Generic Community Management Functionality for the Cobricks
Community Platform Toolkit

Michael Koch
Technische Universität München, Munich, Germany
kochm@in.tum.de

Work in Progress Paper

Abstract

Community management functionality is becoming more and more important in the still growing field of community support platforms. We present a conceptualization of “community management” and a first approach of how generic community management support can be provided in the community platform toolkit Cobricks.

1 Introduction

Community support is becoming more and more important in different application domains like the support of communities of practice for knowledge management or the support of customer communities in marketing. In contrast to the early days of community support in the Usenet these new applications have in common, that management of the communities is desirable or even needed to make the community (or more precisely the community platform) a success. While the discussion of general management issues has been the topic of several publications on community support [e.g. Hagel/Armstrong 1997, Figallo 1998], there has not been much work on how to (generically) support community management.

Cobricks is a toolkit for building (Web- and Web-Service based) community support platforms (see www.cobricks.org). In the context of the development of Cobricks we were faced with the question how generic community management functionality should look like. The first results of this quest are presented in this paper – a conceptualization of “community management” resulting in the identification of some generic areas how community management support functionality could be structured, and a first set of ideas of how this can be integrated in Cobricks.

2 Community Support Platforms and the Cobricks Toolkit

In general a community is a group of people who share some interest, identify with a common idea or more generally belong to a common context. Additionally, community
members usually are cooperating to help each other – by exchanging information, providing general resources etc. [Mynatt et al. 1997; Ishida 1998; Wenger 1998].

The use of networked computers to support communities can be traced back to the beginnings of the Internet. Starting with proprietary communication platforms like Usenet News, the last years have brought an explosion of public and restricted community support platforms that are usually based on Web technologies. Examples are different bulletin boards (forums) or knowledge management applications in companies. Additionally, the hype generated by Hagel and Armstrong [1997] resulted in many companies trying to establish platforms for supporting (and exploiting) customer communities.

The problem still remaining when building community support platforms is that usually the software is programmed particularly for the particular community. There is no generic toolkit for building community support platforms, and there is no interoperability among different community support platforms [Koch 2002].

With Cobricks we have started to address these issues: providing a generic toolkit for building community platforms that provides standard interfaces to other Cobricks platforms and other services for guaranteeing interoperability. Cobricks provides a clear architecture and basic data concepts for users, items and categories that can be extended using ontologies.

Cobricks has been developed for seven years and different (large-scale) community platforms have been built using the toolkit during this years. While some community management functionality can be built for each platform individually using low level functionality in Cobricks, there currently is no particular support for community management in the toolkit.

3 Community Management

Communities differ from teams in not having an organization attached, being mainly based on intrinsic motivation of the community members. This however does not say that communities do not have to be managed – in the contrary.

Reviewing existing work on building communities (like [Hagel/Armstrong 1997, Figallo 1998]) one has to conclude that building communities (community platforms) as technical systems only does not work. Communities involve people (social systems) and cannot be engineered like technical systems. So, everything about community building is about supporting or managing communities.

Management characterizes the process of leading and directing all or part of an organization, often a business, through the deployment and manipulation of resources. One can also think of management functionally, as the action of measuring a quantity on a regular basis and of adjusting some initial plan, and as the actions taken to reach one's intended goal. From this perspective, usually five management functions are identified: **planning**, **organizing**, **leading**, **coordinating** and **controlling**.

We have applied management theory to communities and have derived three basic community management functions that can and should be supported in community support systems [Ljepoja 2005]: **monitoring**, **moderation**, and **motivation**

**Monitoring**: The main task of monitoring is to capture the basic structural variables of an (online) community to derive appropriate management actions.
Information gathered in monitoring can be interesting both for community administrators and for community members in general. In the latter case one usually talks about “community awareness” [Koch 2005]. Since community management is often implemented as at least partly self-management of the community, the usage of monitoring information to display to all community members cannot be overestimated.

**Moderation:** The task of a moderator is to keep the interaction in the community in balance with the norms and values of the community. As already mentioned, an important issue in community management is that communities have the potential to self-regulate themselves. Because of this, community platforms often do not need massive external interference - if the members have the tools for doing self-regulation.

The simplest case of moderation is that moderators (especially selected ones or all the members of the community) determine which content should be made available in the community – according to the norms and values or culture of the community. One can distinguish pre-moderation (a contribution has to be approved before it is published), post-moderation (after being published, a contribution can be canceled), both in a centralized or distributed way.

A very extensive example of moderation functionality can be found in the community platform Slashdot.org (see http://slashdot.org/faq/com-mod.shtml).

**Motivation:** There are various reasons to join a community and to contribute to it. Because of that, there is a wide range of online communities. Common to all of them is that the members have to get added value by joining and contributing to a community. However, this added value can be quite different for different members. Some people help because they received help in the past or they are hoping to receive help in the future (*added value, principle of reciprocity*). Others help because they like to be seen as experts (*appreciation*, see [Hall/Graham 2004, Ludford 2004]).

Visibility is the one thing that can help most for motivation: People who like being seen as experts are more motivated if this status is publicly visible to all members. This is supported by the Top-10-Contributor lists for example. People who hope for help are more likely to read or contribute when they see that the community is active (visualization of inflow and outflow, see (Glance et al. 1998) for an example). Providing awareness of the activity in the community is key to different forms of motivation. This is documented in different works, e.g. by Viegas and Donath who found that increasing social presence results in increasing contribution levels in online communities [Viegas/Donath 1999].

### 4 Community Management Functionality in Cobricks

Cobricks already provides very generic functionality for handling data that has been contributed by community members. Starting with the general conceptualization presented in the previous section we have extended this functionality to particularly support the different facets of community management.

**Monitoring:** Functionality for generating and distributing notifications about basic user activities is already available in Cobricks. Currently, this includes only activities on the generic data objects in the platform (generation and reading of items or user attributes). For supporting community management we added the possibility to generate events based on any user actions, and to aggregate events to new ones (for condensing/abstracting the information). This functionality is provided in a new Awareness
Component together with the possibility to store all events in a database. This information can be displayed in different views on the administrator user interface or on the user interface for the all community members.

**Motivation:** The Awareness Component presented above already provides several possibilities for supporting motivation (by making monitoring information available to all users). Additionally, we added a generic incentive system based on user status points. The system supports generic rules to define how users can earn status points, and provides mechanisms for displaying this information (motivation by appreciation) or for basing access permissions and group membership on the number of status points a user has earned (motivation by added value).

**Moderation:** The most generic way to contribute content to a Cobricks platform is the item. In the field of moderation we worked on making a generic hierarchical moderation system available for any kind of item – and therefore for all content classes used in particular instances of Cobricks. The moderation system supports all modes of moderation (pre-moderation, post-moderation, reactive moderation) by generically extending the meta information of items via so called additional moderation items and moderation categories.

Moderators are defined by their membership to the moderation categories, which makes all possibilities of Cobricks to define membership available for defining moderators, including the definition of membership based on user status points.

### 5 Summary and Outlook

In the past sections we have briefly described our approach to think of and build a generic functionality for community management for the community toolkit Cobricks. We have found that the classification of community management in the three categories monitoring, motivation and moderation helps to think about the domain, and helps to come up with generic functionality for the task.

The functionality presented in the previous section is designed to enable community platform builders to mimic all existing forms of community management and to extend them to their own ones. We are currently making the new functionality available on some of the Cobricks live platforms to get feedback for further development of the conceptualization and of the functionality.

### References


