Working with Industry

Stories of successful and failed research–industry collaborations on empirical software engineering

Magne Jørgensen
Simula Research Laboratory
Fornebu, Norway
magnej@simula.no

Abstract—**The software engineering industry should be the laboratory of much, perhaps most, of the empirical software engineering research. Not only would this create a more realistic context and higher external validity of the empirical research, it would also ease the result transfer and make the results more convincing for the industry. Unfortunately, this is currently not the case. About 90% of software engineering experiments are, for example, conducted with students instead of software professionals as subjects. One reason for the lack of industry studies may be that an efficient and sustainable give-and-take-based collaboration between research and industry can be difficult to establish. The collaborations are frequently fragile, end before the research is completed, and lead to a waste of resources for both the researchers and the industrial partners. This paper presents stories and lessons learned from failed and successful research-industry collaborations. It has a focus on experience with the use of non-traditional collaboration types, such as payment to get industry participation in experiments, trade-based collaboration, lightweight collaborations at industry venues, and network-based collaborations. It is argued that empirical software engineering research should more often consider the use of alternative types of research–industry collaborations than those traditionally chosen.**

Keywords- collaboration, lessons learned, empirical studies

I. FIRST STORY: FREDERIK WINSLOW TAYLOR

Before moving into stories about empirical software engineering industry–research collaboration, let us look into one of the first and most famous research–industry collaborations on empirical studies in industry. The head researcher was Frederik Winslow Tailor, later known as the founder of scientific management, the company was Bethlehem Steel, and the collaboration started in 1898. Taylor, although he was from a wealthy family and was accepted to Harvard University, went into industry. He got an extensive industry background by starting as a machinist and working his way up the career ladder to become a shop superintendent at Midvale Steel. Later on, he opened a consulting practice specializing in the improvement of work practices in the manufacturing industry. From the beginning, he was strongly empirically oriented. He thought that most processes, including physical and mental work processes, could and should be analyzed to eliminate non-productive steps, and he advocated empirical studies on the effect of process and tool changes. He was consequently among the very first proponents of evidence-based engineering. His strong focus and research on reducing waste in work processes, which inspired Henry Ford and, in turn, Toyota's , and software engineering production processes, implies that he in many ways is the grand-father of lean and agile software development [1].

One of Taylor's empirical studies at Bethlehem Steel concerned the design of shovels. One may think shovels, used for thousands of years by so many, would get an optimal design just by trial and error. The shovels used at Bethlehem Steel (and elsewhere) were, however, far from optimal. Research on the optimal shovel design was non-existent, or at least not known to those who could make use of it, and both the managers and the workers at Bethlehem Steel seemed to be ignorant about the importance of shovel design. One of Taylor's main observations was that all the workers got the same shovel regardless of the density of the substance (e.g., gravel, sand, or coal) they were shoveling. This could not be optimal, Taylor thought, and he conducted a number of empirical studies on so-called *first-class workers* testing out different shovel designs. These studies found, amongst other things, that 9.7 kg was the optimal weight to put on a shovel and that the shovels should be designed, relative to the density of the substance, to fit that weight [2], see Fig. 1.

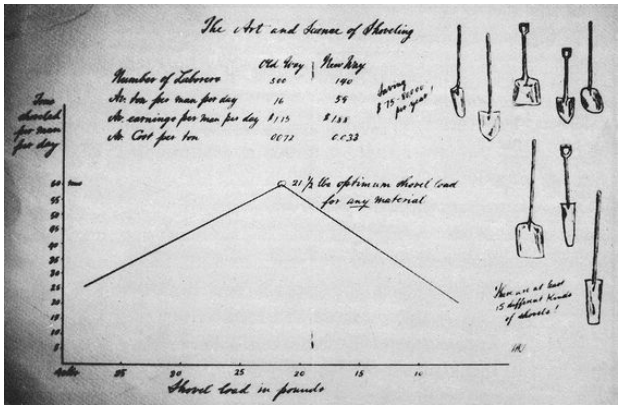

Figure 1. The optimal weight to put on a shovel (from Taylor's notes)

Taylor introduced about 10 new types of shovels, together with several management changes (often not

mentioned by those referring to the effect of Taylor's work), and claimed to have reduced the need for people shoveling, so-called yard laborers, from 400–600 to about 140 at Bethlehem Steel.

