Community integrated research project model for improving the quality and dissemination of new testing tools

Matti Vuori and Antti Jääskeläinen

Tampere University of Technology. Department of Pervasive Computing

P.O.Box 553, FI-33101 Tampere, Finland

e-mail: matti.p.vuori@tut.fi, antti.m.jaaskelainen@tut.fi

Abstract New types of software testing tools are often created by research organisations. The development is done in collaboration with companies. However, the dissemination of the results is often lacking and the created prototypes may even be neglected and not turned into tools available for the industry. This is influenced by current project models that do not form a continuum for the development of a sufficient community of committed practitioners. The prototypes created are also usually not of sufficient quality technically or from the viewpoint of practical use to invite productising. Dissemination and productising could be improved by new modes of collaboration during the research project in which practitioners will be integrated with the development from early on and would form a community with the researchers, much like in the case of open source development. The article presents such model for discussion and presents the rationale for it. Also, the roles and competences of the participants are discussed. The benefits for the model include voluntary involvement of interested and motivated practitioners, good exchange of ideas between research and practice, rapid testing of ideas, rapid prototyping of the tools, and getting development skills into the project early on. Furthermore, by forming a seed for a community that the ownership of the tool could be later transferred to, it would bring the tool to market in a community fashion, and at the same time open opportunities for commercial productisation.

Keywords software testing, software development, constructive research, university, industry, collaboration, innovation

1. Introduction

Constructive research (Crnkovic 2010), where research is aiming to produce new tools, e.g. testing tools for the practitioners, is an important part of academia-industry collaboration. There is a need for researching new approaches and testing technologies to handle the complexity of new technological systems or to create testing tools that are more productive than the last generation. Of course, any such work that aims in producing tools or at least prototypes or specifications for tools must be carried out in collaboration with the industry that will be using the tools, because it is the suitability of the tool designs and implementations that measure the validity of the approach and the value of the theoretical ideas. As a goal of disseminating the new ideas to the industry and almost every project of constructive nature aims to produce a good starting point for that: perhaps a prototype that could be developed into an industrial grade product or an open source offering that others could start developing further.

However, the projects will not often produce readiness for productising or actions toward it. In this article, we will look into some of the reasons for that and analyse common project structures, implementation and dissemination strategies and propose a new model for the projects that has as its core the idea of integrating a community of practitioners into the research project. The model presented includes the general architecture of a research project, the flow of dissemination, the roles of the participants, but will not include specifications for e.g. the development lifecycle used or its practices, though they are expected to be based on principles of agile software development.

The paper is reflective in its nature and based on the experiences of the authors and their colleagues. We don't aim to give hard proof or statistics on anything, but to bring the essential elements of our research and the changing environment into discussion and to give ideas for the research projects of coming years. The ideas are based on the authors' personal experiences in projects of this kind as a researcher and as a steering group representative of a company (a testing house) and discussions about the issues with researcher colleagues who have participated in the same projects, such as the development project of model based testing toolset TEMA (Tampere University of Technology 2012) and the project for developing a robot assisted testing platform (Tampere University of Technology 2013). Those experiences have led to the thinking that there could be possibilities for improving the basic models of carrying out constructive research.

The main thesis of the article is that the development and dissemination of tools created in constructive research could be based on a project model that is built around the idea of integrating a user (tester) community in the project from the beginning. That is a new idea for research projects, but communities are widely used in the development of open source software, so there are plenty of experiences available for making it work. Unique in the research setting is the need to include in the whole the hard core of scientific research and the practical development and testing done by the community of industrial practitioners. If and when that can be made to happen, the resulting project work should be productive and satisfying from the viewpoints of both academia and industry. Perhaps most importantly, the proposed model begins, besides design and implementation of the tool, its dissemination very early and thus organically builds on the visibility and reputation of the tool, making it attractive for a growing user base.

The article is structured as follows. In Chapter 2 we present the basic setting for constructive research and what the role of research institutes is in it. In Chapter 3, the current project structures are described and analysed for their characteristics regarding dissemination and productisation. In Chapter 4, a new approach is presented that is based on using a community of practitioners to collaborate in the development, much in the way they work in open source software development, yet maintaining the integrity of the crucial theoretical research. The characteristics and expected benefits of the proposed model are analysed in Chapter 5, and Chapter 6 contains summarising discussion about the approach presented.

2. The role of research organisations in tool development

Constructive research is one form of scientific research, where theories are turned into working tools that provide a proof or concept for a new approach, preferably in a form that allows a clear route to its development into tools that can be used in practice. It has been understood that trying things is essential for the development of theories and communicating them to the society and industry requires prototyping and demonstration tools. In this sense, the work is also design science (Hevner & Chatterjee 2010) as it as part of the process researches how the implementations of the theoretical ideas should be designed.

Obviously, as the goal is to change the world with the new research results, the results should be disseminated to the industry. Indeed, the research does not only look for scientific results, but ways to make the world a little better, perhaps with new testing tools that would help improve the quality of technological systems.

There is "contract" in the society that this is a work for research institutes, as they specialised in research. Any new approach contains a risk that it might not work in practice. If there is no such risk, the approach is not worth researching. Such a risk is impossible to bear for most companies, with the exception of huge companies whose R&D units have university-like opportunities for researching new ideas. Generally, companies should not take such risks as the research is too far from their core business. Companies have tight budgets and just enough personnel to do their business, there is usually no time to do advanced tool development. New advances are also expected to be based on knowledge that has been built with long durations of research and collaboration with other research parties. Thus, there are many expectations for the research units. First, they are expected to generate and development knowledge and ideas that advance the state of art in the domain. They are also expected to validate those ideas so that the industry can trust them. To get the development of a practical tool started, the researchers are expected to create artefacts that the industry can use to begin in advancing practices, such as documents that describe the new knowledge and documents that describe the new suggested practices that are validated at sufficient level. Additionally, they are expected to produce tools and prototypes of sufficient maturity that can work as models or starting points for industrial-grade tool development.

In general, there is a traditional "social contract" that set the roles of industry and academia in the advancement of technology. This setting is based on a world-view originated in mathematics and physics, where the world is seen as a "problem to be solved" in a laboratory. Obviously, the issues related to creating new testing algorithms is such, but developing tools for actual use, tools that could transform the world, requires seeing the world as a socio-technical system. It is a system where the activity of humans defines the "truths" and successes; the theoretical working of approaches is not sufficient. That requires a more intimate relation with the real activity in the actual activity systems of testing. That is why research institutes need to collaborate with the industry in the constructive research. There are many single motives for the collaboration, including the value of collaboration as such, the engagement of utilisers of research results and the validation of the approach. Practical reasons for the participation of companies include proving the commitment of industry to funding agency; without that there is little hope of getting funding. As the goals are of practical nature, the companies monitor and steer the research based on their knowledge of real needs of the industry and thus help in ensuring the applicability of research. An important element of collaboration is enforcing links between researchers and companies and networking people. The companies may also provide case studies where the ideas and tool prototypes can be assessed in real-life settings. In general, the companies expose the researchers to industry's needs, practice and thinking, which is essential in creating good tool concepts.

