Social Volunteer Computing

Adam MCMAHON and Victor MILENKOVIC
Department of Computer Science, University of Miami
Coral Gables, FL, 33146, USA

ABSTRACT

While both volunteer computing and social networks have proved successful, the merging of these two models is a new field: Social Volunteer Computing. A Social Volunteer Computing system utilizes the relationships within a social network to determine how computational resources flow towards tasks that need to be completed, and the results of these computations are added back into the social network as content. Such a system will provide scientists and artists a new facility to obtain computational resources and disseminate their work. RenderWeb 2.0, a prototype Social Volunteer Computing system, is introduced that allows animations created in Blender to be distributed and rendered within Facebook.

Keywords: Social Networks, Volunteer Computing, Graphics

INTRODUCTION

While both volunteer computing and social networks have already proved to be successful, the merging of these two models is a new field. We call this new field Social Volunteer Computing (SVC).

This paper is divided into two major sections. In the first section, we outline the proposed benefits of the SVC model. In the second section, we apply this model to rendering and introduce RenderWeb 2.0, which is our prototype rendering system that is integrated into Facebook.

This paper represents an initial step into the field of SVC. We will present the model, introduce a prototype, and discuss future directions. As SVC communities begin to emerge, our future work will focus on applying the SVC model towards new artistic and scientific projects, along with testing the proposed benefits of SVC model against other computational and social systems.

1. SOCIAL VOLUNTEER COMPUTING

Before we discuss the proposed benefits of Social Volunteer Computing, we will first briefly review traditional volunteer computing.

1.1 Volunteer Computing

The term volunteer computing was coined in 1996 by Luis Saramenta, who defines it as a form of distributed computing that allows "high-performance parallel computing networks to be formed easily, quickly, and inexpensively by enabling ordinary Internet users to share their computers' idle processing power without needing expert help" [1].

One of the first successes of volunteer computing occurred when a combined effort of 700 volunteer computers discovered the 35th Mersenne prime number [2]. The most popular volunteer computing system, SETI@home, is an attempt to search the skies for intelligent life [3]. SETI@home is now part of the larger BOINC project, which is a framework that joins together multiple research projects and allows volunteers to select among those projects [4]. BOINC currently supports over 30 scientific research projects and has approximately 300,000 active volunteers. In short, volunteer computing is a viable option for harnessing computational power for the sciences and arts.

Within a volunteer computing system, there are four roles that interact with each other:

a) Volunteers – volunteer computer's unused cycles.
b) Submitters – submit tasks to be computed.
c) Developers – develop the code that is executed on the volunteers' computers
d) Facilitators – create the framework that connects developers, submitters, and volunteers.

Using BOINC as an example, the facilitators are those who provide the BOINC framework. The developers then take that framework and apply it to a particular domain (e.g. protein folding). The submitters, who are typically part of the same research project as the developers, formulate meaningful tasks that need to be computed. Finally, the volunteers download the project's modules and provide their own computational resources.

1.2 Progress Thru Processors

In 2009, Intel announced a new project in conjunction with BOINC called Progress Thru Processors (PTP), which has the goal to join BOINC with Facebook [5]. PTP uses Facebook as a portal to download the BOINC software and uses Facebook to display volunteer statistics within in a Facebook application.

Though PTP uses Facebook in a variety of ways, it does not truly integrate social networks with volunteer computing. First, PTP does not actually perform computations directly within Facebook, but instead requires the volunteer to have BOINC separately installed as a desktop application. Second, the relationships within the social network do not direct the volunteers' resources, but the developers control the queue. Third, the results of the computation are not available to the social community but are only available to the developers and submitters of the projects.

While PTP demonstrates an important step towards merging volunteer computing and social networks, the integration is

primarily cosmetic: displaying user statistics and providing links to manually download the BOINC framework.

1.3 Overview of Social Volunteer Computing

Social Volunteer Computing (SVC) is a proposed new form of volunteer computing that is integrated within a social network and brings together volunteers, submitters, developers and facilitators. Unlike previous models, SVC begins to blur the lines between volunteers and submitters, such that any individual or party can enter into these two roles. Moreover, the relationships between the roles of volunteers and submitters are analogous to friends belonging to a group within a social network. Within the context of a social network application, a SVC system utilizes these relationships to determine how computational resources flow towards tasks that need to be completed. We call this Socially-Driven Computation. The result of this computation, which we call Socially-Computed Content, is added back into the social network. This content can be tagged and shared with other members of a social network.

