Model-driven Development of Social Network enabled Applications with WebML and Social Primitives

Marco Brambilla

Politecnico di Milano
Dip. di Elettronica e Informazione
P.za L. Da Vinci, Milano, Italy
marco.brambilla@polimi.it

Andrea Mauri

Università di Trento, DISI
Via Sommarive,
Povo, Trento, Italy
andrea.mauri@disi.unitn.it

Abstract. Social technologies are transforming the Web to a place where users actively contribute to content production and opinion making. Social networking requirements are becoming a core part of the needs of modern enterprises too, which need ad-hoc Web platforms that incorporate the right set of social features for their business. This leads to the need to provide facilities and methods for developing such socially enabled applications. In this paper we propose a model-driven approach that is specifically focused on the development of Web applications that exploit social features. In particular, we describe an extension of the WebML notation (a Domain Specific Language designed to model Web applications), comprising a set of modeling concepts that encapsulate the logic of the interaction with the social platforms. Upon this, we define a set of design patterns that respond to the typical needs of enterprises and we show some sample application scenarios.

1 Introduction

Social technologies are transforming the Web to a place where users actively contribute to content production and opinion making [SR+08, TK10]. While the broad public is aware of only a bunch of world-spread applications (including Facebook, Gowalla, Foursquare, LinkedIn, Twitter), social networking requirements are becoming a core part of the needs of modern enterprises, at the B2C (Business-to-Consumer), B2B (Business-to-Business), and B2E (Business-to-Enterprise, i.e., the connection between the company and its own internal organization and workforce) levels.

Several examples of applications exist at B2C level, spanning from brand management and viral marketing to Customer Relationship Management, while at B2E level, enterprises look at social networking tools as possible means for improving their operations thanks to the unstructured interaction they foster among employees [TK10].

The growth in the need of specific features within social network and collaboration platforms raised the problem of designing and developing Web applications inte-

MDWE workshop, p. 1, 2012.

grating such a heterogeneous set of services into a single application. The purpose of this is to provide enterprises with ad-hoc Web platforms that incorporate the right set of social features to comply with the specific context of the company.

This leads to the need to provide facilities and methods for developing such socially enabled applications. One option is obviously that of applying traditional developing techniques based on manual coding. However, this solution is quite inappropriate both in terms of efficiency and of effectiveness. Indeed, developing an application integrated with social networking platform with a manual approach implies that the developer must know how each platform works and must rely on different external libraries to interact with the social services. This is time consuming and error prone. On the other hand, existing social enterprise platforms like Salesforce Chatter [SF12] or Tibco Tibbr [Tibco12] now exist, which provide a fixed set of features. Another solution could consist of applying general-purpose model-driven approaches to the problem, possibly specifically focused on web application development [CFB+02, RS+01, KR02]. However, these solutions cannot capture the details of the interaction with the social platforms and therefore still require manual modeling of social network API invocations.

To address these shortcomings, we propose a model-driven approach that is specifically focused on the development of Web applications that exploit social features. In particular, we describe an extension of the WebML notation (a Domain Specific Language designed to model Web applications)[CFB+02], comprising a set of modeling concepts that encapsulate the logic of the interaction with the social platforms. Those modeling concepts provide both cross-social platform capabilities and platform-specific ones and allow seamless integration between the ad-hoc application development and the social networking features. The proposed units can be used within a full-fledged model-driven development cycle that covers all the phases from requirement specification, to high level business need design (with notations based on BPMN or similar) down to application design with WebML and implementation and deployment with automatic code generation techniques. The development method is not part of the contribution of this paper, but can be found in [BFV11].

In this paper we discuss the basic modeling artefacts we define in WebML, the design patterns that can be repeatedly used for solving the typical needs, and then a few sample applications that demonstrate the feasibility and advantages of the approach. Our experiments are run within the MDD tool WebRatio [WR12], a modeling tool that allows automated code generation and fast application deployment starting from BPMN and WebML models.

The paper is organized as follows: Section 2 describes the social components (i.e., WebML units) that model social behaviours; Section 3 describes the design patterns that cover the most common requirements of social applications; Section 4 shows some applications developed with our approach; Section 5 discusses the related work; and finally Section 6 draws the conclusions.

Table 1. Operations that can be performed by the cross-platform social units.

