

# A Stigmergy Approach for Open Source Software Developer Community Simulation

Xiaohui Cui, Justin Beaver, Jim Treadwell and  
Thomas Potok  
Oak Ridge National Laboratory  
Oak Ridge, TN 37831

Laura Pullum  
Lockheed Martin Corporation  
St. Paul, MN 55164-0525

**Abstract**—The stigmergy collaboration approach provides a hypothesized explanation about how online groups work together. In this research, we presented a stigmergy approach for building an agent based open source software (OSS) developer community collaboration simulation. We used group of actors who collaborate on OSS projects as our frame of reference and investigated how the choices actors make in contribution their work on the projects determinate the global status of the whole OSS projects. In our simulation, the forum posts and project codes served as the digital pheromone and the modified Pierre-Paul Grasse pheromone model is used for computing developer agent behaviors selection probability.

*Keywords-component; OSS, Pheromone, Agent, Model*

## I. INTRODUCTION

In most online collaborative groups, the online environment provides a shared medium for storing information so that it can be interpreted by other individuals later. This stored information can be represented as stigmergy signals, serving as long-term memory for the group. The stigmergy collaboration approach provides a hypothesized explanation about how these online groups work together. The open source software (OSS) developer community is a new kind of online software development group where participants can read, modify, and redistribute software source code without cost. Given the significant amount of detailed information available on their internal structure, OSS development communities offer a unique opportunity to understand the large self-organized entity. In this paper, we present a stigmergy approach [1] for examining how the behavior of individuals affects the emergence of an OSS project's global status.

## II. STIGMERGY COLLABORATION IN OSS COMMUNITIES

The stigmergy term was first proposed by Pierre-Paul Grasse in the 1950s in conjunction with his research on termites [2]. Grasse showed that a particular configuration of a termite colony's environment can triggered a termite to modify its environment (drop a mud in a particular place for building or maintaining the nest). The modification in turn stimulates the original or other termites in colony to further transform its environment. Grasse made a general definition of stigmergy as: "the stimulation of the workers by the very

performances they have achieved". The stigmergy process has been observed in termites, ants, bees, and wasps in a wide range of activities.

In termite colony, a highly complex termite nest is not caused by the net building knowledge of individual termite. It is just a collective behavior result from large numbers of individual termites performing extraordinarily simple actions in response to their local environment. There are no direct communications between termite workers for coordinating their nest building actions. The modified environment caused by termite simple actions served as the coordinate signals. The state of the nest structure triggers some behaviors, which then modify the nest structure and trigger new behaviors until the construction is over.

The concept of stigmergy provides a theory for explaining how disparate, distributed, ad hoc contributions from individuals could lead to the emergence of large collaborative enterprises. While people are more intelligent than social insects, individuals in open software development groups use essentially the same stigmergic mechanism for collaborating. Mockus et al. [3] point out, OSS developers rarely if ever meet face-to-face or even via telephone. Participants in OSS projects mainly engage in online discussion forums or threaded email messages as a central way to observe, participate in, and contribute to public discussions of topics of interest to ongoing project participants [4]. The newly developed software source codes are uploaded to the community website for being scrutinized by the member of the community. Any bug, error or lacking functionality will be point out and entice community member to take up the short come. These communication messages functions as stigmergy signals, albeit a much more dynamic one than the mud used by termites. The stigmergy signal can served as a long-term memory for the group. It can be picked up by any individual at any time.

## III. RELATED WORKS

In most online hosting environments, the project related actions are logged and the log information can later be mined to understand the community structure and interaction patterns. This log data provided enormous detail information for analyzing [5, 6]. Crowston, et al. [7] proposed a model for effective work practices in OSS development. The model was based largely on an existing model of group

effectiveness initially proposed by Hackman [8] in 1986. Smith, et al. [9] presented an agent-based OSS simulation model that includes software modules complexity, the software's fitness for purpose, the motivation of developers, and the role of users in designing requirements. In the researches of how the OSS developers collaborate, researches consider the OSS movement as a self-organizing system and a collaborative social network [10]. Social network analysis was used for analyzing OSS community structure. Actor relationships are represented as nodes and links. The actor can be user or developer. Every node  $i$  represents an actor within network;  $\text{link}(i,j)$  denotes social tie between actors  $i$  and  $j$ . However, in OSS community, the direct connection between actors is unusual. They exchange the information through the forum or email-list indirectly. Most of time, the actor even don't know who is his message receiver before he send out his message.

