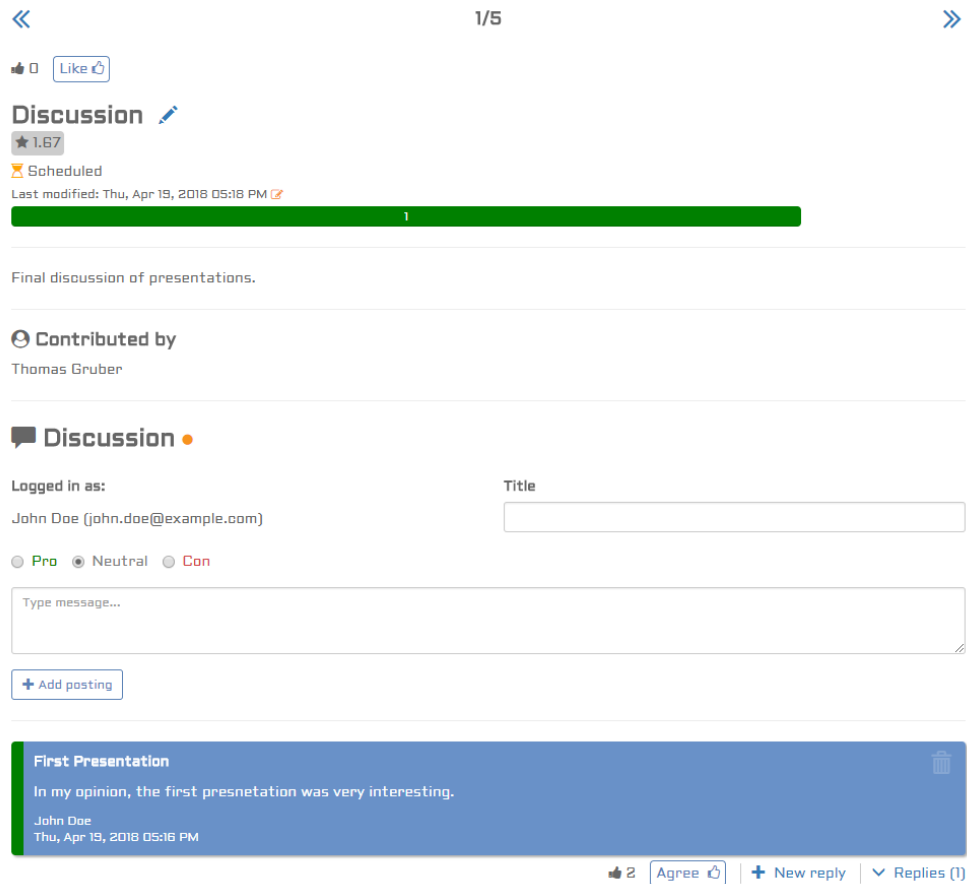


Figure 8.11: Detail page of interactive agenda item including discussion.

for the item is provided so the map is hidden. Furthermore, the appearance of the image upload and discussion interface depends on the settings. Here, discussion is enabled so users are able to create postings (three different types supported) and replies. Furthermore, users can support postings of types *positive* and *negative* to indicate they agree with the posting.

For the creation and modification of agenda items there exist two pages which both use the same form for collecting input data. If users are allowed to create and edit agenda items, those pages can be reached either via the detail page or via the action icons in the list of agenda items on the event page.

8 User Interface and Functionality

Further sub-sections represented as tables

Other sub-sections which are represented as tables are *Side events*, *Feedback*, *Accommodations*, *Restaurants* and *Attachments*. Apart from *Side events*, the items in the other tables contain less information and can be displayed completely on the event page.

For side event items, there exists a detail page which is similar to the one for agenda items in order to present all provided content properly. On this detail page, information regarding the side event and a map (if location is available) is shown. It is also possible for users to participate in side events additionally to the normal event participation. This feature could be especially used to estimate the actual number of participants for particular side events to make preparations easier.

Communication

If the respective parameters in the event settings are set, participants are allowed to invite contacts to the event and send e-mails to other participants of the event. Those two features are also integrated on the event administration page and use the same user interface.

Private notes

In this sub-section, logged-in users are able to add private notes to the current event. Every note can contain a title, a text and an attachment. These notes are only accessible by the user itself and are very useful to add personal data to events.