1.4 Non-Social Vs. Social Volunteer Computing

To make the above definition more clear, we will compare a 3D rendering system that uses traditional volunteer computing to one that utilizes SVC.

In a traditional volunteer computing system, a group which wants an animation to be rendered first needs to recruit volunteers. The volunteers then download the project's application and donate their computer time. In this model, there is a clear distinction between the submitters (those who want the animation rendered) and the volunteers [6]. Moreover, there is little room for organic growth of computation or content, because the control flows down to the volunteers from the facilitators, submitters, and developers. Finally, the volunteers are blind volunteers. They do not control their own queue or choose which projects to render.

Now, let us view the same example from the perspective of a SVC system. In such a system, participants join a Facebook application that is connected to a SVC system. By joining the Facebook application, participants become part of the community and can act as volunteers and/or submitters. As volunteers, they can direct their computational resources towards particular scientists/artists by becoming their friends. As submitters, they can upload their own projects to be rendered by the community. After the rendering is complete, the animation becomes socially-computed content. The entire process is seamlessly woven within the social network and participants do not need to manually download or install any applications. Moreover, there is no gatekeeper who controls the flow of computation, but computational tasks are naturally allocated among the relationships that exist within the social network.

1.5 The Proposed Benefits of a SVC System

We propose that there are four benefits to a SVC system over non-social volunteer computing systems. This section outlines these proposed benefits:

1) *Social Network Integration* – The user experience of a SVC system is completely integrated within a social network. In this sense, users can upload tasks and volunteer their own computer from within the social network. Moreover, this integration provides new methods (such as wall posts or status feeds) of informing users of tasks that need computational resources.

2) *Socially-Driven Computation* – In a SVC system, the task queue is not maintained by the developers, but it is determined by the relationships within the social network. Thus, the more friends a submitter has, the more computational resources will flow towards his or her project.

3) *Socially-Computed Conten*t – The results of the social computations are added back into the system as new content. This content has a new type of value that has not yet been experienced in social networks and could have far reaching ramifications on how communities learn and share knowledge. By sharing the computational results within a social network, participants not only become more invested but also facilitate new discoveries.

4) *New Value Added to Social Networks* – SVC does not merely use social networks as an infrastructure, but SVC adds new value to social networks by contributing new types of content and relationships. For example, allocation of computational resources through the relationship of "friend" gives a new dimension to the role of a friend, which has become a devalued commodity within a social network.

2. RENDER WEB 2.0

In order to test the above ideas, we integrated a volunteer rendering system within a social network. Because rendering is computationally intensive, important for scientific visualization, and a popular hobby among many artists, rendering lends itself towards an interdisciplinary SVC prototype system. The Facebook application of our SVC rendering system can be tested at the following url: http://apps.facebook.com/renderweb.

2.1 Introduction to RenderWeb 2.0

In previous work, we demonstrated that distributed Java applets can efficiently render high quality animations across the Internet in a volunteer computing system we called RenderWeb (http://www.renderweb.org) [7]. RenderWeb used the Java Sunflow renderer [8] within an applet to render tasks downloaded from a Java servlet. In a final experiment, we harnessed 172 heterogeneous computers across the Internet to render an animation approximately 100 times faster than one lab computer

RenderWeb 2.0 retains a similar architecture for client/server communication and distribution logic: an applet embedded within a web page downloads a project from a Java servlet, renders the image, and uploads the image back to the servlet. Apache Tomcat is used as the application server in conjunction with MySQL. While its architecture is very similar to the previous version, there are two major changes in RenderWeb 2.0

In a first change, RenderWeb 2.0 no longer uses the Sunflow renderer but instead uses the open-source Blender renderer [9]. Within a trusted signed applet, the applet downloads the native Blender and executes it within separate process, as detailed in Section 2.2. With this change, artists and scientists can now use a production quality animation program in conjunction with volunteer computing. Blender was selected as our rendering

platform for several reasons: it is open-source, has a small download footprint, and is one of the most widely used animation programs with over 200,000 downloads per release [10]. Though Blender is an appropriate renderer, there is no reason, aside from licensing issues, that commercial animation programs could not also be integrated into RenderWeb. In the future, we hope that commercial vendors will realize the potential of RenderWeb and will construct a licensing mechanism to allow their renders to be utilized.

