

Empirical studies on software development effort uncertainty assessment:

The role of outcome feedback in the learning process

Cand. Scient. thesis

Tanja Gruschke (tanjag@ifi.uio.no)

University of Oslo

Department of Informatics

November 2004

Abstract

To enable properly sized software project budgets and plans, it is important to be able to assess the uncertainty of the estimates of most likely effort required to complete the project. Previous studies show people in general, as well as software professionals, tend to be overconfident when assessing uncertainty over estimated effort. This thesis explores the possibility of learning more realistic uncertainty assessment with the use of outcome feedback. Two experiments, with favorable learning environments, were set up to investigate the issue. The first study focused on whether people in general possess the ability to learn more realistic uncertainty assessment; the second how much, and how, software developers learn to improve uncertainty assessment. The results indicate that people in general are well calibrated initially, and highly capable of adjusting towards realism given favorable learning conditions; i.e. frequent and relevant feedback on performance. In the software engineering setting, using experienced software developers, there was, in comparison, observed a lower degree of learning realism in effort uncertainty assessment. There was found that a necessary condition for improvement of uncertainty assessments on effort estimates may be the use of explicitly formulated uncertainty assessment strategies. In contrast, intuition-based uncertainty assessment strategies may lead to little or no learning.

The implications found for the industry and further research was:

(I) For learning to occur, the learning process may need to be aided by explicitly stated learning strategies, and frequent reminders of the goal of the learning session.

(II) There must be given special attention to the framing of the probability measures used to state uncertainty over effort. Check for adequate understanding of the concept of probability and uncertainty, give proper explanations of these terms, and issue reminders of the agreed upon definitions at regular intervals during the time of learning. It seems beneficial to support mathematical probability definitions with natural language descriptions, and oral consensus through debate of these definitions.

(III) Feedback should be given in such a way that: (1) several kinds of feedback is used and issued frequently, as a minimum it should be given at naturally occurring places; (2) the possibility of subjective interpretations on performance is avoided as much as possible; (3) it can be directly transferable as input to future uncertainty assessments, i.e. framing of the appearance to visually match the uncertainty assessment process to come, history based tendencies should be pointed out.

(IV) There are different qualities and learning strategies that are effective for learning the skill of “know how” versus learning “how uncertain is”. The design and framing, of learning environment and feedback, should therefore reflect how learning uncertainty assessment is best obtained when this is the purpose.

Preface

The fall of 2002 I started on the cand.scient. part of my education; full of hope and anticipations, but with no tangible plan for a thesis. I spent the better part of a year waiting for inspiration for THE thesis of all time to somehow pop into my head. It didn't. Though, I was sensible enough to finish (nearly) all mandatory courses. It was during this time of pondering I realized I wanted to help make things better; a course in Software Process Improvement being the main catalyst for this (as well as trying out being a study group teacher in a software engineering class). I found that, however way I tried to attack the issue, I always wound up with learning being the fundament for any improvement. Alas still no tangible plan in sight, though narrowing the problem somewhat... It may be depicted as an unsettling time, but without this growing period I see now that things may have turned out quite differently, and not necessarily for the better. I'm truly glad I didn't start on something I didn't feel any eagerness towards, just to get going with a thesis.

Coincidence, chance, and a sitcom-like occurrence, lead to a meeting with Magne Jørgensen, on September 16th 2003, where it all finally came together. During this meeting the outline of an extremely tangible thesis, investigating whether and how learning uncertainty over effort is possible, was as good as carved in stone. And not only that, the thesis outline made it possible to combine my interest in human behavior (psychology) as well as, in my view, possibly being a useful contribution to software process improvement. Although the main ideas were set, the path was pretty much threaded as I went along; and at the same time to some degree rekindling some of the once lost hope and anticipations of my former naive(er) self.

When the first day of the first experiment came, in October 2003, butterflies were flying all over the place. And when it was all over, it had left me with the reassurance that this was something I wanted to explore further, and the self-knowledge that this was something I very much enjoyed doing.

The second experiment was originally set to late January, it was held in the middle of March of 2004. The thesis was planned delivered on May the 1st, postponed to August 1st and now, finally, delivered before the next deadline, November 1st. Proving me to be just as human as the participants in my experiments and Murphy's laws, yet again, bitterly ironically accurate (things will take longer than you think; whenever you set out to do something, something else must be done first). But joke aside, the extra time has been nothing but beneficial (except for the sleepless nights and somewhat frayed nerves) to mature both in mind and in writing. And, now, in the light of hindsight's great transparency, I see that it needed to take just as long as it took.

Acknowledgments

I would like to thank the Simula Research Laboratory for funding the experiments; and supervisor Magne Jørgensen for excellent guidance, inspiration and motivation throughout the entire work on this thesis.

Thanks to Kjetil Bovim, for helping with the layout, and putting up with endless whining, and for always being a source of never-ending support and bliss; Liv Ellen Ensby for proofreading the thesis, and being a highly though of contributor on numerous (enjoyable) debates regarding school work and life in general. Special gratitude also goes to Kristin Aaser, Ingvill Fauske, Siri Jonassen, Håkon Ursin Steen, Mari Tylden for being good friends and always lending an ear when needed. Appreciation to the authors J.R. Rowling (for the Harry Potter series) and Douglas Adams (for Hitchhikers Guide to the Galaxy "fiveology") for providing a magical-world escape, when the realities of this world become too real. Lastly tanks to those who in their own way contributed during this period in my life, but didn't get their name on print; and off course, the experiment participants.

TANJA GRUSCHKE, 26TH OF OCTOBER 2004

Empirical studies on software development effort uncertainty assessment:

The role of outcome feedback in the learning process

Cand. Scient. thesis

Table of contents

Abstract	3
Preface	5
Acknowledgments	6
Table of contents	8
1 Introduction	11
1.1 Previous and Related work	13
1.2 Terminology	15
2 Overall Design of Studies	19
3 Experiment no. 1	
– The General Knowledge Questions Experiment (GQE)	21
3.1 Motivation	21
3.2 Design of study	21
3.2.1 Research questions	21
3.2.2 Measures	23
3.2.3 Subjects, Tasks and Material	24
3.2.4 The experiment process	26
3.2.4.1 An example run-through of the experiment	26
3.3 Results	30
3.3.1 Calibration learning ability	30
3.3.1.1 Summary of the calibration learning ability	47
3.3.2 Task difficulty and knowledge level	48
3.3.2.1 Correlation between knowledge level and learning ability	49
3.3.2.2 Differences in learning ability on Hard and Easy Questions	50
3.3.2.3 Initial Knowledge Level	52
3.4 Discussion of Results	53
3.4.1 The nature of the task	56
3.4.2 The nature of the feedback	57
3.4.3 The availability of the probabilities	58
3.4.4 Potential limitations	58
3.4.4.1 Cheating	59
3.4.4.2 Boredom and indifference	59
3.5 Summary and Conclusion	60
4 Experiment no 2.	
– The Software Development Experiment (SDE)	63
4.1 Motivation of study and study design	63

4.2 Design of study	64
4.2.1 Research questions	64
4.2.3 Measures	65
4.2.4 Subjects, Tasks and Material	67
4.2.5 The experiment Process	67
4.3 Results	69
4.3.1 Development and estimation skill	69
4.3.2 Ability to assess relative difference in uncertainty	70
4.3.3 The ability to assess the average level of uncertainty	71
4.3.4 The uncertainty assessment strategies	74
4.3.4.1 Main uncertainty assessment strategies	75
4.3.4.2 The general affecting the particular	76
4.3.5 Perceived skill level, the build of (unjust) confidence	77
4.4 Discussion of Results	79
4.5 Summary and Conclusion	81
5 Industry and Research Implications	83
5.1 Implication no 1:	
Aid the Learning Process	84
5.2 Implication no 2:	
Avoid Probability Angst and Arrogance	87
5.3 Implication no 3:	
Frame Feedback carefully	89
5.4 Implication no 4:	
Acknowledge the difference in learning to “know how” versus “how uncertain is”	91
6 Summary and Conclusions	93
6.1 Further Work	94
Appendix	95
Appendix A	97
Appendix B	129
Appendix C	143
Appendix D	155
Appendix E	163
Appendix F	169
References	181

I Introduction

No one knows what the future holds before they get there, but when they do it's no longer the future but the present and it quickly becomes the past. The world is an uncertain place to go about your daily business, anything can and will to some degree happen to you whether you like it or not. When the alarm clock makes its ungodly sound early in the morning telling you it's time to get up, it could just as well not have perhaps making you late and stressed the rest of "one of those days". The notion that we are puppets in a world controlled by moody, loving or vengeful gods playing with us for their own amusement has just recently started to let go, only to be replaced by the even more unyielding forces of chance (at least you could sacrifice a virgin to secure your harvest in the "good ol' days"). Although often associated with dramatic overturn and existential terror, the concept of uncertainty is far more of an everyday objective reality – neither good nor bad, it just is.

