How to Systematically Conduct Crowdsourced Software Testing? Insights from an Action Research Project

Research-in-Progress

Niklas Leicht¹
niklas.leicht@unisg.ch

Ivo Blohm¹
ivo.blohm@unisg.ch

Jan Marco Leimeister¹,²
janmarco.leimeister@unisg.ch

¹University of St. Gallen
Institute of Information Management
Unterer Graben 21, CH-9000 St. Gallen, Switzerland

²University of Kassel
Research Center for IS Design (ITeG)
Pfannkuchstr. 1, D-34121 Kassel, Germany

Abstract

Nowadays, traditional testing approaches become less feasible – both economically and practicably – for several reasons, such as an increasingly dynamic environment, shorter product lifecycles, cost pressure, as well as a fast growing and increasingly segmented hardware market. With the surge towards new modes of value creation, crowdsourced software testing (CST) seems to be a promising solution to effectively solve these problems and was already applied in various software testing contexts. However, literature so far mostly neglected the perspective of an organization intending to crowdsource tasks. In this study, we present an ongoing action research project with a consortium of six companies and present a preliminary model for crowdsourced software testing in organizations. The model unfolds necessary activities, process changes, and the accompanied roles for crowdsourced software testing to enable organizations to systematically conduct such initiatives and illustrates how test departments can use crowdsourcing as a new tool in their department.

Keywords: Crowdsourcing, software testing, action research, IS development
How to Systematically Conduct Crowdsourced Software Testing?

Introduction

Software testing is an integral part to ensure quality in software development. However, many IT departments face an increasingly dynamic environment, shorter product lifecycles and cost pressure. On top of that, the rapid development of new IT-enabled business models and a fast growing hardware market as well as its segmentation, that is, smartphones, tablets, wearable technologies or the “Internet of things”, lead to an increasing degree of complexity in software testing. Given the increased complexity, the domain of software testing is about to develop manifold approaches to overcome this issue. One approach is test automation, i.e., the automated execution of pre-scripted tests via software (Huizinga and Kolawa 2007). However, since automated testing is still not applicable in many settings (Rafi et al. 2012) and most tasks still require human intelligence to be performed, traditional approaches to software testing are becoming less applicable – both economically and practically (Bacon et al. 2009; Dolstra et al. 2013). With the surge toward collective undertaking of production processes in the context of new information and communication technologies (Boehm 2006; Tajedín and Nevo 2013), the concept of crowdsourcing – that is, the proposal of activities to a group of individuals who voluntarily undertake tasks based on a flexible open call (Blohm et al. 2013) – has gained track to effectively solve business problems (Brabham 2008; Jeppesen and Lakhani 2010) and recently found its application in software testing (LaToza and van der Hoek 2016). In crowdsourced software testing (CST) or crowdtesting, a diverse pool of people test software in real environments using their own devices leveraging the “wisdom of the crowds” (Surowiecki 2005) or as Linus’ Law, one of the mantras of open source movement, states: “Given enough eyeballs, all bugs are shallow” (Raymond 1999).

While the research on this topic is still in its inception, a number of studies have already examined some of the implications of CST. Existing literature predominantly describes crowdsourcing as a system with three components, the crowd, the intermediary, and the IT platform. CST was applied for the testing of graphical user interfaces and mobile applications and was found to be both technically feasible and sufficiently reliable (Amini et al. 2012; Dolstra et al. 2013; Liu et al. 2012). Furthermore, Hossfeld et al. (2014) assess key issues in crowdsourced user experience testing and provide a collection of best practices to address the challenges in crowdsourcing projects (Stol and Fitzgerald 2014). From a system’s perspective, Bacon et al. (2009) provide a comprehensive overview of the application of crowdsourcing in bug and vulnerability reporting. These studies deliver first insights that crowdsourcing can be a feasible option in software testing in different scenarios. Building on that, there are first attempts to identify specific scenarios in which CST might be a superior mode of creation compared to in-house testing (Leicht et al. 2016a). Moreover, there are first findings on how task design, expertise (Leicht et al. 2016b), and other factors such as process support and guidance (Tung et al. 2013) or time constraints (Mäntylä and Itkonen 2013) positively influence the performance of the crowd. The crowd is usually reached through an intermediary who has three main challenges to tackle: managing the settlement process, managing the crowd, and managing the technology (Tajedín and Nevo 2013; Zogaj et al. 2014).