In a second change, RenderWeb is now a Social Volunteer Computing system that is integrated into Facebook and utilizes relationships to drive the flow of computation. Integration with Facebook was accomplished using the Facebook client API, which allows a web application to display within Facebook and allows the application to query Facebook user data. In Section 2.3, we will discuss the benefits of integrating RenderWeb into Facebook.

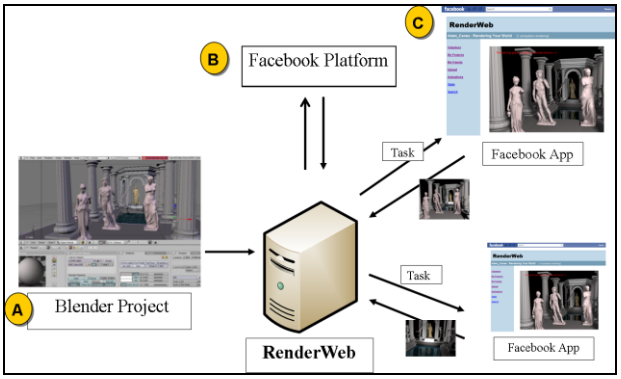

Figure 1: Overview of RenderWeb 2.0

Figure 1 depicts an overview of the RenderWeb 2.0 system and the general work flow: (A) Users upload blender projects to the RenderWeb server via a Facebook application. (B) The RenderWeb server communicates with the Facebook platform to authenticate the user and obtain user data, such as a user's friend list. (C) The computation is distributed among the community and rendered using Java applets that are embedded within the Facebook application. After rendering each frame, the applet uploads each rendered frame back to the server.

2.2 The Rendering Client

Because RenderWeb 2.0's rendering client uses a similar architecture to our previous version. we refer to our previous publication for details regarding the implementation, the scalability, and the experiential results of the RenderWeb architecture [7]. Yet, one significant change is that RenderWeb 2.0 now allows users to upload and render native Blender projects, as opposed to previously using Sunflow projects.

To accomplish this, a trusted signed applet performs the following steps:

a) The user is prompted with a screen informing the user that the applet's code is signed and verified. If the user accepts, the applet will have the permission to run the native Blender code. By using signed trusted code, we have followed the same security mechanism as BOINC.

b) The applet downloads the appropriate native code for the user's operating system (the user's operating system is obtained through the user- agent HTTP header).

c) The applet downloads a Blender task from the server through HTTP GET.

d) The applet performs a check sum on all downloaded native code and projects. The checksum takes a few milliseconds, but is an important security step to make sure that all code and files have maintained integrity.

e) A JNI call is made to render the task in a separate process.

f) When the rendering is complete, the resulting image is displayed within the applet

g) The image is compressed and sent to the server over HTTP POST.

h) Step C is repeated.

2.3 The SVC Model Applied to RenderWeb 2.0

This section applies the four proposed benefits of the SVC model (outlined in section 1.5) to RenderWeb 2.0.

1) *Social Network Integration*: Unlike non-social volunteer computing systems, RenderWeb 2.0 can be accessed directly as a Facebook application (apps.facebook.com/renderweb). Moreover, unlike Progress thru Processors, there is no need for the user to manually download and install a separate application. Instead, the user experience is entirely contained within Facebook. The following images outline the process of uploading and managing a Blender project (Figure 2) and volunteering a computer to render (Figure 3). Social network integration also allows wall and status updates to notify friends of projects that need to be rendered (see Figure 4).

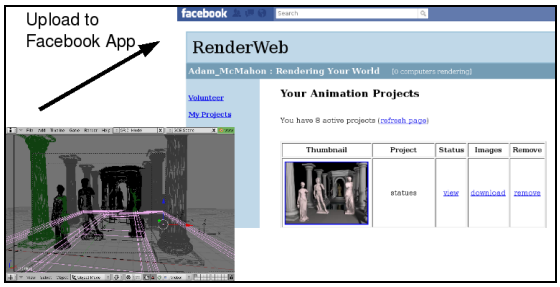

Figure 2: Blender projects managed within Facebook.

Figure 3: Screen shot of rendering an animation within Facebook. Users volunteer by clicking the "volunteer" link.