Newer studies suggest that the optimal weight put on a shovel for most workers is likely to be lower than what Taylor found (i.e., around 5–7 kg and dependent on factors such as shoveling speed and whether you want to minimize energy consumption or maximize production) [3]. This indicates that Taylor's *first-class workers* were stronger than the subjects used in later studies, which is seldom discussed when presenting the results and illustrates the importance of presenting and understanding the context of empirical results in industry.

Although Taylor's empirical research produced many productivity-enhancing results, he seems to have had problems convincing the management at Bethlehem Steel to use several of them. He got into problems with the labor union, and many workers were not happy with the changes he managed to implement. He was also accused of exaggerating the effects of the process changes and that his research was flawed [4]. However, even if the actual impact of Taylor's research findings is much less than he claimed, and his research designs were not optimal, there is no doubt that his ideas about empirical studies on work processes have had, and still have, a huge impact on industry [1].

What can we learn from Taylor's experience with industry collaboration and his success in having long-lasting industry impact? The research was conducted more than 100 years ago, and there are numerous differences between the mainly manual and low-skill work he studied and software engineering today. In spite of the differences, I think there are several observations regarding Taylor's work with industry that may be useful to reflect on and learn from:

- Addressing real, and preferably strongly felt, problems is key to getting industry-management priority and collaboration success. Taylor was originally hired by Bethlehem Steel to solve a strongly felt issue related to a costly machine capacity problem. He seems to have been able to use this problem solving as a stepping stone to do other, more interesting (at least from a researcher's point of view), research. The first step of a successful collaboration may be to identify shortcomings or issues strongly experienced as problematic by the industry itself.
- An extensive industry background, preferably with experience from all involved processes, seems to be a great advantage for success with collaborations. Taylor did not come equipped with an empirically based research method alone but with lots of knowledge about the work processes. Many of the problems he addressed were problems he had experienced himself. He was able to speak the language of industry and understand their problems in depth. That is not to say that Taylor understood all aspects of value. His ability to understand the human and psychological aspects of the workers was, as illustrated by many stories about him, not well developed. This seems to have limited his impact and explains why he, and partly scientific management itself, got the reputation of considering workers to be machines.
- The opportunities to support industry's work processes are numerous. The fact that a process has been in use for many years is no proof of that it is close to an optimal process. There are numerous factors that could prevent a process from improving by trial-and-error or by on-the-work feedback alone.
- Close interaction with industry when conducting the research is essential to getting high external validity of the results, being convincing, and easing the transfer of the results to those making use of them. For example, there seems to have been some previous research on shovel design by the physicist Charles-Augustin de Coulomb. That research seems, however, to have lacked the close collaboration with industry, not being shared with them, and, as a consequence, was unable to impact their shovel design and work processes.
- Management and/or worker resistance toward believing and making use of empirical results on work processes is normal. Every change is a potential threat to someone, and it is always possible to find limitations and problems with how empirical studies are conducted and use these to argue that the results are not valid. A successful impact of results from empirical research may require that at least those making the decisions benefit from putting the results to use. In Taylor's case, the managers frequently benefited from many of the suggestions and, partly for this reason, many of Taylor's findings were implemented.
- Marketing the results is essential. Being very careful not to exaggerate or over-generalize while presenting results (which is considered a good researcher habit), may be a disadvantage for their impact. Taylor was very good at marketing the results, not modest in presenting their effects, and spent considerable time on convincing the industry to put his results to use. Most of his publications (mainly books and presentations) aimed at industry, not academia. His impact on academia and education was mainly through his lectures at Harvard, where his course on scientific management was a cornerstone beginner's course of one of the first business management degrees. Presenting at industry venues, writing about the results in magazines and text books, and continuous support of industry management over years may be needed to have an impact on industry processes. It may be only on rare occasions that a single result changes industry processes.
- Changes in work processes may have negative side-effects. Taylor's results may have contributed to treatment of the workers as salary-aiming objects part of a large machine. Collaborating with industry should include making ethical reflections on how the results will be used and their side effects. Using valid research results in negative ways for workers or consumers is not a new thing and will happen again.