Despite these merits, the perspective of an organization that seeks to crowdsource task has largely been neglected in crowdsourced software testing research, and is only examined on a conceptual level (Afuah and Tucci 2012; Estellés-Arolas and González-Ladrón-de-Guevara 2012). Thus, we follow the call of various researchers (e.g., Leicht et al. 2015; Zhao and Zhu 2014) and investigate crowdsourcing in software testing from an organization’s point of view. Accordingly, this paper aims to answer the following research question: How to use crowdsourcing and systematically conduct crowdsourced software testing initiatives? To answer this question, we report an ongoing action research project with a consortium of six companies that desire to employ crowdsourced software testing to uncover the necessary activities and tasks to be performed to leverage CST in their department. As a result, we present a preliminary model for mediated crowdsourcing. Mediated crowdsourcing is a popular form of crowdsourcing in the field of software testing where an intermediary connects organizations that want to apply crowdsourcing with potential crowdsources and manage the crowdsourcing process (Ipeiritis 2010; Zogaj et al. 2014). Our expected contribution is twofold. First, by investigating crowdsourcing from a seeker’s perspective we help to synthesize crowdsourcing literature that so far mostly neglected this perspective. Second, the developed process model structures the CST process and unfolds necessary activities as well as key levers in the context of software testing. Thus, we create deeper insights into the mechanisms of crowdsourcing in a software testing context. For practitioners, the paper illustrates how test departments can use CST and provides a model as a guideline with precise action taking in software testing projects for managers.
The remainder of the paper proceeds as follows: First, section 2 outlines the conceptual background of our research and introduces crowdsourced software testing as an application of crowdsourcing in practice. Section 3 is dedicated to our research approach and the project setting of our action research project. Following that, we report the first research cycle. Fourth, we give an overview of the planned second cycle. Last, we discuss expected contributions from our research as well as implications for practitioners, such as test managers and executives.

**Related Work**

**Crowdsourcing**

Crowdsourcing represents a new form of value creation and requires different activities and responsibilities compared to in-house approaches or outsourcing (Bernstein et al. 2012). Compared to outsourced testing activities, crowdsourcing represents a fundamental shift and extends the frontiers of available skills or knowledge beyond the boundaries of organizations or suppliers (Geiger et al. 2012). With crowdsourcing, it is possible to mobilize the expertise and creativity distributed among a large panel of people in order to achieve a certain set of tasks (Schenk and Guittard 2011), and not just preselected testers from a supplier (Afuah and Tucci 2012). However, the roles and activities in crowdsourcing can also vary. Figure 1 depicts the different models of crowdsourcing compared to traditional outsourcing. In traditional outsourcing, the principal negotiates the terms of the outsourcing degree, as well as the mode of outsourcing and the outsourcing company delivers the a priori specified services (Dibbern et al. 2004). In mediated crowdsourcing (II), an organization selects an intermediary who acts as a service provider. The intermediary mainly manages the crowdsourcing process including the identification and invitation of the crowdsourcees, the allocation of tasks, as well as the remuneration for the crowdsourcees via its IT-based platform (Blohm et al. 2016; Zogaj et al. 2014). The tasks are then processed by the crowdsourcees on the intermediary’s IT platform. The organization usually pays a fixed fee for the use of the service and interacts only directly with managers from the intermediary. Of course, an organization can choose to perform crowdsourcing without an intermediary in place (III). In this scenario, the organization takes the responsibilities of the intermediary and has to perform various other tasks and governance mechanisms, such as the motivation and incentives of crowdsourcees, communication with crowdsourcees, and many other (see Zogaj et al. 2015 for details).

![Diagram of sourcing forms](image)

**Figure 1. Sourcing Forms (adapted from: Hoßfeld et al. 2012; Zogaj et al. 2014)**

**Crowdsourced Software Testing**

Crowdsourced software testing (CST) or crowdtesting is a specific application of crowdsourcing in the domain of software development. It refers to the outsourcing of software testing activities to the crowd. It grants access to a diverse pool of people who voluntarily test software in real environments using their own devices (Leicht et al. 2016a; Zogaj et al. 2014). We define software testing as a verification process for the...
assess assessment of software quality and a process for achieving that quality by supporting the interests of all stakeholders of an application, that is, end-users, developers, software designers, and software testers (Bertolino 2007; Naik and Tripathy 2011). To achieve that goal, it becomes clear that different types of testing by various stakeholders at different times (during or subsequent to development) need to be done (Roggio et al. 2013). That means, that software testing in this sense is much more than just “the process of executing a program with the intent of finding errors” (Myers et al. 2011, p. 6).