Social Login Unit	Login through social network credentials. It supports in a transparent way all the needed handshaking with the platform and allows to get or reuse an authoriza-

	tion token and to get information on the user. The main actions that can be performed are: starts authorization process and produce URL for redirection; define "landing" custom URL name receiving the authorized user back from the social network; and verify the status of a given authorization token.
Social Search Unit 	Keyword search over social network contacts. It retrieves a set of people whose profiles match the search criteria.

2 WebML Extension: the Social Units

The first contribution we propose in this paper is an extension to the WebML notation for covering social network integration requirements. WebML [CFB+02] is a visual language for designing data- and service-intensive Web/SOA applications. A WebML model consists of one or more site views, which represent hypertext application used to publish or manipulate data and interact with the back end business logic. A different site views can be defined for each process actor; internally, a site view consists of a set of pages, atomic units of interface, containing units, representing data publishing components. Units are related to each other through links, representing navigational paths and parameter passing rules. Additionally, the WebML application model may comprise the definition of backend operations, parallel and independent threads (which can be activated manually, automatically or based on temporal triggers), Web services, REST APIs and their invocations.

In order to enable the development of social applications using a model-driven approach, we extended the WebML notation by adding a set of units that encapsulate the logic of the interaction with the social platforms. These units are designed as wrappers of the social platform APIs and hiding the underlying complexity from the developer, reducing the cost of designing new applications. The units are divided in three sets: cross-platform units, social platform-specific units, and collaboration platform units.

First we designed the conceptual definition of the units, analyzing the most common functions provided by the social networks, then

we implemented their behavior within the WebRatio tool [BBF10]. This has been obtained by implementing new WebRatio components and model transformation rules that allows automatic code generation from models. The code generated from WebML models is a standard Java application, which can be deployed on any Java application server. Connectivity to the social software is realized by APIs calls to the external platforms, which is the concrete way to implement the Social units.

2.1 Cross-platform Units

The units belonging to this group can perform operations (enumerated in Table 1) on multiple social networks at a time. These units are thought as conceptual representations of behaviors that are common among all the social networks. The units we implemented up to now provide two basic behaviors: login through social network credentials, and search over the set of contacts in the social network. The social networks we cover with our implementation are: Facebook, Twitter, LinkedIn and Google+. For both units, one can decide to query one or more networks at a time, by choosing the network either at design time or at run time.

Table 2. WebML operations that can be performed by the social platform-specific units.

	Operation	**Description**
Facebook Unit 	Verify Token	Verify the status of the authorization token
	Get User Id	Get the id of the current logged user
	Get Friends	Get the list of friends of the logged user
	Post to Wall	Post a message on the logged user's wall
	Post to Friend	Post a message on the wall of a friend of the logged user
	Post Comment	Post a comment to a given post
	Post Note	Post a note
	Create event	Create a event
	Invite to Event	Send an invitation to a friend to participate to an event
	GetPost fromWall	Get the list of posts form the wall of the logged user
	Get Comments	Get the list of comments of a given post
	Upload photo	Upload a photo to the user's Facebook account
	Tag photo	Tag a photo
	Get Groups	Retrieves the list of groups of the logged user
Twitter Unit 	Get User Id	Get the id of the current logged user
	Get Friends	Get the list of friends[1] of the current logged user
	Send Message	Send a direct message to a friend
	Tweet	Post a tweet
	Search	Perform a keyword search on Twitter
	Get Tweets List	Get the tweet list of a given user
Linkedin Unit 	Keyword Search	Perform a keyword search over the user's connections
	Get connections	Get the first-level connections of the current logged user
	Message	Send a message to a connection
	Get User Id	Get the user id of the logged user

[1] We define "friends" in Twitter the people that follow and are followed by the user. This is also the necessary condition for sending direct messages.

2.2 Social Platform-specific Units

The units belonging to this group encapsulate all the operations specific of one social network, and up to now they implement the set of API functions enumerated in Table 2 The units developed up to now cover the networks of Facebook, Twitter and Linkedin. Each unit allows invoking an operation within a large set of actions available through the API.

2.3 Collaboration platform units

The units belonging to this group represent the interaction on services that enhance the collaboration between users. In particular, the units that have been developed so far include interaction with Doodle, Google Docs and Google Calendar and implement the functions enumerated in Table 3. These units do not address social networking, while instead focus on information sharing and collaboration, which are some additional crucial aspects of the Web 2.0 paradigm.

Table 3. WebML operations that allow integration with collaboration platforms.

