# TweetGames: A Framework for Twitter-based Collaborative Social Online Games

Markus Esch*, Aleksandrina Kovacheva*, Ingo Scholtes† and Steffen Rothkugel*

* University of Luxembourg
Computer Science and Communications Research Unit
Luxembourg, Luxembourg
Email: markus.esch@uni.lu, aleksandrina.kovacheva.001@student.uni.lu, steffen.rothkugel@uni.lu

† University of Trier
Department of Computer Science
Trier, Germany
Email: scholtes@syssoft.uni-trier.de

***Abstract*—Today social networks and microblogging services attract much attention and their importance and pervasion is constantly increasing. This trend is fostered by the increasing prevalence of mobile internet devices, which enable users to be online every time and everywhere. A popular kind of applications provided via social network platforms are games. While existing social network games basically realize common online game principles where each user controls a single game entity and acts on its own behalf, this paper presents a framework for the creation of a novel kind of social online games. The idea is to enable games where users do not act as individual game entities, instead groups of users control one game entity whose behavior emerges from the collective behavior of all group members. The framework is based on the microblogging service *Twitter* and users interact with the game via Twitter messages. These collaborative social online games are not just an innovative social network application, the intention of our framework is rather to provide a useful and powerful tool to complex social network researchers to study emergent and collective user behavior on a large scale utilizing the huge user base of a social network service like Twitter. In addition to presenting the idea of collaborative social online games along with the so-called *TweetGame* framework, this paper presents a sample application that has already been realized on top of this framework as proof of concept.**

***Index Terms*—Social Networks, Collaborative Games, Twitter, Collaboration, Emergence, Collective Behavior, Social Dynamics**

## I. Introduction

The increasing prevalence of social network services like *Facebook*[1], *Twitter*[2] or *LinkedIn*[3] has changed and accelerated every day live and social interactions. On the other hand social aspects yield the demand of novel network infrastructures which consider emerging phenomena and the high dynamics in social networks. By this means, technical and social systems exert mutual influence and the field of socio-aware technical network systems establishes a new interdisciplinary field of research incorporating computer science, complex network science as well as social sciences.

From the computer science perspective, one of the key challenges in this field is to answer the question, how emergent behavior of socially interconnected groups can be analyzed and modeled in order to dynamically adapt technical network systems to social dynamics [1] and emergent phenomenas like internet memes or slashdot effects. For this purpose it is important to acquire a deep knowledge about the collective user behavior in social networks. This includes studying and modeling of dynamic process in social networks like agreement processes [15], collective behavior [8] [16], influence propagation [12], consensus dynamics [19] [18], collective motion [22] [23], opinion dynamics [11] [10], social dynamics [5], emergent phenomenas [6] etc. The Twitter-based gaming framework presented in this paper enables complex social network researchers to study and analyze these aspects of social networks. The framework introduces the concept of collaborative social online games, a novel and innovative kind of social network application. The basic idea is to create games where users do not act as individual game entities as in traditional online games. Instead groups of users act as one game entity and the behavior of this entity is emerging from the collective behavior of the group members. The *TweetGame* framework presented in this paper provides a platform for the implementation of this kind of online games based on the microblogging service Twitter, which means that users participate in these games by sending twitter messages to the game. The framework offers extensive support for the implementation of TweetGames in terms of providing an application server, handling Twitter messages, managing user groups and so on.

A simple example for such a TweetGame is the Gorillas game presented as proof of concept in this paper. This game features two competing teams each represented by a gorilla. The goal for both teams is to hit the other gorilla with a banana that is thrown at the opponent. For this purpose the game

[1] www.facebook.com
[2] www.twitter.com
[3] www.linkedin.com

COLLABORATECOM 2011, October 15-18, Orlando, United States

DOI 10.4108/collaboratecom.2011.8

defines two input parameters: the angle and the power of a throw. All users are able to send Twitter messages with their guess for the input parameters. The actual parameters for a throw are then given by the mean values of all group members. While this scheme is a quite simple example for the mapping of individual input onto group behavior more sophisticated schemes are realizable as well.

