8th Conference on Systems Engineering Research
March 17-19, 2010, Hoboken, NJ

Paper 1569249279

# Observations from the Industry-as-Laboratory Research Project Darwin[1]

**Piërre van de Laar**
Embedded Systems Institute
TU/e Campus, Laplace-building 0.10, Den Dolech 2,
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
pierre.van.de.laar@esi.nl

## Abstract

In 1993, Potts observed that research done using the research-then-transfer paradigm failed to influence industrial practice. Potts advocated that by applying the industry-as-laboratory paradigm, research can make a significant practical difference. Darwin is an industry-as-laboratory research project that is almost finished. In this article, we describe observations and evidence from the Darwin project related to Potts' hypothesis. Furthermore, we describe observed challenges in our industry-as-laboratory research project, together with how we addressed them.

## Introduction

According to (Potts, 1993), the influence of software engineering research on solving industrial problems is currently little. He proposes to use the industry-as-laboratory paradigm instead of the research-then-transfer paradigm to improve the situation. He advocates that the industry-as-laboratory approach offers several benefits due to intervention coming early on

- The definition of the problem to be solved comes more directly from a detailed understanding of the application environment.
- There is less emphasis on a separate technology-transfer phase. Instead, interim results are applied to practical problems throughout the research process. Feedback shapes later investigations.
- As research progresses, it becomes increasingly problem-focused.

Furthermore, the industry-as-laboratory approach shows features, i.e., scale and complexity, unpredictability, and dynamism, that a laboratory experiment cannot exhibit. And finally, in the industry-as-laboratory approach technology evaluation and transfer processes are underway at an earlier stage of the research programme which considerably reduces the transition time from research innovation to state-of-practice. Potts' hypothesis is, however, not proven yet.

The Embedded Systems Institute is founded by a number of large Dutch high-tech companies and by the three Dutch technical universities. Since its foundation in 2002, the Embedded Systems Institute applies the industry-as-laboratory paradigm not only for software engineering research but also for systems engineering research. Darwin is the fifth software and systems engineering research project started by the Embedded Systems Institute. Darwin is a collaborative project between Philips Healthcare, Philips Research, five Dutch universities (Delft, Eindhoven, Groningen, Twente, and the VU University of Amsterdam) and the Embedded Systems Institute. The project started end of 2005 and will run until the end of 2010. The size of the staff of the project is equivalent to 20 full-time people, and includes 10 PhD students. In the Darwin project, we aim to solve an industrial

[1] This work has been carried out as part of the Darwin project at Philips Healthcare under the responsibility of the Embedded Systems Institute. This project is partially supported by the Netherlands Ministry of Economic Affairs under the BSIK program.

problem. In particular, the goal is to develop specific methods, techniques, and patterns to improve the evolvability of product families within industrial constraints and while maintaining other qualities. See also Figure 1.

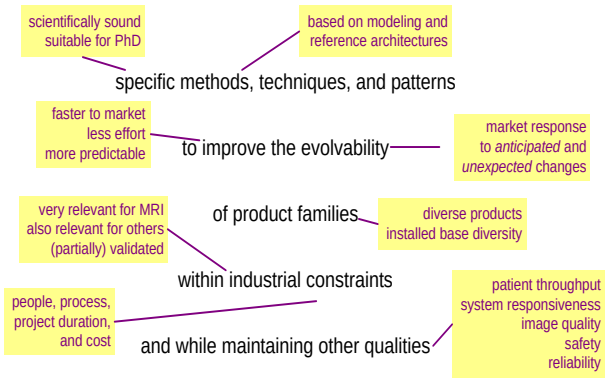


**Figure 1. The goal of the Darwin project.**

Why would companies and universities cooperate in a single project to solve an industrial problem? This question is especially relevant, since it is well-known that a gap between them exists. See, for example, (Zelkowitz et al. 1998) (Muller 2005) and (Punter et al. 2009). Before answering this question, we first elaborate on the gap between industry and academics and problem solving in general.

**The Gap between Companies and Universities.** In this section, we describe the goals and needs of companies and universities. The gap between companies and universities can be understood by the difference in their goals and needs. Furthermore, mutual understanding of each other's goals and needs may help to bridge the gap (Muller 2005) or, at least, to understand challenges in cooperations.

