[ simula . research laboratory ]

# How to Become an Excellent IT-research Institute

## (with substantial impact on the IT-intensive industry)

*”By Thinking Constantly About it”*
*“Basic, applied research”*
*“The industry is our lab”*

Magne Jørgensen
Simula Research Laboratory

[ simula . research laboratory ]

# The Story About
# Simula Research Laboratory

# In the beginning there was …

- the decision to close down the airport near Oslo (early 1990s)
- political discussions about what to do with the premises of the old airport.
- an industry forum suggesting that an IT-park was a good idea. The ship owner and investor Fred. Olsen – he also produces the Timex watches and is the role model for Montgomery Burns in Simpsons! – was the main initiator of this idea. The idea got political support in 1997.
- the vision was to make the IT-park the most attractive knowledge centre in Europe by the year 2005 and a transfer of Norway into a knowledge society less dependent on export of natural resources.
- a minister of education (a man with only elementary education!) who decided that the IT-park needed a research center (➔ Simula Research Laboratory)

# Achievements of Simula in the 10-year period

- Simula’s future existence has always been dependent on good evaluations!
- Every fifth year we are evaluated by international (independent) experts. The highest evaluation (”excellent”) is very hard to get.
- The evaluations shows a very progress on the research quality:
  - 1999-evaluation (pre-Simula): One group was ”excellent” (SC), the other two (SE and ND) were ”good”. [”Good” is not really good, but average or below …]
  - 2004-evaluation: One group is ”excellent” (SC), one group is ”very good, on its way to excellent” (SE), one group is ”good, with some very good elements” (ND).
  - 2009-evaluation: Two groups are ”excellent” (SC and SE), one group is ”very good, with some excellent elements” (ND).
- The Software Engineering (SE) group was recently ranked as the second best software and systems research institute worldwide by Journal of Systems and Software.

# Achievements of Simula in the 10-year period

- Seven companies generated in the period (none of them, however, are big successes, yet …)
- Substantial (but difficult to measure) impact on software processes in the Norwegian software industry.
  - Examples of very high return of investment, e.g., related to our fault prediction and testing research.
- An increase in external funding (from industry and government) enabled an increase in number of employees, now at about 100.
- Simula's reputation has the last years enabled the attraction of international highly recognized software engineering researchers:
  - Strong industry background/understanding
  - Strong research record
  - Dedicated to research goals corresponding with Simula's goal

# The platform of Simula’s success

- Full-time research! *[at least that is the goal ….]*
- Basic, applied research.
- Quality culture.
- Organization of activities are more like those in private companies, e.g., less bureaucratic hiring processes.
- Free research within a directed topic (less free than at the universities).
- Good contact with the politicians - explaining them why we are doing what we are doing and what they get from their investments.
- Creation of new businesses based on our research.
- Good PhD-students, recruited internationally.
- **Strong collaboration with the industry. They fund more and more of our research, without we becoming consultants.**

# Full time researchers

- How did you come up with the law of gravitation?
  - ”*By thinking constantly about it ….*” (attributed to Isaac Newton)
- The best researchers are those who get fully absorbed by what they are doing and are not too much “disturbed”.
- Golden rule in project management: At least 80% effort on your main deliveries leads to the best quality and efficiency. The same goes for research.
- Universities:
  - Many other obligations, little time for research
  - Can easily kill the research enthusiasm
- How do we find dedicated, enthusiastic researchers?
  - Headhunting (through networking)
  - Luck
  - Create dedication (not easy, if the personality/trait is not there)

# Basic, applied research ….

“*Our aim is to conduct long-term basic research with a clear view to application of the research results. The projects focus on fundamental and complex challenges that are important for society at large.*”

Projects that are interesting, but not sufficiently fundamental or without an important applications will not be started, even when there is funding available. [*well, this is at least the ideal – there are exceptions ….*]