This kind of collaborative social online games enables various experiments and empirical studies in the field of social dynamics as well as complex social networks. It is for example intended to utilize the Gorillas game for studies on the wisdom of crowd effect [7], [14], [21]. In this regard the huge user base of social networks and the popularity of online games provides the opportunity to conduct studies on a large scale and to aggregate large data sets about collective user behavior, agreement process, opinion spreading etc. Social science can profit from our framework by the opportunity of analyzing social process on a larger scale than this is often possible in traditional empirical studies.

The rest of this paper is organized as follows. After introducing the concept of collaborative online gaming in Section II, Section III describes the TweetGame framework in detail. Subsequently Section IV presents the Gorillas game that has been implemented as prove of concept on top of the TweetGame framework. Section V discusses the scientific applicability of our approach and presents usage scenarios. Related work is presented in Section VI. The paper concludes with a discussion of our main contributions, open issues and future work in section VII.

## II. Collaborative Social Online Gaming

The basic idea behind collaborative social online gaming is to provide games where users act collaboratively in groups in order to reach certain common goals. The main difference to existing online games is the fact that users do not act as individual game entities, rather groups of users form one game entity which is controlled by the emergent behavior of all group members. By this means, no user is able to perform game actions solely by individual activities. Instead the collective behavior of a group is considered.

Figure 1 depicts the basic principle and the general control flow of games based on the TweetGame framework. All games are round-based and users interact with the games by sending Twitter messages (*Tweets*). Each game defines two or more different teams. Before being able to play users have to register to one of the teams. The actual number of different teams depends on the specific game. The sample game presented in section IV for example supports just two different teams. Users can join teams at any time during the runtime of a game. It is also possible to switch to another team later on. But, in order to prevent cheating, a team memberships persist for at least one round. In order to join a certain team in a certain game, a user has to send a Twitter message with a join command to the Twitter account of the game. Note, that each game has an associated Twitter account and user commands are sent as Twitter messages directed to this account using

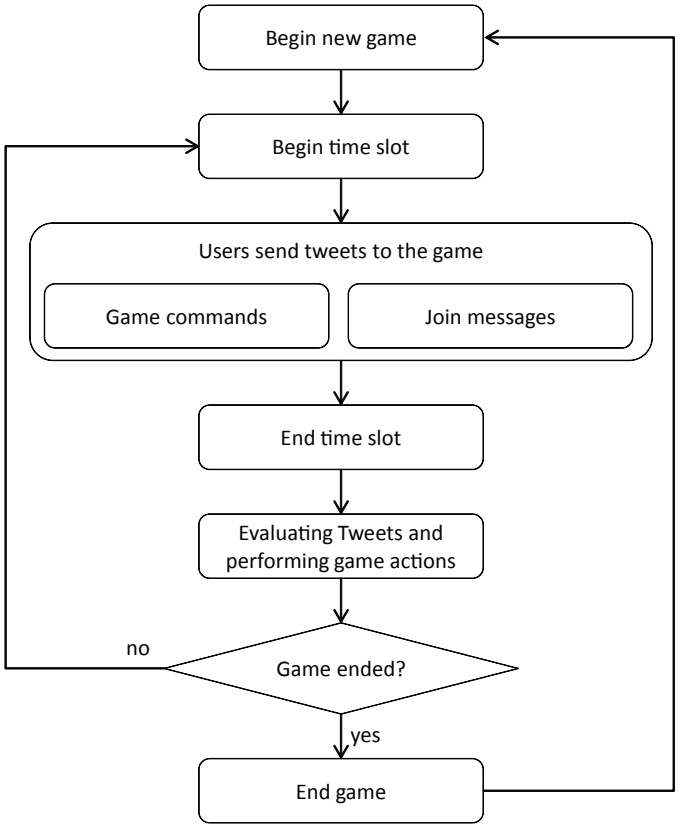


Fig. 1. Basic Principle of TweetGame-based Collaborative Games.