Consider the every-day business of figuring out what to eat for dinner. On your way home from work or school you stop by the grocery store. After patrolling the aisle for a while you decide on making some kind of fish dish. You try to remember if you have something in your fridge that needs to be used soon, or would go nicely with the fish you've picked out. You also wonder if your roommate is home and also would like some dinner. If he is, you'll need to buy more fish and ingredients. To reduce the possibility of buying too much food that may be spoiled, you pick up your mobile to get an answer. The battery is flat. "Oh well", you think, and decide to make dinner for two anyway. You check the time and see that if you just hurry it along a bit, you can make dinner and have it ready to eat in front of your favorite TV-show. The store is also out of sour cream, a key ingredient you need. This isn't crucial, you'll pass another store on your way home; you can buy there. After battling with the after work rush line in the cash register, the time schedule has become considerably tighter. On your way home it starts to rain, not much but enough to make you completely forget about the sour cream when picking up the pace to get home. Well inside the dry comforts of home, you find out a mutual friend of you and your roommate is contemplating ordering a pizza.

After debating the issue, you all decide that fish would be nicer than pizza. It is now, however, very uncertain if you have enough food for everybody. It depends on how hungry you all are. Discovering the missing sour cream, as well as milk and coffee you completely forgot about in the hassle in the store, and nobody wanting to go out in the rain, you'll have to rethink what to make of your fish. Checking the cupboard, the once so rash decision to buy 5kg rice comes once again in handy. Making the fish dish on the fly, you find yourself with a vast amount of decisions to be made all effecting each other as well as the outcome: how spicy should it be, is anybody allergic to something, would tomatoes overpower the taste of the fish, how hot should the pan be, how long should it simmer, in which order should you add your ingredients, and the list goes on and on. In the end everybody likes the fish and fills up on rice to compensate the small amount. After a little while you decide ice cream is worth going out in the rain for, and you buy the coffee previously forgotten but substitute the milk for orange juice.

Every step of the way in making dinner is filled with uncertainty, you either plan for risks that may happen or compensate when an unexpected situation occurs. This is a easy process: dictions are either being made intuitively, e.g. setting the hot plate on full; others need some consideration, e.g. where to get a hold of the missing sour cream; others need to be debated with others, e.g. fish versus pizza for dinner, and so on. Any number of things could have happened making the outcome completely different. Your friend might not have been hungry as he had eaten before he came over; you might have decided that a bit of rain never hurt anybody and went back to the store to by sour cream; nobody was home; not wanting to make a big deal out of dinner after all you toast some bread and curl up in front of the TV; the bargaining of fish versus pizza could have resulted you all making pizza with fish on; the lid on the chili bottle might have fallen off making dinner uneatable thereby having to ordering pizza. In any rate what to make for dinner isn't scary, and you compensate relatively well according to the feedback you receive from the world around you, as we all have from the dawn of mankind. However the outcome, or the path to get there, turned out, the main goal of finding what to eat for dinner (to add the daily minimum of nutrients to the body so that it can continue functioning) was reached.

In software development, as in any field concerned with producing a finished product, the need for knowing the uncertainty connected to estimates of time and money is fundamental. To illustrate, imagine a project estimated to cost \$1.000.000. Before starting the project the customer may want to know how likely it is that the cost will exceed the expected benefit of the project, which is expected to be about \$ 1.500.000. This requires knowledge about the uncertainty of the cost estimate.

It has been demonstrated that people tend to underestimate uncertainty [1, 2] and that the learning from typical on-the-job experience is poor [3]. This thesis focuses on people's, in particular software developers, ability to improve their uncertainty assessments in learning friendly environments. One motivation for this is that if learning friendly environments do not lead to learning, we should not expect learning to take place in more realistic job environments. Another focus of the thesis is the analysis of learning strategies and characteristics of those who learn from feedback.

The thesis is structured as follows. Section 2 covers the overall design of the studies. The detailed design, the results and the discussions of the first and second experiment are included in section 3 and 4 respectively. Section 5 discusses the implications of the results. Section 6 provides a summary, concludes and outlines further work.

1.1 Previous and Related work

This thesis is of multidisciplinary nature. The main domain is software engineering, but as the focus is the possibility of learning uncertainty, the need for psychological and pedagogical insight is high. If overlooked the value of the analyses done in this study would have been greatly diminished (if not useless due to extreme naivety), as they would have missed fundamental issues in human behavior affecting the results.

The accuracy of software effort intervals has only been studied a few years, as pointed out in [4]. The number of studies done in the context of software development and engineering is also relatively few, compared with the vast amount of research done in e.g. decision-making and human judgment concerning the possibility of, and causes for, (un)realism in uncertainty assessment. The need for a greater body of studies focusing on the software engineering field, particularly realism in uncertainty of effort estimates, is in great demand, as called for in [1, 2, 4-12]. It is also directly linked to and useful for the software industry, as the industry itself has issued concern on the lack of precision in estimation cost and time, stated already in 1968 by Pietrasanta at IBM Systems Research Institute: *"Anyone who expects a quick and easy solution to the multi-faceted problem of resource estimation is going to be disappointed. The reason is clear: computer program system development is a complex process; the process itself is poorly understood by its practitioners; the phases and functions which comprise the process are influenced by dozens of ill-defined variables; most of the activities within the process are still primarily human rather than mechanical, and therefore prone to all the subjective factors which affect human performance"* [13]. There has been no sign of a "silver bullet" since then, and there presumably never will be.

Overconfidence and optimism seems to be the weak link in effort estimation, as studies has shown that when software managers are asked to create 90% certainty confidence intervals, the actual hit rate is about 60-70% [1]. Research in human judgment has also arrived at this high level of overconfidence, see e.g. [14-17]. Forecasting in subjective probability also show overconfidence and the lack of realism, see e.g. [18-20]. Jørgensen et al. [8] found the most frequently used method in the software industry to arrive at a uncertainty assessment of a effort estimation to be expert judgment. As this implies that many decisions are made on the basis of gut feeling and intuition, i.e. a non retrievable and to some degree uncontrollable mental process, there is a need for the understanding of how human cognition processes work both in general and within experts in a specific field. There has been much written and researched on the topic of expert performance, e.g. [21-23], both because they represent a small almost special case of performance and that it is believed that experts outperform the lay person [21]. In contrast, other studies have shown that expert performance doesn't always exceed the performance of the naive practitioner or simple models, described e.g. in [22].

The software industry has been haunted by overconfidence in many aspects, especially project cost overrun by 30-40% [12]. The popular Chaos Report by the Standish group indicates cost overruns to be much higher, with most projects suffering 189% overrun. This report is however something of a media stunt rather than based on freely accessible sound evidence of their numbers, as criticized in [12] and [24]. Be that as it may, the point however is probably to add an extreme inkling so as to draw attention to the sad state of the software industries ability to be realistic. 30-40% cost overrun is high enough to cause alarm. There is a need to get a better overview of the uncertainty aspect. It's possible that an overrun could to some degree be expected and therefore either planned for or at any rate not cause an upheaval in the organization and the marked, provided managers had a tangible level of uncertainty (rather than a vague "fairly sure") to relate to. It is therefore crucial to uncover the mechanisms that lurk underneath the overconfidence and optimism if something constructive is to be done with the de facto state. Pulling the forces from several fields of study on decision making, human behavior, forecasting, experts, cognitive processes etc. is a step towards understanding and ultimately reducing this crippling overconfidence.

1.2 Terminology

Analytic process “A series of steps for processing information according to rules. An analytic process is explicit, sequential, and replicable.”[25]

Bias “A systematic error; that is, deviations from the true value that tend to be in one direction. Bias can occur in any type of forecasting method, but it is especially common in judgmental forecasting. Researchers have identified many biases in judgmental forecasting. Bias is sometimes a major source of error.” [25]

Calibrate “To assess the extent to which estimated probabilities agree with actual probabilities.”

Cognitive dissonance “An uncomfortable feeling that arises when an individual has conflicting attitudes about an event or object. The person can allay this feeling by rejecting dissonant information. For example, a forecast with dire consequences might cause dissonance, so the person might decide to ignore the forecast.”[25]

Confidence The state of being or feeling certain, or having certainty of an act or event.

Confidence interval “An expression of uncertainty. The likelihood that the true value will be contained within a given interval. In forecasting, the term refers to the uncertainty associated with the estimate of the parameter of a model while the term *prediction interval* refers to the uncertainty of a forecast.”[25]

Correlation (r) “A standardized measure of the linear association between two variables. Its values range from -1, indicating a strong negative relationship, through zero, which shows no relationship, to +1, indicating a strong positive association.”[25]

Decomposition “The process of breaking a problem into sub problems, solving them, and then combining the solutions to get an overall solution.”[25]

Effort estimate “Forecast (predictions) of expected effort. Without any further description, the precise meaning of this term may be unclear, e.g., whether ‘estimate’ means the ‘modal’ (‘most likely’), the ‘median’, or, the ‘mean’ value of a distribution of possible effort usage [5]. We therefore try to avoid this term when we need to be precise, e.g., we use the term ‘*estimate of most likely effort*’ when the modal value of the distribution of possible effort usage is meant.”[26]

Effort prediction interval “A minimum-maximum interval for effort, with a connected confidence level of including the actual effort value. For example, an estimator may estimate the most likely effort to be 1 000 work-hours and the probability of including the actual effort in the effort interval from 600 to 1 500 work-hours to be 90%. Then, the 90% confidence effort prediction interval is [800; 1 500] work-hours.