In an ideal case, the first domain of collaboration in product development of this kind is the fuzzy front end (see Koen et al. 2002) for the research, where there should be open ideation about the problem areas, applications, design characteristics, usage principles and so on. The fuzzy front end is a product development concept used in situations where new approaches are searched, and that should in many cases be the situation in research-led development that searches for unique, even disruptive innovations. In this kind of research the fuzzy front end, where there is chaos and where ideas flow freely, is still grounded in something: the new theory and the understanding that it could provide value if applied. In this phase it would be good to have very agile dynamism in finding where and how to apply the new theory (and where to test the prototypes), and that can only happen if there is good, open-ended collaboration between researchers and practitioners, while still keeping the "hard core" or research intact so that it will not be diluted in the wish of gaining fast benefits by shallow implementations. Scientific research works differently than other development paradigms on the fuzzy front end. Its main goal is to study the possibilities and opportunities for various disciplines in the target domain (such as software testing). The relevant disciplines include mathematics, computing science, computer science and formal logic. In addition to those, usability and user experience research and anthropology and organisational science have a lesser role. Also, domain specific disciplines are obviously utilised, of which in this case testing research has a central role. The researchers have responsibility to work mostly on the special disciplines, as they are supposed to be the experts on those. This is in contrast to traditional software development where the developers should be experts on the development process. In both situations, interaction with people who know the domain and its substance is needed to fill in the input in the non-expertise areas. That is where the end users are needed in the traditional development situations and the industrial practitioners are needed in the constructive research type of tool creation. In the collaboration, the theories and the "reality" represented by the practitioners are put into interaction where ideas and views are "bounced" and hopefully will lead into making sense of the opportunities, yet in the frame of the project's goals.

The next product development phase is the conceptual design phase, which should include planning of the overall approach of the software tool, defining the user stories or use cases it should support and forming a deeper understanding about the needs of the users. This is where the input from the practitioner is critical. Without that working properly, the tool has no basis and will fail to produce value. This is the phase where the agile development emphasises customer collaboration and where user research is highly recommended in all development approaches.

This phase will naturally lead into deeper design and implementation and testing of the tool, in an iterative and incremental manner. In this phase, the developers need to consider more practical issues such as interoperability with other tools and environments. Those are areas normally neglected by researchers working in isolation. Here is also a danger of misusing the researchers' ideas and turning the project into something it should not be: a productising project. That needs a change of the mindset and thus has traditionally been a separate project altogether and done in a spin-off company or similar. Note that the researchers can also change their mindset and re-focus their skills when the context is clearly changed. Good development benefits from changes in context and mindset and thus, separate phases or domains of activity can be valuable.

Both parties, the researchers and the practitioners, need some critical elements for the collaboration to succeed. First, they need to have the competence related to their roles. They should have high motivation for the mission, the creation of a new tool of value. Finally, they need an opportunity to do their work, including having allocated time, a working environments, all necessary support, tools and information of various kinds. For a project to get funding, the researchers need to be, and therefore will be competent in the theoretic parts of the development mission. Their motivation may, however, be more aligned with the theoretical goals and that is again where the practitioners are an invaluable part of the whole. The practitioners will carry with them to the researchers the meanings and values of the real world, several kinds of information, and an opportunity to have empathy and association with the humans who later would use the tools. So the collaboration is not only "objective" and rational exchange of information, but a highly psychological process.

When the elements of collaborative tool creation are in place, there is an opportunity for developing a good tool prototype that all parties have urgency to see disseminated to the industry, either by commercialisation or by some community process.

The characteristics of tool creation are something that any organising model for constructive research should offer and now we shall look into some current models before looking into a suggestion for improvement.

3. Models of academia-industry collaboration in constructive research projects

In this chapter we look into how the constructive research is organised in Finland. For that, we describe the common organising models for the projects that are used in Finland, which can be expected to be quite similar in many other countries. As there are in the reality of various models that vary in detail, the models are approximate averages, based on the authors' understanding of what elements are the most descriptive for the purposes of this article.

3.1 Roots in history

The basis for collaboration between research institutes and industry is laid in the national strategies for how research should be carried out, the instructions and guidance of the funding agencies, such as in Finland Tekes (Tekes 2015) and Academy of Finland (Academy of Finland 2015a) and in the researchers' understanding of what could be feasible.

We have seen different ages in this thinking. The relevant ones in this context are the "traditional years" from 1990's to 2010, during which there was a stable view of how roles in research should be divided and how companies should produce and utilise innovations, and the years after that when the structures in industry and the whole society have changed. We will not look into those changes in detail, but the main factors are, as perceived by the authors:

- Faster changes. Research can't spend many years in producing a new tool.

- Changing companies. Society is not built on large companies that have resources to implement the rough prototypes and raw ideas from research through their organisation structures, but need to have more ready-made tool infrastructure.

- Open source software. People are not expecting to spend thousands of euro per one workstation license for a tool, but expect to find free open source tools that they can use and share as needed.

- Communities. Software development has partly moved from companies to communities, where software (including testing tools) is created as a fast response to needs and either created further or neglected.

- More complex world. The world is not as simple as it was before. There is a diversity of cultures and needs and the development of tools needs to be sensitive to those and use the opportunities for development wherever a need and good environment emerges.

- The view of the essence of software tools has evolved. Software tools used to be considered sufficient if they performed their task technically, but today we understand that they also need to have good usability and produce good user experience. Even security should be considered when developing software development tools.

When the environment has changed, the ways of doing research that is in deep collaboration with the industry or aims to product tools for it, need to change also. After the traditional model of carrying out constructive research had been formed, much has been learned about how software development should be done, and those are critical things to consider. It is now generally understood that the research and development process should have agile characteristics, be faster than before due to the more dynamic environment, and consider more the overall quality of the resulting tools. Even though the end results are just prototypes, they need to enable trying and testing in real use scenarios, have sufficient usability and a user experience that produces a wish to use the tool. For the technological basis, they should have sufficient technical quality to enable further development easily. Those qualities need support both in the approach and goals for the research project, but also for the structures and processes used in the research and in the collaboration between researchers and industrial practitioners. Most importantly, it needs to be accepted that the researchers, who traditionally do the most design and implementation work, necessarily have skills and knowledge focused on their research issues. Thus, the process must have means to include the additional required skills of product development into the process in a fruitful way.

3.2 Traditional research project with research unit and industry partners

In this section, we present the process of a typical traditional research project with research unit and industry partners, first as it is supposed to work ideally, and then how it often works – or fails. Here we concentrate on the dissemination part of the process, as the whole point is usually to produce something that the industry can use. Before that, let's look into the whole project flow.

Research projects in the field of software engineering in Finland are traditionally started by a research group developing a research idea, writing it into a research project plan and then asking potential companies whether they would like to participate. Participation may have several levels, including having a case study in the project or just joining the steering group. One part of this is the provisioning of funding to the project. The funding agencies usually fund the project only partially (50-70 %) and while the research institute provides some internal funding, industry will usually need to come up with at least 10 % of the budget (all numbers are illustrative, not necessarily accurate). The exception for that is mostly strategic scientific research that does not yet aim at industrialisation. Besides aiding in the collection of the necessary funds, funding from the industry also proves the commitment of some companies to the project's execution and a serious expectation of added value from the project.

The means of industry's participation are varied. Sometimes the companies act as equal participants with their own sub-project or a case study for the researchers to assess their ideas. Case studies are usually private and the company decides what is published about them. In the case of tool development, they may involve gathering requirements for the tool in the special case of the company, tailoring the tool prototypes to support the company's workflows, data and integration to other tools and carrying out its piloting in the company, including targeted training in the company.

The companies usually have a member in the steering group of the project. Steering groups are usually composed of representatives of the research unit, the funding agency and the companies. Steering groups are a collaboration forum where the progress of the project is monitored and decisions about steering it are made. Often the decisions are related to schedule and details of the work packages, but when aiming for agility in longer projects, steering groups provide a mechanism for deciding larger changes in the scope of the project – after all, it makes no sense to do research and create tools that are no longer needed, if technology or another element has changed. Researchers are sometimes invited as a group to the steering group meetings, as that allows them to hear the views of the companies directly. That has been the usual practice in most of our tool development projects.

A technical advisory group is sometimes used, composed of experts from companies. Its task is to carry out deeper technical discussions and perhaps coordinate development. Private or public project seminars are often arranged. In the seminar, case studies and research results are presented. Presentations and seminars are sometimes also held in the participating companies' premises for their personnel.