Images

In this sub-section it is possible to upload images to an event. The same user interface which is depicted in Figure 8.12 is also used for the image upload on the detail page of agenda items. After some images have been selected for upload, a preview of each image including an additional field for a caption is shown. If the upload of the images has been successful, they will be added to the already existing images immediately. The upload as well as the deletion of images for participants is only allowed if the corresponding parameter in the event settings is set properly.

8 User Interface and Functionality

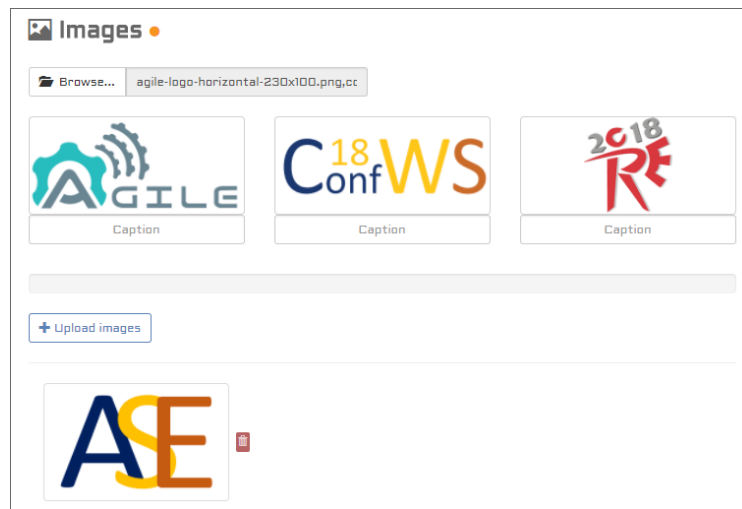


Figure 8.12: User interface for uploading images to an event.

Postings

The user interface for the creation of posting and replies in the event context is basically the same as shown at the bottom of Figure 8.11. The only difference is that postings on the event level do not have three different types but are neutral by default. If the creation of a posting has been successful, it will be added to the already existing postings immediately. The creation as well as the deletion of postings for participants is only allowed if the corresponding parameter in the event settings is set properly.

Participants

In the sub-section *Participants*, a list of all event participants is displayed. Although the participation in an event is handled via the event participation page, it is possible to remove a participation or to re-participate. If the parameter *Participation in event is only possible for logged-in users with invitation* in the event settings is set, users will not be able to access the event page without an existing participation and get redirected to the event participation page automatically.

8.6 Managing Events

The advantage of using EVENTHELPR with a user account is the ability to manage events at a central point. All created events and events in which a user is participating can be assigned to a user if and only if the user had been signed in at the time the corresponding action was performed.

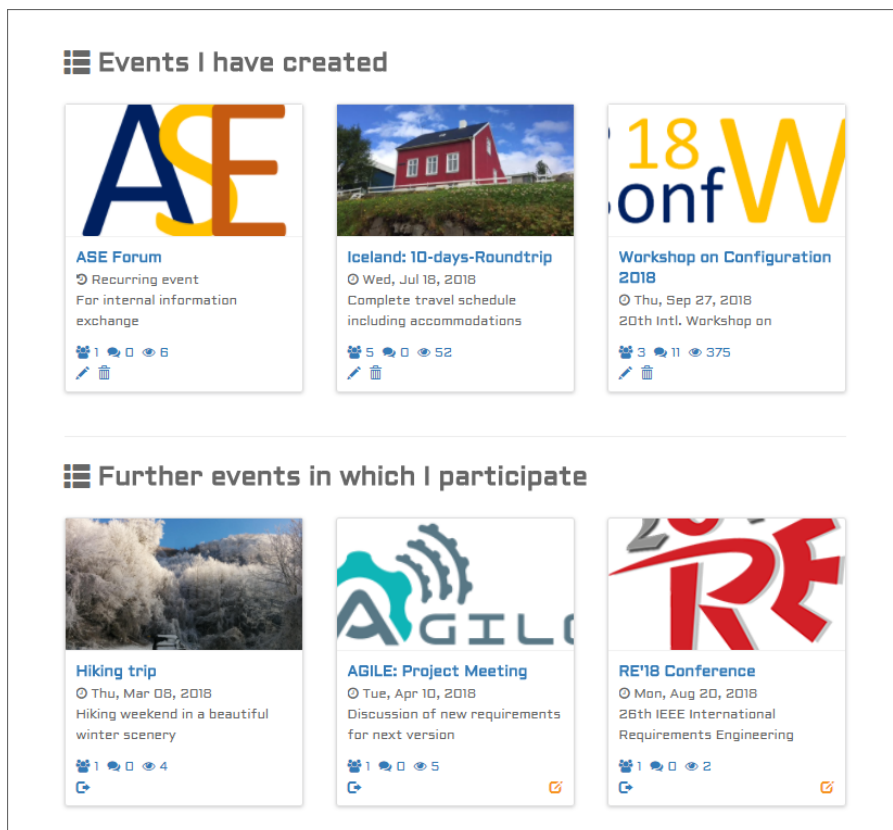


Figure 8.13: Snippet of event overview page for user.