Effort prediction intervals are used frequently in the planning and budgeting of software projects [27].”[26]

Effort Uncertainty “A description of the expected uncertainty in use of effort. The type of description of uncertainty applied in this paper is based on *effort prediction intervals*.”[26]

Environment “Condition surrounding the situation. The environment includes information about the ranges and distributions of cues, the correlations among them, and the relations between the cues and the event being judged. The environment also includes constraints on information available to the judge and on actions the judge may take, as well as time pressures, requirements for documentation, and anything else that might affect cognitive processes.”[25]

Environmental feedback “(Or task feedback) is information about the event to be predicted, including the factors that may influence the event and their relationship to the event.” [26]

Estimate of most likely effort “The effort value believed to have the greatest chance of being equal (or close to) the actual effort.” [26]

Estimation outcome feedback “Information about the discrepancy, if any, between the actual effort (the outcome) and the estimated most likely effort. The information about this discrepancy can be used to improve the accuracy of the assessed level of effort prediction intervals. Estimation outcome feedback is frequently the only type of feedback received in software projects, i.e., there is typically no systematic investigation of reasons for higher or lower uncertainty.” [26]

Experiments “Changes in key variables that are introduced in a systematic way to allow for an examination of the effects that one variable has on another.”[25]

Expertise “Knowledge or skill in a particular task.”[25]

Feedback “Information that experts receive about the accuracy of their forecasts and the reasons for the errors. Accurate, well-summarized feedback is probably the primary basis experts have for improving their judgmental forecasts. The manner in which feedback is provided is critical because people tend to see what they want to see or what they expect. When feedback is well summarized, frequent, and when it contains explanations for the events, judgmental forecasters can become well-calibrated.”[25] Benson and Önköl [18] lists the most relevant types of feedback in judgmental forecasting as: Outcome feedback, Performance feedback, Process feedback and Environmental feedback.

Framing “The way a question is asked or a statement stated. Framing can have an important effect upon subjects’ responses, so it is important to ensure that questions, instructions, or feedback are worded properly.”[25]

GQE The General Knowledge Question Experiment

Heuristic “From the Greek word, meaning *to discover* or *find*. Heuristics are trial-and-error procedures for solving problems. They are simple, mental operations that conserve effort.”[25]

Hindsight bias “A tendency to exaggerate in hindsight how accurately one predicted, or would have been able to predict by foresight. Sometimes referred to as the “I knew it all along” effect. Forecasters usually remember that the forecasts were more accurate. Because of hindsight bias, experts may be overconfident about later forecasts.”[25]

Hit rate “The percentage of forecasts of events that are correct.”[25]

Intuition “A person’s immediate apprehension of an object without the use of any reasoning process. An unstructured judgmental impression. Intuitions may be influence by subconscious cues. When one has much experience an there are many familiar cues, intuition can lead to accurate forecasts. Based on the research literature, however, it is difficult to find published studies in which intuition is superior to structured judgment.”[25] In this thesis intuition is regarded as the capacity or ability of direct knowledge and immediate insight without observation or reason. “Intuitive thinking is perception-like, rapid, effortless, [while] deliberate thinking is reasoning-like, critical and analytic” Kahneman restated in [28].

Optimism “A sate of mind that causes a respondent to forecast that favorable events are more likely to occur than is justified by the facts.”[25]

Outcome feedback “Information about an outcome corresponding to a forecast. For example, how often does I rain when the weather forecaster says the likelihood is 60%”[25]

Overconfidence “A state of mind that causes a forecaster to think that the probability that a forecast is correct is greater than the actual probability. This leads prediction intervals to be too narrow. Experts are overconfident because of various biases, such as an unwarranted feeling of control or a desire to see things turn out well.”[25]

Performance feedback Information about the accuracy of the forecaster’s predictions. It is derived from the forecaster’s predictions and the outcomes that occur; e.g. calibration feedback.

Prediction “A statement regarding future events that are unknown to the forecaster. Generally used as a synonymous with forecast.”[25]

Prediction interval “The bounds within which future observed values are expected to fall, given a specified level of confidence. For example, a 95% prediction interval is expected to contain the actual forecast 95% of the time. However, estimated prediction intervals are typical too narrow for quantitative and judgmental forecasting methods.”[25]

Process feedback Gives information about the forecaster’s cognitive processes. It includes information about the evidence perceived by the forecaster,

how the forecaster utilizes evidence in developing predictions, and information about the predictions themselves.

SDE The Software Development Experiment

Uncertainty “The lack of confidence associated with a forecast, which can be represented by a prediction interval. Also, the lack of confidence about a parameter estimate, which can be represented by a confidence interval. Uncertainty *cannot* be represented well by statistical significance.”[25]

2 Overall Design of Studies

The two studies conducted in this master thesis research are experiments. There are two main reasons for the use of experiments, rather than e.g. case study in a real industry setting. The experiments are tailored so that examining the ability to learn uncertainty is not distracted by other less interesting factors, as well as minimizing the potential biases due to too many influencing factors. The use of an artificial experiment environment enhances two important issues. First, it helps to eliminate factors that are less interesting in this context, that otherwise could have clouded the results in an uncontrollable manner and would possibly be untraceable later. By stripping the environment of superfluous influence it is believed that it is more likely that the results display a more accurate picture of the ability to learn. This is not to say that the potential contamination of the results has been totally eliminated from unwanted factors, as an artificial setting brings its own baggage, however the risk is reduced. The second rationale of using experiments concerns the actual ability to learn. There is considerable noise found in any real world setting, which makes learning difficult. Consider e.g. the lack of feedback in an industry setting (feedback being one of the key factors of learning) as well as interruptions and other pressing issues that need attention before learning can be addressed by the individual. The focal point of the studies is the ability to learn, specifically the possibility of learning uncertainty assessment. If the subjects display poor signs of learning or progress in a learning stimulating context, the likelihood of observable learning in a realistic setting is possibly very small. By using the experiment setting, we can in a controlled manner test if it is at all possible to learn uncertainty assessment. It is believed that this is a meaningful way of studying necessary factors of learning, although not for studying sufficient conditions.

Both experiments have a dual purpose, being an independent study and being a preliminary study for the next experiment. “The General Knowledge Questions”-experiment (hereafter referred to as GQE) is the first study, and attacks the issue in a very general perspective. It was the first possibility to test whether more realistic uncertainty assessment is learnable by using

outcome feedback. If it had turned out that given the designed circumstances learning was poor or nonexistent the next experiment would have had to further aim at uncovering necessary factors of learning. The second study, “the Software Development Experiment” (hereafter referred to as SDE), moves from a general perspective of uncertainty learning to the more narrow and specific field of Software engineering, with its special challenges and quirks. This experiment uses the lessons learned and findings of the first study in its design. This study used students of informatics as subjects of investigation. In a third study, that was supposed to bring the research closer to the real world, professional developers were to be used. It however became apparent during the course of the work with the two first experiments that the work load of conducting this third experiment was beyond the demanded possible work load of a cand.scient. thesis.

The two research questions worded here are the general questions of the thesis.

RQ1: Given a favorable learning environment, can people learn to better calibrate uncertainty estimates when provided with feedback on performance?

RQ2: Is there a relationship between a participant’s learning strategy and the amount of learning observed?

3 Experiment no. 1

- The General Knowledge Questions Experiment (GQE)

3.1 Motivation

In this study the objective was to find if it was at all possible to learn uncertainty assessment. This was not given as obvious initially. Other studies indicates that if learning is to occur it will strongly dependent on how feedback is given, the kind of feedback given, and when and how often it is given to the participants, see e.g. [3, 18]. Because of the openness and many possible outcomes due to highly different design decisions in other studies, an experiment tailored to the investigation of learning uncertainty was strongly needed. By designing and conducting an experiment tailored to meet the research questions in this thesis, greater control over factors contributing to the results was made possible. It also made possible to exclude factors that may have contributed in a negative or distracting manor on the subjects and or the results. The experiment's sub-objective was to serve as a training exercise and learning experience in experiment design, planning, organizing and execution. In addition, it was to contribute with design input to the upcoming SDE. This input was to be of an informal nature, mainly first impressions and observed action-reactions from this experiment.

3.2 Design of study

The design of the study was approached in an evolutionary manor, i.e. several smaller pilot studies investigating possible effects of different design decisions were held.

3.2.1 Research questions

The research questions in this experiment are a refinement to a particular situation based on the general research question RQ1: *“Given a favorable learning environment, can people learn to better calibrate uncertainty estimates when provided feedback on performance?”*. This experiment's research questions concretize into a specific tangible setting in investigating the possibility of

learning uncertainty assessment. The favorable learning environment can be subject to many possible realizations, and in this respect is as much as a subject of investigation as the possibility of learning realistic uncertainty assessment. The construction and shape of the learning environment is alpha and omega for the results, both positive and negative. This is due to the fact that the outline of the environment is the key to discovering the possibility of learning, in that only a environment that holds the necessary factors for acquiring a skill will eventually expose it. Whether or not the learning of uncertainty is possible can be hidden by an inadequate learning environment. A major aim of this experiment is therefore to find sufficient learning enabling design features. If eventual learning is observed, one can assume that the environment is favorable in the context of learning. In that event, the environment can be stripped of different factors to eventually discover what the minimum requirements for uncertainty assessment learning are.