Figure 1 shows the traditional model of research dissemination in how it is ideally thought to happen.

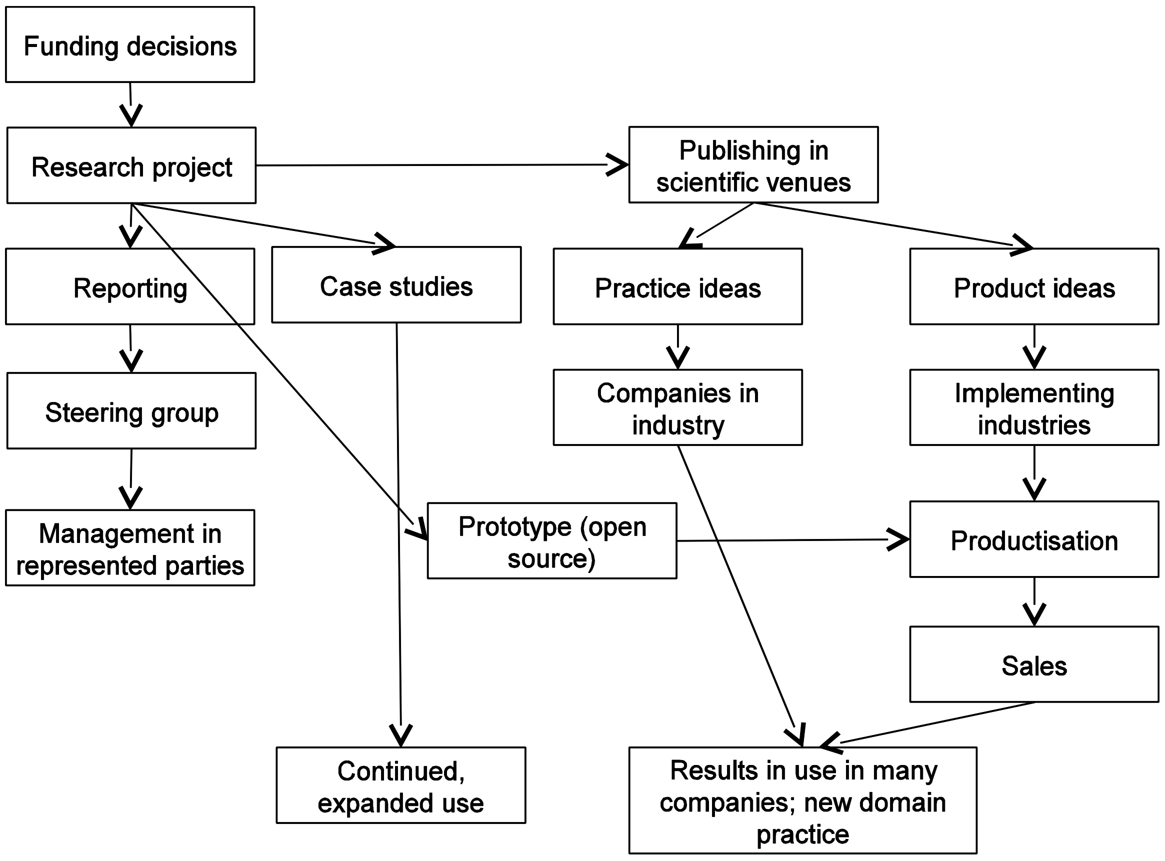

Fig. 1 The traditional model of research flow and dissemination in an ideal case [Editor: File: Fig_1.eps]

Let’s look into the dissemination part of the project. The research project will produce results, which are assumed to be scientific by nature, as they are produced by a research unit. Therefore they are expected to be assessed by the scientific community, and are mostly published in scientific venues, preferably the more respected ones. The idea is that the produced ideas are also assessed by industrial companies that would use the ideas in their processes, and possibly by companies that would productise the ideas. Those companies would take the ideas and tool prototypes and turn those into commercial, useful and reliable products and bring those to market for all companies to use, thus helping the whole industry get better tooled and more productive in their business. Also, the case study companies would continue to use their prototype systems, evolve them and work as influencers in their local markets, helping the adaptation of the ideas and new types of tools, even ones that utilise a new paradigm (such as model-based testing). Additional public dissemination during the project will be done in varying amount by semi-public and public seminars arranged by the project and non-scientific publications.

However, things don't always work that way. Figure 2 shows the often seen practice being not quite the ideal it is expected to be.

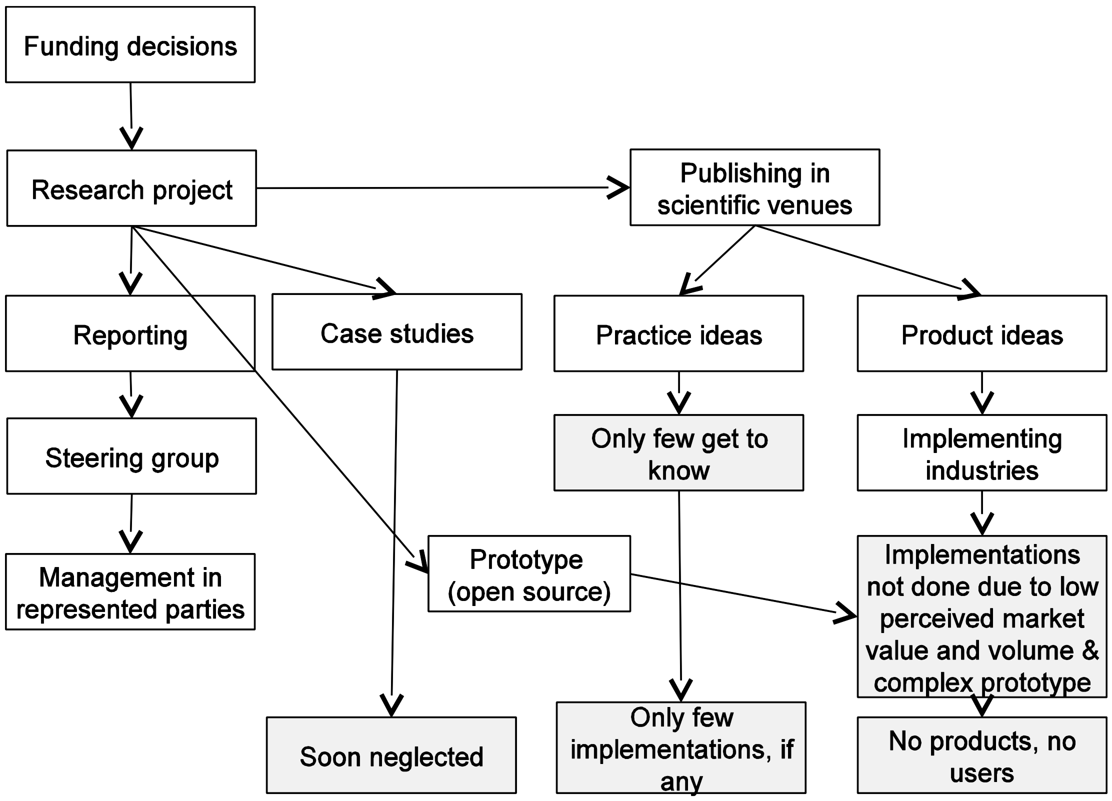

Fig. 2 Traditional model of research dissemination in practice [Editor: File: Fig_2.eps]

There are numerous problems in that type of research. Recently (2014-2015) there has been plenty of discussion in the media about the state of research publications and how they are rarely read. That is because they are mostly published in closed channels, and someone who would like to read a paper would need to buy it, which would involve dealing with company bureaucracy – getting permission to buy and having someone do the buying using the company's credit card, usually including creation of an account on the publisher's site. That means that the ideas are not communicated to the industry. It may be that of the participants of the project, the publications are read only their authors at most a couple of other people. This is a common problem in all research. Every couple of years this issue is raised up in media and social media, such as by Eveleth (2014) and Rehmer (2014). Also, in 2014 a former study made in the University of Indiana was raised up (Meho 2006) that states bluntly: "(…) as many as 50% of papers are never read by anyone other than their authors, referees and journal editors". Of course the actual numbers vary, but the situation is quite alarming. Now the research funding institutes are starting to take action and beginning to require open access publishing from the projects they fund. If Finland, Academy of Finland (2015b) requires that "(…) Academy-funded projects commit to open access publishing. We urge projects to make their research data and methods freely available. A plan on open access publishing and open data must be outlined as part of the research plan." This should help future projects in their dissemination.