II. SECOND STORY: CONSULTANCY-BASED COLLABORATION BUILDING AN EXPERIENCE DATABASE

More than 15 years ago, I was part of a research project successful in recruiting software companies to collaborate on software process improvement and in receiving financial funding. My task was to collaborate with one of the companies in establishing an experience database. The company's official goal was to implement an experience database that enabled continuous learning from their software projects. My main research goal was to analyze the potential positive effect of process improvement on experience databases and understand the learning mechanisms involved. It seemed to be a good alignment of industry and research goals.

As part of my previous industry experience, I had already been in charge of implementing an experience database in another company. I had also been working as a programmer, a project manager, and a general manager, so I felt that I understood the context and the work processes involved pretty well.

The work with the experience database in the new company went well. The project included three engineers from the company itself and three researchers. A minor worrying factor was that it soon became evident that the company's real motivation was to be certified at CMM-level 3, which was important to gain and keep US clients. This meant that our research goal—to find out to what extent the experience databases would lead to positive effects on their projects—risked being a lower priority for the company than we originally hoped for.

Less than one year after we started, the first version of the experience database was released. It already included experience from many previously completed projects, collected and included by us. The management support was good, probably due to the importance of the CMM certification. It was decided that it should be mandatory to include experience from all the finished software development projects in the experience database. The experience database was already in use by several new projects to support planning and cost estimation. Strategies for the company's process-improvement work were partly based on analyses from the information in the experience database. The collaboration atmosphere was good, and we, as researchers, had gained a lot of goodwill through our support. We were now on the verge of starting to measure, we assumed, the positive impact of using an experience database. We had collected data about previously completed projects, our baseline data, and were now waiting to collect information about the projects that had used the experience database in their planning and to find out to what extent the experience database had had a positive effect. After more than one year of extensive company support, in many ways consultancy work for the company paid by the research council, we were now starting to hope for research results. This really looked like a successful industry–research collaboration.

Then, as is perhaps pretty normal in industry, something happened. In this case, the company had to face serious cost cuts. Consequently, central people working on the experience database project were replaced or even lost their jobs, and the management support was not there to give the project as high a priority as we previously had. We quickly learned that an experience database without strong management support and enthusiastic, internally employed people quickly gets a low priority, and its use fades. The fact that use of the experience database was mandatory meant nothing in reality, given the lack of management priority. Very soon, the experience database was hardly used at all. Suddenly, the collaboration was not a success anymore, neither for us nor the company.

What I learned from this is that:

- It is risky to couple research goals closely to longer-term industry goals (longer term is, in this context, more than one year ahead). During a period of 1–2 years, many companies will have reorganized, cut costs, or done other things that may lead to down-prioritizations of non-core activities, frequently including research collaborations. As a consequence, it is risky to plan to first invest a lot in building goodwill and then, 1–2 years later, be able to harvest research results.[1]
- In addition to the longer-term research goal, it is perhaps important to look for shorter-term research results when collaborating with industry. In this collaboration, we were able to collect information about a few projects, which we later used to generate hypotheses and, to a minor extent, as part of later research analyses. Not much support for our research, but better than nothing.
- It is sometimes very hard to make an industry–research collaboration so sustainable that it survives strong company turbulence. A risk analysis before a collaboration should examine how likely turbulence is to happen, and risk management during the collaboration should aim at reducing the risk of the collaboration getting down-prioritized. Support from the management group is essential. Relying on the support of mainly one person in the management group, as we did, is risky.
- Demonstrating company benefits from the collaboration early and often could help keep management support. We did this through the use of the content of a pilot version of the experience database as input to their strategy work. This was possibly a reason for the good support we got, at least until the organizational turbulence took place.
- There is a need for researchers with industry experience or at least people who speak the language of the industry and understand their problems well in this type of collaboration. Even if the research grant or university is paying your salary for the work done for the company, and not the company itself, there is a strong need to be considered competent and useful by the company to get the research done properly and for the collaboration to work. We were partly successful at this, which contributed to us getting as far as we did.

[1] Although I thought I had learned, I have experienced the same reason for collaboration failure twice later on. Sometimes it is hard to separate wishful thinking from what is the true risk of such research-industry collaborations.