A company is a business institute. Typically, the goal of a company is to generate a sustainable profit by developing and maintaining a competitive product or service portfolio. To realise this goal, a company needs to regularly introduce product improvements that meet changed and increased customer requirements. For this, multiple disciplines must be combined, such as marketing, development, sales, and service. Many projects in companies have heterogeneous teams. Interactions in these teams and with (potential) suppliers and customers tend to be multi-disciplinary.

A university is a publicly funded organisation. The goal of the university is to extend and distribute the body of knowledge. To realise this goal, a university needs to perform research and provide education to bachelor, master, and PhD students. For this, mono-disciplinary excellence is needed. Many projects in universities only contain one PhD student and his supervisor. Interactions in these teams are mono-disciplinary focusing on a single technology[2]. Interactions outside the team with customers typically only happen when the technology is considered mature.

**Problem Solving.** Solving a problem is a creative, iterative process. This process includes problem analysis and technology selection from the available body of knowledge. Problem analysis is difficult, since real-world problems are seldom where you expect them to be (Potts 1993). Even worse, one might not even be aware of the problem. For example, people believing the earth was flat were unaware of the possible inefficiency in their travelling, since on a globe the shortest path is not the same as in a plane. Iterations in which the problem is analysed using different viewpoints and technologies are needed, since selecting the right technology can make the difference between a complex and expensive solution and a simple solution. For example, a milk factory once had a too high concentration of milk residue in its cleaning water. This milk factory first considered water filtering as the solution technology, which would require an expensive filtering installation. However, the cheapest solution technology turned out to be contamination prevention, which could be easily realised by tilting the milk bins slightly longer when emptying them. By reducing the amount of milk in the bins before cleaning and thus preventing milk to contaminate the water, the milk residue in the cleaning water was reduced to an acceptable level.

**Why Would Companies and Universities Cooperate?** Solving an industrial problem, of course, has value for the problem owner, i.e., the company gets a competitive edge, such as new functionality or improved performance. Yet, it has also value for the technology provider. A benefit

[2] Similar to (Pfleeger, 1999), we use technology to refer to methods, techniques, tools, procedures, or paradigms.

for the technology provider, in particularly for universities, is that the industrial applicability of the technology is proven. This proof is needed for scientific publications and a healthy science (Tichy et al. 1995). Another benefit is that continuity of the technology and its provider are stimulated. Especially commercial technology providers need real-world validation to convince similar problem owners to apply their technology (Zelkowitz et al. 1998). Trying to solve similar problems results either in additional proof of general applicability, or in explicitly known limitations of the technology. These known limitations can trigger and justify further research to overcome them. Summarizing, solving industrial problems has value for both companies and universities. Whether it is valuable, i.e., the benefits outweigh the drawbacks, is another question.

**Structure of the Article.** In this article, we present evidence from the industry-as-laboratory project Darwin related to Potts' hypothesis: Cooperation using the industry-as-laboratory paradigm is more valuable, for both company and university, than using the research-then-transfer paradigm. After discussing the related work, we describe the observed benefits and drawbacks for universities and companies. Next, the challenges we experienced in Darwin and possible resolutions are described. We end with the conclusions and future work.

## Related Work

(Muller et al. 2007) describe the lessons learned from the first Embedded Systems Institute project Boderc. One recommendation is that PhD students should follow multi-disciplinary classes instead of mono-disciplinary classes. By following multi-disciplinary classes, PhD students will be better equipped for the multi-disciplinary industrial setting.

(Muller 2005) observed that the needs and interests of industrial and academic people are often opposing and conflicting. This results in a need for different publications for industry and academia.

Many research projects in which academic and industry collaborate exist. See for example (SAVE-IT) (Henshaw et al. 2009) (Philips Research).Whether these projects apply the industry-as-laboratory paradigm is however not made explicit.

(SAVE-IT) is an industrial graduate school that organizes research studies for a number of industrial PhD students, and works for an increased cooperation and exchange between companies and universities. An industrial PhD student is a PhD student employed at a company and formally registered as a PhD student at a department of a university.