CST can be applied in numerous types of testing, but research so far usually applied CST in dynamic testing scenarios where a written code is executed and examined by the crowd. Further, the crowd is mostly concerned with output given specific inputs since they do not know or see the source code. This is referred to as black box testing (or “end-user testing”) where the software itself is seen as a blackbox and only inputs and outputs are visible for the tester. This form of testing not only covers functional aspects, it also covers non-functional aspects such as performance, reliability, and security of a software (Roggio et al. 2013). The most common type of testing where CST gained increasing popularity is verification, validation and acceptance testing. While verification testing aims to eliminate defects and faults that cause error states (functional black-box testing), in validation testing the end-user runs tests to determine if specific inputs result in specific outputs to ensure no failures are experienced (Stutzke 2005). Another important part of software testing is concerned with the usability of a software that is defined as “the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use” (ISO 9241-11). Subsequently, usability testing is the test of software with the goal of improving it (Dumas and Redish 1999). In this vein, beta testing can be considered as both validation and acceptance, as well as usability testing depending on its goal. Table 1 gives an overview of the application of CST in different types of testing.

<table>
<thead>
<tr>
<th>Type of Testing</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functional and Verification Testing</td>
<td>Dolstra et al. (2013); Leicht et al. (2016a); Mäntylä and Itkonen (2013); Chen and Luo (2014)</td>
</tr>
<tr>
<td>Non-Functional Testing</td>
<td>Chen and Luo (2014) (performance testing)</td>
</tr>
<tr>
<td>Validation and Acceptance Testing</td>
<td>Amini et al. (2012); Leicht et al. (2016a); Tung and Tseng (2013);</td>
</tr>
<tr>
<td>Usability Testing</td>
<td>Hoßfeld et al. (2014); Liu et al. (2012)</td>
</tr>
</tbody>
</table>

As with other crowdsourcing applications, companies, meaning the crowdsourcer, can either directly interact with the crowd or they can use intermediaries who provide this service for a fee (Chanal and Caron-Fasan 2010). These intermediaries act as brokers who connect the organizations that want to apply crowdsourcing with potential crowdsourcerees and manage the crowdsourcing process (Ipeirotis 2010; Zogaj et al. 2014). Depending on the type of testing (e.g., (non-)functional testing, usability testing), these tasks as well as the targeted crowds can be very diverse (Stol and Fitzgerald 2014).

**Methodology**

**Action Research**

In order to study crowdsourced software testing initiatives in an organizational context and to derive a process model, we applied action research. Action research is future oriented and does not strive for distanced and generalizable explanations or the prediction of coherences, but the joint understanding and learning by researchers and subjects as well as the changing of actual conditions based on a real problem (Baskerville and Myers 2004; Susman and Evered 1978). Thus, the essential purpose of action research is to address concerns to individuals or organizations with the broader purpose to contribute to the solution of this concern (Reason 2006). Action research combines the theoretical knowledge of the researchers with the subject’s practical and situated insights and has established as viable method in information systems research in general (e.g., Kohler et al. 2011; Lindgren et al. 2004; Puhakainen and Siponen 2010) and also for implementing crowdsourcing mechanisms in organizations (Haas et al. 2015). It is especially suited when the researcher needs to become deeply involved with the subject’s problem and environment (Füller
et al. 2011; Street and Meister 2004). To invoke the deep involvement, action research is a cyclic and multiphase process consisting of five iterative phases: (1) diagnosing, (2) action planning, (3) action taking, (4) evaluating and (5) specifying learning (Aguinis 1993; Baskerville and Wood-Harper 1996). However, because the researchers are directly involved in the project, action research is about choices. The quality of action research rests on the ability of the researchers to see the choices that are made and understand the consequences, as well as to make the choices transparent and communicate them to a wider public (Reason 2006).

Project Setting

The research project is built on the basic principles of consortium research (Österle and Otto 2010). Consortium research aims to develop solutions for a problem class within a collaborative environment. It profits from the collaboration and exchange of expertise from researchers as well as partner companies who all share the same goal. The process is iterative in its nature and thereby fits into the action research approach with two cycles. To identify how to profit from crowdsourced software testing, we started the initiative with six companies that monitored CST for a while but had no experience with crowdsourcing so far and subsequently struggled to use this new form of software testing in their departments. These companies all believe that testing with the crowd can help them to improve on the overall testing performance. To tackle this issue an interdisciplinary project team was composed consisting of researchers specialized in crowdsourcing and several experts in software testing, that is, test managers, head of testing services, and testers from the involved companies.