The favorable learning environment designed in this experiment contained, in short, these features:

- The use of general knowledge questions with four alternatives as the task to be performed by the subjects. The believed correct answer to these questions can be used to measure uncertainty assessment by the participant. The correctness of the task, and the completion of the task, can be measured objectively.
- A stable level of difficulty on a pile of questions; i.e. questions of varying difficulty are not mixed together.
- The level of difficulty is known to the participant, given by an understandable money label.
- Immediate feedback of correctness of answer to the questions, when answer and uncertainty assessment has been given by the participant.
- Individual performance feedback given on natural occurring pauses in the question answering process:
 - After the answer of a question is given.
 - After the completion of questions of the same difficulty.
 - Before the beginning of the second day of the experiment.
- The individual performance feedback is given immediately after a natural occurring pause in an unbiased formal statistical form.
- Informal coaching is given by the experimenter on the interpretation of the statistical performance results tailored to the individual.
- Thorough run through of the experiment process and motivation given individually before a participant starts the experiment.

Based on the RQI, the research questions specific for this experiment are as follows:

1. Given the favorable learning environment outlined above, how good are people at assessing uncertainty initially?
2. Given the favorable learning environment outlined above, are people able to learn to be better calibrated if initially being over or under-confident?
3. Does the level of difficulty on a group of questions affect the possibility of learning to be better calibrated?
4. Does the personal level of knowledge affect the possibility of learning to be better calibrated?

3.2.2 Measures

The different measures conducted in the experiment are worded in this section. Both, measures used to calculate the feedback given to the participants, and the measures used in the analysis of the results, are described.

The uncertainty assessment the participants were to do on their given answer to a question was given in a predefined list. The levels set by this list are given in table 1. These levels are also the basis for all calculations and analysis of calibration performance.

Level	Uncertainty Intervals
Level 0	[0.25]
Level 1	(0.25, 0.40]
Level 2	(0.40, 0.60]
Level 3	(0.60, 0.75]
Level 4	(0.75, 0.90]
Level 5	(0.90, 0.99)
Level 6	[0.99, 1.00]

TABLE 1 UNCERTAINTY LEVELS

Definitions relating to the use of the uncertainty levels:

${}_{up}Lev_n$ = upper limit Level n

${}_{lo}Lev_n$ = lower limit Level n

$QLev_n$ = the number of questions with assessed uncertainty of Lev_n

$HitLev_n$ = the number of correct answers with assessed uncertainty of Lev_n

$HitRateLev_n = HitLev_n / QstLev_n =$ ratio of portion correct on a $Lev_n =$ hit rate

When calculating the feedback to be given to the participant, as well as calculations used in the analysis of the results, these definitions were used:

$AcceptedRate(Lev_n) = {}_{lo}Lev_n < HitRateLev_n < {}_{up}Lev_n =$

the hit rate is inside the level boundaries

$Overconfidence(Lev_n) = HitRateLev_n < {}_{lo}Lev_n$

$Underconfidence(Lev_n) = HitRateLev_n > {}_{up}Lev_n$

When analyzing the task difficulty, calculations were performed on piles of questions. Definitions of calculations used in this thesis:

Q_p = number of questions in a pile

Hit_p = number of correct answers in a pile

$HitRate_p = Hit_p \setminus Q_p =$ percentage of questions correct answers in pile p

The feedback given to the participants were automatically calculated when a pile was finished. The participants were helped in interpreting the results. As small sample size occurred, the informal natural language description of each uncertainty level was also considered during coaching sessions (see table 2).

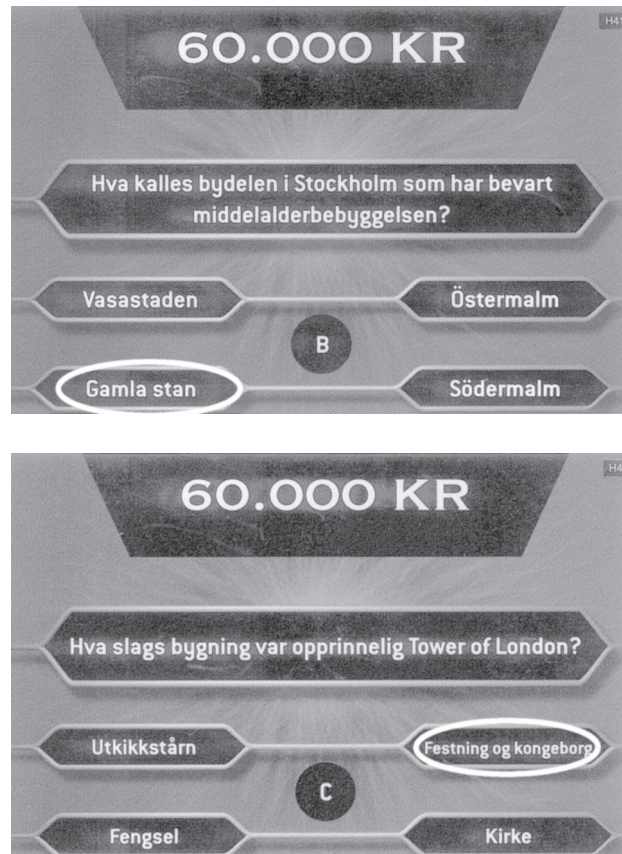
3.2.3 Subjects, Tasks and Material

Several lines of communications were used to find participants for the experiment. Notices were put up at different locations at the University of Oslo, it was advertised in a software engineering class, different mailing lists were used, and the grapevine. In all forms of communications it was south after people interested in trivia question games. The 15 first participants wanting to participate were used. The pay was set to a 1000kr. The work was piloted to take about 8 hours over two days.

The task was to answer trivia questions from the board game “Who wants to be a millionaire?™”. The game contains question cards with four alternative answers to each question. The correct answers are provided on the back of each card. Each card has an odd and an even side. The questions are grouped into difficulty levels labeled by a money amount. A pile with the same money amount printed on the question cards contain about 160 questions, about 80 on the odd side and about 80 on the even side. In this experiment the piles with 10.000, 20.000, 40.000, 60.000, 80.000 and 100.000 kr printed on them

were used. The cards of the board game ranges from 1.000kr to 2.000.000kr, indicating the levels used in the experiment correspond to less than intermediate difficult questions. The themes of the questions in the game are general knowledge genre like sports, history, politics, science, literature and so on. See figure 1 for a example card.

Figure 1 A question card used in the experiment, here you can



see the front side and back side of the same card. The letter in the center of the card indicates the correct answer to question on the opposite side (here it has been highlighted by a ring for clarity).

The experiment was conducted by handing out piles of questions to the participants. The participants themselves register the money amount of the received pile and the experiment day (day 1 or day 2) on the web support system. The experiment was held in one of the Department of Informatics' computer labs. The internet browser installed on the computers in the lab was used to access the web support system used in the experiment by the participants.

3.2.4 The experiment process

The experiment took place over a period of about 5 days. The participants partook on two consecutive days, using on average 4 hours each day. Each participant answered either the odd or the even side of a question pile on day one, and then the opposite side of that pile on day two. The sequence the money labeled question piles were handed out to the participants were unsystematic; as the sequence was determined by which piles were available and had not yet been answered by the participant. The same sequence was used on day two. On the first day each participant was coached individually and given an instruction booklet containing in depth information about the experiment and practical information on the web support system (appendix C). They were explained that their ratio of correct answers on a questions assessed to be e.g. level 2 (41-60%) should lie in this interval at the end of a round.

3.2.4.1 An example run-through of the experiment

When handed a pile of questions the participant registered the money label and the experiment day (day 1 or day 2) on the web support system, and started answering the questions. For each question she was required to decide on an alternative and assess the uncertainty attached to the given answer. The level of uncertainty was predefined to seven set levels; the levels can be seen in table 2. In figure 2 a screen shot of the window the participant was to register his answer and uncertainty level is shown. As can be seen from the screen shot, the participants were presented with the uncertainty levels and their language description when answering every question.