Doodle Unit 	Create Poll	Create a poll with the given options
	See Poll Details	See the details of a given poll
	Vote Poll	Select an option of a given poll
	Close Poll	Close a given poll
	Comment Poll	Comment a given poll
Google Calendar Unit 	Create Calendar	Creates a calendar
	Create Event	Creates an event on a given calendar
	Get Events	Get the list of events from a calendar satisfying some conditions
Google Docs Unit 	Get Documents	Get the list of the documents owned by the user
	Upload Documents	Upload a document to Google Docs

3 Social design patterns

In this section we show how the units we developed can be used to implement reusable design patterns that address the typical requirement of Web 2.0 applications. We refer to the needs presented [B12] as a starting point for our analysis and we identify a set of social design patterns with the aim of covering the most common requirements that a social application must fulfill. Notice that the patterns we present are derived from typical usage we registered in the design of various social-enabled applications.

The most important patterns we identify are: Post, Comment, Send Message, Like/ Vote/ Rate, Login, Group Management, Event Management, Content Management, People Search, and Content Search.

Since the different units implement different operations, not all the units can be used to implement a specific pattern. This is either due to a missing feature in the unit conceptualization, or because the social platform doesn't provide a specific feature, or because the platform is not addressing the issue at all. The choice of implementing a feature within a unit is also based on orthogonality reasons. For instance, while it could have been possible to implement a Facebook login function within the Facebook unit, we decided to provide the login only within the cross-platform Login unit. Table 4 shows the mapping between the identified patterns and their implementability within the various units.

The following subsections describe each pattern and show how they can be implemented using the social units. Notice that this section is not meant to show the usage of the units, but instead aims at describe reusable design solutions to the problem of expanding the features of a Web application to the social networking needs. As such, they represent conceptual models that can be easily represented with other alternative notations such as UWE, OOWS, or others.

Table 4. Social Design Pattern vs. Unit type. Y: unit supports the pattern; N: unit does not support the pattern; ND (not defined): the specific platform does not support the pattern.

WebML Units / Patterns	Cross Platform	Facebook	Twitter	Linkedin	Doodle	GDoc	GCalendar
Post	N	Y	Y	N	ND	ND	ND
Comment	N	Y	N	N	Y	ND	ND
Message	N	Y	Y	Y	ND	ND	ND
Like/Vote/Rate	N	N	ND	N	Y	ND	ND
Login	Y	N	N	N	ND	N	N
Group Management	N	Y	ND	N	ND	ND	ND
Event Management	N	Y	ND	N	ND	ND	Y
Content Management	N	Y	N	N	ND	Y	ND
People Search	Y	Y	Y	Y	ND	ND	ND
Content Search	N	N	Y	ND	ND	N	N

3.1 Social Login

This pattern implements the interactions for allowing the login through the credentials of a social platform. For this purpose we use the cross-platform Social Login unit. It implements the Oauth authentication protocol in order to obtain the login response and to get permission to use the service API. The protocol is implemented using the units as shown in Figure 1.

In the *Social Login Sample* page the user see a form (the unit *Social Network*) in which he can choose which social network use to perform the login; then the operational unit *Social Login* redirects the user to the social network authentication page. Eventually the user is brought back through the green OK link. In case of error, the user is redirected to the *Social Login Failed* error page. If the Oauth token is already present and valid the authentication phase is skipped.

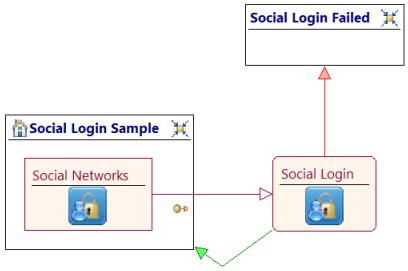
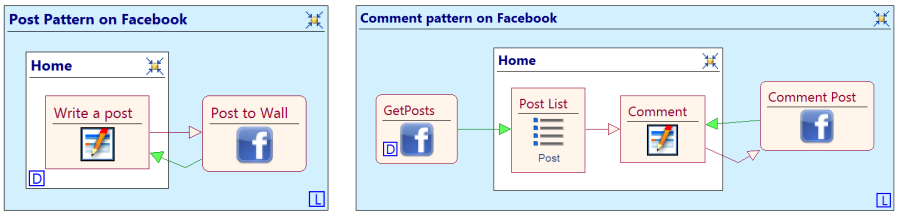


Fig. 1. The *social login* pattern modeled in WebML with the cross-platform Social Login unit.