As shown in Figure 8.13, events are arranged in a tile layout to provide a compact overview of individual event properties. Every tile consists of a title image, a title, a date (if provided), a status bar including links and one or more action buttons. Further, an orange edit icon indicates whether an event has been modified since the last login of the user. The events are divided into the following three categories:

8 User Interface and Functionality

Events I have created. Those are all events which were created by the user itself and have either no date or a date in the future (currently running or getting started) specified.

Further events in which I participate. Those are all events in which the user is participating and which have either no date or a date in the future (currently running or getting started) specified.

Past events. This category is a mixture of both previous categories with the only difference that it consists of past events.

In addition, users have the opportunity to archive events instead of deleting them permanently. Archived events are listed on an extra page and can be accessed or restored if needed.

9 Evaluation

In this chapter, the evaluation of a specific part of EVENTHELPR is described. Since the application has many different aspects and features, the decision was to focus on the understandability of the event settings and on the participation process in general. At first, the procedure of the evaluation is explained before the results are discussed in the next step.

9.1 Procedure

The basic procedure of the evaluation was as follows. People, who had no or little previous experience with EVENTHELPR, were asked to perform tasks within the application and were observed by an evaluator during the execution of these tasks. The author of this thesis acted as evaluator. At the beginning, the test persons just got a short description of the task. Then, the evaluator was only allowed to answer occurring questions and to note what the test persons did understand or did not understand. After the completion of the task, the test persons were asked to give informal feedback and to answer two more general questions which were the following:

What are general suggestions for improvements?

What are further scenarios, in which EVENTHELPR could be used?

In order to get a comprehensive feedback, eight different scenarios (types of events) were used for the evaluation:

- Team meeting
- Project meeting
- Workshop
- Holiday travel

9 Evaluation

- Football camp
- Birthday party
- Collective order of customer cards
- Collective order of t-shirts and shorts

For every scenario, a minimum number of test persons was required. The number of test persons varied between the two parts of the evaluation, which are explained right now.

Creation of events

In this part of the evaluation, two different test persons per scenario were asked to create an event. The evaluator only provided a short description of the event including some definitions on how the configuration should look like. The main purpose of the first part was to figure out whether the test persons were able to recognize and set the correct parameters in order to achieve the intended behavior of the event. Especially, the test persons were advised to configure the participation in such a way that participants of the event exactly understand what to do.

For the reason of simplicity, the event descriptions were written in such a way that the test persons did not have to enter too much data. The detailed descriptions for every scenario, which were handed out to the test persons, are listed here:

Team meeting:

- Create an event for a team meeting of your choice (for example a board meeting) which takes place ongoing (no fixed date) and in which open issues should be treated
- Participants should be allowed to add files
- Instead of the default *Agenda*, "Open issues" should be displayed as headline
- Participants should be allowed to create and edit open issues
- Participants should be allowed to comment on open issues and rate them with likes
- Only participants with an existing EVENTHELPR account and a valid invitation should be allowed to participate in the event
- Participants should be able to see who is already participating

9 Evaluation

Project meeting:

- Create an event for a project meeting with a fixed start date (including a start time), in which requirements for the project should be discussed
- Participants should be allowed to send e-mails to other participants and to add postings on the event page
- Instead of the default *Agenda*, “Requirements” should be displayed as headline
- Only participants with an existing EVENTHELPR account and a valid invitation should be allowed to participate in the event
- E-Mail address should be required for participation

Workshop:

- Create an event for a workshop with the topic data protection which takes place in Linz and lasts two days
- E-Mail address should be required for participation
- Participants should be asked to provide following further information if they participate (Experience with data protection YES/NO, education, job)

Collective order of customer card:

