Progress in IS

More information about this series at http://www.springer.com/series/10440

Wei Li · Michael N. Huhns
Wei-Tek Tsai · Wenjun Wu
Editors

Crowdsourcing

Cloud-Based Software Development

Springer

Editors
Wei Li
State Key Laboratory of Software Development Environment, School of Computer Science and Engineering
Beihang University
Beijing
China

Michael N. Huhns
Department of Computer Science and Engineering
University of South Carolina
Columbia, SC
USA

Wei-Tek Tsai
Department of Computer Science and Engineering, School of Computing, Informatics and Decision Systems Engineering
Arizona State University
Tempe, AZ
USA

Wenjun Wu
School of Computer Science and Engineering
Beihang University
Beijing
China

ISSN 2196-8705 ISSN 2196-8713 (electronic)
Progress in IS
ISBN 978-3-662-47010-7 ISBN 978-3-662-47011-4 (eBook)
DOI 10.1007/978-3-662-47011-4

Library of Congress Control Number: 2015938742

Springer Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer-Verlag GmbH Berlin Heidelberg is part of Springer Science+Business Media (www.springer.com)

Foreword

The last few years have seen a sea change in how computer science proceeds as a profession. This change is motivated not only by advances in technology and innovations in business models, but also by the emergence of new attitudes toward technology, business, and work. In part, this has to do with lowered barriers to entry into the profession; in part, with a new generation of software developers coming of age, many of whom are not formally trained as computer scientists; in part, with businesses pushing toward open innovation; and in part, with the continuing internationalization of the economy and contributions by active participants from virtually anywhere.

Regardless of the underlying causes, we are now seeing a new pattern of software development gaining prominence. In this pattern, a programming project is broken down into pieces of work; each piece is performed by one or more people; the results come together; and the project is assembled. This, the idea of *software crowdsourcing*, is the theme of this book.

A difference between software crowdsourcing and traditional crowdsourcing is the presence of significant structure in the problems being solved, as well as in the teams drawn from the "crowd" that solve these problems.

A difference between software crowdsourcing and traditional software contracting is that each step of the above-mentioned pattern now may be achieved in a proactive and team-oriented manner. For example, the selection of a problem to solve, its breakdown into subproblems mapped to concrete tasks, the assignment of tasks to different teams, the testing of their solutions or products, and the assembly of the products into a solution to the original problem are all tasks that the crowd would accomplish, potentially with limited or no control from a central party.

The chapters in this book address the numerous challenges to be overcome in advancing the vision of software crowdsourcing and helping make it a dominant approach in software development. In skimming a draft of this book, I see chapters that take up major challenges.

- "Crowdsourcing for Large-Scale Software Development" and "The Five Stages of Open Source Volunteering" provide a historical perspective and tutorial

material on the key concepts and best practices of software crowdsourcing, such as they are established today.

- Some of these challenges apply to Internet applications in general, but are made more acute in the case of software development. In this category I would include the selection of workers, trust, and reputation, and broadly the nature of work. "Worker-centric Design for Software Crowdsourcing: Towards Cloud Careers" introduces these concepts.
- Some challenges pertain to new models for collaboration, including interaction protocols, human participation, incentives, and team work as are required for this setting. "Bootstrapping the Next Generation of Social Machines"–"An Evolutionary and Automated Virtual Team Making Approach for Crowdsourcing Platforms" address these challenges through approaches that introduce a broad range of concepts.
- Some challenges pertain to motivating humans to participate actively and positively. "Collaborative Majority Vote: Improving Result Quality in Crowdsourcing Marketplaces" describes a voting method as a basis for crowd-based quality assurance and "Towards a Game Theoretical Model for Software Crowdsourcing Processes" an incentive mechanism for encouraging high quality.
- Some challenges apply to software development in general. Here, I would include "TRUSTIE: A Software Development Platform for Crowdsourcing", which describes a software environment that realizes some of the concepts introduced in "Crowdsourcing for Large-Scale Software Development", "The Five Stages of Open Source Volunteering", and "Worker-centric Design for Software Crowdsourcing: Towards Cloud Careers".
- Some challenges apply to maintaining communities of participants as sustainable ecosystems. In this category I would include "Social Clouds: Crowdsourcing Cloud Infrastructure" and its notion of a Social Cloud.
- Some challenges become clearer when one attempts to pull together various technical ideas into deployed software systems. "Recommending Web Services Using Crowdsourced Testing Data" shows how to predict the quality of service offered by a web service instance based on crowdsourced performance data. "A Cloud-based Infrastructure for Crowdsourcing Data from Mobile Devices" illustrates the practical challenges in connection with a crowd-sensing application.