3.2 Post

This pattern covers all the cases in which a message is published to an undefined number of users (e.g., on the Twitter timeline or Facebook wall). Fig. **2**.a shows the WebML diagram describing the implementation of the Post pattern using the Facebook unit. In the *Home* page to the user can enter the post text through the *Write a post* form. Once the form is submitted, the message is posted on the user's wall by the *Post to Wall* operation unit.



(a) (b)

Fig. 2. The *post* pattern (a) and the *comment* pattern (b) implemented with the Facebook unit.

3.3 Comment

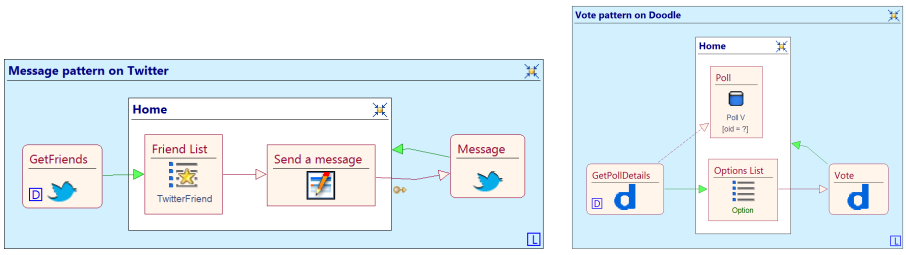
This pattern describes the workflow that is executed when a message is published as a comment to another content on a social network. Fig. **2**.b shows the Comment pattern implemented using the Facebook unit. In the *Home* page, the index unit *Post List* presents to the user the list of posts from his Facebook wall, retrieved by the *GetPosts* operation unit. The user can select one post, write his comment in the *Comment* form, and submit and publish it with the *Comment Post* unit.

3.4 Message

This pattern describes the workflow that is executed when a message is sent to a single, identifiable user (e.g., a Twitter / Facebook direct message). Fig. **3**.a shows the implementation of the Message pattern using the Twitter unit. In the *Home* page the index unit *Friend List* shows to the user the list of his Twitter friends retrieved by the *GetFriends* unit. The user can select a friend and compose the message through the form *Send a message*, which submits it through *Message* unit

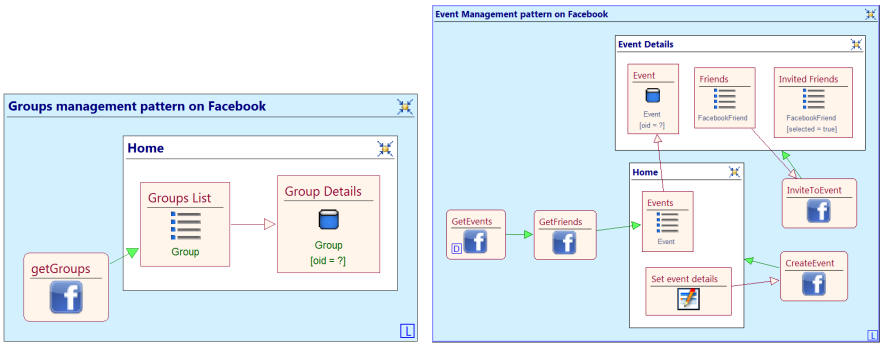
3.5 Like-Vote-Rate

This pattern describes the social interaction for expressing of a preference (liking) or the assignment of a score to an item. Fig. **3**.b shows the WebML design pattern implemented with the Doodle unit. The *GetPollDetails* unit retrieves the details of a given poll and shows the poll question with the options the user can vote using the *Vote* unit.



(a) (b)

Fig. 3. The message pattern implemented using the Twitter units (a) and the vote pattern implemented using the Doodle units (b).



(a) (b)

Fig. 4. The groups management (a) and event management (b) patterns in WebML.