Related to that is the problem that for historical reasons researchers do not publish their findings using blogs or other self-controlled media, and if they do, they will often just write about the existence of the results. Communication requires repetition to succeed. Re-publishing results is hindered by the first publishers wanting copyright to the papers, which is quite often understood too tightly to mean the substance and not just the paper as it is published (mostly the form and text verbatim). After publishing the results are forgotten, because there are no resources for implementing the results in practice or developing them further.

The participants of the project will get some information about results during their participation, but the industrial participants do not often have sufficient time to study the areas under research due to their daily jobs, not to mention participating in the research and any experiments in it. Sometimes the results are disseminated to the industry along with the researchers who at some point find a job in the industry. But for that to bear fruit, they should be able to find jobs where they can apply and develop the results further.

The division between researchers and practitioner exist, even though case studies and technical collaboration groups are sometimes used. Agile steering and serious critique on the approaches of the project may be lacking, making the results less applicable in practice than desired. Also, the clearly defined use cases in case studies do not allow for a proper "fuzzy front end" for the development.

As noted earlier, both the researchers and the practitioners need some critical elements for the collaboration to succeed: competences, motivation and an opportunity to do good work. In the traditional project model, the competences are based on choosing the right people for the collaboration. Steering groups do not necessarily have the best competences if the tool under development is disruptive (such as utilising a new testing paradigm). Direct contacts between researchers and practitioners are always needed to support technical decisions. For those contacts, case studies, workshops or something else, people should be selected who really have a motivation to take the issue at hand forward. Then we come to the opportunity part. Companies may not always have plans for implementing the developed technologies or using the tools under development and correspondingly the experts assigned to the project may not have sufficient resources allocated for collaboration with the researchers. Note that this kind of development would benefit from the participation of the best experts of a company and those usually have their time fully allocated for the company's own tasks.

The prototypes may be difficult to turn into products, for various reasons. They may even be made quite immature due to the requirements of the funding agency, which may just want the research to produce a proof of concept (a version that shows that the ideas are really working) and let the companies do the rest. Ideally, research could produce a developable open source product which others could just "fork" and start developing further. The products may be published under an open source license, but may be technically too immature to enable productising. They may have a complex architecture, a user interface with lacking usability, fragile source code that has not been tested properly, undocumented source code – all those are common types of “technical debt”. Sometimes the specification or the prototype implementation is simply done only halfway due to time running out in project and no mechanisms for continuing the work.

This situation is as should be expected, because researchers are not professional software developers (although they may be excellent programmers and have other product development skills). Clearly there could be opportunities for using the skills of software development professionals in the research project from early on.

Due to the nature of the prototypes, the productisers would usually need to rewrite the whole product and that may not be a viable option when the size of the market is small. This has been more problematic because a viable business case has been hard to find since the turn of century when open source tools started becoming mainstream in the testing tools market. This means that companies are not willing to pay for testing tools as readily as before, as they can today just download good tools from the web – just think of JUnit (JUnit 2015), Selenium (SeleniumHQ 2015), Robot Framework (Robot Framework 2015) and others.

3.3 Evolutionary version of the traditional model

These problems were understood by all participants in the dissemination system. Tekes, the Finnish Funding Agency for Innovation, (Tekes 2015), the most important funding agency in Finland for projects of this kind, took an evolved model seen in Figure 3 into use in 2011. The funding agency does not define the detailed structure of the model, just the elements of participation and the defining elements of managing the process. The rest of the elements are a cultural convention based on how the participants see such a process should be executed. Thus, the figure is the authors’ interpretation of the reality. It should be noted that there are various actual project types where the details will vary and the situation will also evolve every year. The synthesis here is adapted into a form that suits the setting of practical tool development oriented research projects and presents its essential processes and phases.

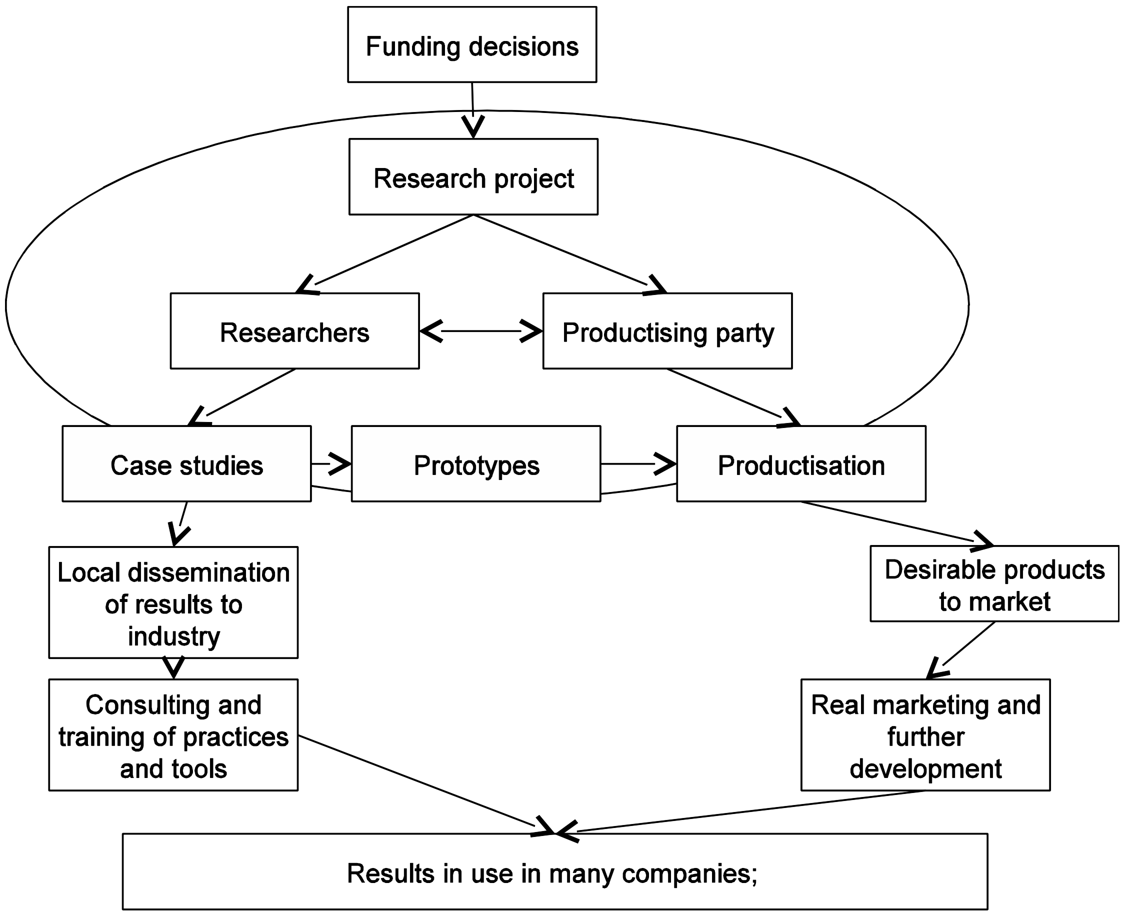
Funding decisions
Research project
Researchers
Productising party
Case studies
Prototypes
Productisation
Local dissemination of results to industry
Desirable products to market
Consulting and training of practices and tools
Real marketing and further development
Results in use in many companies;

Fig. 3 Integrated model of research dissemination gradually taken into use (as of 2011-) [Editor: File: Fig_3.eps]

The main idea for improvement is that a commercialising should be planned in the project from its start at least. That would mean that dissemination of the results and bringing the tools to the market would be considered from the very beginning instead of when the project has ended. During the recent years there has been an emphasis on finding new businesses and creating disruptive innovations to build those, so ideally the projects should find a unique new tool concept and not only a better version of an existing approach. Of course, finding novel concepts is exactly what is expected of research institutes.