(Henshaw et al. 2009) observed that the outcome of systems engineering research should not be business as usual: Industry should prepare a change management programme to fully utilize the research outputs. This agrees with (Punter et al. 2009), who view transfer of software engineering technology as a change process, which aims at making a business success out of research results. One perspective of change management is evolutionary versus revolutionary change (Paulk 1999) (Bril et al. 2005).

Large industries often have a separate research department; see e.g. (Philips Research). These departments often collaborate with academics, and researchers of these departments regularly obtain a PhD degree. We refer the interested reader to (Bril et al. 2005) who describe not only the interactions of Philips Research with industry and academics but also three case studies in which Philips Research was involved. Note that the first two authors of that paper obtained their PhD degrees while working at Philips Research.

(Gill 2009) defined the knowledge, skills, experience, and personal attributes needed to be a "Systems Person". Experience in various jobs was one of the prerequisites.

(Bril et al. 2005) and (Punter et al. 2006) observed different roles of humans in the transfer of research results. Key roles of humans for transfer are champions and company angels. Champions will match research ideas to industrial problems and will motivate their colleagues to work with a particular technology. Company angels will sponsor the transfer with money and effort.

## Benefits and Drawbacks for Universities

In this section, we describe the observed benefits and drawbacks for research in general, and universities in particular. The benefits are access to industrial repositories, access to state-of-practice solutions, observation of practitioners, insight in industrial problems, and early feedback. The drawbacks are involvement of more persons and limited commitment.

**Benefits.** The first benefit for research we observed is the access to industrial repositories. Artefacts of software and systems are expensive and time-consuming to build, see also (Glass 1994). Hence, the industry-as-laboratory approach provides an effective way, both time wise and money wise, to gain insight into and prove theories about software and systems engineering. Note that open-source repositories are no alternative to prove industrial applicability, since differences between development in open-source and industry exists (Raymond 2001) (Paulson et al. 2004).

The second benefit for research we observed is the access to state-of-practice solutions. The interplay of theory and application — with applications often leading the way — is long known, but too often forgotten (Frey et al. 2006). Software and, especially, systems engineering are young research fields (Valerdi et al. 2009). Practitioners have, however, created point solutions for a large number of problems they experienced. The goals of companies stimulate the practitioners to protect their solutions, and thus maintain a competitive edge, instead of publishing them. Access to these state-of-practice point-solutions provides insights needed to arrive at general solutions for software and systems engineering problems. Building upon the state-of-practice also increases the appreciation of research by its practitioners.

The third benefit for research we observed is the opportunity to observe practitioners. Observation provides answers to many questions. Just to mention a few:

- What activities are performed by practitioners?
- How effective are these activities to obtain their associated goals?
- How much time is spent on these activities?
- Which activities lend themselves to automation?

Answers to these questions are relevant for research. Especially, for researchers who are not Systems Persons (Gill 2009), for example, since they did not have various jobs and thus cannot answer these questions from personal experience in the past.

The fourth benefit for research we observed is insight in industrial problems and their urgency. Showing the industrial applicability of research results is often required by research funding bodies (Punter et al. 2009). Hence, insight in industrial problems and their urgency helps to write more successful research proposals. Also, at a personal level, focusing on urgent industrial problems can be a motivating factor. Since the amount of publications is already so large that even if you work in a rather specialized area, keeping up-to-date by reading all the journals is almost impossible (Dybå et al. 2005), I prefer relevant industrial research over shelfware research.

The fifth and last benefit for research we observed is early feedback. By closely working with practitioners, the researchers get feedback already during problem analysis and technology selection instead of after publication. This feedback helps to determine whether the goal is realistic, achievable, and desired, and whether the solution fits in an industrial environment.

**Drawbacks.** The first drawback for research we observed was the involvement of more persons in the research team than just the PhD student and his academic supervisor. The involvement of more people resulted in more discussions, among others, to obtain and maintain a common goal. Furthermore, a larger team is more sensitive to change, i.e., people retiring or moving to other jobs. (Bril et al. 2005) observed that company angels and champions that leave the project have a negative influence on the sustainable embedding of technology. We observed the negative influence of leaving also before and during the initial embedding.