# Basic, applied research

- It is a myth that applied research cannot be on basic (core) problems, e.g.:
  - Mathematical problems that improve the simulation of heart attack (to prevent it)
  - Methods for higher quality of service in communication networks
  - Understanding the mental steps in people’s judgment of time to complete work.
- If your research is both basic and applied, you may (as we did) experience that this enable:
  - More industry attention (funding, partners, collaborations)
  - Longer term focus (applied IT-research is frequently short-term)
  - Higher quality research
  - Expertise and knowledge that is not outdated or irrelevant when the technology changes

# Organization as a limited company & more directed research

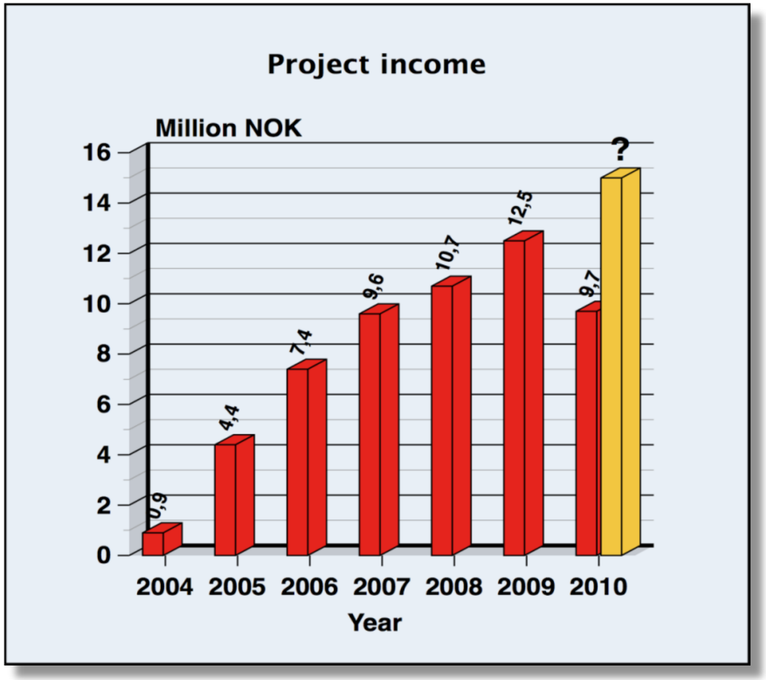
- University characteristics (in Norway):
  - Research freedom (nearly unlimited, as soon as you've got a permanent position)
  - Hiring people at the universities takes months (typicaly 8-12 months)
  - The organization is not designed (managed) to provide support for high quality research, but more towards teaching obligations.
- Simula breaks with all these three university characteristics:
  - Less freedom. Not much freedom to choose the topic, but large degree of freedom when it comes to how to do the research and problems addressed within the topic.
    - Good researchers will typically get the opportunity to do side-activities, which may evolve into new research topics.
  - Quicker hiring processes (much based on headhunting)
  - Management processes more designed to support high quality research

# Collaboration with the industry

- Enabled through:
  - Basic, applied research
  - Industry experience and understanding of their problems
  - Good reputation as researchers doing relevant, high quality research
- Challenges:
  - Avoidance of short-term process improvement for the company
  - Partner re-organizes frequently (lack of stability of collaboration)
  - Lack of industrial understanding of the researchers (you have to speak their language!)
- Our main categories of industry collaboration:
  - One big company funds our research (more an more common)
  - Publicly funded research involving several industry partners (who fund their own work and sometimes some of our expenses)
  - Informal collaborations with companies, not in need of funding

# Example 1: Collaboration with the Oil Industry Scientific Computing

- Focus on hydrocarbon exploration
- 100% funded by Statoil. Total 45 MNOK (8-9 Mill USD) by end of 2009.
- Long-term research goals, that require both basic and applied research on computation (50/50). The applied part generated a spin-off company.
- Collaboration enabled through the SC group's world-leading research on numerical methods and software for solving partial differential equations.



# Example 2: Testing of software

- Det Norske Veritas (DNV) provides certificates for the Maritime and Energy sector
- The safety of, for example, the vessels/ships depends on the software for the steering and navigation.
- The SE group at Simula collaborate with DNV on methods for providing evidence for the safety of the software and other issues related to verification and validation of embedded software.
- DNV finance the research (PhD-students, etc.) made by Simula personnel.
  - Research on core problems (e.g., how to provide evidence of software safety).
  - Nevertheless, highly applied.