Figure 4: Status updates inform friends of projects that need rendering.

2) *Socially-Driven Computation*: In RenderWeb 2.0, the relationships within Facebook drive the flow of computational resources. That is to say, the priority of volunteers' resources will be allocated towards friends who have projects that need to be rendered. Within RenderWeb 2.0, there are three relational levels that drive computation:

a) *Self*: RenderWeb 2.0 gives a higher priority towards allocating to a user his/her own projects. That is to say, when a user volunteers a computer, the system first checks to see if that user has a project that needs to be rendered. If so, the system allocates a user's own project to be rendered on his/her computer.
b) *Friend*: If the user does not have a project in the queue, RenderWeb 2.0 then checks to see if any of the user's friends has a project in the queue. If so, the friend's project will be allocated to be rendered on the user's computer.
c) *Community*: If neither the user nor a friend of the user has a project in the queue, then RenderWeb 2.0 will allocate a random project from community.

3) *Socially-Computed Content*: After an animation is rendered, the video is placed as content back into the social network for the entire RenderWeb community to view, tag and comment on. In a sense, the entire community owns the animation, because the community provided computation resources towards creating the animation. Yet, like all content within a social network, the original creator of the animation can remove the animation from RenderWeb. Figure 5 depicts a screen shot that shows recently rendered animations that have been added as Social-Computed Content.

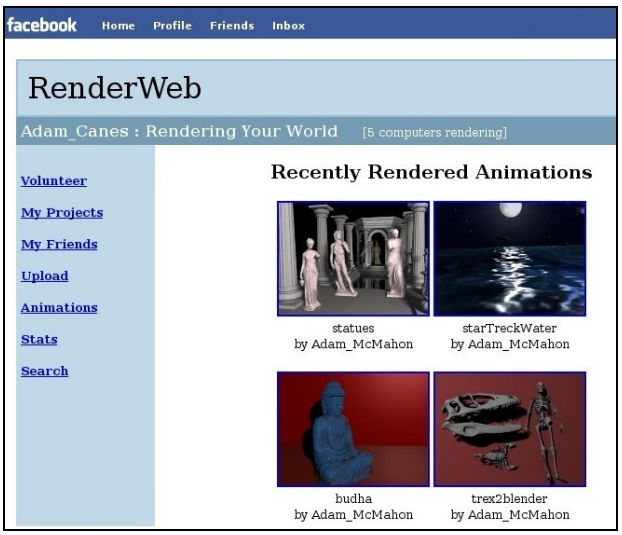

Figure 5: Rendered animations are added back into the Facebook as Socially Computed Content

4) *New Value Added to Social Networks*: Within RenderWeb 2.0, not only is volunteer computing benefited by the social network, but the social network is benefited from the SVC model and given new value. First, a friend within Facebook (which has often become a devalued commodity) now has the value of contributing computational power towards rendering and sharing computational resources. Second, the many Facebook groups centered around Blender will now have ability to easily browse each other's animations that were rendered across the social network. Third, through a community effort of sharing computation, the community will have a new value of ownership of this new type of content.

3. RENDER WEB 2.0 - IN THE ARTS AND SCIENCES

Though it is a new platform, RenderWeb 2.0 is already being utilized to render scientific data. For example, Figure 6 depicts a raytraced animation of a 3d configuration space. In robotics, a configuration space represents the permitted translations and rotations that will allow a robot to safely navigate a room [11].


Figure 6: Configuration Space rendered in RenderWeb 2.0

In another example, RenderWeb 2.0 is being used to visualize geological data that was obtained after the recent Haiti earthquake (Figure 7). This visualization may assist researchers in determining sections of Haiti that will be susceptible to future earthquakes.

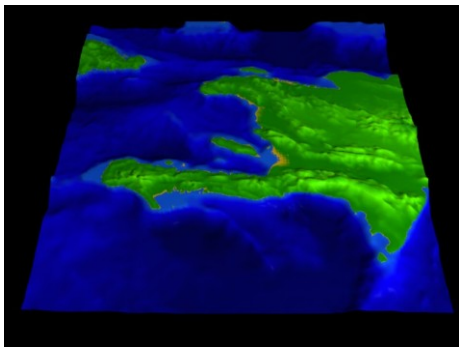
Figure 7: Haiti earthquake data visualized in RenderWeb 2.0

Starting this year, RenderWeb 2.0 will be utilized as an educational component in computer graphics courses at the University of Miami. By utilizing RenderWeb, students will have the opportunity to share computational resources and content within a Facebook community. We believe this social

dimension will enhance their collaboration and their sharing of knowledge. During these courses, we will begin to study how a SVC system affects users' experience of sharing resources and content through social networks.