Research Cycles

In order to explore the mechanisms of crowdsourced software testing and develop a systematic approach, we intend to conduct two action research cycles. For the first cycle, we chose to investigate mediated CST. The reason we chose this setting for our first cycle is twofold. First, this setting represents a very common scenario in crowdsourced software testing where an intermediary acts as a service provider and is managing the crowdsourcing process. Hence, the test manager of the organization is only communicating with the intermediary who manages the crowd and the testing process. Second, since companies are accustomed to having software testing performed externally, the shift to using mediated crowdsourcing becomes less complex but at the same time unveils key insights on the differences between traditional sourcing and crowdsourcing. As for the second cycle, we intend to expand the model for crowdsourcing without mediation that is more complex in its nature because the organization has to perform several activities performed by the intermediary in the mediated CST approach.

Action Research Cycle One

Phase 1: Diagnosing

With the intent of creating an approach to systematically conduct crowdsourced software testing, we first investigated the existing testing processes and practices. We monitored the testing process in the companies and reviewed documentation to gain a deeper understanding of existing practices. Further, we wanted to develop a deeper understanding why test managers did not apply CST so far. For this reason we conducted exploratory interviews with the involved test managers (N=3). Moreover, we interviewed three intermediaries who are all specialized in different kinds of testing (e.g., usability testing, functional testing) to understand the mediation process and the requirements to a company from an intermediary’s perspective. Our interviewees (N=3) were experienced test managers who execute and manage crowdtasks with different clients on a regular basis. Finally, to ground the diagnosis in theory and further enhance our understanding, we reviewed existing literature regarding these matters. This triangulation of data allowed us to derive pivotal requirements and challenges to be addressed by our solution.

The interviews with the test managers revealed that they either did not know how to frame a test to use crowdsourced testing but also were not sure what tasks they have to perform in mediated crowdsourcing. Statements from managers of intermediaries supported this by stating that many customers were unsure about their objective and the test scope and that these two activities are vital to project success. Another concern by the test experts was, that the defect management process could become excessively extensive
due to many submissions of low quality (Ipeirotis and Paritosh 2011) or duplicate bug reports. Furthermore, the diagnosis revealed that the involvement for test managers in the test itself and defect management are different in CST. Although the intermediary communicates with the crowd, there are often specific issues where support from the companies' test manager is needed during the tests.

Besides that, security and intellectual property is another issue in crowdsourced software testing (Leicht et al. 2016a; Stol and Fitzgerald 2014). By opening to a large, anonymous crowd, companies are afraid that crowdsourcers could, instead of testing the software, try to hack the application and harm the company. In this regard, the test managers were also concerned about industrial spying, as well as public exposure due to low software quality of the test object. This leads to the conclusion that it is of high importance to check if crowdsourcing can be applied taking IT and data security regulations in account. Moreover, test managers need to carefully consider in which software testing phase to use crowdsourcing and how to scope the test.

**Phase 2: Action Planning**

Based on the diagnosis and informed by standard project management literature (Cleland and Ireland 1994; Wysocki 2011) we framed the CST initiative as a small sub-project in the testing process and created a four-phase process (cf. figure 2). As a next step we attempted to address the expressed concerns and necessary steps and built high-level activities within every phase.

![Figure 2. Preliminary CST Model](Image)

This high-level model was elaborated and further specified in two expert workshops with test managers of the involved companies as well as in single interviews with the test managers (N=4) and ultimately led to the description of 14 activities in mediated crowdsourced software testing. Table 2 explains the particular activities for each phase.