- Create an event for a collective order of a customer card (for example a discount card from a sporting goods manufacturer)
- The event should only serve for the order, after completion of the order the participants should not be able to reach the event page
- E-Mail address should be required for participation
- Participants should be asked to provide following further information if they participate (Address for delivery, subscribe to the newsletter of the company YES/NO)
- In addition, “Order card” should be displayed on the participate button

Collective order of t-shirts and shorts:

- Create an event for a collective order of t-shirts and shorts
- The event should only serve for the order, after completion of the order the participants should not be able to reach the event page

9 Evaluation

- E-Mail address should be required for participation
- Participants should be asked to provide following further information if they participate (Shorts YES/NO, size of shorts, t-shirt YES/NO, size of t-shirt)
- In addition, “Add order” should be displayed on the participate button

Holiday travel:

- Create an event for a holiday travel to a desired destination and specify a date (from - to)
- Participants should be allowed to upload images
- Instead of the default *Agenda*, “Trips” should be displayed as headline
- Participants should be able to see who is already participating
- E-Mail address should be required for participation
- Participants should be asked to provide following further information if they participate (Booked pub-crawl wanted YES/NO, interested in a day trip with a ship YES/NO)

Football camp:

- Create an event for a football camp in Kapfenberg which lasts 4 days
- Participants should be allowed to upload images
- E-Mail address should be optional for participation
- Participants should be asked to provide following further information if they participate (Number of days, can go there with my car YES/NO)

Birthday party:

- Create an event for a birthday party at your home and specify a date and begin time
- Participants should be able to see who is already participating
- Participants should be asked to provide following further information if they participate (Coming in company YES/NO, barbecue wishes [steak, sausages or anything else], favorite drink)
- In addition, “Count me in!” should be displayed on the participate button

9 Evaluation

Participation in events

For the second part of the evaluation, 5 to 10 test persons per scenario were asked to participate in the event given to them. In this case, the evaluator only provided the participation link for the event and the test persons were instructed to complete the participation. In order to get better results, events which had been created by the evaluator (for each scenario), were used. The main purpose of the second part was to figure out whether the test persons were able to understand the participation and to provide all necessary data.

The test scenarios *Team meeting* and *Project meeting* only allow the participation with an existing EVENTHELPR account and a valid invitation to the event. Due to this configuration, the test persons were asked to provide an e-mail address where to receive an invitation. Furthermore, the test persons had to create an account for the given e-mail address first in order to be able to complete the participation.

9.2 Results

In this section, the results of the evaluation are summarized. The results are based on the informal feedback given by the test persons and the notes which were taken by the evaluator.

What did the test persons understand?

Most of the test persons assessed the usability of EVENTHELPR as very good. The user interface was also described as intuitive and easy-to-use in general.

In the first part of the evaluation, no test person had problems with the parameters in the general settings when creating the events. That was to be expected because these parameters are nearly self explanatory. The parameter for declaring an event as recurring was also recognized correctly. A majority of the test persons did understand the configuration of the agenda instantly, because the purpose of the parameters is obvious. However, the test persons did only partially understand the configuration of the

9 Evaluation

participation. The parameters which are realized as a check box were easy to understand, but the more complex ones led to entanglements. Only the definition of a custom text for the participate button was perceived as intuitive. One test person had no problems in adding additional fields for participation and highlighted this feature as very useful.

The second part of the evaluation turned out to be very easy for the test persons. All of them described the interface for participation as clear and were able to fill in the provided input fields without any doubts. In the two scenarios where participation was only possible with an account and an invitation, the test persons also had no problems. In these scenarios, they figured out how to get to the sign up page and how to create a new account.

What did the test persons not understand?

Although the test persons were able to complete all of the given tasks, there have been some cases where they did not know what to do immediately. Only by asking questions and by repeatedly reviewing the parameters which come into question, the intended behavior could be achieved.

In the first part of the evaluation, a few test persons had problems in renaming the agenda. The reason was that they were looking in the wrong place and tried around in the event subsection for agenda items and not in the event settings. Most of the questions came up in the configuration of the participation. The test persons did not understand the difference between the parameters *Show list of event participants on event page* and *Show list of event participants on participation page*. A reason for this ambiguity could be that users, which had never used EVENTHELPR before, do not know that the event page can only be accessed after the participation was completed on the participation page. Another big issue was the definition of additional input fields for participation. Most of the test persons did not recognize the possibility of adding new fields due to a confusing textual description. Some of them tried to specify the additional fields via the info text field for the participants. Two test persons got unsettled because after clicking on *Add field*, an input field called *Name* appears which leads to a misunderstanding regarding the header text *Input fields in addition to standard fields (Name/E-Mail)*. A suggestion in order to improve understandability was to formulate the text differently.