The second and last drawback for research we observed is limited commitment. The industrial members in the research team not only need to support the researcher with respect to providing

access to relevant industrial data and persons in the organisation. They also have to continuously pull the researcher in the desired direction, i.e., to solve a relevant industrial problem in its actual context. Without this pull, the necessary industrial feedback is missing. The lack of feedback seriously jeopardises the industrial value and the justification of the research, the technology transfer, and the goal of the cooperation. We observed limited commitment when industrial members were only partially assigned to the research project and when industrial members assigned by management did not feel ownership of the industrial problem and its solution.

## Benefits and Drawbacks for Industry

In this section, we describe the observed benefits and drawbacks for industry. The benefits are focus on the long term, focus on system overview and optimisation, efficient insight in technology advancements, early feedback on solution directions, and challenging of assumptions. The drawbacks are inappropriate documents and the mono-disciplinary excellence.

**Benefits.** The first benefit for industry we observed was a focus on the long term. Companies tend to focus on the short term, among others, due to market pressure. They often act like fire fighters that drive from one fire to the next. Due to the many fires, no time is taken to reflect on them: Why do these fires break out? And how could they be prevented? Researchers are outside the daily development activities, and hence are better able to reflect on them. Researchers tend to focus more on root-causes than on symptoms and the suppression thereof. And, researchers can more easily pay attention to problems with no quick, straightforward solution. Note that it is often easier for a company to pay attention to the long term by cooperation than doing it yourself, since the daily activities are less affected by interacting with a researcher to come up with a solution than by finding a solution yourself.

The second benefit for industry we observed was a focus on system overview and optimization. During product specification and design, the focus is on the system as a whole and global optimisations: the goal is to build the right product. During product development, test and verification, and bug fixing the focus is on details: the goal is to build the product right. Since customers will only buy the right product, even when it is not built (completely) right, companies should pay considerable attention to product specification and design. However, the time spent on product specification and design is negligible compared to the time spent on product development, test and verification, and bug fixing in the current state-of-practice in software and systems engineering. The balance becomes better when cooperating with software and systems engineering research focusing on general principles, on the architecture and design phase, and on system overview and global optimizations.

The third benefit for industry we observed was efficient insight in technology advancements. As already said before, the amount of publications is already so large that even if you work in a rather specialized area, keeping up-to-date by reading all the journals is almost impossible (Dybå et al. 2005). In other words, keeping up-to-date is expensive. Consequently, a company experiences the following tension with respect to insight in technology advancements. On the one hand, to survive in a competitive market, a company cannot afford to keep up-to-date by employing experts in all technologies, but needs to focus on its core activities and capabilities. On the other hand, a company cannot afford to miss an advancement in a (supporting) technology that yields a substantial advantage. Missing such advancement gives the competition a competitive edge that results in loss of market share or even bankruptcy of the company. Cooperation with universities, especially related to supporting technologies, is an effective way to balance the previously described tension. Cooperation provides valuable insights in technology advancements without substantial investments.

The fourth benefit for industry we observed was early feedback on solution directions. When a company has a problem and is considering particular solution directions, universities can provide valuable insights on them. For example, a solution direction might already be used in practice, but research is lagging with theory to accurately explain and support practice. That solution direction thus has a considerable risk: the company will be in the dark when the realization is not performing as needed and improvements are needed. (Frey et al. 2006) describe an example of

this situation from the medical domain: psycho-stimulants are prescribed for the calming effect on humans with ADHD, despite the fact that this result is logically inexplicable. In order to improve the treatment, i.e., more effective and/or fewer side effects, no theoretical support is available.

The fifth and last benefit for industry we observed was challenging of assumptions, for the greatest fool may ask more than the wisest man can answer. Over time a company will build up many implicit and company-wide shared assumptions. Employees in the company will not challenge these assumptions, despite the fact that due to environmental changes some of them will have become invalid. People with no experience in the company have an unbiased view and are more likely to challenge these implicit and company-wide shared assumptions, especially when they are not valid anymore. However, the role and goal of people new to the organization largely determines the likelihood that assumptions are actually challenged. The goal of a new employee is to become productive quickly. Challenging company assumptions typically does not help to achieve this new employee's goal. The goal of a researcher is to improve the current practice in a company. Challenging company assumptions typically does help to achieve this researcher's goal.