Twitter's *@reply* notation. A join message could for example look like this: *@samplegame join team_a*. More details about game commands will be described in section III. After joining a game, a user is able to send game commands in order to influence the game flow. Game commands again have to be directed to the Twitter account of the game using the *@reply* notation. As mentioned above TweetGames are round-based. The length of a round is adjustable and application specific. After each round the framework gathers all Tweets directed to the game's account during the last time slot. Thereupon these Tweets are analyzed and the resulting game actions are calculated and performed. How the collective game commands of a team are merged to one team action depends on the concrete implementation of each game. The sample application presented in this paper for example just calculates the mean value of all commands. But, of course more sophisticated mappings of user commands to team actions are realizable as well. For example it could be required that the members of a team did agree upon a game command in order to perform an action at all.

After each round, the framework checks whether the game is over or not. In case the game is over a new instance is started, otherwise the next round begins. The criterion for the decision when a game is over is application specific. The game developers can e.g. define a maximum number of rounds or any other criterion for the end of a game (e.g. one of the teams has reached a certain amount of points).

Having presented the basic idea of collaborative Twitter games the following sections presents the details of the TweetGame framework. Afterwards Section IV describes a sample application developed based on this framework.

## III. The TweetGame Framework

This section presents the TweetGame framework a framework developed for the realization of Twitter-based collaborative social online games as described above. This framework basically consists of two components: a) A *Java*-based application server that hosts the game and represents the back-end of the architecture. b) A *HTML 5* and *JavaScript* based front-end displaying the game state on a website. The framework is built entirely around Twitter in terms of using the microblogging service as central communication channel. Configuration commands as well as user commands are sent via Twitter. Moreover the communication between the application's back-end and front-end is based on Twitter messages.

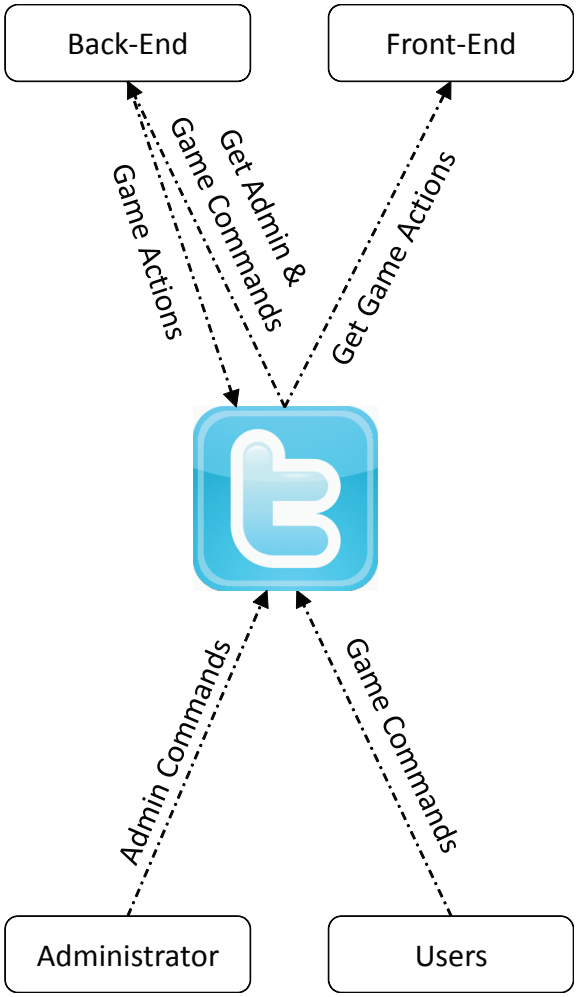


Fig. 2. Components of the TweetGame Framework.

Figure 2 shows the components of the TweetGame framework and how they interact. The basic components are the back-end that realizes the game logic and manages the game state, the front-end that visualizes the progress of the game as well as users and administrators that interact with the game via Twitter messages. Users participate in the game by sending game commands via Tweets. The back-end periodically (always at the end of a game round) fetches these Tweets and calculates the resulting game actions. These are thereupon published by sending a public Tweet describing the game action from the game's Twitter account. The front-end gathers these game action Tweets and visualizes the game actions.