3.6 Group Management

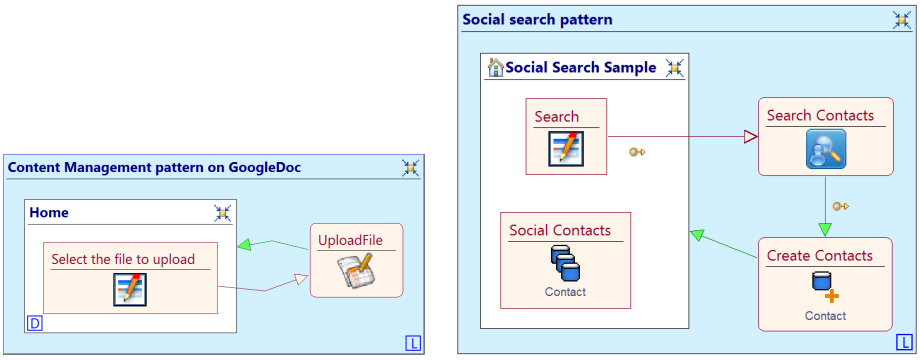
This pattern describes the interaction with the social platform for the management of user groups, for instance retrieving information about the groups a user belong to. **Fig. 4**.a shows the WebML diagram for groups management using the Facebook units. In the *Home* page the *Group List* index unit shows the list of the groups the user belongs to (retrieved with the *getGroups* unit). By clicking on a specific group, the *Group Details* data unit shows to the user the details of the group.

3.7 Event Management

This pattern comprehends all the actions that enable the management of social events like meetings, or others. For example scheduling events, inviting people, and so on. **Fig. 4**.b shows the WebML diagram describing the event management pattern implemented using the Facebook unit. In the *Home* page the *Events* index unit shows to the user the events retrieved by the *GetEvents* unit. Clicking on an event the user is brought to the *Event Details* page, where he can invite some friends (through the *InviteToEvent* unit) by selecting them from the *Friends* index unit. The *Set event details* form in the *Home* page lets the user create a new event through the *Create event* unit.

3.8 Content Management

This pattern describes the interaction with the social platform in order to manage binary content (i.e., photos or documents). Fig. **5.a** shows the WebML content management pattern using the GoogleDocs units. The *Select the file to upload* form in the *Home* page lets the user browse his computer for a file he wants to upload. Then the *UploadFile* unit uploads the file to the user's Google Docs account.



(a) (b)

Fig. 5. The content management pattern implemented using the GoogleDocs unit (a) and People search pattern implemented using the Cross-platform units (b).

3.9 People Search

This pattern describes the search of contacts in a social network based on text search criteria. Fig. **5.b** shows the pattern implemented using the Cross-platform units. In the *Social Search Sample* page the user can insert the keywords to be searched within his social contacts. The *Search Contacts* unit carries out the actual search. The *Create Contacts* unit creates the *Contact* objects in the user session and finally the *Social Contacts* multi-data unit shows the list of the retrieved contacts.

4 Sample Application Scenarios

In this section we show 5 simple applications built using WebML extended with the new social units proposed in this paper and used according to the above patterns.
All these applications are modeled within WebRatio tool, which allows automated code generation and fast deployment of the applications, thanks to the fact that specific code generation rules have been devised for the new units.

4.1 Twitter keyword search

This simple application allows the user to perform a keyword-based search over the Twitter timeline. **Fig. 6** shows the WebML diagram of this application. In the *Home* page to the user is presented a form (the *Search Form* entry unit) where he can submit a search criterion or ask for more results for the last search.

In the former case the keyword is passed directly to the *Search* unit that performs the search, while in the latter the *getLastTweet* unit returns the id of the last tweet retrieved in the last search that is passed to the *Search* unit as the starting point in the result list, together with the keyword used. In both cases, the retrieved tweets are stored in the user session and then shown in the *Tweets List* index.