4. CONCUSIONS

We have introduced the Social Volunteer Computing model, a field which combines volunteer computing and social networks. This model introduces new forms of computation and content to social networks. We have also proposed that there are four benefits of SVC that improve volunteer computing: Social Network Integration, Socially-Driven Computation, Socially-Computed Content, and New Value to Social Networks.

We have also introduced RenderWeb 2.0, which is a prototype SVC system that is integrated into Facebook and uses social relationships to drive the priority of task allocation. Moreover, we discussed how RenderWeb 2.0 is starting to be utilized by the scientific community for visualization, and we discussed our plan to use RenderWeb 2.0 as an educational component within graphics courses.

Our future work will touch upon three major areas. First, we will continue to reach out towards artistic and scientific communities to integrate SVC with existing and emerging projects. Second, we will develop measures to test the proposed benefits of the SVC model against other computational and social systems. Third, we will explore how the roles of developer and facilitator can be integrated into the SVC model. With this integration, scientists will be able to seamlessly enter into the role of developer by uploading custom code that will be distributed among the relationships in a social network.

Of course, allowing developers to submit custom code opens a security risk that must be addressed. We are currently exploring how technology, such as the Google Native Client [12], will allow distributed native code to be securely executed within a web page of a social network. By expanding to include all four roles (volunteer, submitter, developer, and facilitator), volunteer computing will truly become social. It will allow new groups and projects to emerge overnight without the intervention of a gatekeeper that controls the flow of computation or the priority of projects. For example, in such a system, there could be many Facebook groups corresponding to numerous projects, and volunteers can contribute their resources by simply joining such a community. This would allow users not only to join a group interested in finding a cure to breast cancer but also to volunteer their computer towards that cause as an effect of joining that community.

In summary, the relationships and constructs within social networks will continue to provide a rich and innovative platform for sharing computational recourses.

5. ACKNOWLEDGEMENT

This work supported by NSF grant CCF-0904707.

6. REFERENCES

[1] L. Sarmenta. “Volunteer Computing”, Ph. D. diss., Massachusetts Institute of Technology, 2001.
[2] I. Peterson. “Another Record Prime.” 1996. http://www.maa.org/mathland/mathland_12_16.html
[3] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, D. Werthimer. “SETI@home: An Experiment in Public-Resource Computing.” Communications of the ACM, 45(11), November 2002, 56-61.
[4] D.P. Anderson. “BOINC: A System for Public-Resource Computing and Storage.” 5th IEEE/ACM International Workshop on Grid Computing, pp. 365-372, Nov. 8 2004, Pittsburgh, PA.
[5] Intel News. “Intel Helps Convert Unused PC Processor Power into an Instrument to Fight Disease and Study Climate Chang.” 2009. http://www.intel.com/ pressroom/archive/releases/20090803corp.htm
[6] Nov, O., Anderson, D. and Arazy, O. (2010). Volunteer Computing: a Model of the Factors Determining Contribution to Community-Based Scientific Research. Proceedings of the 19th International World Wide Web Conference (2010).
[7] A. McMahon, V. Milenkovic, “Rendering Animations with Distributed Applets.” Proceedings of the International Conference on Computer Graphics and Virtual Reality, 2009.
[8] Sunflow web page - http://sunflow.sourceforge.net
[9] Blender web page - http://www.blender.org
[10] Blender Manual - http://wiki.blender.org/index.php/ Doc:Manual/Introduction
[11] S. Trac. “Robust Explicit Construction of 3d Configuration Spaces Using Controlled Linear Perturbation”, Ph. D. diss., University of Miami, 2008. www.cs.miami.edu/students/strac/PhDThesis/thesis.pdf
[12] B. Yee, D. Sehr, G. Dardyk, J. Chen, R. Muth, T. Ormandy, S. Okasaka, N. Narula, and N. Fullagar, “Native Client: a Sandbox for Portable, Untrusted x86 Native Code,” IEEE Symposium on Security and Privacy, 2009.