Elliott [1] argued that collaboration in small groups (roughly 2-25) relies upon social negotiation to evolve and guide its process and creative output. Collaboration in large groups (roughly 25-n) is enabled by stigmergy. Heylighen [11] proposed to distinguish the stigmergy in OSS community as direct and indirect. In OSS development, the unfinished jobs, served as the direct stigmergy, which stimulate other actors to come to finish the jobs. Indirect stigmergy can be recognized in forums where bugs or function requests are posted. These forums are regularly consulted by the developers, thus attracting their attention to tasks that seem worth performing.

#### IV. TECHNICAL APPROACH

Our approach to reproducing the complex environment of OSS software development community was to develop an agent-based stigmergy collaboration model. The model will represent how the OSS community collaboration and how each individual developer's chose of what software element to develop and how many effort to contribute will impact the status of the whole OSS project. Our hypothesis is the collaborations of individual OSS developers and users are stigmergy collaborations. The forum posts, email list and unfinished source codes serve as the digital stigmergy. The peer-to-peer communications between individual OSS members are not occurring very often. The measures used in this research for representing OSS project status are listed in Table 1.

To simplifier the simulation, we assume there are only two kinds of agents in the simulation, the developer and the user. User agent can change to developer agent if they want to. The agents do not interact with each other directly. Instead, they go through the forums for information exchange. There are two kinds of forums, the public forum and the developer forum. The public forum can be access and post message by any agents who are interesting in the software project. Most of time, it served as the message exchange board between users and developers. Users can post questions about how to use the software, bugs they found during the software using and functions they wish to be included in the software. Each problem will be represented by one forum thread. The other users and the

developers occasionally go through the forums, answer the questions and get the first hand information about the bug problems and the wish list functions.

TABLE I. OSS COMMUNITY STATUS MEASURES

| OSS Community Status Metric | Font size and style  |
|-----------------------------|--|
| Number of Developers        | A count of the group's core membership   |
| Number of Software Releases | A count of the number of orchestrated actions that the group has performed               |
| Number of Downloads         | Measures the degree to which the group's actions are found to be useful in the community |

In this simulation, the developers use a modified Ant Colony Algorithm model to choose which forum thread problem he/she will contribute to solve. In this algorithm, each forum thread serves as one potential digital trail to different software development directions and the post messages in this thread represent the digital pheromone laid down. Every time, when user or developer posts a new message in this forum thread, a new pheromone is deposit on the trail. The pheromone content of a forum thread can be updated and decayed.

Pheromone update: when a message is posted in a forum thread, the pheromone for this thread is incremented by a constant,  $\gamma$ . The nominal value of  $\gamma$  is one. Equation (1) describes the pheromone update procedure when a message is posed by actor  $a$  in a post thread  $d$  at time  $t$ .

$$P_d^{t+1} = P_d^t + \gamma \quad (1)$$

Pheromone decay: to account for pheromone decay, each thread pheromone values are periodically multiplied by the decay factor,  $\epsilon$ . The decay rate is  $\tau > 0$ . A high decay rate will quickly reduce the amount of remaining pheromone, while a low pheromone decay rate will degrade the pheromone value slowly. The nominal pheromone decay interval is one day; we call it decay period. Equation (2) describes pheromone decay.

$$P_d^{t+1} = P_d^t * \epsilon^{-\tau} \quad (2)$$

If no message has been posted in a thread in quite some time, the pheromone for this a thread will be decay to a near zero value. The thread will be removed from the developer's potential thread select direction. According to the pheromone theory, forum users will most likely join and post message in the thread that has highest pheromone content.

Thread selection: Agents will randomly chose a thread based on the amount of pheromone present on each forum

thread. The equation (3) describes the thread  $d$ 's probability  $\rho_d$  being chosen.

$$\rho_d = \frac{(P_d^t + K)^F}{\sum_{i=1}^N (P_i^t + K)^F} \quad (3)$$

$N$  is the total number of forum threads. The constants  $F$  and  $K$  are used to tune the behavior of forum users. The value of  $K$  determines the sensitivity of the probability calculations of small amounts of pheromone. If  $K$  is large, then large amounts of pheromone will have to be present before an appreciable effect will be seen in the message posting probability. The nominal value of  $K$  is zero. Similarly,  $F$  may be used to modulate the differences between pheromone amounts. For example,  $F > 1$  will accentuate differences between links, while  $F < 1$  will deemphasize them.  $F = 1$  yields a simple normalization. The nominal value of  $F$  is two.