An example of how I collaborate with the software industry in generating knowledge and improve processes

**My research question:**

How to make the assessment of the uncertainty of effort estimates more realistic?

# Number of inhabitants in Norway

Minimum

Maximum

Be 99% confident to include the true value in the minimum-maximum interval!

# Radius of the (dwarf) planet Pluto

Minimum

Maximum

Be 99% confident!

# Overconfidence in the accuracy of effort estimates

## Step 1: Motivate research and review existing work

- Findings from case studies and systematic review of existing knowledge gave that:
  - Overconfidence regarding accuracy of estimates leads to poor plans, budgets and investment analyses in many software projects.
  - Inherent uncertainty in effort estimates.
  - Neglected topic. The current method (PERT) used in the software industry lacks evidence that people are actually able to produce meaningful uncertainty (minimum-maximum) interval on the required format, e.g., 90% confidence intervals.
  - Related work in psychology, forecasting and cognitive science show a general bias towards overconfidence in judgment accuracy.
- This is clearly a basic (overconfidence in time estimation), applied (improving project plans) research topic.
- We started a collaboration with researchers in forecasting and psychology.

# Overconfidence in the accuracy of effort estimates

## Step 2: Improved understanding of the problem

- Completion of several case studies of state-of-practice in software companes suggested that:
  - Most companies did not have any formal assessment of effort uncertainty (just added a expert judgment-based risk buffer)
  - Some companies provided minimum-maximum values without any indication of confidence level.
  - Very few provided effort prediction intervals, e.g., “90% confident in including actual effort in the interval [1000; 1500] work-hours”.
  - “90% confident” or “almost certain” minimum-maximum interval included actual effort in only 60-70% of the cases.
  - Narrow, overconfident prediction intervals were rewarded by the management.
  - Poor learning opportunities. The feedback (if any) was on a format difficult to learn from.
- Overconfidence led to frustrated clients and poor project control.

# Overconfidence in the accuracy of effort estimates

**Step 3: Small scale experiments (with software professionals and students) to better understand the underlying cause-effects.**

- **Example of experiment**: Does the developers understand what “90% confident” means?
- **Experimental design**: Same development task. Different confidence levels.
  - Group A: Minimum-maximum interval, when 50% confident.
  - Group B: Minimum-maximum interval, when 75% confident.
  - Group C: Minimum-maximum interval, when 90% confident.
  - Group D: Minimum-maximum interval, when 99% confident.
- **Main result**: No! The minimum-maximum values were about the same for all confidence levels.
- **Implications**: Meaningless to ask people to be “90% confident” without training/support? We need to change how we elicit uncertainty and how we train software developers.

# Overconfidence in the accuracy of effort estimates

**Step 4: Invention of alternative uncertainty assessment processes. Evaluation of them through small scale experiments with software professionals**

- Experiment: Does a change in elicitation format improves the realism?
  - Traditional approach (PERT): “What is the minimum and the maximum effort? (Be 90% confident to include the actual effort)”
  - Alternative 1: “How frequently have similar projects overrun their estimated effort with more than 50%?”
  - Alternative 2: “What is the maximum effort? How likely is it that the actual effort will be higher?”
  - **Main finding**: Alternatives 1 and 2 were better than the traditional approach

# Overconfidence in the accuracy of effort estimates

## Step 5: Evaluate the most promising processes in representative field settings

- A randomized controlled trial in field setting showed the same positive effect of using Alternative 1 and 2 compared to use of the PERT approach.
- Experiment:
  - The company randomly allocated the use of one of the three alternatives to uncertainty assessments of new software projects.
  - We paid them for the extra work (actual work on the uncertainty assessment following the alternative processes + administration).
- Studies in field settings are required to be convincing!