**Drawbacks.** The first drawback for industry we observed was inappropriate documents. (Muller 2005) observed that different publications for industry and academia are needed. Many researchers, and in particular PhD students, have little to no experience in industry, making writing for an industrial audience a real challenge. Yet, researchers only get education in academic writing. Furthermore, researchers already have to write a number of conference and journal papers. Reusing (parts of) these academic papers for industrial publication seems time-effective. The researcher's background and education, and the reuse of earlier work targeting a different audience cause that industrial publications are somewhat inappropriate. In a few cases, the only difference between the academic and industrial publications was that some names were anonymized in the academic paper to protect company sensitive information.

The second and last drawback for industry we observed was mono-disciplinary excellence. The problem analysis is typically not completed when the cooperation with a researcher is started. Insight from the researcher on the technology is needed to evaluate the solution in sufficient depth. The evaluation might lead to the conclusion that the researcher's technology should be rejected as a potential solution. This conclusion will end the cooperation, except in two cases. In the first case, the technology is still applicable to other problems in the company. In the second case, the researcher changes technology. The latter case is highly unlikely since achieving mono-disciplinary excellence is not easy. Summarizing, especially in long-term industry-as-laboratory research projects, a company might lose the cooperation with researchers due to their mono-disciplinary excellence, since the company can come to the conclusion that their technologies are not appropriate to solve its problems.

## Challenges Experienced

In this section we describe the challenges we experienced in the Darwin project:

- Home of the researcher
- One shared goal
- Revolution or evolution
- Alignment of the different time scales

We discuss possible resolutions to these challenges, with a focus on our resolutions in the Darwin project and the experience with them.

**Home of the Researcher.** In an industry-as-laboratory project, researchers have (at least) two possible homes: at the company and at the university. We experienced a challenge in our project to make the researchers feel at home at both places to ensure the benefits of both worlds. The researcher should feel at home at the university to learn about the technology, to be informed about appropriate workshops, conferences, and special issues, and to be stimulated to reflect, abstract, and generalize the specific industrial details. The researcher should also feel at home at the company to learn about the problem and its context, to ensure a suitable solution, and to observe and experience the relevant details. A lot of the learning experience can not be planned, but happens "accidentally", e.g. behind a test machine or at the coffee machine.

During the Darwin project, we experienced the challenge of making researcher feel at home at the company and at the university. Therefore, we tried to make the university and the company a comfortable place to work from the first day onwards. This includes the computer infrastructure, furniture, and the working space. In addition, since the travel time to the company and university should not be prohibitive long, researchers were stimulated to live close to them. We experienced that short travel time to the company seems more important for overall success than short travel time to the university. Note that this might be caused by working with academic PhD students who are intrinsically more at home at the university and less at the company than industrial PhD students (SAVE-IT).

We believe that following classes can also contribute to feeling at home. Therefore, we make a slightly different recommendation than (Muller et al. 2007): when following classes is required, we recommend choosing those classes that make researchers feel most at home at both places. An academic PhD student with no industrial experience might still only follow industrial-oriented, multi-disciplinary classes, as recommended by (Muller et al. 2007). Yet, for industrial researchers with years of industrial experience, we think that following academic-oriented, mono-disciplinary classes is more appropriate.

**One Shared Goal.** In an industry-as-laboratory project, many stakeholders are involved. In the Darwin project, each researcher interacted with academic supervisors, representatives of the company, and Embedded Systems Institute employees who focus on the industrial applicability in general, support the industrial research and transfer of research results to industry, and manage the project.

A researcher needs a clear, single, and concrete goal to be effective. The goal should be considerably more concrete than improve the multi-disciplinary product development in the company using the mono-disciplinary excellence of the research institute. Hence, we experienced a challenge to present that clear, single, and concrete goal to the researcher despite the fact that three kinds of stakeholders are involved, with different, maybe even conflicting, goals and requirements. We handled this challenge as follows: First, we made clear to all researchers that they should signal disagreement between stakeholders instead of trying to (sufficiently) satisfy all the stakeholders despite their different goals. Second, we defined a shared long-term goal, and we agreed on concrete actions for the upcoming half year to arrive at that goal. Final, we brought all stakeholders together regularly, every six working weeks, to discuss the common goal, current status, recent progress, and next steps in considerable details. A time slot of two hours was allotted for these meetings. Although the two hours were not always needed, planning a shorter time slot would have prevented a good discussion with sufficient, relevant details.