Another forum in the simulation is developer forum. It serves as the internal forum and used by developers post their ongoing works. It represents the email-list and SVN repository in real OSS project. Each developer's contribution is stimulated by other developers' ongoing works posted in the developer forum. The probability  $c$  for developer  $i$  continually contribute on a software element developing is modeled as termite mud drop probability and is given by:

$$c_i = n_i * \left( \frac{f(i)}{k + f(i)} \right)^2, \quad (4)$$

$$f(i) = \max(0.0, \frac{1}{\sigma^2} \sum_{j \in L} (1 - \frac{\delta(i, j)}{\alpha}))$$

Here,  $\delta(i, j) \in [0, 1]$  is the dissimilarity value between developer  $i$  contributed post and the developer  $j$  contributed post message in email-list and SVN repository,  $n_i$  is the message length.  $\alpha \in [0, 1]$  is a data-dependent scaling parameter, and  $\sigma^2$  is the total number of post messages in developer forum in pre-defined time period  $L \in [1, 15]$ .

The equation (3) and equation (4) decide the behavior of agents in the simulation. These individual's behaviors can change the OSS project status. In our simulation, one of the measures for the status of the OSS project, the software utility (number of downloads), is modeled by the equation (5).

$$\frac{dU}{dt}$$

represents the number of the software been downloaded each day.

$$\frac{dU}{dt} = \theta * \sum_{i=1}^N \frac{u_i * p_i}{Z} \quad (5)$$

$d$  is constant.  $u_i$  is the number of users who post messages in forum thread  $i$ .  $p_i$  is the number of developers who post messages in forum thread  $i$ , and  $Z$  is the total number of users and developers in the forum.

The membership measure (number of developers), is

$$\frac{dP}{dt}$$

modeled by the equation (6).  $\frac{dP}{dt}$  represented the increasing rate of the number of the developers.

$$\frac{dP}{dt} = \alpha * \sum_{i=1}^N \frac{u_i}{Z * p_i} \quad (6)$$

$\alpha$  is constant. Other parameters are same as equation (5).

## V. EXPERIMENTS

We used detailed OSS community log data on SourceForge to illustrate and validate the model's theoretical mechanisms. SourceForge, an online center for OSS development communities, provides collaborative resources for approximately 200,000 projects. This data consisted of all the activity information of OSS software developers and users that registered on SourceForge from 2003 to 2008. We developed scripts that query the SourceForge Research Data Archive hosted by the University of Notre Dame [6], for project data that meet our criteria. We were interested in OSS projects that reached a minimum team size of 20 developers at some point in the project lifetime in order to insure that the social and technical factors were well represented. In addition, we eliminated projects that did not appear to use the SourceForge collaboration tools as a significant means for communication and coordination. In all, we identified 67 projects as viable for use as training data for our OSS model. From these data, we can reconstruct how the local behavior of setting up the project teams that created individuals led to the emergence of the project global status measures.

### A. Determining Model's Accuracy of Fit to Empirical Data

We first trained the proposed agent-based model with the data collected from the log files of OSS project web site and then compare the closeness of the simulated results to the empirical data. In this exercise, the model was tuned with actual OSS project data for 9 time slices (one time slice = one month), and then simulated for 1, 3, 6, 9, and 12 additional time slices. At each interval, an assessment of the "Accuracy of Fit" is recorded. The Accuracy of Fit measure is a quantitative determination of how well a model represents the underlying data. It is a measure that indicates the correctness of the model with respect to the data that was

used to construct the model. The Accuracy of Fit is determined through an analysis of the Equality of Means and the Equality of Variances between the modeled values of the OSS project measures and the associated actual OSS project measures values.

The Equality of Means Hypothesis Test [17] quantifies the confidence that the mean values of two given datasets are equivalent. The equation is shown in Figure 1. By comparing the Equality of Means between actual open source software community status values and the status value generated by OSS community simulation, the ability of the OSS model to accurately characterize the underlying data set is revealed. If the Test Statistic for the given quality measure is less than the critical value for that measure, then the null hypothesis must be accepted, the means are determined to be equivalent, and the model is said to provide an accurate fit for the underlying data. For this study, a confidence of =0.9, or 90% was used for all Equality of Means calculations. Thus, there is 90% confidence that all Equality of Means determinations are correct.