Uncertainty levels	Description in natural language (translated from Norwegian)
25%	No idea
26-40%	Some idea
41-60%	Fifty-fifty
61-75%	Pretty sure
76-90%	Fairly sure
91-98%	Really sure
99-100%	Without a doubt

TABLE 2 THE UNCERTAINTY LEVELS AND THEIR NATURAL LANGUAGE DESCRIPTION



Figure 2 When answering a question, the uncertainty levels are shown with a natural language description of each level (in Norwegian, see table for English translation)



Figure 3 The performance feedback given immediately after a pile of questions is finished

After choosing the preferred alternative and the uncertainty level, the participants themselves check the correctness of the answer, and registered it on the web page before moving on to the next question. When all the questions in the pile (either the odd or even side) are answered, feedback was given; see figure 3 of an example snapshot of the feedback given after a pile. As the snapshot in figure 3 shows, the participants were given the number of questions they had labeled with the different uncertainty levels and the number of correct answer on each of these levels. The hit rate (the percentage correct answers for a given level) is also shown, making it easy to determine if the hit rate was in comparison to the level boundaries. The experimenter helped interpret the statistical results displayed and occasionally gave tips on future uncertainty assessment, e.g. “you’re a bit under-confident on the lower levels and overconfident on the upper” The participant was then handed his next pile of questions. At the beginning of day 2 the participants were given a paper summary of all their results from day one, i.e. all the statistical feedback given for each pile as shown for one pile in figure 3. Commentary and coaching from the experimenter was given orally when the summary was handed to a participant. This rundown contained a general analysis on their level of confidence, making the participants aware of their predisposition to be pessimistic, optimistic or already relatively adequate at the different levels.

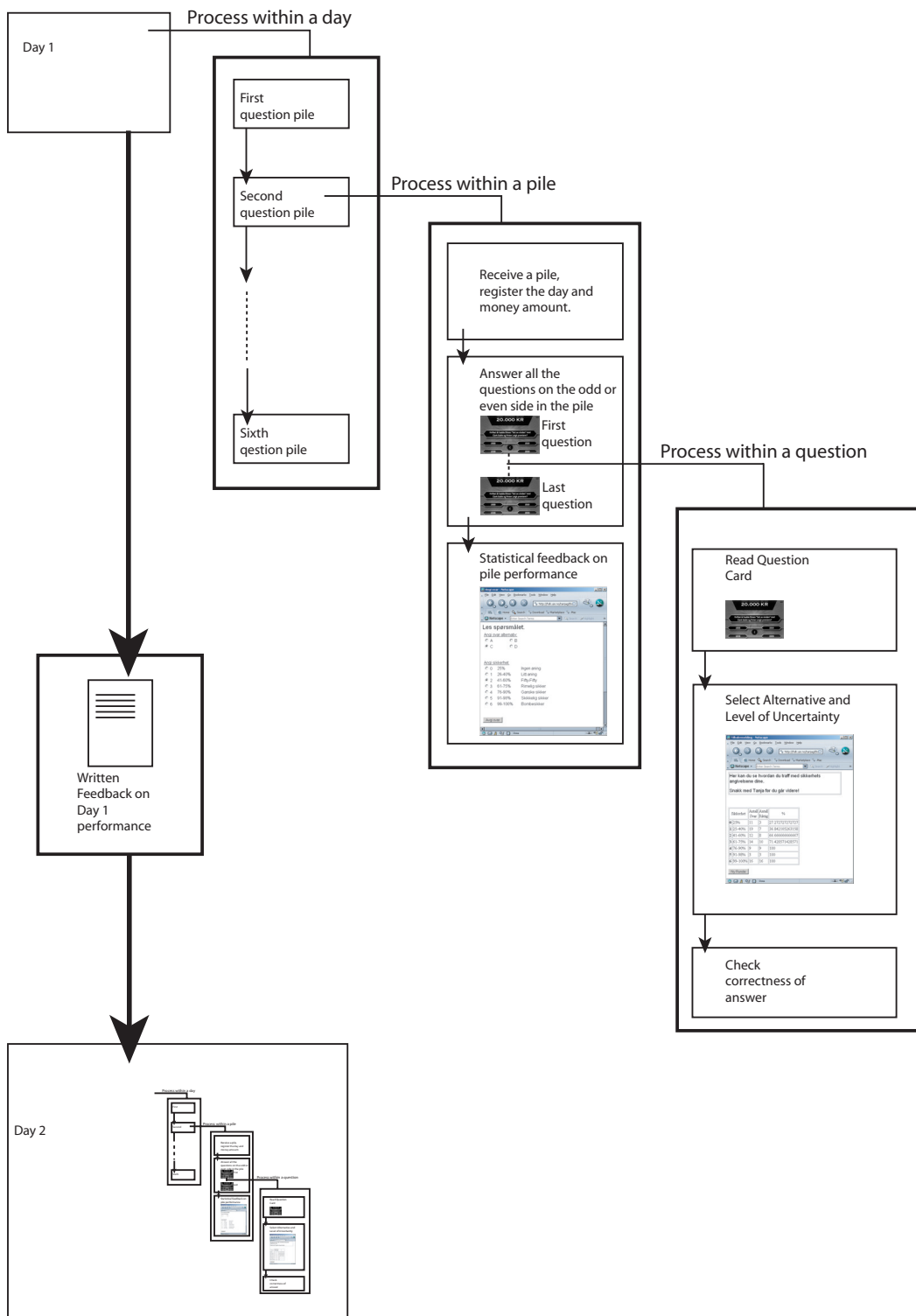


Figure 4 Visualization of the experiment process

3.3 Results

3.3.1 Calibration learning ability

To analyze the ability to assess uncertainty and the calibration improvement (learning) after outcome feedback, graphs showing adjustment over time were created for each of the fifteen participants. The *learning graph* shows the progress for a given participant throughout the experiment. It visualizes how well they calibrated their assessment of uncertainty to the real level of uncertainty, during the course of the experiment, by plotting the hit rate. The hit rate is the number of correct answers on a level divided by the total number of questions on that level, i.e. the percentage of correct answers. Determining if a participant was actively trying to adjust according to the feedback received is much easier when performance is visualized, as opposed to only viewing the numerical data. The learning graph consists of comparison points for each uncertainty level.

The comparison points in the graph are defined as follows:

- 1) The HitRateLev_n is compared with the limits for Lev_n for P
(Comparison Point 1).
 $\text{Lev}_n = \{\text{level } 0, \dots, \text{level } 6\}$ and $P = \{1^{\text{st}} \text{ pile of the day}, 2^{\text{nd}} \text{ pile of the day}\}$
- 2) The comparison at Comparison Point 2 is conducted similarly,
but for $P = \{3^{\text{rd}} \& 4^{\text{th}} \text{ pile day one}\}$
- 3) The comparison at Comparison Point 3 is conducted similarly,
but for $P = \{5^{\text{th}} \& 6^{\text{th}} \text{ pile day one}\}$
- 4) The comparison at Comparison Point 4 is conducted similarly,
but for $P = \{1^{\text{st}} \& \dots \& 6^{\text{th}} \text{ pile day one}\}$
- 5) Similarly for day two.

The rationale for including two piles in a comparison point was to ensure a large enough sample size at each level. As the number of questions per pile is about 80, and these are spread over the seven uncertainty levels, the sample size became frequently too small. This could make the calculated hit rate untrustworthy as percentiles easily become biased when sample size is small. To support the process of determining the calibration ability of a participant, and if that participant improved performance (learning), calibration graphs was also used in addition to the learning graphs. The calibration graphs display the total hit rate for each level on day 1 and 2. This is the same information displayed in the day 1 and day 2 columns in the learning graphs, performance improvement from day to day is however easier to spot in the calibration graphs. The two graphs complement each other in that one shows information that the other hides, and vice versa. The sum hit rate, plotted in the calibration graphs of a day, can hide unfavorable fluctuations through

the course a day. This can in fact point to a conclusion of (no) learning for the participant in question, when the opposite may be the case. By only viewing the sum, it can give the visual impression of bettered calibration from day to day if the comparison points of a day are both over and under the accepted rate. The amount of information displayed in the learning graphs, as performance for each uncertainty level is displayed, can lead to missing overall performance tendency due to the high level of detail.

There should be issued a word of caution to the comparison points in the learning graphs. Some of them have small samples; the percentile may lead to a biased visual and numeric outcome (the hit rate). The numeric numbers of questions answered and questions answered correctly on a level were used as support when sample size at a comparison point was small. Points with sample size lower than 12 questions are marked in the graphs with a jagged line underneath. Points with sample size lower than 5 only state “NeD” (Not enough data), and does not give any information on hit rate. The coached guidelines, i.e. the natural language descriptions, are also taken into account when analyzing the graphs.

As it would be distracting to the flow of this thesis, the analysis of each participant is placed in Appendix A. The learning and calibration graph for each participant are given here.