9 Evaluation

The evaluation of the participation process yielded a little ambiguity. If the parameter *Show link to event page after successful participation* is not set for an event, the test persons did not know what to do after successful participation because no link to the event page is appearing. In this case, a suggested solution was to inform the user that no event page is displayed because the data entered for participation is the only purpose of the event (for example in the collective order scenario where the creator of the event just wants to get data from the participants).

What are general suggestions for improvements?

The suggestions for improvements and extensions are mostly related to the configuration of the event participation and the participation page, because the test persons were really focused on the tasks and did not explore other aspects of EVENTHELPR. The most useful ones are listed and shortly described here:

- Add a new boolean attribute *Required* to additional participation fields
- Add a new type *Select box* to attribute *Type* of additional participation fields
- Introduce a way to make the additional participation fields dependent on each other
- Put the area *Input fields in addition to standard fields (Name/E-Mail)* directly beneath the parameter for the e-mail address
- Show contact information of the event creator on the participation page in order to enable users to ask questions regarding the participation, if there are any
- Introduce a new parameter for participation settings which specifies whether the provided information from the additional input fields is shown in each individual item in the list of participants or not
- Improve the display of the additional input fields, which were entered by the participants, in the list *Overview of participants*

What are further scenarios, in which EventHelpr could be used?

Since it is already possible to cover a variety of different scenarios with EVENTHELPR, the suggestions of the test persons did not bring many new insights. Most of the interviewed people stated meetings of different types as main usage scenario. One test person proposed cross-cutting events in

9 Evaluation

the business context between two or more parties as a proper scenario. Since every institution has its own internal structures for the organization of events (in this case especially meetings), the organization of such events could be improved by using an independent solution. Another idea was to use `EVENTHELPR` for external customer meetings in general.

10 Summary

In today's digital age, people tend to use all kinds of devices and applications to make the organization of their lives easier. There are many event management tools out there, which support users in creating and managing events. The variety of tools range from standalone applications to web based solutions and of course mobile applications. After the analysis of several applications, it turned out that most of the applications are very powerful and extensive. Furthermore, nearly all of these applications require the creation of an account and the payment of some fees in order to be able to use them.

In the course of this thesis, a personalized and lightweight event management environment called EVENTHELPR was designed and implemented. EVENTHELPR is a very easy to use web-based application and does not necessarily require an account or any personal user data in order to create and share events. The main advantage of the application is that all important information of an event is centralized on one single page. Events can be easily shared via e-mails by using auto-generated unique links. Invited contacts are able to participate in events and get the information they need. The creator of an event has many different options available to customize the event for a variety of usage scenarios. Especially, the agenda and the participation can be configured in detail to adapt it to their needs.

The agenda of an event in EVENTHELPR can be extended to an interactive agenda. The interactive agenda enables participants to manage items and to rate items with likes and comments. The utility of an item is calculated on the basis of the number of positive and negative postings and the number of supports for each individual posting. For the visualization of group preferences in the context of the interactive agenda, a new approach has been introduced. This approach uses a horizontal bar on a per item basis

10 Summary

where the number of positive, negative and neutral postings is shown. Additionally, the calculated utility of each item is displayed.

The interactive agenda can help participants to make group decisions and prioritize tasks. Since the items in the agenda may also represent various types of information, the field of use of EVENTHELPR is getting bigger.

During the design of EVENTHELPR, many parameters which enable or disable features within events have been introduced. In order to figure out which combinations of components or features are allowed in which configuration, an approach for modeling variability was discussed in this thesis. This approach is based on a feature model of the EVENTHELPR environment and uses conflict detection [23] and model-based diagnosis [34] to identify inconsistencies [24].

Finally, an evaluation of the web application was conducted. The evaluation consists of two parts and was focused on the understandability of the event settings and on the participation process in general. In the first part, test persons had to create events starting from a short description while in the second part test persons had to participate in existing events. In both parts, the test persons were observed and at the end they were asked to give informal feedback.

10.1 Conclusion

The newly developed web-based application EVENTHELPR was positively perceived and is already used for real-world scenarios. With increasing usage of the application, it came out that the parameters in the event settings fulfill their needs and enable EVENTHELPR to be used in a wide range of scenarios. Due to the high flexibility of the event content and settings, creators of events can easily set up events for meetings, workshops, conferences, projects, travels, parties or the organization of orders.