For the commercialising to happen successfully, an expert party should be involved in the project, as the researchers are not expected to have such competences. Both expertise in business and plans for the actual commercialisation are needed. The expertise as such could be in the form of an external consultant. A company that would commercialise the product could take several actions during the project for that to succeed. First, it could do market research about customers' preferences. During the development, it should demand good architectural choices and documentation that would enable professional software developers to continue the work of the researchers. It could demand user studies and close collaboration with case study companies to ensure that the tool will meet real needs and fit in actual practices in companies. Furthermore, it should promote dissemination early and spread the word about the new approach in the market or the utilising communities. That company could be a software tool manufacturer or a new start-up built especially for commercialising the tool under development.

Of course this model may have its problems. It may lead into market oriented thinking overriding radically new ideas and innovations. The model is just the traditional model with some additions and simple additions do not address the deeper problems in actually carrying out constructive research. The expected collaboration is mostly positioned on the high level of the project, into its focusing and steering, but could ideally also influence the more detailed design. It should give additional value to planning what should happen to the prototypes after the research, and that is absolutely important. The change into this model is still ongoing. However, there could be alternatives and we will present one of them next.

4. A new approach: community integrated tool research

During the past ten years, the landscape of tool development has changed. It is now not that usual for a company to develop a testing tool and bring that to market. More often the case is that enthusiastic practitioners develop a tool as open source, or a company's internal tool is published as open source, and that tool slowly gains momentum and becomes a force in the market. Professionals need good tools with flexible licensing and perfect fit for their testing workflows. Self-made tools make more sense than before, because such tools start as tightly focused, small tools (compared with the culture of commercial tools). Publishing them as open source makes also sense. A community may grow, bringing more resources to the development of the tools, perhaps bringing the original developers new functionality "for free" that they might need for testing the next generation of their products. This idea of open sourcing the developed tool applies for research-developed tools too. For example, it has been a wish that a community could be created around the TEMA Toolset developed at TUT, but even starting the community would require monetary resources that do not currently exist.

The main motivations for this model are:

- A need for faster tool development and faster dissemination. Using the traditional models, tool prototypes are old-fashioned when they are implemented.

- A response to the current ways of producing software products, which is organic and led by communities and individuals instead of large software houses.

- A need to involve more people with the development of tools.

- A need to bring new interaction between research and industry, to handle the current complex world and to make the researchers more knowledgeable of the real needs of the tool users.

So, to bring the tool research up to date with the current way of developing software, we present a community integrated model of tool research, see Figure 4.

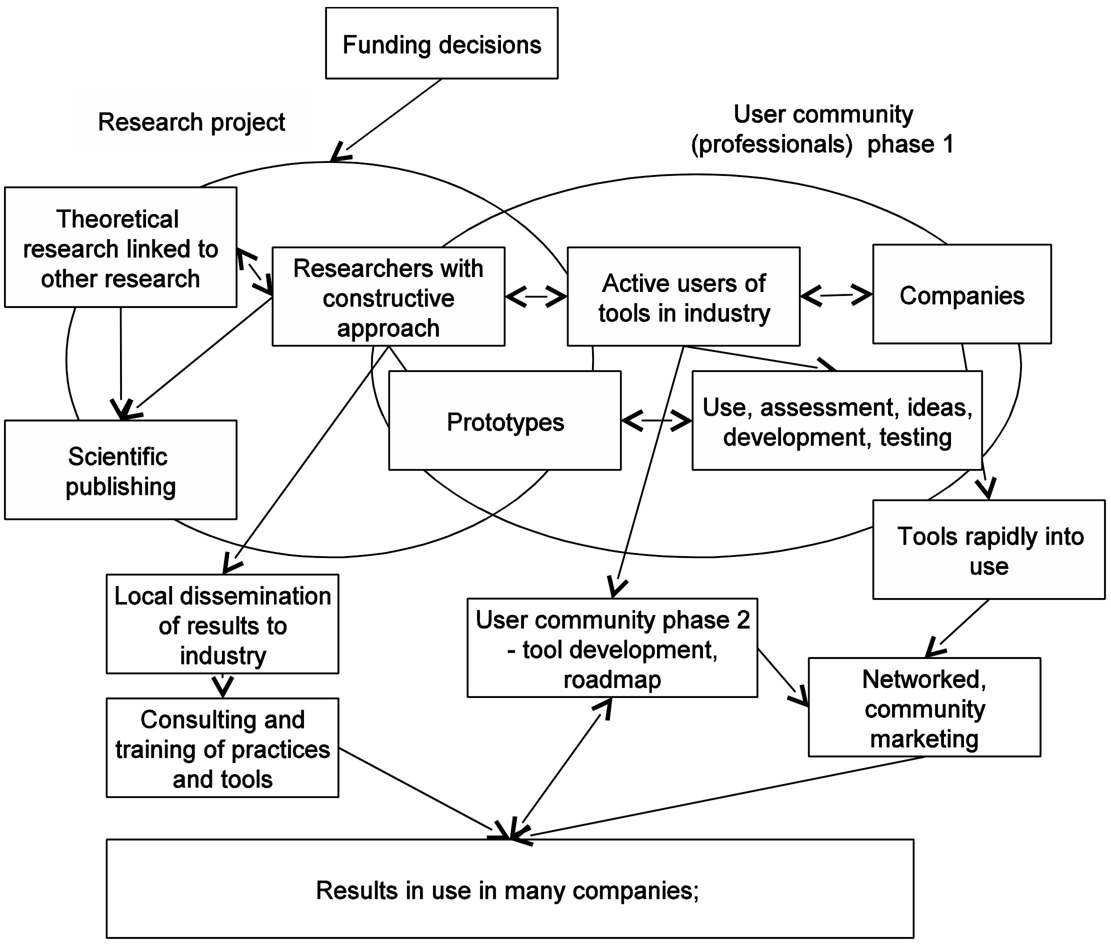
Funding decisions
Research project
User community (professionals) phase 1
Theoretical research linked to other research
Researchers with constructive approach
Active users of tools in industry
Companies
Scientific publishing
Prototypes
Use, assessment, ideas, development, testing
Tools rapidly into use
Local dissemination of results to industry
User community phase 2 - tool development, roadmap
Networked, community marketing
Consulting and training of practices and tools
Results in use in many companies;

Fig. 4 The community integrated model of constructive tool research and dissemination [Editor: File: Fig_4.eps]

The main idea is to involve future users with the tool research early on. Active professionals are invited to an open community that would be in communication with the researchers continuously. Ideas of the new approach to be implemented are discussed in the community between researchers and professionals. Tool ideas are assessed in discussions. The community can try tool demonstrators and prototypes and can test them, suggest improvements and provide alternative implementations. This interaction would lead into a natural growth of the tool in the right direction and for a group of first user population of the tool.