<table>
<thead>
<tr>
<th>Phase</th>
<th>Activity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. Definition</td>
<td>Potential Assessment</td>
<td>Identification and assessment of potential for the application of CST for the specific project, i.e., software object.</td>
</tr>
<tr>
<td>I. Definition</td>
<td>Definition of Objective</td>
<td>Definition of test objective. Crowdsourced software testing can be leveraged to accomplish several testing goals, such as functionality, usability, non-functional aspects, reliability etc.</td>
</tr>
<tr>
<td>I. Definition</td>
<td>Involvement of Test Manager</td>
<td>Depending on the objectives, the test manager can select between various service levels varying in the degree of involvement. Also, the crowdsourcers are defined, i.e., external crowdsourcers, customers, or employees.</td>
</tr>
<tr>
<td>II. Planning</td>
<td>Project Management</td>
<td></td>
</tr>
<tr>
<td>II. Planning</td>
<td>Supplier Assessment</td>
<td></td>
</tr>
<tr>
<td>II. Planning</td>
<td>Test Scope Definition</td>
<td></td>
</tr>
<tr>
<td>III. Execution and Controlling</td>
<td>Test Definition</td>
<td></td>
</tr>
<tr>
<td>III. Execution and Controlling</td>
<td>Application Management</td>
<td></td>
</tr>
<tr>
<td>III. Execution and Controlling</td>
<td>Intermediary Management</td>
<td></td>
</tr>
<tr>
<td>III. Execution and Controlling</td>
<td>Test Monitoring &amp; Support</td>
<td></td>
</tr>
<tr>
<td>III. Execution and Controlling</td>
<td>Defect Management</td>
<td></td>
</tr>
<tr>
<td>IV. Closing</td>
<td>Handover of Results</td>
<td></td>
</tr>
<tr>
<td>IV. Closing</td>
<td>Project Closure Report</td>
<td></td>
</tr>
</tbody>
</table>
### How to Systematically Conduct Crowdsourced Software Testing?

<table>
<thead>
<tr>
<th>Phase 3: Action Taking</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Compliance Check</td>
</tr>
<tr>
<td>5. Project Management</td>
</tr>
<tr>
<td>6. Supplier Assessment</td>
</tr>
<tr>
<td>7. Test Scope Definition</td>
</tr>
<tr>
<td>8. Test Definition</td>
</tr>
<tr>
<td>9. Application Management</td>
</tr>
<tr>
<td>10. Intermediary Management</td>
</tr>
<tr>
<td>11. Test Monitoring &amp; Support</td>
</tr>
<tr>
<td>12. Defect Management</td>
</tr>
<tr>
<td>13. Handover of Results</td>
</tr>
<tr>
<td>14. Project Closure Report</td>
</tr>
</tbody>
</table>

### Phase 3: Action Taking

The project setting allowed us to conduct a crowdsourced software testing project with the developed model and its activities in three different companies. The collaboration and ongoing interaction with the respective test managers proved especially helpful to collect valuable feedback and learnings as the consortium approach enabled a very fruitful interaction between all involved parties. Table 3 depicts the involved companies as well as the test object and the main objective of the test. To enhance generalizability the tests were conducted with different intermediaries per test. To further enhance the generalizability of the results we chose to apply the model in both development paradigms and with different test objects. In addition, the type of testing varied between the companies. The tests itself were conducted by 20-30 crowdsourcers for functional testing and 15 for usability testing. The duration of the tests was about 48-72 hours in each scenario.
Table 3. Test Settings

<table>
<thead>
<tr>
<th>Company</th>
<th>Development Paradigm</th>
<th>Test Object</th>
<th>Type of Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bank</td>
<td>Waterfall</td>
<td>Mobile Banking Application</td>
<td>Functionality</td>
</tr>
<tr>
<td>Insurance Company</td>
<td>Agile</td>
<td>Corporate Public Website</td>
<td>Usability</td>
</tr>
<tr>
<td>Online Retailer</td>
<td>Agile</td>
<td>Online Shop (Web)</td>
<td>Functionality &amp; Usability</td>
</tr>
</tbody>
</table>

Planned Phase 4: Evaluating

For the first cycle, we focused on a formative evaluation of the project. A formative evaluation is especially useful for empirical interpretations that serve as a basis for the improvement of the developed model (Venable et al. 2016). To evaluate and improve the process steps, we interviewed the involved test and project managers (N=6) before and after the project. For these interviews, we created a roughly structured interview guideline with questions regarding every phase of the crowdtest, the overall satisfaction with the process, the test results, and the collaboration with the intermediary. The participants should especially elaborate on aspects that caused disturbance or were perceived as not timely, i.e., too early or too late in the process. All interviews were recorded and subsequently transcribed. In addition, detailed notes were taken during the interviews. Moreover, we conducted a focus group workshop (Morgan 1996) with the consortium (N=13) to identify pitfalls and improvements of every phase for the second research cycle and to synthesize the knowledge from all initiatives, which will determine the learning in the next phase.