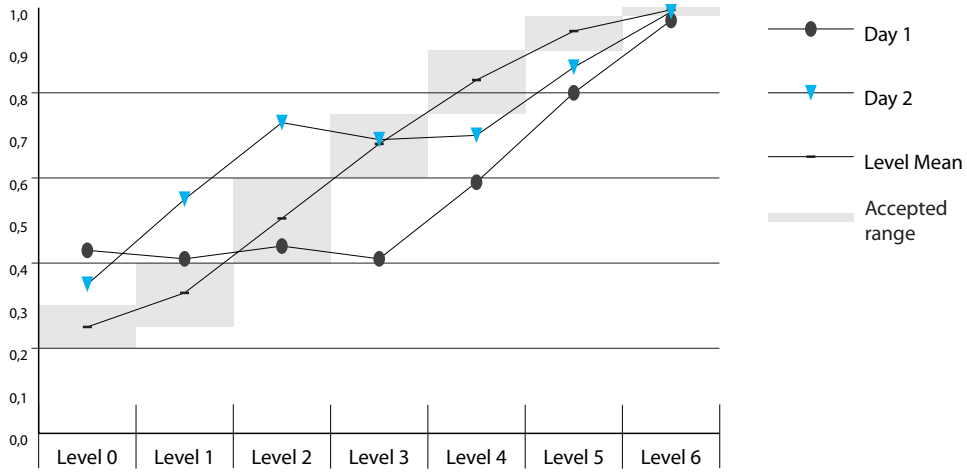


Figure 5 Participant I - Calibration graph

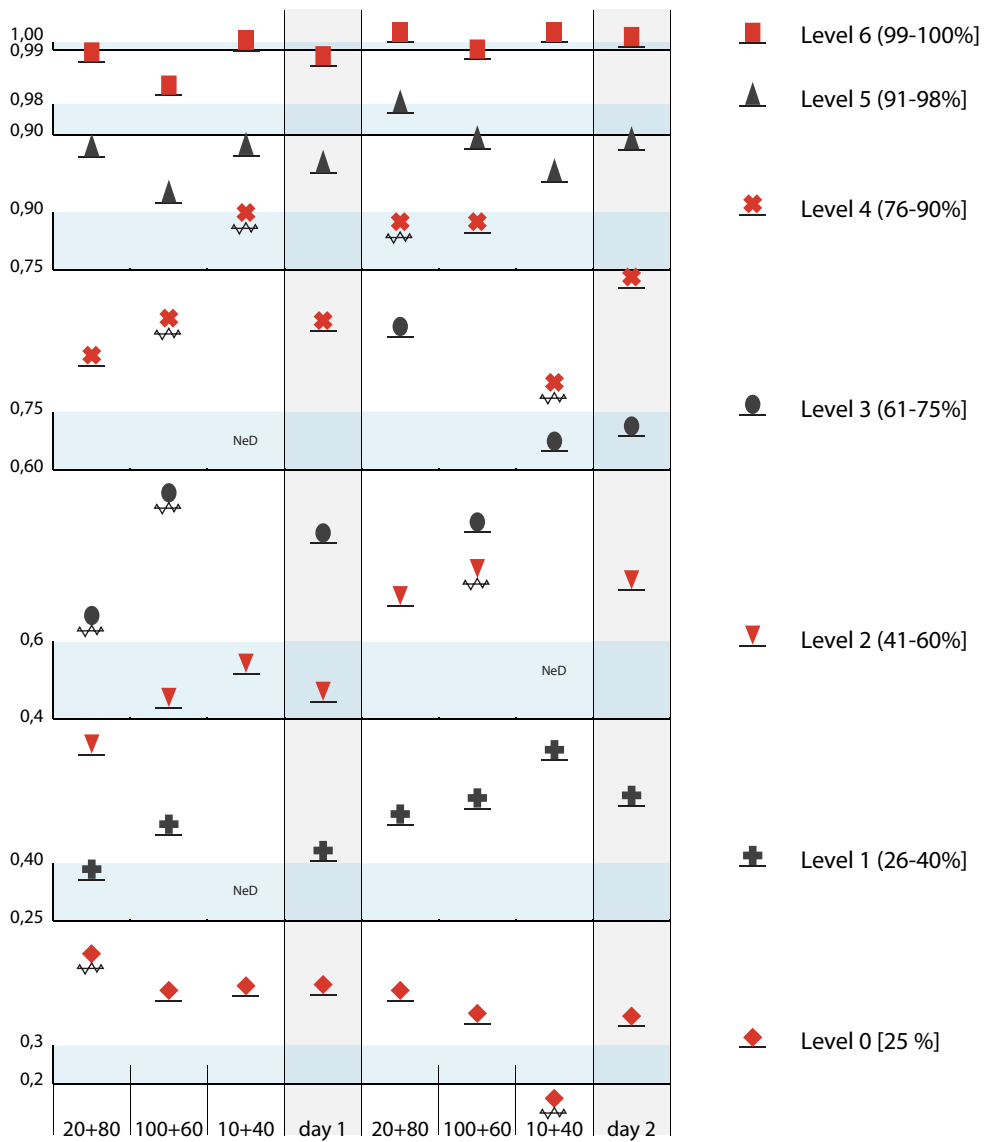


Figure 6 Participant I - Learning graph

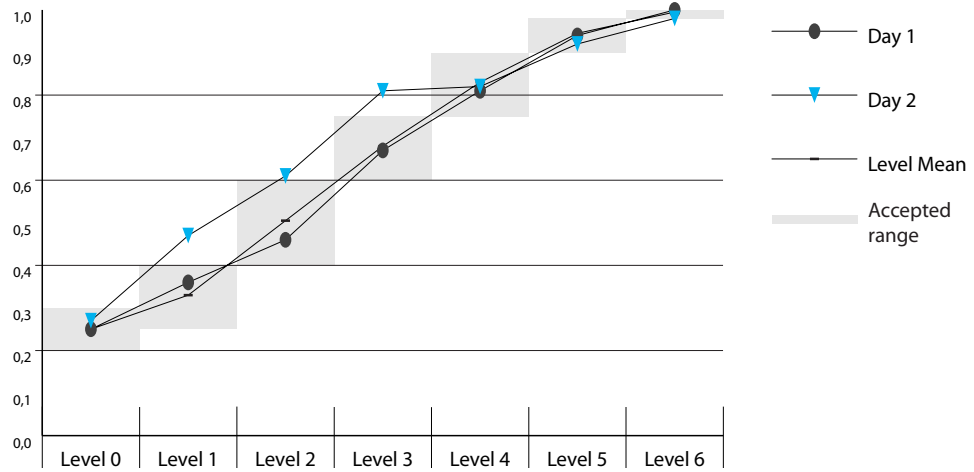


Figure 7 Participant II - Calibration graph

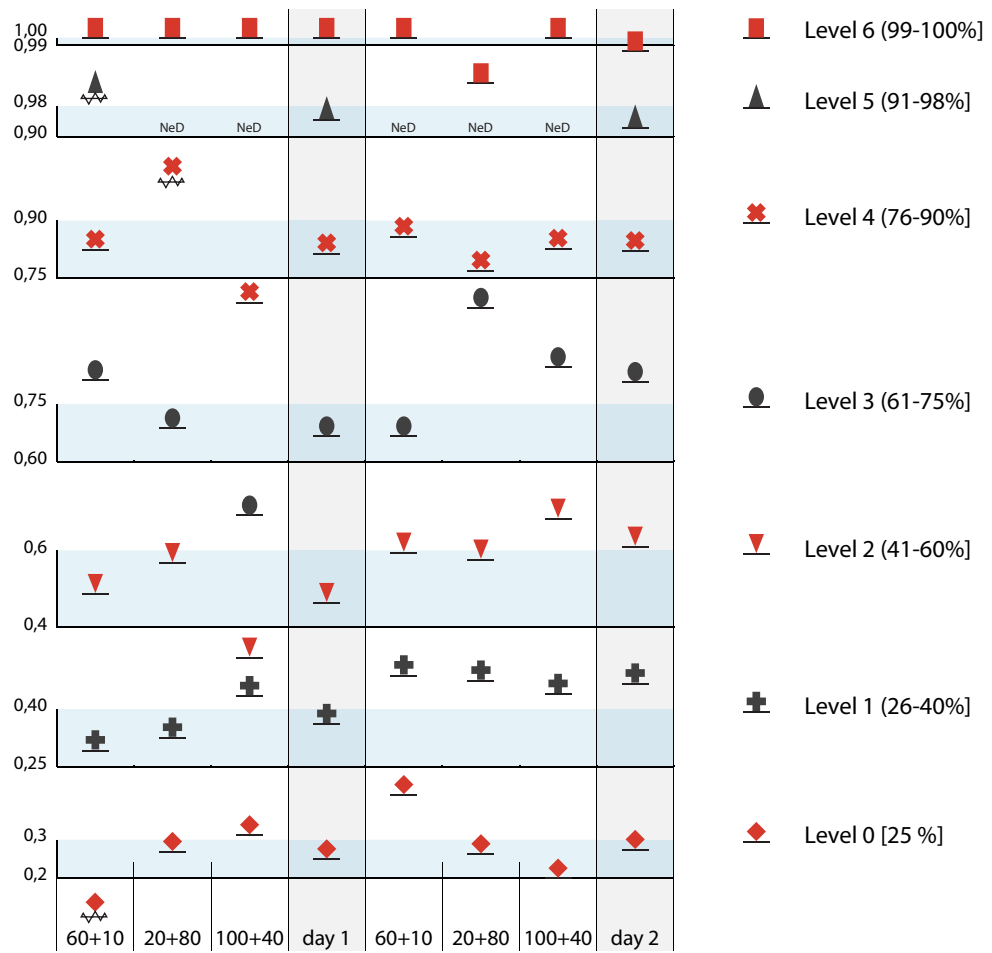


Figure 8 Participant II - Learning graph