The first-level users are "innovators" in the traditional view of how innovations flow through populations (Rogers 2003). They will be active in other communities and the knowledge of the new tool would spread in an organic way. Usually, the kind or professionals who would participate are against commercialism, and therefore it is important that the development marketing of the new tool is done without the participation of commercial entities. They will provide a connection between research and practitioners' needs that is not predetermined, but allow for the agile emergence of application ideas and "testing" of various new opportunities. This will happen in situations where:

- Different kinds of people can connect with each other in a meaningful way and exchange ideas to and from research and practice. Personal connections (even virtual ones on a forum) will make it possible to transfer knowledge, opinions and tools fast and directly. Practitioners will not need to buy research papers or search the Internet.

- Communication can be continuous, but also rhythmic as needed, in the form of more structured session.

- Scope of collaboration can vary from group discussion to private communication.

- Ideas can be bounced between people until they find a spot where they can be developed, or else they will be put aside for a while (perhaps turning back into the deeper research to evolve them a little, fixing the deficiencies).

- Ideas can be co-created (co-designed and co-implemented) into working programs, using the best skills of both parties.

- There is an opportunity to start taking development further, perhaps just a small part, perhaps even on a larger implementation. The opportunity meets a want in both parties' minds, which is a necessary precondition.

- The collaboration is mentally open and free with no formal commitments and minimal exchange of money. When the commitment is personal, based on interest and personal needs and goals, it is true and will more easily lead to action.

Winning over the innovators will not guarantee the later success of the tool, but it is a necessary precondition for the success. The most critical phase is to get the tool into use by the larger population that requires a more mature product (good robustness, good usability, no tailoring in common tasks), and that is a very demanding task. One way to tackle it is stepwise development with the early users who know the practical demands of their testing workflows, practices, workplaces and business contexts. That knowledge can, with good designing, be extrapolated for the larger market.

Kilamo et. al. (2015) note that development work requires communication, coordination, cooperation and collaboration and this arrangement should provide a good environment for all of those. This is a mental space, but necessarily supported by virtual or physical spaces as well. Virtual spaces are common nowadays in the form of net forums and collaboration tools. Physical spaces can usually be arranged only rhythmically, due to costs, but obviously, if funding allows and there is a high volume of collaboration, even permanent spaces could be arranged. Communities of open source development have been studied recently a lot. The research setting in this article is quite close to those communities in its characteristic. Kilamo (2014) defines four essential main elements in the open source communities: people, purpose, product, policies and platform. It is essential to define the policies and make them clear for everyone, especially the rules regarding the interface between science and practice and the rules for influencing the product, and the roles of the participants. The platforms are the digital working and collaboration spaces (containing code repositories, discussion forums, voting tools, project desktops and so on). They are essential for managing the various works-in-progress, packaged configurations and enhancement suggestions. They should also provide a true working space, a true collaboration space.

The roles of the participants need clear definitions too. According to Kilamo (2014), often in the open source development the roles can be, from the most critical core to the less critical roles: project leader, core developers, active developers, peripheral developers, bug fixers, bug reporters and readers.

In the setting of this article, the roles can be similar, but with some critical differences. The project needs a leader. In research projects, there is always a project manager. In addition to pure developer roles, there needs to be roles related to the science part. Running the joint community really requires a role that focuses on coordination between science and implementation. We can call that role "coordinator". It would likely contain the role of community manager, as we don't want too many roles. Here we should only define the key roles for industry, which could be "development contributors" – people who really add to the product, and "advisors" – people who have no development skills or other deep technical competences, but who wish to monitor the progress closely, be part of the community and give advice. Of course, the non-developing pilot users in companies should be part of the community.

In summary, the list of roles could be like this:

- Project manager.
- Scientists.
- Coordinator.
- Core developers, who are most likely researchers, but could be from industry.
- Development contributors ranging from active developers to less active ones.
- Pilot users.
- Advisors.
- Readers.

Critical for the success of the research project are proper skill sets in the roles, so let's recap those. The imagined subject area being the development of a new type testing tool based on computational theories.

The project manager will manage the overall project, and thus, managerial skills are required. She may also be a development manager, if such special role is needed, and look after the development processes, deliverables and their deployments.

The scientists should be proficient in doing research and science, and in the theories applied. Of course, they need to know testing sufficiently well, to understand its goals and current ways of doing testing. Some researchers can stay purely on this side, in order to be able to really concentrate on the "hard" issues, such as new algorithms.

The coordinator will look after the activities of all parties, monitor the success of interaction and see that obligations are met. She may also be a community manager who manages user rights on digital platforms and helps solve conflicts.

The core developers will be tool builders by nature. They are people who can take ideas and turn them into usable programs. The core developers on the researcher side will be good on the theories, but also able to turn them into maintainable programs in collaboration with others. They need skills for eliciting opinions, requirements and ideas from the practitioners and skill in practical software engineering (such as understanding architectures, using version control and similar). They are the key interface between theory and practice. Core developers on the industry side will again be people who like to build things. They will in this case be people who build testing tools for themselves or their companies and are very much interested in implementing new ideas – if they see value in them. They will be good at practical software development. They know testing well and have deep understanding about the requirements for the tools in real life use. They wish to see things getting done.

The development contributors, who will not be developing themselves, will most likely be testing experts and tool users who are interested in the development and like to steer it and contribute with their knowledge about testing contexts: practices, tool requirements, tool integration, competence requirements and so on. Some will be focused on test automation, some are generalists, and some are something else.

The pilot users are people who will mostly just test prototypes in their organisations. Some of them will be development contributors at the same time and will act on that role publicly in the community's forums and privately in the context of their case. They will have the competence of a testing practitioner.

The advisors will be trusted people who will not participate actively, but give their views perhaps in reviews or by other requests. They should have a high competence level in the fields they are advising on.

The readers are just followers of the project, who may join the conversations at times, but mostly just follow it passively. Their competence will vary wildly and is not required for the success of the project.

This approach supports the most important general ideas in agile software development as described by Beck et.al (2001), Cockburn (2007) and Martin (2003). Collaboration is emphasised and seen as a critical element. The community does not work by contracts, but by voluntary participation and contributions. Reality of the implementation of the software is emphasised. For the practitioners, the theories have no value if they are not beneficial in real use. That will lead the researchers to assess the theoretical ideas and their application. This reduces the danger of researchers falling in love with ideas that have no relevance in the real world. Due to the continuous exchange of ideas, re-focusing the tools and tailoring their features is possible. The practitioners may even be able to do their own versions of the tools.

This is one way to achieve tools that a) meet real needs of their users and b) will have real users as early as possible and c) will have a committed community to lead the development into an industrial grade product. It is also possible to develop a whole activity system around the tool, not just the tool. Tools do not live in isolation. They need around them an overall context (formed or emerged) consisting of ideas about:

- Who will be using the tool? In the case of testing tools, are the users generic testers, special expert testers of some kind, or perhaps developers?

- For what purpose it will be used? Why in general would the tool be used?

- In what contexts, situations and environments it will be used? What are the main domains, test types, phases and situations in the development process and the information needs the tool is to fulfil?

- What are the actual ways of using it? What are the practical workflows like? How is the tool integrated into toolchains?

- What are the expected benefits of the tools? Is it better than some current approach? Can it replace some other tool type? Can it supplement something? Cost, effort issues etc.

- What are the associated risks of using it, including over-reliance on the tool?

Those ideas or specification will form the concept for the activity system. The collaboration will be crucial for defining all of these elements. Those are essential not only for the development of the tool, but for the dissemination: the thing that is disseminated is the whole concept of using the tool. That seems to be something that tool developers are only gradually learning.

Still, there is a division between research and the community. The research part will continue to carry out theoretical research in a project continuum and will not be hindered by running after the daily needs of the community. That research will be aided by the feedback loop of the community and the proof it provides from the latest implementations. The research group will need to have people who can start and run the community (to be a community manager) for this to succeed and that is a requirement and a skill set that does not often exist, but needs to be built. So, there are different roles in the research group; some more concentrated in the theoretical work and some more focused on the community and practical issues.