**Revolution or Evolution.** The result of successful research is change (Henshaw et al. 2009) (Punter et al. 2009). One perspective of change management is evolutionary versus revolutionary change (Paulk, 1999) (Bril et al. 2005). In the Darwin project, we experienced a challenge to prevent the researchers from inappropriately choosing revolutionary over evolutionary change. One reason is the immaturity of software and systems engineering (Glass 1994) (Tichy et al. 1995) (Tichy 1998). Due to the lack of validation in these particular research fields, researchers are tempted to have poor validation or even no validation at all. Another reason is the fact that revolution seems easier than evolution. We will discuss this counterintuitive fact in more details in the remainder of this section.

Solving an industrial problem in its context often seems more difficult than necessary. For example, in a company that uses a relative old programming language, like C, a solution that needs advance programming features, such as reflection and attributes, seems unnecessary difficult: effort has to be invested in programming features that are available in more modern languages, like C#. However, a threat of validity remains when the conclusion that the industrial problem is solved based on a solution in another context. In the medical domain, it is well known that experiments in different contexts, e.g., in vitro and in vivo experiments, require great care with respect to inferences about their results (Frey et al. 2006). So, if we do not want to throw out the baby with the bathwater, we need to prove that the solution really solves the industrial problem in its context without undesirable side effects.

Therefore, to prove the solution in an industrial context in the previous example, we either have to build the solution and (parts of) the artefacts of software and systems in a modern programming language, which is expensive and time-consuming (Glass 1994), or build the solution in the relative old programming language guided by the concepts and realizations of the advanced programming features. The latter choice, i.e., evolution, is typically more effective than the first, i.e., revolution!

Focusing on evolution instead of revolution also helps to find the added value of research without imposing unnecessary requirements. By focusing on the added value, not only the solution, but also the cost of experiments (Tichy 1998) and risk of adoption are minimized. For example, in the Darwin project, we discussed the generation of state models instead of code from control requirements to minimize the impact of research and to reduce the risk of adoption (Theunissen et al. 2009). See also Figure 2.

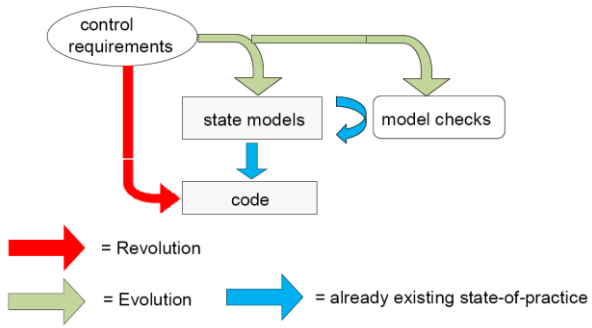


**Figure 2. Two approaches for code generation from control requirements: either directly or as a two step approach using state models as the intermediate step.**

**Alignment of the Different Time Scales.** Companies typically produce new versions of their product on a (half) yearly basis. The new versions have improvements in functionality and performance. Whilst realising improvements expected and unexpected problems are encountered. Companies have a short-term interest in problem solving: these problems have to be solved quickly to ensure meeting the product release date. To solve a problem, companies prefer technologies known to them, since the knowledge with those technologies ensures a fast and effective resolution of the problem. Yet, if no known technology can solve the problem, they do not care what particular technology is used, as long as it effectively solves the problem. For companies, it is acceptable to stop a particular improvement, i.e., not to solve the associated problems, especially when other improvements are more profitable.

Researchers typically publish increments on a technology on a (half) yearly basis. To be able to extend the body of knowledge, a considerable amount of time and effort has been invested to understand that particular technology in great depth. While validating a particular technology in an industrial context, one either gets proof that a solution technology is applicable or finds limitations of that technology in that context. Both results are equally valuable for a researcher. A researcher has a long-term interest in solving problems: as long as the body of knowledge can be extended, research is considered worthwhile.