The evaluation showed that users do not have noteworthy problems in creating events, but they do not get it immediately that events can be configured in such a way that more specific scenarios could be covered with EVENTHELPR. Furthermore, the participation in events was an easy task for

10 Summary

the users. The only aspect that was not intuitive was the fact that in some cases no link to the event page was provided after successful participation (for example in a collective order scenario where the participation itself is the only purpose of the event).

10.2 Future Work

Although EVENTHELPR is currently in use and offers a simple way to create events of all kinds, there is still a lot of potential for further improvements and extensions. As with all web applications, the styling of the pages and the user experience is always part of a continuous improvement process. There are many possible small improvements which have appeared permanently and are still going to appear in the future, but the main focus of this section is on the extension of existing features and the introduction of new features for the application. There exist many suggestions that would be very useful for EVENTHELPR, but here only the two most important ones are discussed in detail.

The first new feature which will be discussed is the support of a social sign in for the application. With the integration of this feature, users may get encouraged to sign up to EVENTHELPR more likely. Users would then be able to sign up by using a social provider of their choice (for example Facebook, Google or LinkedIn). It would be also possible to connect an existing EVENTHELPR account with one or more social accounts. The benefit of this feature for the application would be that information from the social account could be used in order to get interests of the user (of course only if the user approves the access to his data). This data could be used for the implementation of recommendations within the application.

The second interesting feature would be the integration of recommendations of different types into EVENTHELPR. On the one hand it would be possible to recommend similar events to users, if there exist public events. And on the other hand it would be possible to recommend other participants to an event participant, if they have common interests or common events they participated in. There are even more possible recommendations that come to mind when looking at the content of events.

Bibliography

- [1] J. Arthur and S. Azadegan. "Spring framework for rapid open source J2EE Web application development: a case study." In: *Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Network*. May 2005, pp. 90–95. DOI: [10.1109/SNPD-SAWN.2005.74](https://doi.org/10.1109/SNPD-SAWN.2005.74) (cit. on p. 34).
- [2] R. Bakker et al. "Diagnosing and Solving Over-determined Constraint Satisfaction Problems." In: *13th International Joint Conference on Artificial Intelligence*. Chambery, France, 1993, pp. 276–281 (cit. on p. 26).
- [3] D. Batory. "Feature Models, Grammars, and Propositional Formulas." In: *Software Product Lines Conference*. Vol. 3714. Springer Lecture Notes in Computer Science. 2005, pp. 7–20 (cit. on p. 27).
- [4] Ankur Bawiskar et al. "Spring Framework: A Companion to JavaEE." In: *IJCEM International Journal of Computational Engineering & Management*. Vol. 15. 3. May 2012, pp. 41–49 (cit. on p. 36).
- [5] D. Benavides, S. Segura, and A. Ruiz-Cortes. "Automated Analysis of Feature Models 20 years Later: a Literature Review." In: *Information Systems* 35.6 (2010), pp. 615–636 (cit. on pp. 26, 27).
- [6] D. Benavides et al. "Automated Analysis in Feature Modelling and Product Configuration." In: *13th International Conference on Software Reuse*. Lecture Notes in Computer Science 7925. Pisa, Italy: Springer, 2013, pp. 160–175 (cit. on p. 26).
- [7] Li Chen and Feng Wang. "Explaining Recommendations Based on Feature Sentiments in Product Reviews." In: *IUI*. 2017 (cit. on p. 10).
- [8] Ray Corrigan. *Digital Decision Making*. first edition. Springer-Verlag London, 2007. ISBN: 978-1-84628-673-5 (cit. on pp. 7, 8).