5. Assessment of the community integrated model

In this chapter we assess the proposed community integrated model from various viewpoints. First, we take a look into its general characteristics, then the expected effects on the quality of the tools is assessed, and finally we discuss the effects on the dissemination of the tool to the industry.

5.1 General characteristics of the model

The proposed model has some clear differences with the traditional type of constructive research. This approach is compared with the traditional one in Table 1. A comparison of this kind is illustrative and always somewhat a caricature.

Table 1 Comparison of the characteristics of the traditional model and the community integrated model

Characteristic	**Traditional model**	**Community integrated model**
Idea of the best way to do constructive research	Defined by the structures based on national scientific strategy	Depends on what we are researching. Depends on context. There are many best ways and the community can self-direct to it.
Word view in research	Researcher’s main paradigm defines the world	Systemic, multi-paradigmatic, holistic. Produces benefits for the community.

Characteristic	Traditional model	Community integrated model
Researchers' interests	Science, abstraction, tools for their own sake	How ideas and tools work where they are used in reality Putting theory into practice Testing a theory
Researcher participation	Isolated	Participatory & dualistic (partly isolated, partly in close collaboration with the community)
Lifecycle of research & construction	Linear waterfall process	Agile, iterative and incremental
Idea of how real products, based on the ideas, are built	Someone productises a prototype if it is good enough	With a continuous community process. Building a chain of working tools.
Time span from idea to first use	At least a year	As fast as possible
When are results tested	At the end of the project	Starting at the concept phase and in small steps during the project
Who defines success	Scientific circles	People who use the tools (practical success) Scientific circles (scientific success)
Idea of how ideas are disseminated	Research produces scientific information. Managers and experts in companies implement the ideas. People are exposed to ideas through processes and mature tools.	Through communities, collaboration, culture, incremental trying out, organically

We can see that the model should support a more modern approach to tool development and produce overall process benefits by enabling faster development into usable and assessable tool version. That is very crucial, as the main ideas – the theory and the tool concepts – can be tested fast. When the people who are collaborating understand the value of the ideas and have a mutual understanding of the main characteristics of the tools, they can be further developed for many use cases and in many environments. That is critical for any practical tool for today's market. A community-based activity can also self-define the working methods, but they are expected to form around some agile types, such as the ones based on rhythmic development as in Scrum (Scrum.org 2015). However, there is special emphasis on protecting the core theoretical research and finding good roles for many types of researchers where they can use their skills and orientation as fully as possible. The dualism of the model is an important distinction.

In order to further analyse the benefits of the model, we used a model for the context of human work used in action research (Engeström & Miettinen 1999) as a template. The core element of the model is the concept of an activity system that consists of seven essential elements. The (1) subject is the person who does the work, in this case the tester. She does the work on some (2) object, such as a system under test, which results in (3) outcomes, which often are information about defects or a tested software version. She uses some (4) mediating artefacts or instruments in that work. They can be testing tools, test systems, data and documents. The work is carried out in a (5) working community – a team, an organisation. In that community there is a (6) division of labour – the tester does some things, the managers and developers other things, based on their jobs and roles. All this is guided by (7) rules, which include non-written organisational norms, process instructions and cultural conventions. The idea is that in an activity system there needs to be a balance between these elements and if one of those is changed – by process improvement or some other reason – the others need also be assessed to regain this balance.

The analysis below shows the benefits and potential problems of the new activity model. The potential problems include things that can be influence in the design of any particular community.

The main benefit for the subject (which can be a person or a team) is that each person can utilise her best skills. There is a place for many kinds of people who complement each other. However, some scientists may find this model difficult, but this should be rare nowadays.

As for the mediating artefacts (the tools and methods used in the development), it can be seen that a richer set of methods and their users can be utilised to tackle the software engineering issues. Still, there is a danger that development gets too method and tool oriented with the cost of less work on ideas. Methods must be easy for the researchers so they can concentrate on their method set.

The object is the product, the tool under development. All aspects in the product get care from persons who are qualified for the task (e.g. theoretical parts from researchers, practical requirements and steering for design from practitioners), thus there is potential for good overall quality and a good basis for continuous development after the funded project. There is a danger that theoretical work can get diluted due to a too dominating community, but there are structures for handling that.

For the outcome, the goals of development, we can assess that the practical goals get the priority they need for the results to be usable. For that to happen, the practical goals need to be articulated and they need to be brought to critical discussion. That requires good coordination of the community activities and practices for gathering the needs and ideas from the practitioners. If the community has participants from various contexts, the goals will be balanced and will support dissemination to various types of use. Again, there is a danger that theoretical goals can get diluted due to a too dominating community.

An important element of the activity system is the community. A true community gives motivating energy to all participants. Participation in it is based on interest, not representation of a party. There are many roles available and all who are interested can find a suitable role. A well-formed community by default continues working together after the funded project stops. However, forming a community and maintaining it requires special skills and there should be very serious emphasis on that.

5.2 Quality effects on the tools produced

The quality should generally mirror that of generic open source communities, where there obviously are variations as some projects are more quality-driven than others and some have more proper quality practices in place (such as testing processes and acceptance processes for any modification). Technical quality (especially functional quality and maintainability) should start at prototype level and improve if the tool designs in various implementations converge. Usability is often seen a weak spot in open source products (Nichols & Twidale 2003, Andreasen et. al 2006), because the developers develop for themselves and not the average user and because usability experts don’t often participate in the projects. That problem remains here, and usability should get proper attention when the tools are gaining wider interest outside the first phase community. Any quality assurance practices should, however, have a secondary priority in relation to the factors that help in forming the community and can emerge in the community as it gradually forms.

There are many foreseeable benefits for the quality from the community integrated model compared to traditional research settings. The quality characteristics are assessed here using the ISO/IEC 25010 quality model (ISO/IEC 2011), which is a generic quality model and quite widely used for purposes of this kind.

The functional suitability should be improved due to various use cases in testing and development and the open source approach. Performance efficiency is about the characteristics about the relationship between the level of performance of the software and the amount of resources used. It should be improved due to various test environments. Compatibility between other software and hardware should be greatly improved due to various test environments. Usability is expected to be improved due to many developers, even if usability experts would not participate in the process. Reliability of the tool should be improved due to various use cases and the general volume of testing. Security improvements to the tools may vary, depending on the tool type, its security challenges and security skills of the participants. Maintainability should be improved, as the community development leads to modular design and control practices for it to succeed. Lastly, portability of the tool to different platforms should be improved due to various test environments and workflow integrations in those.

Overall, the effects on quality from the community integrated model should be very positive.

5.3 Effects on dissemination

The first requirement on the dissemination of the tool and its related practices is that they really provide better value than any alternatives. We noted in the previous section that better quality and thus greater value should be expected.

The dissemination of any open source tool is dependent on a strong community that develops the tool and makes it visible in the larger user population. That is not easy and can fail. Traditionally, a community is an afterthought realised when the project ends and more based on hope than any actions. But when the community is carefully built from the beginning, there is much more hope of making it a living, organic, growing entity.

When that succeeds, the dissemination is not done by "marketing", but by building the visibility and reputation of the tool in a grass-root manner. The participants of the community will use the tool in their environments and as they are likely to be active persons, even opinion leaders, their choice will affect others in their circles of influence. The participants of a digital community will be active in other Internet communications and sometimes in the physical activities of the tester community and will make the tool and its benefits visible in those channels.