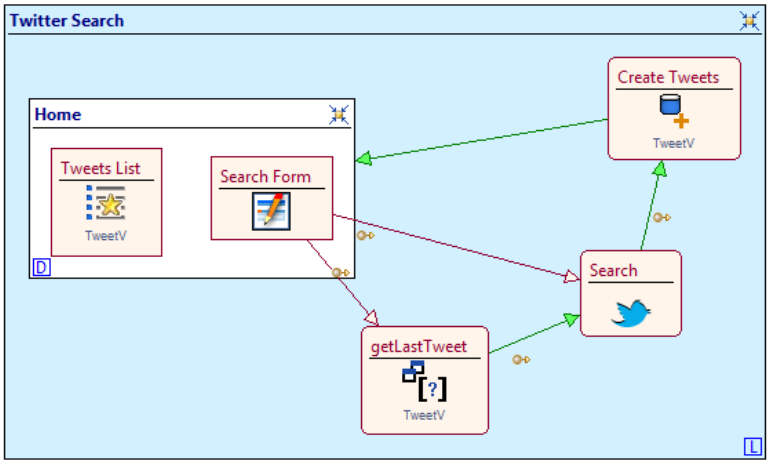
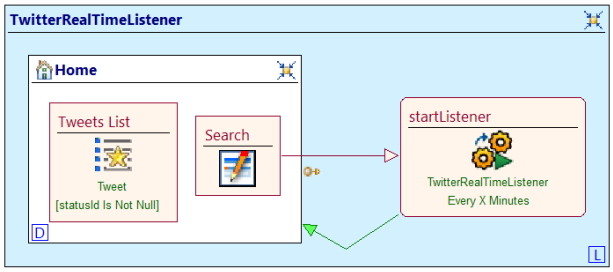
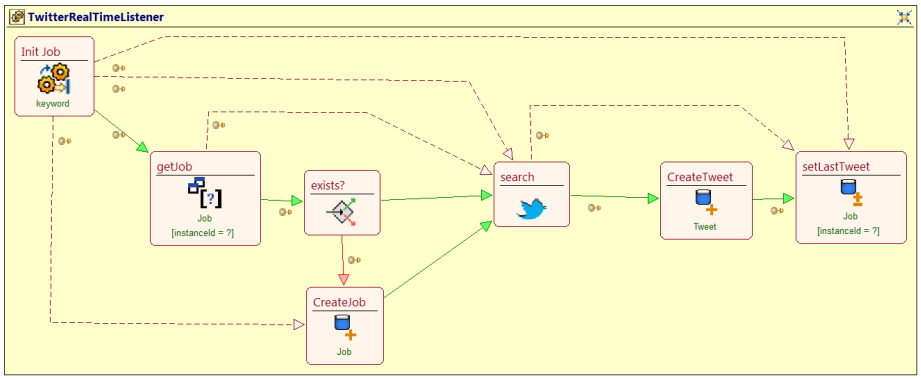


Fig. 6. WebML diagram of the Twitter Keyword Search application.



(a)



(b)

Fig. 7. WebML diagram of the Twitter real-time listener application (a) and of the WebML *thread* that retrieves the tweets from the timeline every X minutes (b).

4.2 Twitter real-time listener

This application allows the user to follow the stream of tweets that talk about a given topic. **Fig. 7** (a) shows the WebML diagram of the application. The user, through the *Search* form, starts a thread that every X minutes gathers the tweets that contain the words specified in the form. **Fig. 7** (b) shows the diagram of the thread, which selects the last tweet retrieved in the previous iteration and performs the search over the Twitter timeline starting from the last tweet. Then the tweets are stored in the database and the last tweet is updated. In his *Home* page, the user sees the new tweet list.

4.3 Meeting setup

This application allows the user to create meeting with his Linkedin contacts, with the possibility of deciding the date of the event with a poll created in Doodle, as shown in **Fig. 8**. In the page *Search Contacts* the user can search over his Linkedin connection for people to invite to the meeting. Then, in the second page (*Add Time option*), the user can add different date options for the meeting. In the *Define Meeting Details* page, the user configures some aspects of the meeting (topic, description, location). Finally a poll is created on Doodle with the *Create Poll* unit, and a message is sent to the contacts invited to the meeting with the *Send Message* unit. At the end to the user is shown a page that summarizes the details of the created event.

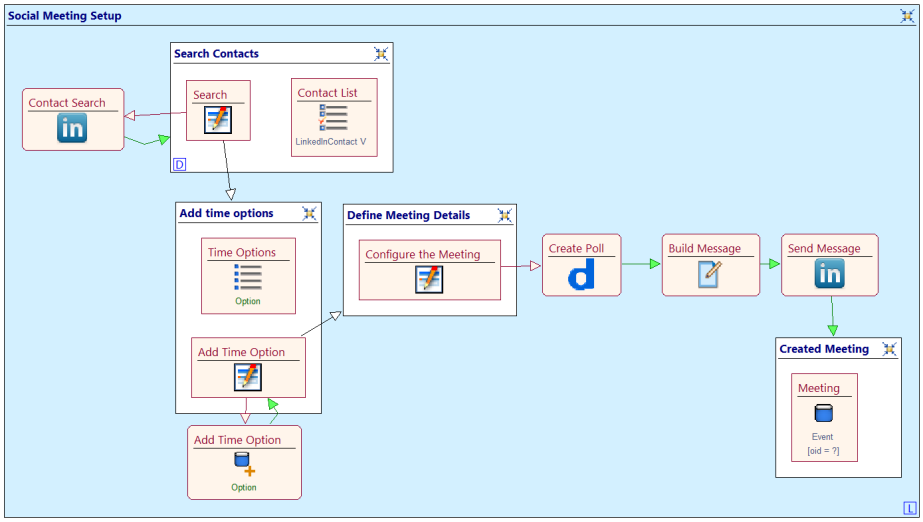


Fig. 8. WebML diagram of for the creation of meetings using Linkedin and Doodle.