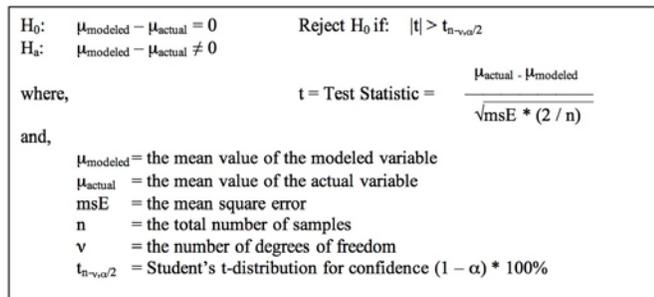


Figure 1. Hypothesis Test for Equality of Means.

The approach to calculating the Equality of Variances is to use a textbook rule of thumb test recommended in [18], where in comparing the modeled data to the actual data, if the ratio of the maximum variance value to the minimum variance value must be less than three to consider the variances equivalent. The application of this rule of thumb is appropriate in that it bounds the relationship of the variances. The goal for the Equality of Variances is not to get a quantifiable confidence on the accuracy of the modeled data (which is already accomplished through the Equality of Means test), but to get a discrete indication that the variance of the modeled data is on the order of the variance of the actual data.

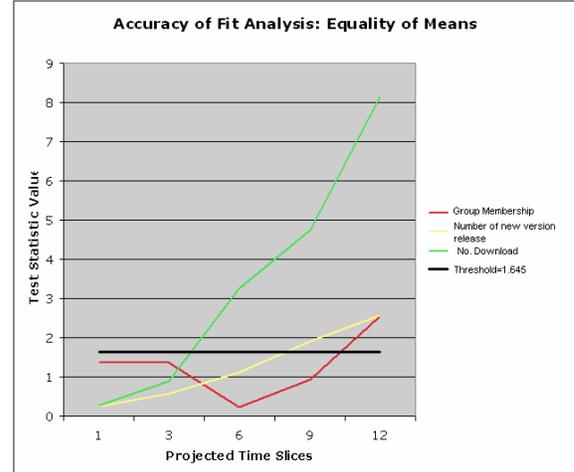


Figure 2. Equality of Means Validation.

## B. Results

The results of applying the Accuracy of Fit statistical tests, Equality of Means, and Equality of Variances are shown in Figure 2 and Figure 3. The Equality of Means analysis, shown in Figure 2, highlights the statistical threshold for this test as a black line. The interpretation of this graph is that those data points below the threshold line indicate that the simulation was able to maintain a mean value consistent with the underlying data for those metrics. Similarly, data points above the threshold indicate that the simulation diverged from the distribution upon which the simulation is based. Figure 2 shows that the mean values of all three OSS community status measures were consistent with their actual mean values for up to three time slices (three months). In the case of Group Membership, the mean value was consistent for nine simulated time slices. Thus the model performed related well in remaining consistent with the underlying mean values of the data for short-term simulation, but became less consistent as the simulation progressed.

The results of the Equality of Variances test, shown in Figure 3, are similar to the Equality of Means in their interpretation. Values above the threshold line indicate points of unequal variance between actual and simulated data, and values below the threshold indicate points of consistency across the variances. The model did well at simulating the variances for modeled values consistent with the underlying OSS data.

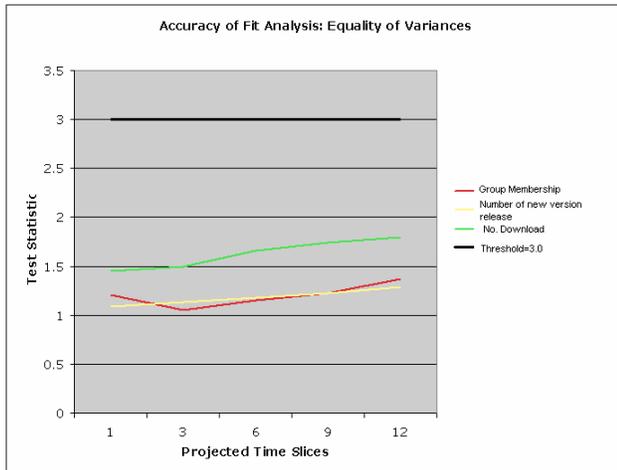


Figure 3. Equality of Variances Validation.