III. THIRD STORY: THE NEED FOR DIFFERENT COLLABORATION TYPES FOR THE SAME RESEARCH TOPIC

Typically, the uncertainty of effort estimates is assessed through minimum–maximum effort intervals (e.g., that the effort to complete a software project with 90% confidence is 100–200 work-hours). The effort interval is then used as input to plans, budgets, bids, and other purposes. My research on this topic, which lasted for several years, required several quite different types of industry collaborations:

- To better understand the problem—in this case, the lack of realistic assessment of the uncertainty in effort—I needed to observe real-life estimation work. This required a collaboration on collecting project information, interviewing project leaders, and observing estimation and uncertainty assessment in practice.
- To evaluate ideas on how to improve, I needed to conduct controlled experiments. This required software professionals to collaborate as participants in controlled experiments comparing traditional and new methods and tools.
- To really see if ideas from the promising results of the experiments led to improvements in practice, I needed a collaboration in which I trained people in the use of the method and where they used the new methods in real-life industry situations.

Problem identification and understanding: Consultancy-based collaboration

The main challenge in the collection of project information is often recruiting companies with relevant data of acceptable quality. This is a hard part, and requires a good network and/or willingness to try and fail. After a few failed attempts, a couple of companies were identified that both had the required information and were willing to share it. The collaborations were initiated with focus on my research goals, and I got access to written project documentation and was allowed to interview selected project leaders. The companies' benefits from the collaboration were that I gave them a report on their estimation processes and suggestions on how to improve them (i.e., sort of consultancy work) and a seminar on effort estimation. The collaboration was rather short-lived, only 2–3 months. The main research finding, supporting findings in other domains, was that the effort prediction intervals were much too narrow to reflect the actual level of uncertainty, (i.e., there was a strong level of overconfidence in the accuracy of their effort estimates) and some indications on reasons behind overly narrow intervals. This motivated further research on this topic.

First phase of method evaluation: Payment-based collaboration

Based on own ideas and ideas from other domains, I wanted to test several alternative methods potentially beneficial for a more realistic assessment of effort uncertainty. This required a set of experiments with both students (pilot experiments) and software professionals. While it would have been possible to create a research–industry collaboration based on giving industry consultancy services or process evaluation feedback in return, I determined it would be more time efficient (based on previous experience) to base the collaboration on paid participation.

A number of software professionals were hired and paid on an hourly basis to complete the estimation work using processes and in contexts similar to what they normally experienced. Typically, the software professionals were allocated to either a control group (work as normal) or a treatment group (use the uncertainty assessment methods devised by us). To keep the cost down, software professionals in low-cost countries, typically from Eastern Europa and Asia, were hired for participation in the experiments.

The initial challenge of this type of payment-based collaboration, which is not common in empirical software engineering research, is getting acceptance for spending research grants or internal funding on paid industry participation. In our research institute, the understanding, at least internally, was that this was an acceptable use of research grants. One of our arguments, which seems to have been convincing, was that our laboratory *is* the software industry work processes, and it should be just as natural to spend money on our laboratory as it is to spend money on buying expensive laboratory hardware or software to do research. In many situations, hourly payment for work is the natural context of software professionals. Not paying for software development or project planning work, but instead trying to get volunteers to use new processes, would be an unusual situation limiting the external validity of the results.

Several, but not all, of the experiments conducted this way were successful in terms of research output and in what we saw as discoveries of innovative, realistic advancements in effort uncertainty assessment processes [5].

Use of a new method in real life: Payment-based collaboration

To be really convincing, it would be helpful to test out new methods on real projects in real-life situations. The challenge was, consequently, to make companies use the method. Unfortunately, it was difficult to find companies that felt an urgent need to improve their uncertainty assessment processes, and it was not likely that companies would hire us or spend effort on implementing new methods just because we claimed to have some promising experimental results.

As with the earlier studies, the solution was found in the use of payment-based collaborations. Two companies that currently used the traditional effort uncertainty assessment method were identified. These companies were willing to evaluate the new method on their projects, given that we compensated them for the extra work involved in making the new method work and the work on logging the extra information we needed.