4.4 Crowdsourcing-based Search

CrowdSearcher [BBC12] is an application that allows posting questions with structured objects over different social networks. **Fig. 9** shows the WebML diagram of the query creation phase, in which the user creates the query to be posted on the social networks, by defining a textual question, creating a schema and adding a list of structured object. In the *Responder Selection* page, the user must select among his friends the recipient of his question.

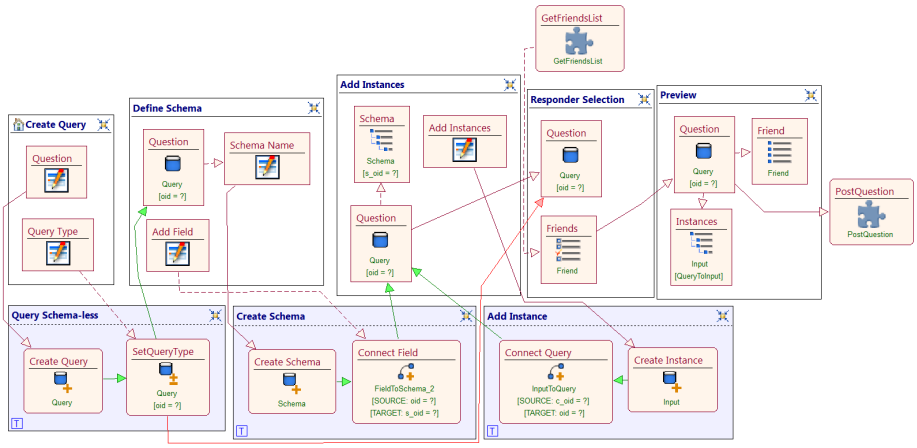


Fig. 9. WebML diagram of the query creation phase of CrowdSearcher

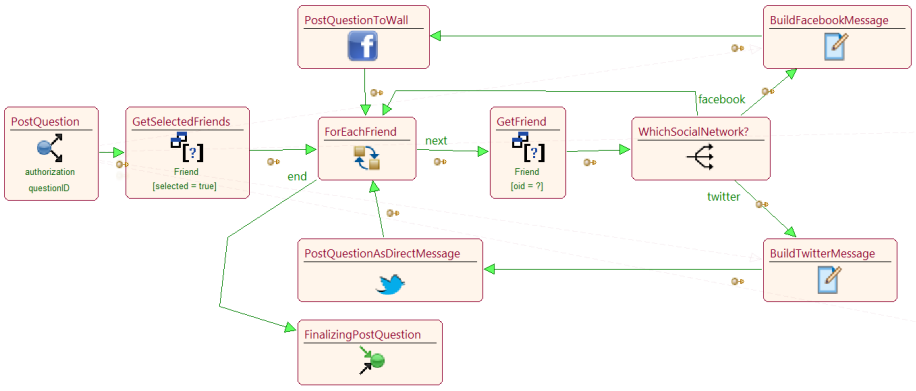


Fig. 10. WebML diagram of the module that post the question on various social networks

The friend list is retrieved at run time by the *GetFriendList* module, which selects the friends of the logged user in each social network. This module is not shown here for space reasons; the reader can easily figure out the general behavior of the module,

which is inspired to the People Search design pattern presented in Section 3.9. After selecting the recipients, the query is posted on different social networks by the *PostQuestion* module (whose WebML diagram is depicted in **Fig. 10**).

The module receives as input the id of the query and the authorization level required by the user (public or private question), then for each selected friends, depending on the social network he belongs to, it either sends a private message on Twitter or posts it on the Facebook wall.

5 Related Works

This paper applies MDD techniques to a specific domain of Web applications, namely Social-networking enabled applications. Other Web engineering approaches have tackled these problems. Among the various Web engineering methods, we can mention OOWS as one of the most complete approaches with respect to requirement specification and social interaction coverage [VP+09]: the method proposed there provides a set of design patterns, covering also social aspects and also a mapping to an executable model.