One interesting feature of the framework is the fact, that administration and configuration of the game can be done via Twitter as well. For this purpose for each game one or multiple dedicated administrator Twitter accounts are registered. Administrators are able to configure a game by sending private messages containing administration commands from an administrator account to the game's Twitter account. The back-end gathers these Tweets and performs the corresponding configurations. This feature allows administrators to access and administrate the server easily every time and everywhere just by sending a private Twitter message. The security of this approach is given by the fact that private messages are not readable for a third party and Tweets are sent over a secure *https* channel. Since only registered Twitter accounts from administrators are authorized to send configuration messages, the security of the system is inherently given by the security level provided by Twitter. By this means our framework automatically profits from Twitter's security measurements.

An important feature of our framework is the fact that the application server hosting a game does not necessarily require a public IP address, because the entire communication with the application server is based on Twitter messages. The application server on the one hand gathers Tweets from administrators as well as users and on the other hand sends Tweets in order to publish the resulting game actions.

The details of the back-end as well as the front-end of the TweetGame framework will be highlighted in the subsequent paragraphs.

*a) Back-End:* The back-end of our architecture is responsible for managing the game state and for realizing the game logic. It has to fetch and evaluate the user commands and triggers the corresponding game actions. For this purpose our framework provides a Java-based application server that hosts TweetGames and provides the basic functionality for the development of the back-end component. While the implementation of the actual game logic obviously is part of each concrete game realization, the framework provides means for handling the Twitter messages, managing user groups, connecting to the game's front-end as well as application configuration. The application server framework basically provides the following components to application developers:

- **Application Manager:** The application manager is a container for the execution of TweetGames. The manager is responsible for hosting the applications (games) running on a server and provides facilities to execute, start, stop and pause the applications.
- **Applications** The application component represents one

dedicated Twitter game, it contains the game logic and manages the game state. Each TweetGame has a related Twitter account that is used for the entire communication with the game. Moreover one or multiple Twitter accounts are registered as administrator accounts. The administrator accounts can be used by application administrators to send private configuration commands to the game account in order to administrate the game. A TweetGame developer basically has to extend the `Application` base class in order to implement its own application. The base class already provides the fundamental functionality and common user commands. Game specific behavior and commands have to be added by the game developers. More details about game commands will be discussed below.

- **Groups** As described above groups and teams respectively are a central concept of the TweetGame framework. The idea behind the collaborative Twitter games is that users do not act as individual game entities but rather as groups whose game actions emerge from the collective behavior of the members. For this purpose the game framework already provides extensive group and user management facilities. The group component of the framework allows to create and remove groups as well as to handle registered users and their group memberships.

As mentioned above, the entire communication within our framework is based on Twitter. We distinguish the following types of commands that can be sent as Tweets: (a) game commands, (b) admin commands and (c) game actions. These commands will be described in detail in the following:

- **Game Commands:** Game commands are directed from the users towards the back-end in order to interact with the game. At the end of a game round the application server gathers these messages and calculates the resulting game actions. The framework provides two different game modes in order to support different game concepts. It is configurable whether game commands have to be sent via public Tweets or via private Twitter messages. This is important since some game concepts may require users to see the commands of their teammates while it may not be desired for other games. Since game commands are application specific each game developed on top of our framework has to define its own game commands along with the command syntax. A game command basically exists of a keyword specifying the game action along with an argument list of arbitrary length: `keyword argument_1 ... argument_n`. An example for a game command is the *throw* command defined by the sample application presented in Section IV. The framework supports the developer in the creation of new game commands by fetching the game command Tweets and by providing a parser that extracts the keyword as well as the argument list and calls a handler method that has been registered for a given command.
  While most game commands are application specific and hence have to be implemented by the application developer, the following two commands are common for all applications. For this reason these are already implemented by the framework:
  - `join team_name`: Joins a user to the given team in the game that receives this message. Of course the command is only executed if the given team exists in this game.
  - `chteam team_name`: Moves a user that has already been registered to a game into the given team. Note, that team memberships persist for at least one round in order to prevent cheating.
- **Admin Commands:** Admin commands are sent by a game administrator in order to configure the games running on a TweetGame application server. As already mentioned, admin commands are only accepted from registered administrator accounts. Since many configuration commands are required for all games the GameTweet framework already defines and implements the most common admin commands. Of course it is possible for each game to define additional game specific admin commands. Here two different kinds of administrator commands need to distinguished: a) commands dedicated to an application server in order to administrate the entire server and b) commands dedicated to a specific game running on an application server. In order to handle the application server specific administration Tweets an application server has its own Twitter account. The application server specific administration commands already implemented by the framework are the following:
  - `aistart game_name`: Starts a new instance of the game with the given name. Note that the application with the given name needs to be present on the application server to execute this command successfully.
  - `aipause game_name`: Pauses the game with the given name. This command just disables the game but does not delete the game instance hence the entire game state still exists. For this reason it is possible to continue the game later on using the following command.
  - `airesume game_name`: Resumes the given game if it has been paused before.
  - `airestart`: Restarts the instance of the game with the given name. This commands deletes the old game state and starts a whole new instance of the game.
  - `aistop game_name`: Stops the game with the given name. In contrast to the pause command this command deletes the game instance and with it the game state. Resuming a game is not possible after executing this command.

  Of course it is possible to extend the framework in order to add further application server admin commands. Additionally the following game specific admin commands are already implemented by the TweetGame framework:

- `aitslot duration`: Using this command it is possible to set the duration of a game round in the game that receives the message to the given value.
- `gradd group_name`: Creates a new group with the given name in the game with that receives the message.
- `grdel game_name group_name`: Deletes the group with the given name from the game that receives the message.

- **Game Actions:** At the end of a game round, all game commands of the users are fetched and the resulting game actions as well as game state changes are calculated. The resulting game actions are thereupon publish by sending a Tweet via the game's Twitter account. This Tweet contains the command corresponding to the game action along with an arbitrary number of command arguments. Obviously the game action commands are application specific and have to be implemented by game developers. The front-end needs to be able to parse these commands and to visualize them in an appropriate manner. While the front-end just visualizes the game actions the game state is managed solely by the back-end.

*b) Front-End:* In addition to the application server which represents the back-end of the TweetGame framework, a front-end visualizing the game state and the game actions is required. The main task of the front-end is to fetch the Tweets with the game commands and to visualize the progress of the game. While the front-end can be realized with arbitrary technologies, the visualization for the Gorillas game is based on HTML5 and JavaScript. As described above the back-end publishes game actions by sending public Tweets from the game's Twitter account. The front-end gathers these commands using a JavaScript and visualizes the game actions on a HTML5 Website.

Our framework provides a weak coupling between game logic and game visualization which has several advantages. At first it is easily possible to provide multiple visualizations for a game by just implementing different websites displaying the game. This allows for example TweetGame mashups by seamless integrating games into arbitrary websites. Moreover it is possible to create front-ends based on different technologies and for different devices as well as platforms. It is for example possible to realize a *Flash*- or a *Silverlight*-based game front-end. Mobile devices can be supported natively by implementing *Android* or *iOS* applications visualizing the game state. It would also be possible to integrate Twitter into the front-end application in order to improve the user experience by providing a integrated user interface. The game front-end could even hide the Twitter communication from the user by providing intuitive game interaction facilities and automatically generating and sending the corresponding Twitter messages. In summary the weak coupling between back-end and front-end provides extensive opportunities to customize the front-end as desired.

While the implementation of the front-end is for the most part application specific our framework provides JavaScript based facilities to connect to a Twitter account and to gather the game actions periodically. Handling and visualizing the game action appropriately is then part of each individual implementation.