In the Darwin project, we experienced a challenge in aligning the different time scales of industry and academia. We addressed this challenge by working on case studies that ranged in time from six months to one year. By using multiple case studies, researchers are automatically prevented from creating a point solution. A new case study is also a natural vehicle to prove the general applicability of a technology. In hindsight, we should have communicated from the start to the researchers that they study a technology that will be applied on different case studies in industry over time to stimulate and prove its generality. Since, especially the first time, the change in case studies was not immediately experienced as contributing to their research by many researchers.

A lesson we learned with respect to aligning the different time scales is looking for the windows of willingness to change. First, an industrial project that could benefit from the research results must be defined. In other words, a project that experiences a problem is needed to be able to validate and/or transfer solution technologies for it. Second, the project should have capability to absorb the results. In other words, the project should not already be in the danger zone: late and understaffed.

## Conclusions

According to (Potts 1993), the influence of software engineering research on solving industrial problems is currently little. He hypothesized that using the industry-as-laboratory paradigm instead of the research-then-transfer paradigm will improve the situation. The Embedded Systems Institute applies the industry-as-laboratory paradigm not only for software engineering research but also for systems engineering research. Both Potts' hypothesis and the Embedded Systems Institute generalization are, however, not proven yet. In fact, the industry-as-laboratory paradigm is a concept for which many different realizations are possible. Variations in the realization of the industry-as-laboratory concept include:

- How the challenges as observed in the Darwin project and described in this article are handled.
- Kind of researchers, e.g. academic PhD student, industrial PhD students, academic Post-Docs, or industrial researchers; and their amount of academic and industrial experience.
- Kind of research, e.g., explorative or applied, and problem- or solution-driven.
- Kind of company involvement, from strongly pulling to resisting change, and from only a single manager, the one that signed the contract, to all employees.
- The knowledge, skills, and abilities of the researchers, including those related to teamwork (Stevens et al. 1994), that are, according to us, more relevant due to the many multi-disciplinary interactions in industry-as-laboratory projects.

Our mainly positive experience with just a single realization of the industry-as-laboratory concept of (Potts 1993) in the Darwin project is therefore definitely not enough to prove Potts' hypothesis. Even worse, due to the small number of researchers, only eleven, confidence of any conclusion is even low from a statistical point of view. However, by making the variations in the industry-as-laboratory paradigm explicit, Potts' hypothesis changes from having a binary answer to an "optimization" problem. Further research is thus needed not only to prove Potts' hypothesis and the Embedded Systems Institute generalization to system engineering research, but also to find the realization of the industry-as-laboratory concept that yields the most benefits for both academia and industry.

## Future Work

In the future, we will combine the observations and experiences of multiple industry-as-laboratory projects. These projects need, however, not to be limited to those executed at the Embedded Systems Institute. By combining these observations and experiences, we get not only a better insight in possible differences in realization of the industry-as-laboratory concept, but also better statistical support for conclusions and theories.

In addition, we would like to support and quantify our observations and experiences better. Among others, we would like to use a questionnaire in which PhD students in industry-as-laboratory projects are compared to PhD students in regular research-then-transfer projects. Getting statistically comparable sets of these PhD students is a challenge. During the conference, we would like to discuss our future work not only to look for cooperation with other industry-as-laboratory projects but also to improve the support and quantification of observations and experiences from such projects.

## Acknowledgements

I would like to thank Pierre America, Gerrit Muller, Jacques Verriet, Teade Punter, and Dave Watts for feedback on an earlier version of this paper.

## References

Bril, R.J., Krikhaar, R.L., and Postma, A., "Architectural support in industry: a reflection using C-POSH." *Journal of Software Maintenance and Evolution: Research and Practice*, Vol. 17(1), pp. 3-25, 2005.

Dybå, T., Kitchenham, B.A., and Jørgensen, M., "Evidence-Based Software Engineering for Practitioners." *IEEE Software*, Vol. 22(1), pp. 58-65, January/February 2005.

Frey, D.D., and Dym, C.L., "Validation of design methods: lessons from medicine." *Research in Engineering Design*, Vol. 17(1), pp. 45-57, June 2006.

Gill, K., "Systems Thinking or Systems Engineering." *Proceedings of the 7th Annual Conference on Systems Engineering Research* (Loughborough, UK, April 20-23) 2009.