The book is a timely contribution to computer science that is at once both practical and scholarly. I applaud and congratulate the authors and editors on a job well done. Enjoy!

Munindar P. Singh
North Carolina State University
Raleigh, NC, USA

Contents

Overview

Summary of the Book

This book, *Cloud-Based Software Crowdsourcing*, brings together research efforts on many areas such as software engineering, service oriented computing, social networking and cloud computing, which are driving and shaping an emerging research field C software crowdsourcing. In the chapters of this book, you will find the perspectives of pioneering researchers on the fundamental principles, software architecture, development process, and a cloud-based architecture to support distributed software crowdsourcing.

Crowdsourcing software development or software crowdsourcing is an emerging software engineering approach. Software development has been outsourced for a long time, but the use of a cloud to outsource software development to a crowd of developers is new. All software development tasks can be crowdsourced, including requirements, design, coding, testing, evolution, and documentation. Software crowdsourcing practices blur the distinction between end users and developers, and follow the co-creation principle, i.e., a regular end-user becomes a co-designer, co-developer, and co-maintainer. This is a paradigm shift from conventional industrial software development, with developers distinct from users, to a crowdsourcing-based peer-production software development in which many users can participate. A cloud provides a scalable platform with sufficient resources, including computing power and software databases, for a large crowd of developers. With the increasingly powerful cloud software tools, it significantly reduces the amount of manual labor needed in setting up software production environments, thus empowering peer developers to perform software crowdsourcing tasks efficiently in design, coding, and testing. By taking advantage of the elastic resource provision and infrastructure, software crowdsourcing organizers can swiftly orchestrate distributed and large-scale development over highly dynamic communities.

Preliminary crowdsourcing practices and platforms including Apples App Store, TopCoder demonstrate this advantage of crowdsourcing in terms of software ecosystem expansion and product quality improvement. Recently, multiple seminars and workshops have been held to start theoretical and empirical studies on software

crowdsourcing. Many open questions need to be explored: What are the tenets for the crowdsourcing development of socio-technical ecosystems? What are the unique characteristics of the crowdsourcing method that distinguishes it from other classic software development methods? What can one align the software architecture with crowdsourcing organization of software ecosystems? How can one govern the social structure of the socio-technical ecosystem to manage the community, regulate the development activities, and balance the potential conflicts between project budget and time constraint?

This book summarizes the state-of-art research in the emerging field and introduces the important research topics including fundamental principles, theoretical frameworks and best practices of applications and systems. The book is a collection of papers written by pioneers and researchers who attended the Dagstuhl seminar in 2013. As this book is contributed by multiple authors with different perspectives, each paper will have its own unique views and notations, but collectively, these papers provide a comprehensive look of this new and exciting field.

Book Organization

The book is divided into three parts: Part I ("Crowdsourcing for Large-Scale Software Development"–"Worker-centric Design for Software Crowdsourcing: Towards Cloud Careers") describes the basic concepts and notation in software crowdsourcing. Part II ("Bootstrapping the Next Generation of Social Machines"–"Towards a Game Theoretical Model for Software Crowdsourcing Processes") covers the theoretical frameworks and models on software crowdsourcing from different perspectives. Part III ("TRUSTIE: A Software Development Platform for Crowdsourcing"–"A Cloud-based Infrastructure for Crowdsourcing Data from Mobile Devices") presents the software crowdsourcing platforms and technologies.

Part I: Software Crowdsourcing Concepts and Design Issues

"Crowdsourcing for Large-Scale Software Development" defines the notation and principles of software crowdsourcing and introduces a 4-level maturity model for assessing software crowdsourcing ecosystems in terms of platform architecture, community scale, organization fabric and development approaches. "The Five Stages of Open Source Volunteering" reviews best practices of the open source projects, and identifies a five-stage process for volunteering and recruitment that can significantly increase the chances for a successful open source project.

Software crowdsourcing projects largely depend upon the global labor forces consisting of temporary crowd workers. But it will be problematic if we expect software crowdsourcing to become a sustainable industry where workers only need to maintain reduced levels of commitment with their commissioners. "Worker-

centric Design for Software Crowdsourcing: Towards Cloud Careers" analyzes the relevant issues including: trust and reputation development between workers, team selection and building for specific tasks, and contextualization of software crowdsourcing projects as a motivating factor for workers.