The evaluation process we agreed on was as follows:

- We were notified every time they planned to estimate the effort of a new project, or larger maintenance task. We then interviewed the person in charge of the estimation about essential project or task characteristics.

- We randomly allocated the project to either a control group (following their traditional process for creation of minimum–maximum effort intervals) or the treatment groups (following our new uncertainty assessment method). Those in charge of the estimation work in the treatment groups were given training and asked to follow our instructions for the process and the documentation of the use of it.
- We were notified when the projects or maintenance tasks were finished and interviewed the project leader about the effort used and other project-outcome information.

As can be seen, this is a randomized controlled trial (RCT) in a field setting. Such experiments are rare in empirical software engineering. In this case, because we paid for the company's extra work involved, it was not only possible but quite easy to manage a RCT. RCTs in field settings are, in many ways, the most powerful type of study and are considered the *gold standard* of empirical study designs in many domains. Software engineering should, I think, aim at having more such studies.

The experimental evaluation turned out to be successful [6]. Those following our new method had more realistic assessment of the actual uncertainty of the effort usage. Measured as immediate industry impact, the collaboration was not a success (i.e., the companies did not start using our method). The results have, however, been used in other contexts and have had some influence on other company processes and even national governmental estimation guidelines.

Lessons learned

Several lessons may be drawn from the various collaboration types used for this research topic:

- The efficiency of a type of research–industry collaboration depends, not surprisingly, on the stage of the research and the type of research problem addressed. Regardless of collaboration, a situation where both the researchers and the industry have expected gains higher than the expected costs is essential.
- Research–industry collaboration does not require shared gains or goals. Instead, in many collaborations, the gains of the industry will, and should, be quite different from those of the researcher. The gains of the industry could be consultancy work by researchers, training, or access to the research results but could also be ordinary payment for effort spent.
- It should be just as natural to base a laboratory of empirical software engineering studies on payment to software professionals as it is to base the laboratory of, for example, data analytics research on payment for hardware and software equipment. Funding for payment of software professionals should be part of many applications for research grants in empirical software engineering.
- Randomized controlled trials in field settings, which are considered to be the gold standard of empirical research in many domains, is clearly feasible for software engineering research. More research studies should consider using this study design. This study type makes the results more convincing to the industry and potentially results in larger impacts. Payment-based industry participation may be needed to enable such studies.

IV. Fourth Story: Trade-Based Collaboration for Research on Software Project Bidding

One of the most undoubtedly successful collaborations in human history is trade (i.e., collaboration in which one party buys and another party sells a service or product). One of the main advantages of this type of collaboration is that it is relatively clear what the different parties take and give. Here is a story about one such research–industry collaboration.

Bidding with negotiation is commonly observed in contexts where a client searches for and selects a provider to complete a software project. One element of the negotiation is the scope (delivered functionality) of the project. A client may want less or more scope, dependent on the price (i.e., based on a cost-benefit analysis). From interviews and previous experiments, we hypothesized that a reason for some over-optimistic bids, often connected with quite problematic software projects, was that the original bid was given based on a larger specification and then reduced due to removal of some of the requirements.

To test this hypothesis in field settings, we invited offshoring companies, again from low-cost countries in Eastern Europe and Asia, to bid for a software development project specified by us. The software development project's goal was to develop software that we planned to use in further experiments on software bidding. The participating companies were randomly allocated to one out of two groups. One group of companies first made a bid based on the full requirement specifications and were then asked to update the bid for only a subset of that functionality. The other group of companies first made a bid based on the reduced specifications and then updated the bid for the full requirement specifications. In total, 29 software development companies ended up giving their bids for the two variants of the project.

The experiment was successful, and we found that different bidding sequences led to large differences in bids. For example, the companies that started with bids based on the full specifications and then updated their bids based on the reduced specifications gave, on average, 40% lower bids for the reduced specifications than those starting with bids for the reduced specifications [7].

The companies were not informed about the research-oriented purpose of the bidding process. They behaved as they would in other project-bidding processes. Not informing them about the main purpose would be an ethical problem if we did not really intend to select any of the bidding companies to develop the software. To ensure fairness, especially given the high number of bidders, we selected and paid not only one but four of them to develop the software. Furthermore, to ensure fairness regarding bidding group allocation (i.e., one group of companies was manipulated to

give higher bids), we decided to select two companies from each treatment group. The selection within each group was based on traditional criteria, such as price, ability to respond to the request, and their documented competence.