Some works already tried to collect and classify design patterns for social web applications. Among them, we mention [FT+10], collecting WebML design patterns describing the most used social interactions within online platforms. The paper [B12] presents a requirements-engineering driven approach to customized social network development. Other preceding works focused on pattern-based development [KR02] [RS+01] and have been inspirational for this work too.

6 Conclusions and Future Works

In this paper we described a model driven approach focused on the development of Web applications that exploits social services. We presented the social units developed as an extension of the WebML notation, which encapsulates all the business logic of the interaction with the social platform. Based on these units, we identified a set of design patterns that solve the most common requirements for socially enabled applications. Subsequently, we demonstrated the validity of our approach by describing five application scenarios developed using the new units and patterns within the WebRatio MDD tool.

Future works will aim both at refining the current set of social units, by extending the behaviour of the existing units and by creating new units in order to model the interactions with other social platforms and social needs. A very important step will be to factorize the API calls that are currently spread all over the set of the network-specific units into a set of appropriate cross-social network units, in order to group functions that share the same semantic.

Acknowledgements. This research is partially supported the BPM4People project, funded by the 7th Framework Programme of the European Commission. We thank all the project contributors for their efforts and useful discussions.

7 References

[BBC12] Bozzon, A., Brambilla, M., Ceri, S.: Answering search queries with CrowdSearcher. Proc. WWW Conference 2012, Lyon, France, pp. 1009-1018.

[CFB+02] Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., Matera, M.: Designing Data-intensive Web Applications. Morgan Kauffman, 2002.

[B12] Brambilla, M.: From requirements to implementation of ad-hoc social Web applications: an empirical pattern-based approach. IET Software, 2012, in print.

[BBF10] Brambilla, M., Butti, S., Fraternali, P.: Business Process Modeling and Quick Prototyping with WebRatio BPM. Proc. of BPM Demonstration Track 2010, Hoboken, USA, September 14-16, 2010, Vol. 615 CEUR-WS.org. online: http://ceur-ws.org/Vol-615.

[BFV11] Brambilla, M., Fraternali, P., Vaca, C.: BPMN and Design Patterns for Engineering Social BPM Solutions. International Workshop Series on Business Process Management and Social Software (BPMS2) 2011, co-located with BPM 2011, Clermont-Ferrand, France, in print.

[FT+10] Fraternali, P., Tisi, M., Silva, M., Frattini, L.: Building Community-Based Web Applications With a Model-Driven Approach and Design Pattern. Handbook of Research on Web 2.0, 3.0, and X.0: Technologies, Business, and Social Applications. San Murugesan (ed.), IGI Global, 2010.

[F10] Fuchs, C.: Social Software and Web 2.0: Their Sociological Foundations and Implications. Handbook of Research on Web 2.0, 3.0, and X.0: Technologies, Business, and Social Applications. San Murugesan (ed.), pp. 763-789, IGI Global, 2010.

[KR02] Koch, N. and Rossi, G.: 'Patterns for adaptive web applications'. Proc. 7th European Conference on Pattern Languages of Programs, 2002

[RS+01]Rossi G., Schwabe D., Danculovic J., Miaton L.: 'Patterns for Personalized Web Applications'. Proc. of EuroPlop, 2001, pp. 423-436

[SF12] Salesforce Chatter, http://www.salesforce.com/chatter/whatischatter/

[SR+08] Subrahmanyama, K., Reich, S. M., Waechter, N., Espinoza, G.: Online and offline social networks: Use of social networking sites by emerging adults. Journal of Applied Developmental Psychology. Volume 29, Issue 6, November-December 2008, Pages 420-433.

[Tibco12] Tibco Tibbr, http://www.tibbr.com/

[VP+09] Valverde, F., Panach, I., Aquino, N., Pastor, O.: Dealing with Abstract Interaction Modelling in an MDE Development Process: a Pattern-based Approach. New Trends on Human-Computer Interaction. Springer, London (2009)

[TK10]Yang, T. A., Kim, D. J.: A Comparative Analysis of Online Social Networking Sites and Their Business Models. Handbook of Research on Web 2.0, 3.0, and X.0: Technologies, Business, and Social Applications. San Murugesan (ed.), pp. 662-672 , 2010.

[WR12] WebRatio, http://www.webratio.com/