Part II: Software Crowdsourcing Models and Architectures

"Bootstrapping the Next Generation of Social Machines"–"Towards a Game Theoretical Model for Software Crowdsourcing Processes presents multiple approaches to modeling and analyzing software crowdsourcing systems and processes.

Both "Bootstrapping the Next Generation of Social Machines" and "Multi-Agent System Approach for Modeling and Supporting Software Crowdsourcing" attempt to create multi-agent models of software crowdsourcing systems but with different viewpoints. "Bootstrapping the Next Generation of Social Machines" develops the model on the basis of a conceptual notion called social machine, to describe human behavior and interaction protocols. These social machines emerging from human community and computing resources are governed by both computational and collective social process. "Multi-Agent System Approach for Modeling and Supporting Software Crowdsourcing" proposes an agent-based analytic framework to model the organization and process of software crowdsourcing. Based on the model, the authors have developed a service-based multi-agent system platform called AutoService to simulate software crowdsourcing process and validate the theoretical framework. Adaptive and programmable incentive mechanism is essential for software crowdsourcing systems to support processing of complex and inter-dependent development tasks. "Supporting Multilevel Incentive Mechanisms in Crowdsourcing Systems: an Artifact-centric View" presents a novel, artifact-centric approach for modeling and deploying incentives in software crowdsourcing environments. The proposed framework augments the Artifact's lifecycle model with incentive mechanisms to facilitate team formation, task orchestration, run-time management of data flow and dependencies, collaboration and coordination patterns.

The last three chapters focus on modeling and optimization of software crowdsourcing processes. "An Evolutionary and Automated Virtual Team Making Approach for Crowdsourcing Platforms" proposes a systematic and automated approach to optimize the assignment of crowd workers to a crowdsourcing task. By considering the major constraints in software crowdsourcing processes, the authors formulate this optimization problem with the framework of search-based software engineering. Given the dynamic nature of crowd workers participating in software crowdsourcing processes, it is essential to design effective quality-control management to ensure the quality of crowd submissions. "Collaborative Majority Vote: Improving Result Quality in Crowdsourcing Marketplaces" introduces a collaboration mechanism to extend majority vote, one of the most widely used quality-assurance methods, enabling workers to interact and communicate during task

executions. "Towards a Game Theoretical Model for Software Crowdsourcing Processes" introduces a conceptual framework of software crowdsourcing process and develops a game theoretical model of peer software production to describe competitive nature of software crowdsourcing. The analysis of the model indicates that prize-only awarding mechanism in software crowdsourcing can only motivate dominant developers with superior skills to other crowd developers.

Part III: Software Crowdsourcing Systems

"TRUSTIE: A Software Development Platform for Crowdsourcing"–"A Cloud-based Infrastructure for Crowdsourcing Data from Mobile Devices" describes software architectures and platforms to support Cloud-based software crowdsourcing and demonstrates examples of practices.

"TRUSTIE: A Software Development Platform for Crowdsourcing" and "Recommending Web Services Using Crowdsourced Testing Data" present new ideas about software crowdsourcing platforms from two different aspects. "TRUSTIE: A Software Development Platform for Crowdsourcing" introduces a software development environment named by Trustworthy Software Tools and Integration Environment (Trustie), which supports a community oriented and trustworthy software development framework. This framework integrates many features including crowd collaboration, resource sharing, run-time monitoring and trustworthiness analysis. "Social Clouds: Crowdsourcing Cloud Infrastructure" proposes a novel approach of an infrastructure crowdsourcing model, termed as Social Cloud, to facilitate a user-contributed cloud fabric to host software crowdsourcing activities on which software development services and systems can be hosted.

The other chapters of this part are mainly about the applications of software crowdsourcing. "Recommending Web Services Using Crowdsourced Testing Data" focuses on improving the QoS prediction of service platforms using crowdsourced testing data. "A Cloud-based Infrastructure for Crowdsourcing Data from Mobile Devices" focuses on crowd sensing applications that often involve massive number of smartphone users for collecting large scale of data. The author presents a multi-cloud based crowd-sensing platform named by APISENSE, supporting participatory sensing experiments in the wild.

Wei Li
Michael N. Huhns
Wei-Tek Tsai
Wenjun Wu