## VI. CONCLUSION

We used group of agents who collaborate on projects through forums as our frame of reference and investigated how the choices agents make in contribution their work on the projects determinate the global status of the whole OSS projects. Our hypothesis is that the collaborations of individual OSS developers and users are stigmergy collaborations. We proposed a stigmergy collaboration OSS model to produce a simulation that accurately represents the collaboration in an OSS community. We compared the simulated output to empirically observed behaviors in different OSS project forums. The simulation is able to partially reproduce the forum evolution trend in many OSS projects. The closeness of the simulated results to the empirical data indicates that our model may reflect the processes that occur in OSS evolution. Our next step will be extending the research model to other self organized communities. Our hypothesis is if two systems obey the same mathematical laws, we can perform experiments on one system and infer how another system might behave under similar conditions.

## ACKNOWLEDGMENT

Prepared by Oak Ridge National Laboratory, P.O. Box 2008, Oak Ridge, Tennessee 37831-6285, managed by UT-Battelle, LLC, for the U.S. Department of Energy under contract DE-AC05-00OR22725; and prepared by Lockheed Martin Company. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Oak Ridge National Laboratory, Lockheed Martin Company, the Department of Energy or the U.S. government.

## REFERENCES

[1] Elliott, M., Stigmergy Collaboration: The Evolution of Group Work. *M/C Journal*, 2006. 9(2).

- [2] Dorigo, M., E. Bonabeau, and G. Theraulaz, Ant algorithms and stigmergy. *Future Generation Computer Systems*, 2000. 16(8): p. 851-871.
- [3] Audris Mockus, R.T.F., James D. Herbsleb, Two case studies of open source software development: Apache and Mozilla. *ACM Trans. Softw. Eng. Methodol.*, 2002. 11(3): p. 37.
- [4] Yutaka Yamauchi, M.Y., Takeshi Shinohara, Toru Ishida, Collaboration with Lean Media: how open-source software succeeds. *CSCW*, 2000: p. 9.
- [5] Xu, J., Y. Gao, and G. Madey. A Docking Experiment: Swarm and Repast for Social Network Modeling. in *Seventh Annual Swarm Researchers Meeting (Swarm2003)*. 2003. Notre Dame, IN.
- [6] Christley, S. and G. Madey, Collection of Activity Data for SourceForge Projects. 2005, Dept. of Computer Science and Engineering, University of Notre Dame: Notre Dame, IN.
- [7] K. Crowston, H.A., J. Howison, and C. Masango, Towards a Portfolio of FLOSS Project Success Measures, in *The 4th Workshop on Open Source Software Engineering, International Conference on Software Engineering (ICSE) 2004*.
- [8] Hackman, J.R., The design of work teams. *The Handbook of Organizational Behavior*, ed. J.W. Lorsch. 1986, Englewood Cliffs, NJ: Prentice-Hall.
- [9] Neil Smith, A.C., Juan Fernández-Ramil, sers and Developers: An Agent-Based Simulation of Open Source Software Evolution, in *SPW/ProSim 2006*. 2006.
- [10] Jin Xu, Y.G., Scott Christley, Gregory R. Madey, A Topological Analysis of the Open Souce Software Development Community in *HICSS 2005*. 2005.
- [11] 11. heyligen, F., ed. Why is Open Access Development so Successful? *Open Source Jahrbuch*, ed. M.B.R.A.G. B. Lutterbeck. 2007, Lehmanns Media.
- [12] Dorigo, M., Ant colony optimization and swarm intelligence 4th International Workshop, ANTS 2004, Brussels, Belgium, September 5-8, 2004 : proceedings. *Lecture notes in computer science*. 2004, Berlin: Springer.
- [13] Bankes, S., Exploratory Modeling for Policy Analysis. *Operations Research*, 1993. 41(3): p. 435-449.
- [14] Sargent, R.G. Verification and Validation of Simulation Models. in *Proc. 2003 of Winter Simulation Conference*. 2003. New Orleans, Louisiana.
- [15] Macal, C.M. and M.J. North, Validation of an Agent-Based Model of Deregulated Electric Power Markets, in *North American Association for Computational and Social Organization (NAACSOS) Conference*. 2005: Notre Dame, Indiana.
- [16] Axtell, R., et al., Aligning simulation models: a case study and results. *Computational and Mathematical Organization Theory*, 1995. 1(2): p. 18.
- [17] Sincich, W.M.a.T., *Statistics for Engineers and the Sciences*. 1995, Upper Saddle Ridge, NJ: Prentice-Hall.
- [18] Voss, A.M.D.a.D.T., *Design and Analysis of Experiments*. 1999, New York, NY: Springer-Verlag New York, Inc.