Bibliography

- [9] K. Czarnecki, S. Helsen, and U. Eisenecker. "Formalizing Cardinality-based Feature Models and their Specialization." In: *SoftwareProcess: Improvement and Practice* 10.1 (2005), pp. 7–29 (cit. on p. 27).
- [10] Michael D. Ekstrand, John T. Riedl, and Joseph A. Konstan. "Collaborative Filtering Recommender Systems." In: *Found. Trends Hum.-Comput. Interact.* 4.2 (Feb. 2011), pp. 81–173. ISSN: 1551-3955. DOI: [10.1561/1100000009](https://doi.org/10.1561/1100000009). URL: <http://dx.doi.org/10.1561/1100000009> (cit. on p. 9).
- [11] A. Felfernig and R. Burke. "Constraint-based Recommender Systems: Technologies and Research Issues." In: *Proceedings of the 10th International Conference on Electronic Commerce*. ICEC '08. Innsbruck, Austria: ACM, 2008, 3:1–3:10. ISBN: 978-1-60558-075-3. DOI: [10.1145/1409540.1409544](https://doi.org/10.1145/1409540.1409544). URL: <http://doi.acm.org/10.1145/1409540.1409544> (cit. on p. 9).
- [12] A. Felfernig et al. "Consistency-based Diagnosis of Configuration Knowledge Bases." In: *Artificial Intelligence* 152.2 (2004), pp. 213–234 (cit. on p. 29).
- [13] A. Felfernig et al. *Knowledge-based Configuration: From Research to Business Cases*. 1st. Elsevier/Morgan Kaufmann, 2014 (cit. on pp. 26, 27).
- [14] Alexander Felfernig, Gerhard Friedrich, and Lars Schmidt-Thieme. "Guest Editors' Introduction: Recommender Systems." In: 22 (June 2007), pp. 18–21 (cit. on p. 9).
- [15] Alexander Felfernig, Martin Stettinger, and Thomas Gruber. "Consistency Management Techniques for Variability Modeling." In: *Enterprise Modelling and Information Systems Architectures – International Journal of Conceptual Modeling* (2018) (cit. on p. 26).
- [16] Alexander Felfernig et al. "An Integrated Environment for the Development of Knowledge-Based Recommender Applications." In: *Int. J. Electron. Commerce* 11.2 (Dec. 2006), pp. 11–34. ISSN: 1086-4415. DOI: [10.2753/JEC1086-4415110201](https://doi.org/10.2753/JEC1086-4415110201). URL: <http://dx.doi.org/10.2753/JEC1086-4415110201> (cit. on p. 9).
- [17] Alexander Felfernig et al. *Group Recommender Systems*. first edition. to appear. Springer International Publishing, 2018. ISBN: 978-3-319-75067-5 (cit. on p. 8).

Bibliography

- [18] Alexander Felfernig et al. "OPENREQ: Recommender Systems in Requirements Engineering." In: *2nd Workshop on Recommender Systems and Big Data Analytics*. 2017 (cit. on p. 11).
- [19] Alexander Felfernig et al. "Persuasion in Knowledge-Based Recommendation." In: *Proceedings of the 3rd International Conference on Persuasive Technology*. PERSUASIVE '08. Oulu, Finland: Springer-Verlag, 2008, pp. 71–82. ISBN: 978-3-540-68500-5. DOI: [10.1007/978-3-540-68504-3_7](https://doi.org/10.1007/978-3-540-68504-3_7). URL: http://dx.doi.org/10.1007/978-3-540-68504-3_7 (cit. on p. 9).
- [20] Alexander Felfernig et al. "Recommendation and Decision Technologies for Requirements Engineering." In: *Proceedings of the 2Nd International Workshop on Recommendation Systems for Software Engineering*. RSSE '10. Cape Town, South Africa: ACM, 2010, pp. 11–15. ISBN: 978-1-60558-974-9. DOI: [10.1145/1808920.1808923](https://doi.org/10.1145/1808920.1808923). URL: <http://doi.acm.org/10.1145/1808920.1808923> (cit. on p. 10).
- [21] Dietmar Jannach et al. *Recommender Systems*. Vol. 24. Jan. 2010. ISBN: 978-0-521-49336-9 (cit. on p. 9).
- [22] J. Jiang, T. Liu, and Y. Liu. "The Construction of E-Business Portal Based on Struts, Spring and Hibernate." In: *2009 International Conference on E-Business and Information System Security*. May 2009, pp. 1–3. DOI: [10.1109/EBISS.2009.5138108](https://doi.org/10.1109/EBISS.2009.5138108) (cit. on p. 35).
- [23] U. Junker. "QuickXPlain: Preferred Explanations and Relaxations for Over-constrained problems." In: *19th Intl. Conference on Artificial Intelligence (AAAI'04)*. San Jose, California: AAAI Press, 2004, pp. 167–172. ISBN: 0-262-51183-5 (cit. on pp. 26, 29, 71).
- [24] K. Kang et al. "Feature-oriented Domain Analysis (FODA) – Feasibility Study." In: *Technical Report, CMU-SEI-90-TR-21*. 1990 (cit. on pp. 26, 27, 71).
- [25] Bart P. Knijnenburg, Niels J.M. Reijmer, and Martijn C. Willemsen. "Each to His Own: How Different Users Call for Different Interaction Methods in Recommender Systems." In: *Proceedings of the Fifth ACM Conference on Recommender Systems*. RecSys '11. Chicago, Illinois, USA: ACM, 2011, pp. 141–148. ISBN: 978-1-4503-0683-6. DOI: [10.1145/2043932.2043960](https://doi.org/10.1145/2043932.2043960). URL: <http://doi.acm.org/10.1145/2043932.2043960> (cit. on p. 9).