Planned Phase 5: Specifying Learning

Although our evaluation is a work-in-progress, first findings indicate that crowdsourced software testing with an intermediary requires a different workflow for the test manager. In contrast to outsourcing or in-house testing where the manager receives the results from professionals, crowdsourced software testing is often performed by semi-professional or even untrained testers. This fact leads to an increased effort in defect management. Another fact that intensifies the efforts is that in most cases, crowdsourcées are not testing against a specification and the intermediary usually also does not have a specification of the software at hand. These circumstances produce bugs professional testers would not have submitted and, in conclusion, lead to an increased effort in defect management. Also, the input of other stakeholders, for example compliance or security department, is needed and the examination process often requires time and additional action taking by the manager. This is especially true if the organization has no previous experience with crowdsourcing. Hence, the findings suggest that a new project role, the “crowdtest manager”, should be established, especially in larger projects with different development and test streams.

Planned Action Research Cycle Two

As for the second research cycle, we intend to apply and further develop the preliminary model from the first cycle for settings where the intermediation is performed by the crowdsourcing company itself. In this scenario, complexity is higher because the company has to perform additional activities once performed by the intermediary, such as, crowd management or remuneration of the crowdsourcées. For this purpose, we also intend to apply the model in three real-world testing projects of our consortium. Table 4 summarizes our approach for the second cycle.
Table 4. Planned Cycle Two

<table>
<thead>
<tr>
<th>Phase</th>
<th>Planned Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diagnosing</td>
<td>In the first cycle, the intermediary is seen as a “black box” performing a variety of important tasks and governance mechanisms to enable crowdsourced software testing. Building on literature (e.g., Zogaj et al. 2014; Zogaj et al. 2015), we intend to “white box” the intermediary and clarify which tasks have to be performed by the organization. Further, exploratory interviews with managers from crowdsourcing intermediaries will be conducted to shed light on that issue.</td>
</tr>
<tr>
<td>Action planning</td>
<td>Based on the learnings from the first cycle, the model will be further developed and enhanced to be applicable for scenarios without intermediation as well as hybrid forms. That includes necessary changes in existing activities, but mainly the inclusion of activities performed by an intermediary such as crowd and community management, and others. Further, the results revealed that existing roles in the testing process must be adjusted and new roles, e.g., a crowd manager responsible for the communication with the crowd need to be considered and employed.</td>
</tr>
<tr>
<td>Action taking</td>
<td>The adapted model including the specified learning will then again be applied in three different companies with different scenarios to ensure generalizability. We intend to apply the model for internal CST with employees where only the platform is provided, as well as completely without mediation with customers of the company as a crowd.</td>
</tr>
<tr>
<td>Evaluation</td>
<td>In order to evaluate the practicability and overall performance of the developed model, we intend to apply a summative evaluation approach. Summative evaluations focus on meaning an support decisions regarding the applicability of the model (Venable et al. 2016). We will conduct semi-structured interviews with the test managers as well as the project managers with a focus on the real world usefulness and realized benefits of the applied CST approach.</td>
</tr>
<tr>
<td>Specified Learning</td>
<td>The model developed in the action research project intends to provide an exhaustive and complete guideline on how to conduct CST initiatives and thereby claims to be applicable for a variety of different crowdsourcing scenarios in software testing. Depending on the results of the evaluation, we will decide if these criteria are met or a third action cycle should be conducted.</td>
</tr>
</tbody>
</table>

Expected Contribution and Future Research

To the best of our knowledge, this paper is the first to investigate crowdsourced software testing from an organizational point of view. Traditional software testing, performed in-house or outsourced, heavily relies on the knowledge of specific and very few people with a limited locus and an ex-ante selection of results. Crowdsourcing flips this concept by broadening the locus due to the open nature of the call and follows the mantra of open source code, which has proven to be a feasible concept over the last decades. This study will help to create deeper insights into the mechanisms of crowdsourcing from an organization’s point of view. Thus, we help to synthesize literature of crowdsourcing that so far mostly focused on the perspective of the crowdsourcer or intermediary.

Preliminary results reveal that, although the crowdsourcing process itself is managed by an intermediary, crowdsourcing still requires other process structures and tasks to be performed than traditional outsourcing of test activities. Hence, the developed model unfolds necessary changes regarding process design and the accompanied roles in the test process with the use of crowdsourced software testing. Further, the expected model identifies activities that are of particular importance for the success of CST initiatives. However, we argue that our results are not just limited to CST. In fact, due to the complex nature of the task and the modularizability as well as the interdependence of activities in software development, we propose that our results are transferable to other activities in the development process, that is, code production or even earlier stages of development such as design evaluation and even requirements elicitation. For practice, the paper illustrates how test departments can utilize crowdsourcing as a new tool in their collection. On top of that, we expect to provide a process model as a guideline with precise action taking in software testing projects for test managers to enable crowdsourced software testing in testing departments.
References