## IV. Sample Application

In order to show the usability of the developed framework and to prove the viability of the concept of Twitter-based collaborative social online games, we have developed a sample application named *Gorillas*[4]. This game follows the model of the traditional Gorillas video game from 1991. Figure 3 show a screenshot of the game's website. The basic idea of the game is simple. Two teams each represented by a gorilla throw bananas onto each other in a round-based game. For this purpose two parameters need to be determined: The angle of the throw as well as the power of the throw. The Gorilla that first hits its opponent three times with the banana wins the game. While the traditional video game is played by two individual players, our TweetGame-based version of the game features two competing teams. The angle and the power of the actual throws is determined by the mean value of all individual estimations.

Users that have registered to one of the teams by sending a *join* command are able to participate in the game by sending *throw* messages to the game's Twitter account as follows: *@PlayGorillas throw power_value angle_value*, while *power_value* and *angle_value* are the parameters for the power and the angle of the throw chosen by the user. At the end of a time slot the framework fetches all Tweets sent during the last round and calculates the resulting game actions. The game actions are thereupon published by sending a Twitter message from the game's Twitter account similar to the following one: *Kiki on the right throws banana with power: 209 and with angle: 49*. The website-based front-end gathers these message and displays the throws on the website. A game is over as soon as one of the teams has three strikes.

## V. Scientific Application

This section discusses the usability of the TweetGame framework in the field of complex social network science. As mentioned above, the intention of our concept is not just to provide a novel kind of online games, the framework can rather be utilized by complex social network researchers in order to study collective phenomenas and social dynamics. To utilize our framework for this purpose it supports two different game modes: a) Game commands are send as public Tweets which enables users to see the game commands of their teammates. b) Game commands are sent as private messages and are not visible for teammates. The availability of these two modes does not only allow different game scenarios, it is especially important for the execution of certain empirical studies, because it allows to execute the same experiments once with mutual awareness of the teammates actions and

[4]http://mocca.uni.lu/gorillas

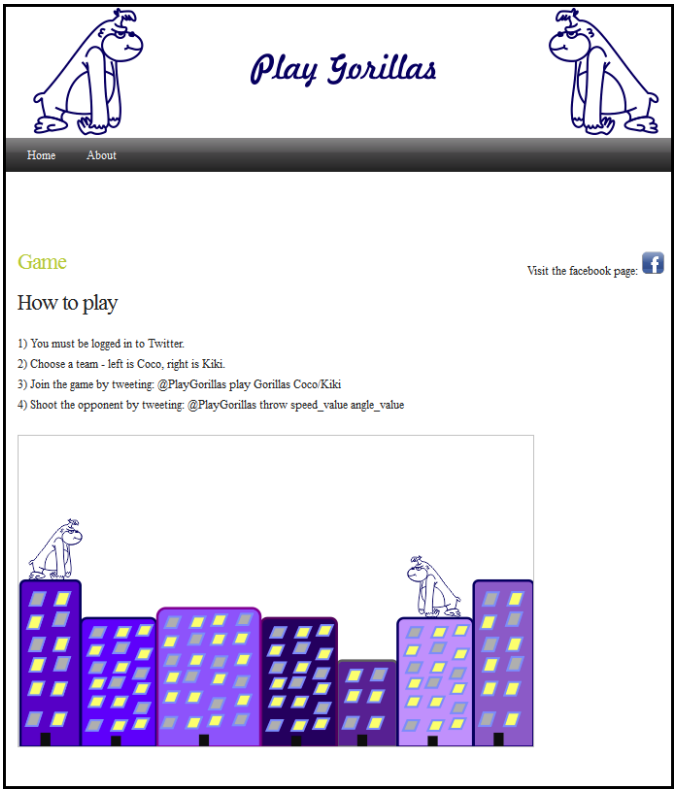


Fig. 3. Screenshot of the Gorillas sample game.