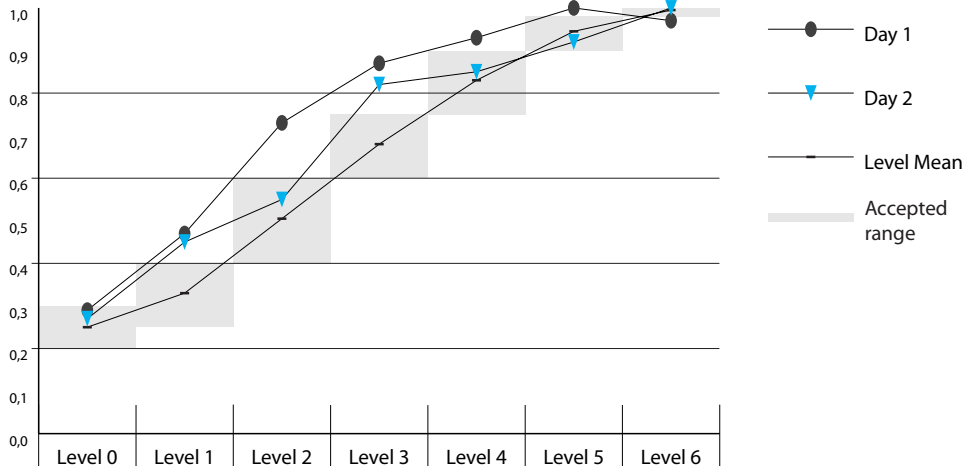


Figure 9 Participant III - Calibration graph

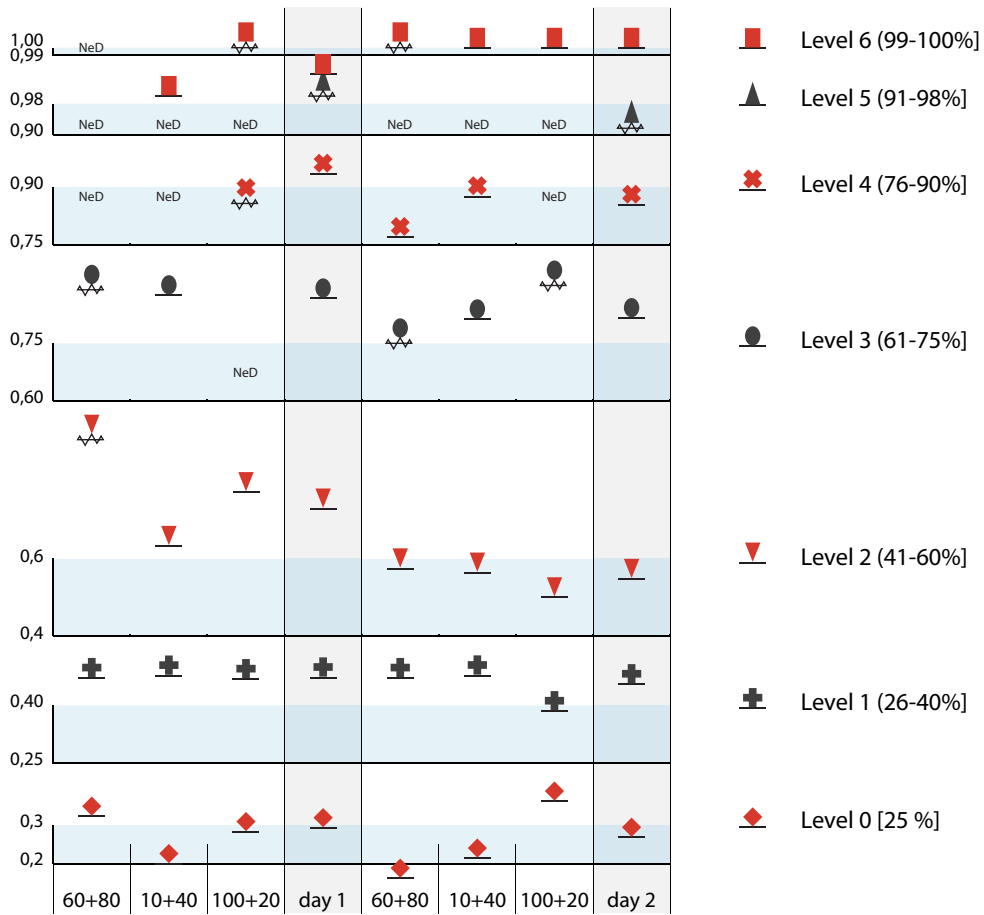


Figure 10 Participant III - Learning graph

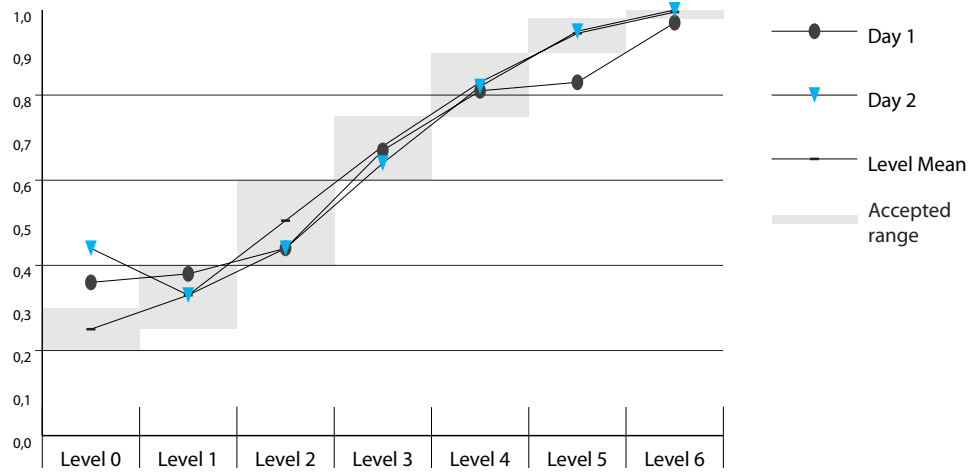


Figure 11 Participant IV - Calibration graph

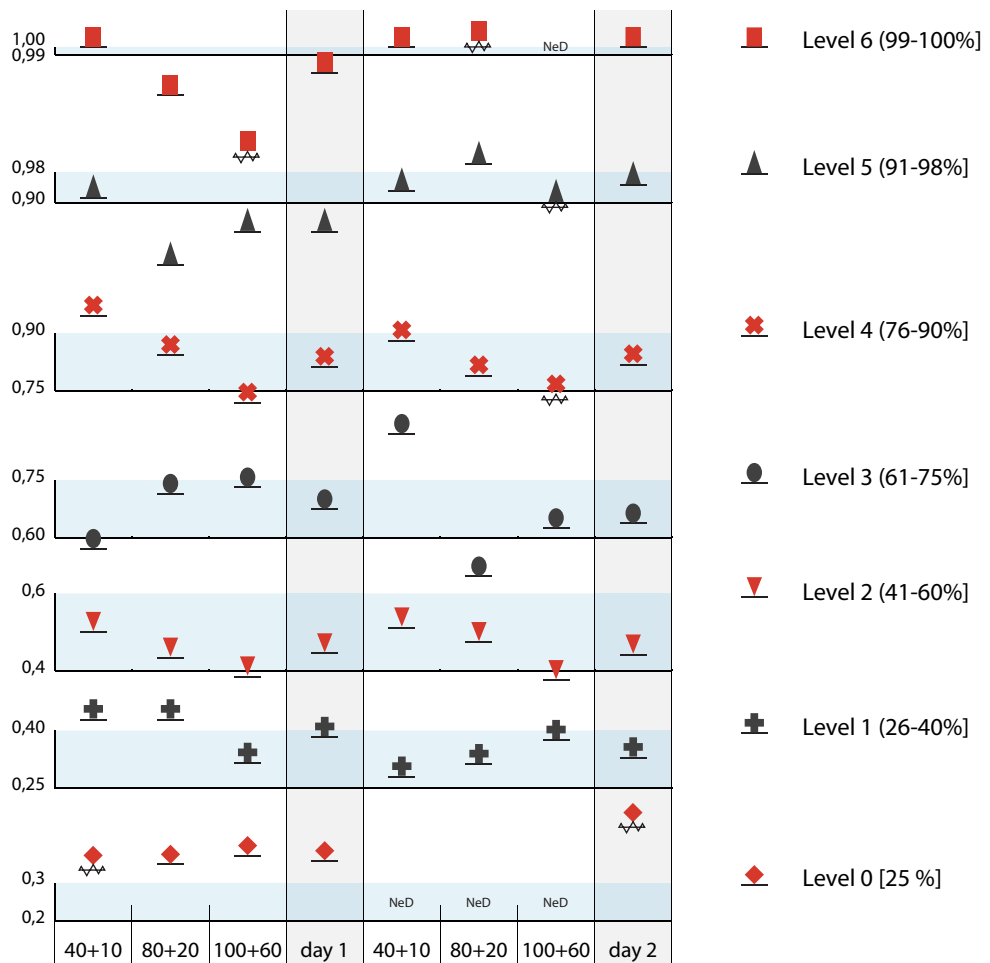


Figure 12 Participant IV - Learning graph