Glass, R.L., "The Software-Research Crisis." *IEEE Software*, Vol. 11(6), pp. 42-47, November 1994.

Henshaw, M.J.D., Gunton, D.J., and Urwin, E.N., "Collaborative, academic-industry research approach for advancing Systems Engineering." *Proceedings of the 7th Annual Conference on Systems Engineering Research* (Loughborough, UK, April 20-23) 2009.

Muller, G., "Industry and Academia: Why Practitioners and Researchers are Disconnected." *INCOSE International Symposium*, 2005.

Muller, G., and Heemels, M., "Five Years of Multi-Disciplinary Academic and Industrial Research: Lessons Learned." *Conference on Systems Engineering Research*, 2007.

Paulk, M.C., "Structured approach to managing change." *Crosstalk: The Journal of Defense Software Engineering*, Vol. 12(11), pp. 4-7, November 1999.

Paulson, J.W., Succi, G., and Eberlein, A., "An Empirical Study of Open-Source and Closed-Source Software Products." *IEEE Transactions on Software Engineering*, Vol. 30(4), April 2004.

Pfleeger, S.L., "Understanding and improving technology transfer in software engineering." *The Journal of Systems and Software*, Vol. 47(2-3), pp. 111-124, July 1999.

Philips Research, http://www.research.philips.com/

Potts, C., "Software-Engineering Research Revisited." *IEEE Software*, Vol. 10(5), pp. 19-28, September 1993.

Punter, T., Krikhaar, R.L., and Bril, R.J., "Sustainable Technology Transfer." *Proceedings of the international workshop on Software technology transfer in software engineering* (Shanghai, China, May 22) pp. 15-18, 2006.

Punter, T., Krikhaar, R.L., and Bril, R.J., "Software engineering technology innovation – Turning research results into industrial success." *The Journal of Systems and Software*, Vol. 82(6), pp. 993-1003, June 2009.

Raymond, E.S., "The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary." ISBN 0-596-00108-8, January 2001.

SAVE-IT, http://www.mrtc.mdh.se/projects/save-it/

Stevens, M.J., and Campion, M.A., "The Knowledge, Skill, and Ability Requirements for Teamwork: Implications for Human Resource Management." *Journal of Management*, Vol. 20(2), pp. 503-530, 1994.

Theunissen, R.J.M., Schiffelers, R.R.H., van Beek, D.A., and Rooda, J.E., "Supervisory control synthesis for a patient support system." *European Control Conference* (Budapest, Hungary) 2009.

Tichy, W.F., Lukowicz, P., Prechelt, L., and Heinz, E.A., "Experimental evaluation in computer science: a quantitative study." *The Journal of Systems and Software*, Vol. 28(1), pp. 9-18, January 1995.

Tichy, W.F., "Should Computer Scientists Experiment More?" *Computer*, Vol. 31(5), pp. 32-40, May 1998.

Valerdi, R., and Davidz, H.L., "Empirical Research in Systems Engineering: Challenges and Opportunities of a New Frontier." *Systems Engineering*, Vol. 12(2), pp. 169-181, 2009.

Zelkowitz, M.V., Wallace, D.R., and Binkley, D.W., "Culture Conflicts in Software Engineering Technology Transfer." *NASA Goddard Software Engineering Workshop*, 1998.

## Biography

Piërre van de Laar was born in Bavel, the Netherlands, in 1971. Between 1989 and 1998 he studied at the Catholic University of Nijmegen and was awarded his master degree (cum laude) in both theoretical and computational physics in 1994 and his PhD on 'Selection in Neural Information Processing' in 1999.

From 1998 until 2006, he was employed by Philips Research in Eindhoven and spent two years following the course 'from doctor in science to computational scientist'. From 2000 onwards, he investigated the exploitation of architecture description languages for product families, visualization, verification, aspect-orientation, and dependability. Since 2006, he has been employed by the Embedded Systems Institute as a Research Fellow. After performing a study for DaimlerChrysler, he became a member of the Darwin project. Darwin is a collaborative research project between the Embedded Systems Institute, Philips Healthcare, Philips Research, and five Dutch universities and aims to improve the evolvability of complex systems.