once without in order to compare the results and to measure the impact of social influence. This is for example interesting to study the wisdom of crowd effect [7], [21] as it has been done in [14]. The wisdom of crowds effect refers to the phenomenon that the average estimation of a group often is more accurate than the estimations of individuals. But in [14] it has been shown, that social influence in terms of knowing the guesses of the other group members, reduces the accuracy significantly. The reason is that social influence causes humans to adjust their opinion to those of others [3], [4], [17], [2]. This in turn influences the statistical aggregate and has a negative effect on the wisdom of the crowd. Utilizing the two modes provided by the TweetGame framework it is possible to accomplish such experiments on a large scale. It is in fact our intention to reproduce this experiment based on the Gorillas game presented in the previous section. While this is just one example for the applicability of our framework it is possible to utilize TweetGames for numerous studies in this field (e.g.: [24], [20], [9], [8]). The huge user base of a popular service like Twitter provides the opportunity to conduct such studies on a large scale with a meaningful number of participants.

## VI. Related Work

Gaming in general is a very popular application provided by social network platforms. Especially Facebook provides numerous online games from various game genres like *Online Role Playing Games*, *Strategy Games*, *Simulations* and so on. Though these games are played on a social network platform and involve user interaction these can not be compared to our approach since these games for the most part just realize traditional online game principles and do not consider the collective behavior of user groups.

Closer to our approach are games that have been developed based on the microblogging service Twitter. These games also utilize the idea of interacting with the game via Twitter messages, but also feature a traditional online game scheme where users act as individuals on their own behalf. Most of these games are quiz games where users have to answer questions via Tweets, examples are *BeatMyTweet*[5], *@Libs*[6], *Twitbrain*[7], *twivia*[8] or *WhoseTweet*[9]. Other populare Twitter-based games are *Tweetbomb*[10], *Artwiculate*[11] or *Tweetfight*[12].

While the idea behind our approach is to utilize online games in order to study the collective behavior and social dynamics of the users, there are other approaches that utilize online games in order to have certain task solved by the collective effort of the game users. One popular example is described in [13]. Here a online game has been utilized in order to find the optimal molecular structure of a certain enzyme, a problem that is hard to solve by computers. Another example is the *ESP Game*, a collaborative game that is used to label images with keywords.

## VII. Conclusion

This paper presented the concept of collaborative social online games. This idea has been realized as proof of concpet on top of the microblogging service Twitter. The basic idea is to provide round-based online games where users can participate just by sending Twitter messages containing game commands. The essential difference to traditional online games is the fact that users do not act as individual game entities, rather the game events are controlled by the emergent behavior of all users belonging to the same team. In addition to introducing this game concept the paper presents the TweetGame framework, a Java-based framework for the development of such collaborative online games. The framework offers the basic functionality required for the development of TweetGames, like providing an application container to host the games, fetching and processing Tweets, managing Groups and so on. Utilizing this framework TweetGame developers can focus on the implementation of the actual game logic. Based on this framework we developed a sample application, the so-called Gorillas game as proof of concept.

In addition to defining an interesting new kind of applications for microblogging services and social network platforms our concept allows researchers in the field of complex social network to study emergent behavior and social dynamics in

[5]http://www.officeevil.com/beat-my-tweet/
[6]http://atlibs.com/
[7]http://ajaxorized.com/twitbrain/
[8]http://www.timdorr.com/twivia/
[9]http://www.whosetweet.com/
[10]http://tweetbomb.wordpress.com/
[11]http://artwiculate.com/
[12]http://www.tweefight.com/

large social networks. The paper also described a Tweet-based application configuration mechanism. This approach is not limited to our scenario but rather may be applicable in other scenarios as well in order to provide a convenient way of Twitter-based administration.

As future work it is intended to perform experiments on collective user behavior and social dynamics based on the TweetGame framework. As a first step it is planed to reproduce the results from the studies about the social influence on the wisdom of crowd effect presented in [14]. As described above, our framework provides all required means for the execution of such experiments. While the original study has been done with a rather small group of just 144 test persons the large user base of the Twitter network provides the opportunity to reproduce the study on a larger scale in order to provide more meaningful results.

## REFERENCES


[1] S. D. andf Peyton Young. *Social dynamics*. Brookings Institution Press MIT Press, Washington, D.C. Cambridge, Mass, 2001.