The collaborations with the companies, both in the bidding and the construction phases, were successful. To a large extent, this was experienced to be a result of the type of collaboration chosen. There are commonly accepted rules for trade and relatively high clarity about expected deliveries in bidding rounds and software projects. In other words, everyone knew what to do and what they could expect from the other party—a good starting point for collaborations.

Lessons that may be drawn from this industry–research collaboration include:

- Trade-based industry–research collaboration, where the research party acts as a normal client, may in some cases be suitable for successful research studies. A clear advantage compared to many other types of collaborations is in the clarity of the give-and-take expectations of both parties.
- There are ethical issues to consider in every industry–research collaboration, especially when the industry participant is not fully informed about the main motivation of the process they are participating in. The main factor to consider may be to what extent there is any (even potential), harm done to the participants, including waste of their time.
- Acting as a client, beyond having additional research goals and having a university affiliation, involves careful research design thinking. For example, it is important to create a context in which the additional information needed is a natural part of the process or, if that is not possible, to include extra payment for work not normally done. Most providers will google your organization and you as a contact person. It is not a good idea to try to act as something you are not. Instead, try to use the context you are in to motivate the requests you make. It is important to the validity of the study to actually behave as a client. It may be necessary to hire other people to represent you as the client to ensure realism of the client role.

V. Fifth Story: Collaborations Based on Seminar Participants and Industry Networks

Over the last 10 years, I have conducted quite a few shorter research experiments and surveys at industry seminars and training courses for industry. Of the numerous industry participants, who in most cases did expect to participate in a research study when attending, only a couple of them have given me negative responses, these responses have been pretty mild, and many more have given very positive responses. My experience is that software professionals find it interesting to participate in such experiments and surveys and that this is a great collaboration opportunity for much of our software engineering research.

The following four conditions are, however, essential to make this type of collaboration work:

1) The participants should always get something back. In my case, this involves giving them some of the most relevant research results from the experiment in which they participated, usually in the end of the seminar or course, or, worst case, in an email the day after. Before I found web tools that made such responses quick and easy (currently I'm using the tool Qualtrics), some of my colleagues or I quickly analyzed the paper-based responses during the seminar or course.
2) The topic of the experiment or survey, at last part of it, should be of real interest to them and related to the topic at the seminar or course. If the industry relevance of your main research question is not obvious, it is necessary to include other questions that are more relevant.
3) The experiment or survey should be short. Experiments or surveys should, in the context of seminars and courses, seldom take more than 15–20 minutes to complete. Surprisingly, many research questions may in my experience be properly addressed in that short time. Clearly, the short time frame limits the experimental tasks and number of questions in surveys. Stay focused.
4) Ensure anonymity of the participants' responses. Anonymity is not always needed, but my experience is that it increases the quality of the responses and sometimes is even required to make people willing to share experience.

Recently, much of my industry collaborations of this type are done through an industry-dominated collaboration network.[2] The network regularly holds about four seminars each year, usually with about 100 industry participants, on various software management topics. On every seminar, we spend 15 to 20 minutes on empirical research with the seminar participants as subjects. The use of this network to complete experiments and surveys has so far produced several interesting research results (e.g., results on what makes IT projects successful in delivering client benefits) [8].

The network has also been very useful in recruiting members for collaboration on other software project topics and the number of members is still increasing. Industry collaboration through this type of network can be very useful, but also requires substantial administration and continuous focus on how to make it useful for industry to participate. Repetitions and lack of renewal lead network-based collaborations to die quickly.

The main lessons from using seminars and courses, lately through the use of an industry network, to enable collaborations may be summarized as:

- There is no reason to be reluctant about using software professionals participating in seminars or courses for surveys and small experiments. Just remember to give them relevant result feedback, and do not spend more than 10 to 20 minutes of their time. Use experiment/survey software that helps you with quick result feedback to the participants, preferably live or at the end of the seminar or course.