A tool that has a growing user base will attract commercial actors who may build supporting services around it. Those include training for companies, general consulting, tailoring the tool for any special workflows and environments and other services. It is also possible to create special commercial versions of the tool, should its chosen license allow for it. Both types of commercialisation are an opportunity for existing companies and also for a start-up built by the researchers. This kind of dissemination suits well software products such as testing tools used and bought by experts. They relate today more to the voluntary recommendations in the communities than to the marketing claims of manufacturers.

All in all, the dissemination processes match the realities and opportunities of today's ecosystems for testing tools. Still, any successful dissemination is uncertain, but at least the project model presented by its main characters gets the dissemination process started, in a cost-effective way. Without starting it, there is no hope of succeeding.

6. Discussion

In this article, the current problems of constructive research are presented. They are rooted in the traditional scientific practices that have clear deficiencies from the viewpoint of how any software tools should be developed by current knowledge. Clearly, there are possibilities for a new type of research project model. The one presented uses ideas form open source communities and agile development and aims to improve the collaboration between researchers and industrial practitioners. That should improve the quality of the produced prototypes and produce a flow from theoretical ideas into various implementations. That makes it possible to thoroughly assess the new ideas in context and start wider dissemination, if the ideas are shown to be worth it – research always has a risk that the ideas may not be as good as the researchers thought in the beginning.

The model is not just a copy of how open source communities work. One of its main characteristics is the differentiation of the theoretic context and the practical context. Research always needs a possibility for researchers to concentrate on the hard technological issues. Without the differentiation, research is in danger of being diluted.

The main motivation for the development of the model is to produce a continuum of development where the prototypes will be turned into implementations that would spread in wider use by the community's processes. That should help in reducing the gaps in current practices. Yet we must remember that making a community work is hard work and requires special skills and it is never sure that a community will really form and grow. But when that succeeds, the new approach can spread into the industry and also offer opportunities for commercialisation in the form of services and specialised tool versions.

The ideas presented were based on reflection of past projects and shared in a wish that they would give ideas for those who are thinking of similar process developments and thus renewing the culture of constructive research in the field of software engineering. The ideas should also not be restricted to the development of testing tools, but should be applicable in the development of various kinds of tools.

7. Bibliography

Academy of Finland (2015a). [Web site of Academy of Finland.] http://www.aka.fi/en. [Checked 2015-09-09]

Academy of Finland (2015b). Academy of Finland's September 2015 call text out now – new guidelines for open access and open data. http://www.aka.fi/en/about-us/media/press-releases/2015/academy-of-finlands-september-2015-call-text-out-now--new-guidelines-for-open-access-and-open-data. [Checked 2015-10-20]

Andreasen, M.A., Nielsen, H.V., Schrøder, S.O. & Jan Stage, J. (2006). Usability in open source software development: opinions and practice. information Technology and Control, 2006, Vol. 35, No. 3 A, p. 303-312.

Cockburn, A. (2007). Agile Development – The Cooperative Game. Second Edition. Addison-Wesley. 467 p.

Crnkovic, G. D. (2010). Constructive Research and Info-computational Knowledge Generation. In: Magnani, L. et al. (Eds.): Model-Based Reasoning in Science & Technology, SCI 314, pp. 359–380.

Engeström, Y. & Miettinen, R. (1999). Activity theory: A well-kept secret, in Engeström, Y; Miettinen, R. and Punamäki-Gitai, R. L. (eds) Perspectives on Activity Theory (pp 1-15). Cambridge University Press.

Eveleth, R. (2014).Academics Write Papers Arguing Over How Many People Read (And Cite) Their Papers. Smithsonian.com. http://www.smithsonianmag.com/smart-news/half-academic-studies-are-never-read-more-three-people-180950222/?no-ist [Checked 1015-10-28]

Hevner, A. & Chatterjee, S. (2010). Design Research in Information Systems, Integrated Series 9 in Information Systems 22, DOI 10.1007/978-1-4419-5653-8_2, p. 9-22.

ISO/IEC (2011). ISO/IEC 25010. Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models. 34 p.

JUnit (2015). [Web site of JUnit.] http://junit.org/. [Checked 2015-09-09]

Kilamo, T. (2014). Essential Properties of Open Development Communities. Supporting Growth, Collaboration and Learning. Doctoral thesis. Tampere University of Technology, Publication 1194. 67 p. + appendices.

Kilamo, T., Leppänen, M. & Mikkonen, T. (2015). The Social Developer: Now, The, and Tomorrow. Proceedings of the 7th International Workshop on Social Software Engineering. p. 41-48. http://dx.doi.org/10.1145/2804381.2804388

Koen, P.A., Ajarnian, G.M., Boyce, S., Clamen, M., Fisher, E., Fountoulakis, S., Johnson, A., Pun, P. & Seibert, R. (2002). Fuzzy Front End: Effective Methods, Tools, and Techniques. In: Belliveau, Pul, Griffin, Abbie & Somermeyer, Stephen (ed.). 2002. The PDMA ToolBook 1 for New Product Development. John Wiley & Sons; 1 edition (4 April 2002). 482 p.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J. and Thomas, D.

(2001). Manifesto for Agile Software Development. http://www.agilemanifesto.org/. [Checked 2011-03-01]

Martin, R.C. (2003). Agile Software Development, Principles, Patterns, and Practices. Pearson. 529 p.

Meho, L. (2006). The Rise and Rise of Citation Analysis. School of Library and Information Science, Indiana University. arXiv preprint physics/0701012. 15 p.

Nichols, D.M. & Twidale, M.B. (2003). The Usability of Open Source Software. First Monday, Vol. 8, Number 1, January 2003.

Rehmer, D. (2014). Are 90% of academic papers really never cited? Reviewing the literature on academic citations.

http://blogs.lse.ac.uk/impactofsocialsciences/2014/04/23/academic-papers-citation-rates-remler. [Checked 2015-10-28]

Robot Framework (2015). [Web site of Robot Framework] http://robotframework.org. [Checked 2015-09-09]

Rogers, E. M. (2003). Diffusion of Innovations. Simon & Schuster International; 5th Revised edition. 512 p.

Scrum.org (2015). What is Scrum? https://www.scrum.org/Resources/What-is-Scrum [Checked 2015-10-28]

SeleniumHQ (2015). [Web site of Selenium.] http://www.seleniumhq.org. [Checked 2015-09-09]

Tampere University of Technology (2012). TEMA. http://tema.cs.tut.fi/index.html. [Checked 2015-09-09]

Tampere University of Technology (2013). RATA – Robot Assisted Test Automation. http://wiki.tut.fi/RATA. [Checked 2015-10-20.]

Tekes (2015). Tekes – the Finnish Funding Agency for Innovation. http://www.tekes.fi/en/tekes. [Checked 2015-09-09]

Matti Vuori is a Doctoral Student in Software Engineering at the Department of Pervasive Computing in Tampere University of Technology (TUT). The topic of his thesis is future competences in testing and quality assurance in the Finnish environment. He is a seasoned researcher, having previously worked at VTT Technical Research Centre of Finland where he in research projects produced software tools for accident statistics, risk analysis tools and tools for decision making, and researched among others development of future products and risk management. Between the research careers, he worked in the industry as a quality manager and consultant. During that phase was a steering group member of two TUT's research projects, so he has seen the two sides of the problem area at hand. He was also at one time a board member of COSS – the Finnish Centre for Open Systems and Solutions, and got acquainted with the problematics of open source development. Currently he is also the chairman of the steering group of Finnish Association of Software Testing and chairman of the board of Information Processing Association in Pirkanmaa Region.

Antti Jääskeläinen is a Post-doc Researcher at the Department of Pervasive Computing in Tampere University of Technology. He has worked on model-based software testing for close to ten years, on both modelling methodology and tool design, and was involved in the creation of the TEMA toolset. Over this time, he has witnessed research prototypes being forgotten once a project is over, but also successful stories in open-source testing tool development.