[2] S. E. Asch. Opinions and social pressure. *Scientific American*, 193:31–35, 1955.

[3] A. V. Banerjee. A Simple Model of Herd Behavior. *Quarterly Journal of Economics*, 107(3):797–817, 1992.

[4] S. Bikhchandani, D. Hirshleifer, and I. Welch. A Theory of Fads, Fashion, Custom, and Cultural Change as Informational Cascades. *The Journal of Political Economy*, 100(5):992–1026, 1992.

[5] C. Castellano, S. Fortunato, and V. Loreto. Statistical physics of social dynamics. *Reviews of Modern Physics*, 81(2):591, 2009.

[6] V. Darley. Emergent Phenomena and Complexity. *Physical Review*, 1994.

[7] F. Galton. Vox populi. *Nature*, 75(1949):7, 1907.

[8] R. L. Goldstone and T. M. Gureckis. Collective Behavior. *Topics in Cognitive Science*, 1(3):412–438, July 2009.

[9] B. Golub and M. O. Jackson. Naïve Learning in Social Networks and the Wisdom of Crowds. *American Economic Journal: Microeconomics*, 2(1):112–149, Feb. 2010.

[10] J. Holyst, K. Kacperski, and F. Schweitzer. Social impact models of opinion dynamics. In *In: Annual Reviews of Computational Physics*, volume 48, pages 253–273. Citeseer, 2001.

[11] I. Kanovsky and O. Yaary. Model of Opinion Spreading in Social Networks. June 2011.

[12] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 137–146, New York, NY, USA, 2003. ACM.

[13] F. Khatib, F. DiMaio, S. Cooper, M. Kazmierczyk, M. Gilski, S. Krzywda, H. Zabranska, I. Pichova, J. Thompson, Z. Popović, M. Jaskolski, and D. Baker. Crystal structure of a monomeric retroviral protease solved by protein folding game players. *Nature Structural & Molecular Biology*, advance online publication, Sept. 2011.

[14] J. Lorenz, H. Rauhut, F. Schweitzer, and D. Helbing. How social influence can undermine the wisdom of crowd effect. *Proceedings of the National Academy of Sciences*, May 2011.

[15] Q. Lu. PROPAGATION , CASCADES , AND AGREEMENT DYNAMICS IN COMPLEX COMMUNICATION AND. *Social Networks*, 2009(November 2009), 2009.

[16] M. W. Macy and R. Willer. From Factors to Actors: Computational Sociology and Agent-Based Modeling. *Annual Review of Sociology*, 28(1):143–166, Aug. 2002.

[17] A. E. Mannes. Are We Wise About the Wisdom of Crowds? The Use of Group Judgments in Belief Revision. *Management Science*, 55(8):1267–1279, 2009.

[18] Z. Néda, E. Ravasz, T. Vicsek, Y. Brechet, and a. L. Barabási. Physics of the rhythmic applause. *Physical review. E, Statistical physics, plasmas, fluids, and related interdisciplinary topics*, 61(6 Pt B):6987–92, June 2000.

[19] W. Ren and E. M. Atkins. A survey of consensus problems in multi-agent coordination. *Proceedings of the 2005 American Control Conference 2005*, 3(5):1859–1864, 2005.

[20] M. J. Salganik, P. S. Dodds, and D. J. Watts. Experimental Study of Inequality and Cultural Market. *Science*, 311(February):854–856, 2006.

[21] J. Surowiecki. *The Wisdom of Crowds: Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies and Nations*. Doubleday, May 2004.

[22] T. Vicsek, A. Czirok, E. Ben-Jacob, I. Cohen, and O. Shochet. Novel type of phase-transition in a system of self-driven particles. *Physical Review Letters*, 75:1226–1229, 1995.

[23] T. Vicsek and A. Zafiris. Collective motion. *Arxiv preprint arXiv10105017*, 19(8):13, 2010.

[24] I. Yaniv and M. Milyavsky. Using advice from multiple sources to revise and improve judgmentsâ. *Organizational Behavior and Human Decision Processes*, 103(1):104–120, May 2007.