[2] The network is called the HIT-network, is based in Oslo, Norway, led by me, and focuses on IT-management (http://www.hitledelse.com/)

- Many interesting research questions do not require more than 10 to 20 minutes of time per participant. Sometimes the short time frame may even help to focus the research and avoid the inclusion of too many hypotheses, which by the way typically lower the statistical power of each of the hypothesis tests in a study.
- Often, it is of more value to design a sequence of connected surveys or experiments, rather than conducting one large experiment. Lightweight, shorter, experiments in an industry network is a good means for this for purpose.
- In my experience, what the software professionals remember best from the seminars, even years later, are the results from the empirical studies. This is perhaps not surprising, since they were part of it (e.g., the experimental treatment had an effect on them). The pedagogic effect of such experiments to affect the software industry is consequently an additional argument in favor of using it.

VI. CONCLUSIONS

When discussing industry–research collaboration, it is easy to focus on the type of collaboration exemplified by Frederik Winslow Taylor (i.e., research in which the researchers come to a company, a problem is identified, and the researchers try to solve the problem using research methods and gain knowledge that can be reused in other companies and/or that will lead to innovations). For lack of a better term, this is called *consultancy-based collaboration* in this paper. While clearly being useful, this type of collaboration frequently involves a high risk of investing much research effort without getting research results, especially when the research goal needs long-term investments before any results can be collected.

Other types of collaborations are, in many research contexts and for many research questions, more suitable and perhaps under-utilized in empirical software engineering research. The stories told in this paper illustrate the usefulness of a wider set of types of collaborations, including collaborations based on:

- Payment (or other rewards) of software professionals to participate in empirical studies.
- Payment (or other rewards) of software companies for the extra work required to invest in structured evaluation of process changes and for logging required information.
- The researcher acting as a client (i.e., a purely trade-based collaboration).
- The use of software professionals participating in industry seminars and networks for shorter experiments and surveys.

The core elements of successful industry–research collaboration in software engineering are, as far as we have experienced, the same regardless of collaboration type: i) A reasonable balance of give and take for both the researchers and the industry participants, ii) Researchers with industry competence and a good understanding of the processes, products, and organizations they study, iii) Research questions and results of perceived relevance for the industry, and, at the same time, iv) A research question that is relevant and interesting from a research point of view. The importance of these core elements has not changed since Frederic Winslow Taylor conducted his empirical studies on work processes and tools in Bethlehem Steel in 1898.

Many types of research–industry collaborations are not covered in this paper, and most of what is presented here as lessons learned are not based on solid empirical research but, rather, on the perceived lessons learned from my own experience from more than two decades of empirical software engineering research. It is easy—and valid—to criticize the lessons learned as based on too few observations or lack of a control group. In spite of these limitations, I think that it is worthwhile to consider the collaboration types illustrated in this paper when starting new collaborations on empirical software engineering research. Hopefully the lessons learned and stories will inspire researchers to consider a wider variety of collaboration types and study types than is currently the case in empirical software engineering. In any case, good luck!

REFERENCES

[1] Savino, D.M., "Frederick Winslow Taylor and His Lasting Legacy of Functional Leadership Competence." *Journal of Leadership, Accountability and Ethics*, 2016. **13**(1): p. 70.

[2] Taylor, F.W., *The Principles of Scientific Management*. 1914: Harper.

[3] Freivalds, A., "The Ergonomics of Shoveling and Shovel Design—A Review of the Literature." *Ergonomics,* 1986. **29**(1): p. 3-18.

[4] Stewart, M., *The management myth: Why the experts keep getting it wrong*. 2009: Andrea Garcia.

[5] Jørgensen, M., K.H. Teigen, and K. Moløkken, "Better sure than safe? Over-confidence in judgement based software development effort prediction intervals." *Journal of Systems and Software*, 2004. **70**(1-2): p. 79-93.

[6] Jørgensen, M., "Realism in assessment of effort estimation uncertainty: It matters how you ask." IEEE Transactions on Software Engineering, 2004. **30**(4): p. 209-217.

[7] Jørgensen, M., "The effects of the format of software project bidding processes." *International Journal of Project Management*, 2006. **24**(6): p. 522-528.

[8] Jørgensen, M., "A survey on the characteristics of projects with success in delivering client benefits." *Information and Software Technology*, 2016. **78**: p. 83-94.