# Overconfidence in the accuracy of effort estimates

## Step 6: Transfer of results to the software industry

- We are still not there to see that this has impacted many companies – although it should. The results should also lead to a re-writing of parts of project management books about minimum-maximum prediction intervals.
- Some positive results:
  - The methods are implemented in at least two software estimation tools.
  - The results have been included as one of the forecasting principles at the main forecasting site (www.forecastingprinciples.com).
  - The results are spread through articles in practitioner's magazines (IEEE Software), industry conferences and industry seminars. It is however hard to say how many companies that actually use the alternative methods.
- It takes time to transfer results ….

# An Example of Collaboration Made Possible When the Funding is Good:

“*The Impact of Irrelevant and Misleading Information on Software Development Effort Estimation*”

# “Clouds Make Nerds Look Better”

- Sunshine increases tipping, impacts stock-market, and, increases happiness.
- Study of university applicants:
  - 12% higher chance when sunshine compared to worst cloudcover.
  - Nerds had significantly higher chance compared to non-nerds on cloudy days.
    - Nerd-factor measured as academic rating divided by social rating (e.g., leadership).

# Irrelevant information is everywhere …

- Requirement specifications and other information provided in an estimation situation typically include
  - some misleading information (on purpose or accidentally)
  - much estimation irrelevant information
  - much information of low importance for the estimation work
- There are good (and not so good) reasons for this, e.g.,
  - information may be relevant for other purposes than effort estimation,
  - ”copy-paste” of general information about the clients’ processes and organization from previous specifications,
  - lack of competence in how to write a good requirement specification
- Are we more rational than stock investors and university applicant assessors, or do we get impacted by irrelevant information when estimating effort?

# A randomized, controlled trial in field settings

- Forty-six companies from various countries estimated the same five projects: Russia (15 companies), Ukraine (5), India (7), Bulgaria (4), Romania (3), Pakistan (5), Belarus (2), Moldovia (1), Poland (1), Serbia (1), Slovakia (1), and Vietnam (1).
- We accepted only estimators with professional experience from projects similar to those to be estimated, i.e., we allowed only reasonably experienced estimators.
- The companies were hired and paid for their estimation work, i.e., they did not (seen from their point of view) participate in an experiment.
  - The companies were on average paid about 1500 USD for the estimation work, ranging from 400 to 4000 USD.
  - The effort a company estimated to spend on the estimation of the five projects varied from about 40 work-hours to about 200 work-hours.
  - They were told that they would not be invited to develop the systems, but that their job was to provide realistic effort estimates.
- Random allocation to different “manipulations” of requirement specification.

# Length of specification ...

- **Hypothesis 1:** A reduction in number of pages of the requirement specification leads to lower effort estimates, even when the written content is exactly the same.
  - Manipulation: Text identical. One version 3 pages, the other 12 pages.
  - Length of specification is clearly not relevant for the development effort, but will it be used as an indicator?
- We had previously found an effect of the length of the specification on the effort estimation in laboratory settings with computer science students. The question was whether this was a relevant effect in more realistic settings, as well.

# Results: Length of specification (H1) [System: DocAssist]

**The Effect of the Reduced Length of Specification**

| **Group** | **Median** |
|---|---|
| Manipulated (3 pages spec.) | 295 work-hours (n=24) |
| Ordinary (12 pages spec.) | 330 work-hours (n=22) |

A small effect – perhaps not even that ...

Effect seems to be reduced (perhaps removed) with more time to do the estimation work and expertise.

# Client expectation ...

- **H2:** Presenting the actual effort of the system to be replaced (a low numerical value in our case) early in the requirement specification leads to lower effort estimates.
  - The following text was included early in the manipulated requirement specifications: "*The preliminary budget of the new system is $10 000 [corresponding to about 100 work-hours with typical pricing in the country in which it will be built]. The preliminary budget is not built on any knowledge about the actual cost of developing the new system, and will, if needed, be extended to cover the expenses necessary to build a quality system with the desired functionality.*"
  - 100 work-hours is a very low value for this project and the companies were instructed to not use this as input to their effort estimate, but they may use it unconsciously.
- Several experiments in laboratory settings found large effects of client expectations on the realism of the effort estimates.

