AA Distributed Software Development Methodology

Renato Fabbri Ricardo Fabbri Vilson Vieira
Alexandre Negrao Lucas Zambianchi Marcos Mendonca
Danilo Shiga

February 18, 2013

Abstract

We present a new self-regulating methodology for coordinating distributed team work called Algorithmic Auto-regulation (AA), based on recent social networking concepts and individual merit. Team members take on an egalitarian role, and stay voluntarily logged into so-called AA sessions for part of their time (e.g. 2 hours per day), during which they create periodical logs — short text sentences — they wish to share about their activity with the team. These logs are publicly aggregated in a Website and are peer-validated, as in code review. A video is generally recorded by the members to make their work logs more understandable. This methodology is well-suited for increasing the efficiency of distributed teams working on what is called Global Software Development (GSD) through asynchronous on-demand communication, reducing the need for central management, unproductive meetings or time-consuming reports. The AA methodology also legitimizes the activities of a distributed software team. It thus enables entities to have a solid means to fund these activities, allowing for new and concrete business models to emerge for very distributed software development. AA as a methodology is also at the core of having self-replicating hacker initiatives. These claims are discussed in a real case-study of running a distributed free software hacker team called Lab Macambira.

1 Introduction

One of the defining features of modern times is the widening geographical distribution of software teams [6] creating what is called Global Software Development (henceforth GSD) [4]. An example is the free software movement. Projects and institutions like Mozilla Foundation has several employees and thousands of voluntary developers distributed across many countries. The same is true for GNOME [4], OpenBSD, MySQL or Apache Software Foundation, to cite just a few of the most active projects.[1] Along the free and open software projects, GSD is growing popularity in every niche of the software industry as a whole, even on those distributing their software with proprietary licenses. This phenomenon is attributed to a variety of factors such as a larger labor pool, natural globalization of software companies and foundations or even the premise of cheaper cost of production [5].

Despite the advantages of GCD, we have noticed how difficult it is to coordinate and fund free software on a larger scale than currently available, when teams are very heterogeneous containing not only volunteers and very experienced developers, but also contractors and freelancers from different backgrounds and cultures. Our observations are founded on the factors suggested by Carmel [2] as main difficulties for GCD: distance, time and cultural differences. In the case of free or open software projects, all these factors are involved.

Another problem faced by modern software companies and other collectives are frequent ineffective meetings, which are seldom focused to the interest to any attendant. The result is that it has become the norm to participate in too many meetings with the "laptop open", which can be un-productive. Software developers like to code, to be productive, to have their hands on their project, to do what they are best at. They dislike to forcibly stop for meetings or to do other bureaucratic activities such as writing lengthy reports to justify their funding. [7]

In this paper we propose the AA methodology and an associated software system for coordinating distributed team work, tackling the disadvantages of GSD. Team members take on an egalitarian role, and stay voluntarily *logged* in the system for part of their time (e.g. 2 hours per day), during which they log a periodical short text sentence or *microlog* — similar to a 'tweet' from Twitter — as the status of their activity. Logging is carried out using an easy a series of available UI's: unix shell commands, native GUI or web page, conventional social network posts, or chat messages to a log bot listening to IRC, GTalk, G+, and others. These "microblog sentences" are publicly aggregated and validated by other team members. Through AA, we have a methodology and an associated system to help implement and validate the activities of a distributed software team. It implicitly legitimizes financial support for the expansion of the activity of the developer team. The AA methodology is specially useful for coordinating distributed and decentralized team work, providing effective means to asynchronously update different team members without the need for synchronous unproductive meetings.

A brief overview of current work on GSD methodologies related to AA is presented in Section 2, while in Section 3 we describe the most relevant characteristics of the AA methodology followed by the description of our experience using AA in a team of 9 paid developers since July 2011. In Section 5 we draw some final conclusions and indicate future possibilities for the practical use of AA in other types of software developers teams or organizations working on

[1]Ohloh, the open source network, have a more complete and constantly updated list of the most active projects on-line at www.ohloh.net