Bibliography

- [26] C. Kop and H.C. Mayr. "Conceptual predesign bridging the gap between requirements and conceptual design." In: *3rd International Conference on Requirements Engineering*. Colorado Springs, CO, USA, 1998, pp. 90–98 (cit. on p. 26).
- [27] Dean Leffingwell and Don Widrig. *Managing Software Requirements: A Use Case Approach*. 2nd. Addison-Wesley, 2003 (cit. on p. 26).
- [28] G. Linden, B. Smith, and J. York. "Amazon.com recommendations: item-to-item collaborative filtering." In: *IEEE Internet Computing* 7.1 (Jan. 2003), pp. 76–80. ISSN: 1089-7801. DOI: [10 . 1109 / MIC . 2003 . 1167344](https://doi.org/10.1109/MIC.2003.1167344) (cit. on p. 9).
- [29] H.C. Mayr and C. Kop. "A User Centered Approach to Requirements Modeling." In: *Modellierung 2002*. 2002, pp. 75–86 (cit. on p. 26).
- [30] H.C. Mayr, C. Kop, and D. Esberger. "Business Process Modeling and Requirements Modeling." In: *International Conference on the Digital Society (ICDS 2007)*. Guadeloupe, French Caribbean, 2007, p. 8 (cit. on p. 26).
- [31] Gerald Ninaus et al. "INTELLIREQ: Intelligent Techniques for Software Requirements Engineering." In: *European Conference on Artificial Intelligence, Prestigious Applications of Intelligent Systems (PAIS)*. 2014 (cit. on p. 10).
- [32] Klaus Pohl. *Process-Centered Requirements Engineering*. New York, NY, USA: John Wiley & Sons, Inc., 1996. ISBN: 0863801935 (cit. on p. 10).
- [33] Tom Postmes, Russell Spears, and Sezgin Cihangir. "Quality of Decision Making and Group Norms." In: *Journal of personality and social psychology* 80 (June 2001), pp. 918–30 (cit. on p. 8).
- [34] R. Reiter. "A Theory of Diagnosis From First Principles." In: *Artificial Intelligence* 32.1 (1987), pp. 57–95 (cit. on pp. 26, 29, 71).
- [35] Spring. *Spring Framework*. Jan. 2018. URL: <https://projects.spring.io/spring-framework/> (visited on 01/30/2018) (cit. on p. 34).
- [36] T. Thum, D. Batory, and C. Kastne. "Reasoning about edits to feature models." In: *31st IEEE International Conference on Software Engineering*. IEEE. 2009, pp. 254–264 (cit. on p. 26).

Bibliography

- [37] Thymeleaf. *Thymeleaf - Using Thymeleaf*. Jan. 2018. URL: <http://www.thymeleaf.org/doc/tutorials/2.1/usingthymeleaf.html> (visited on 01/25/2018) (cit. on p. 38).
- [38] Edward Tsang. *Foundations of Constraint Satisfaction*. London, San Diego, New York: Academic Press, 1993 (cit. on pp. 27, 28, 30).
- [39] Katrien Verbert et al. "Visualizing Recommendations to Support Exploration, Transparency and Controllability." In: *Proceedings of the 2013 International Conference on Intelligent User Interfaces*. IUI '13. Santa Monica, California, USA: ACM, 2013, pp. 351–362. ISBN: 978-1-4503-1965-2. DOI: [10.1145/2449396.2449442](https://doi.org/10.1145/2449396.2449442). URL: <http://doi.acm.org/10.1145/2449396.2449442> (cit. on p. 10).
- [40] J. White et al. "Automated Diagnosis of Feature Model Configurations." In: *Systems and Software* 83.7 (2010), pp. 1094–1107 (cit. on p. 26).