# Results: Client expectation (H2) [System: IMWOS]

**Numerical Anchor**

| **Group** | **Median estimate** |
|---|---|
| Manipulated (client’s expectation) | 724 work-hours (n=23) |
| Ordinary | 956 work-hours (n=23) |

A significant, large effect.
However, lower effect than in our previous laboratory experiments.

# Time schedule presssure ...

- **H3:** Information about that the client requires a short development period leads to lower effort estimates.
  - The following text was included early in the manipulated requirement specifications: “[the client] *expects that the system development starts February 3, 2008 and can be launched February 23, 2008. This three week period should include all development and testing.*”
  - A short development period should lead to, if anything, more rather than less use of effort, but may also induce “wishful thinking”.
- Previous experimental results in laboratory condition indicated that there was wishful thinking involved (we don’t have much time ➔ it cannot take much time) when estimating under these conditions.
- Opposite effect of what most people would considered as “normative” estimation behavior, where compressed time typically means more, not less use of effort.

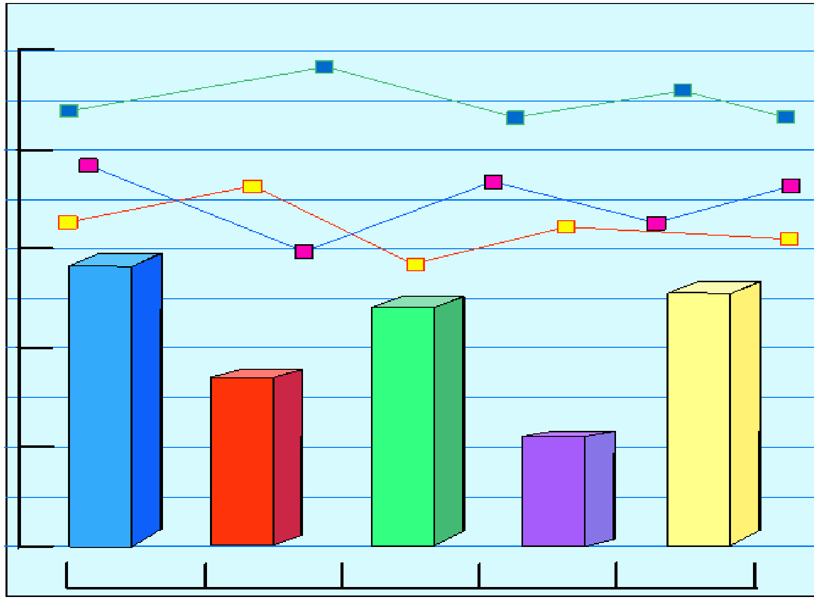
# Results: Time schedule pressure (H3) [System: DES]

**The effect of time schedule pressure**

| **Group** | **Median** |
|---|---|
| Manipulated (Informed that the client expected the system to be developed during 3 weeks period.) | 142 work-hours (n=24) |
| Ordinary | 214 work-hours (n=21) |

Very large, significant effect! Similar to the effect found in the laboratory conditions.

# How to impact the industry with your IT-research?



# Evidence-Based Software Engineering

- Improve the acceptance of Evidence-based SE (EBSE)
  - Teach software professionals EBSE (Training in formulating decidable questions, collection of valid evidence (including experience-based evidence), evaluation of strength of evidence and synthesis of evidence.)
  - Promote evidence-based principles at conferences
  - Train software professionals in completion and understanding of empirical studies
  - Write Software engineering books that are evidence-based
- Like medicine, we should try to get to a stage where the professionals only accept evidence-based principles and methods.
  - It's a looooong way to go, and there may be inherent problems that stop us from reaching the stage where medicine currently is.

# Basic, applied research

- Select research topics where impact is more likely (**basic, applied research!**)
  - Increase the emphasis on relevance. Robert Glass in IEEE Software March/April, 2009 recommends that all studies should go through an “applicability check”.
  - Do not conduct research where there are no opportunity to impact. Timing may be important.
- Include more research with high potential of impact. (Think bigger!)
- Emphasis money saving potential. A rough guideline by innovation advisors is that an idea should be able to save at least 10 times its implementation cost to be convincing for investors.

**There is in my opinion far too much IT-research of low industry relevance! It’s sometimes like doing research on typewriter improvement.**

# High quality + convincing research methods

- Higher quality studies.
  - I think I have reviewed more than 50 studies that show that their own estimation model is better than the other models. Most of these empirical evaluations have in my opinion been poorly designed.
- More convincing studies.
  - Inclusion of real-life success stories, less use of students and small scale systems.
  - Experiments and evaluations in real-life settings, including large-scale software projects.

# Collaborate with the IT-industry

- Conduct the research in collaboration with the software industry.
  - Let them tell convincing success stories. Nothing beats success stories that can be linked to your research results
  - Mean values and statistical significance may convince scientists, seldom software professionals.
  - Make win-win situations out of research results (see picture)

# Learn how to package and sell your results

- Better packaging/wrapping of research results
  - Tools
  - Processes/methods
  - Standards/certificates
  - Courses

# Learn how to transfer your results

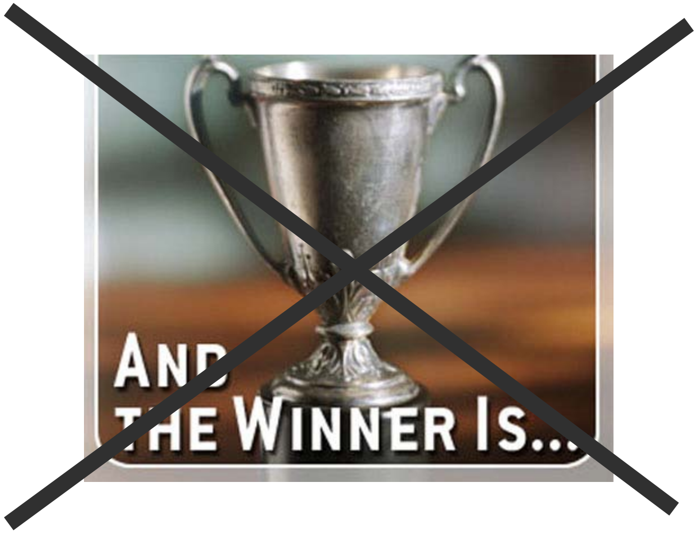
- Better transfer of IT-research results
  - Publish in practitioners' magazines
  - Write books without academic jargon
  - Be were practitioners meet
  - Package the research results as “experience” and “success stories”
  - Educate journalists to write about IT-research (accept that good IT-researchers are not necessarily good communicators)
- Software practitioners are typically not even aware of our studies. If they find them, the studies are in a language they do not understand. This slows (or even inhibit) the impact.

# Timing is important …

- Better timing of research studies.
  - We are far too often lagging behind.
  - When a method already is established, it is difficult to have an impact.
  - Being able to impact sometimes means that the research-based knowledge has to be there (and be known) when (or before) new technology emerges.
  - Example: If I had collaborated with the Planning Poker guru (Mike Cohn) published it, I could have shared with him relevant research results on group-estimation and improved the method before it was distributed.

# Avoid fashion-based research

- Avoid “Is Method A better than Method B”-studies, where the methods consist of many (ill-defined) elements.
  - This “reductionism” may sound like a paradox, since the software industry wants exactly that kind of studies.
  - However, such studies do in my experience seldom produce results that are convincing, they seldom produce insight in cause-effects, they seldom have the timing to enable impact (studies of already established practices are typically used to sell methods, not to improve them).



# Final Words …

- The Simula-model has created a success in terms of research quality and industry collaboration/industry benefit.
  - The model could probably be applied most other places, as well.
- Industry collaboration/benefit is frequently not in opposition to completion of basic research in IT.
- IT-researchers need to improve how they interact with and their skills in impacting the software industry.
- The perhaps most important elements of building a successful IT-research institute (compared to what is currently the case at many universities) are:
  - Direct the research on basic problems of importance to the industry.
  - Enable full time research (or at least 70-80% for key researchers)
  - Recruitment of research talents with industry background.
  - Close collaboration with the software industry, without becoming a consultant.
  - Generous funding that enable the above elements.