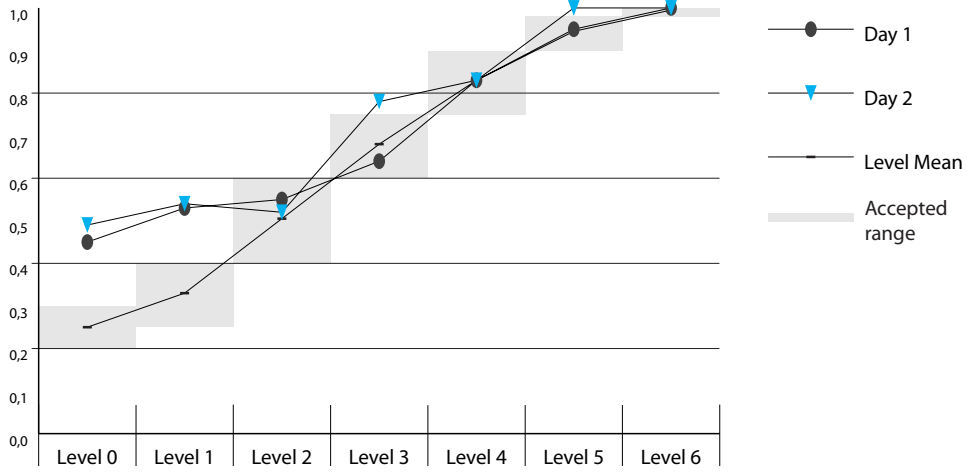


Figure 13 Participant V - Calibration graph

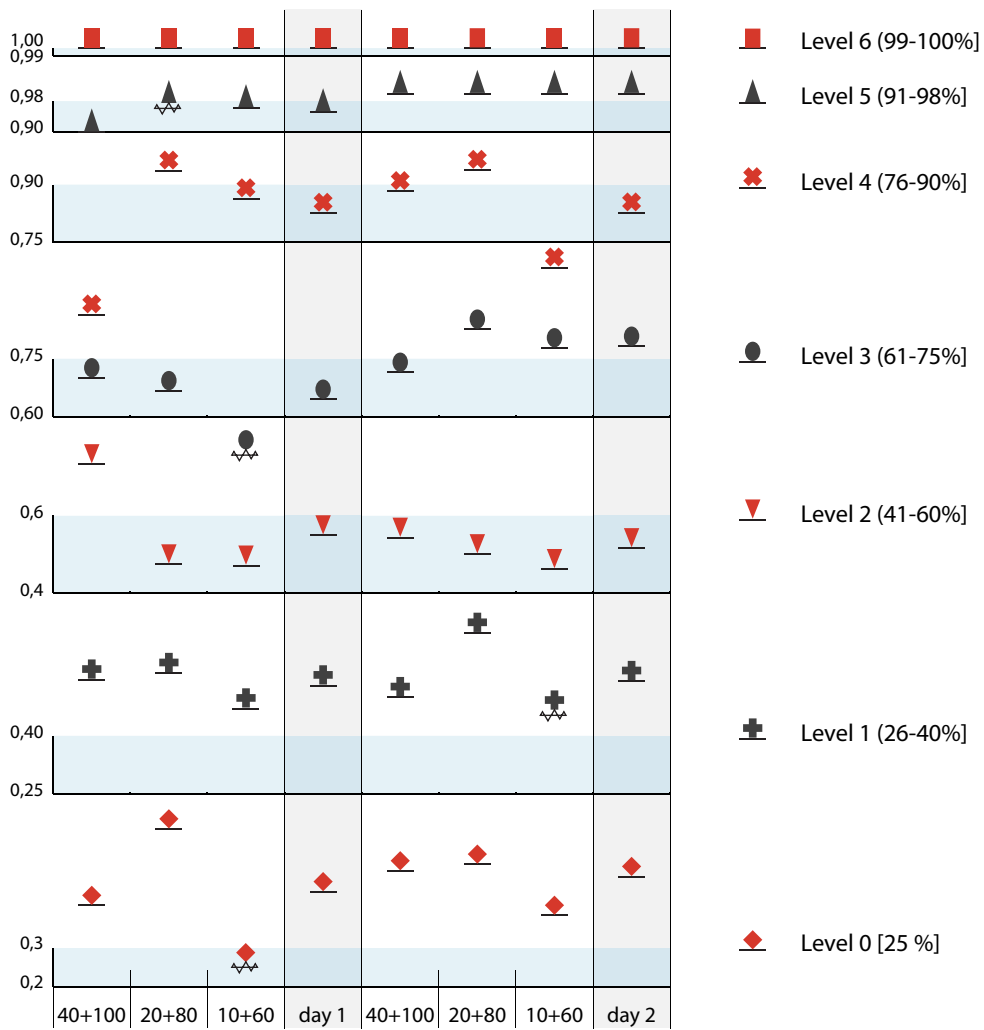


Figure 14 Participant V - Learning graph

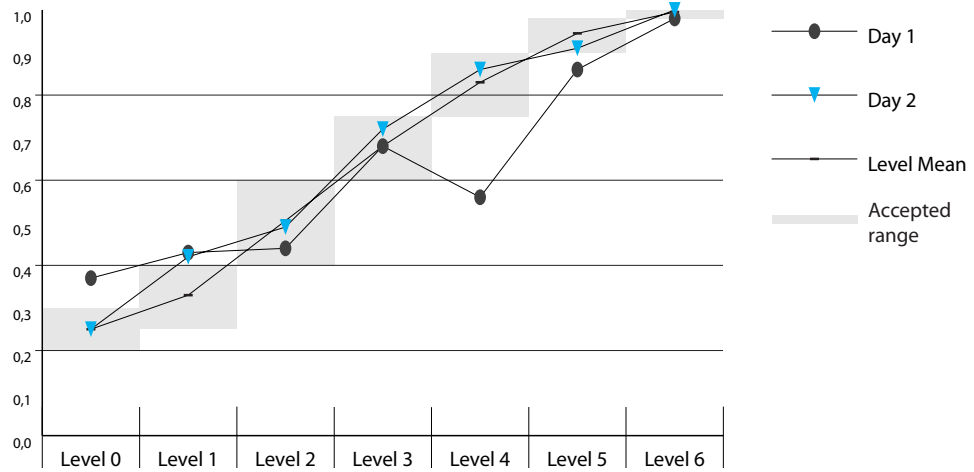


Figure 15 Participant VI - Calibration graph

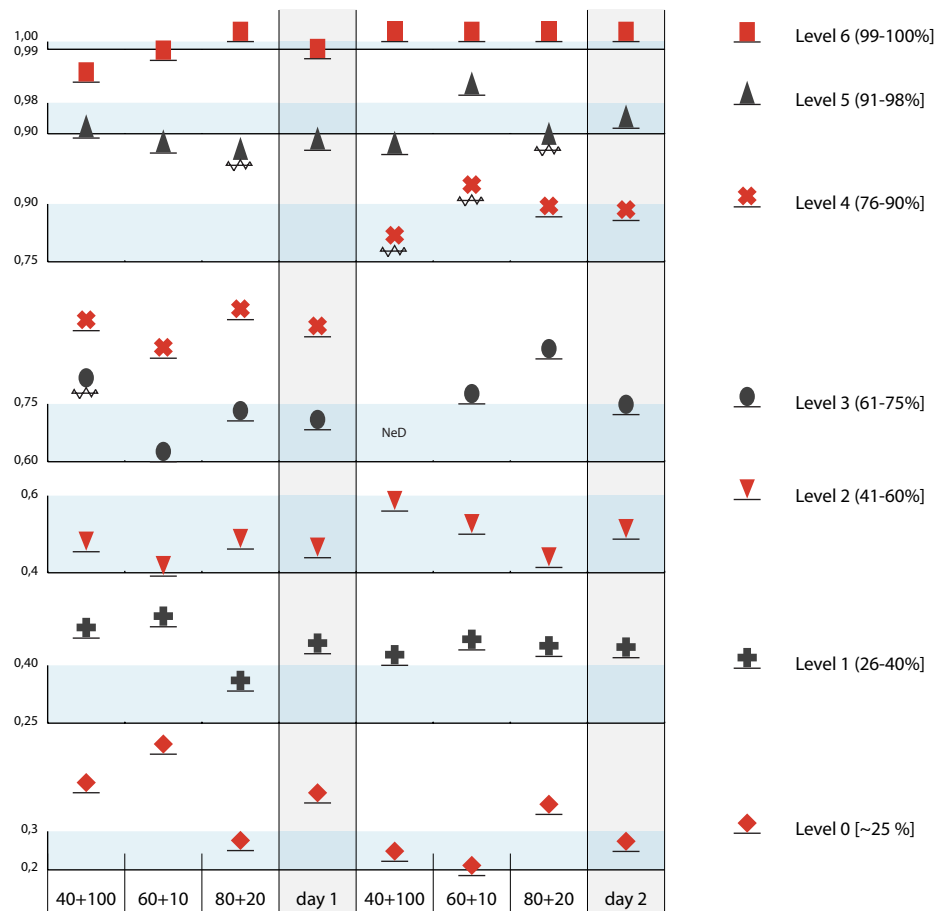


Figure 16 Participant VI - Learning graph

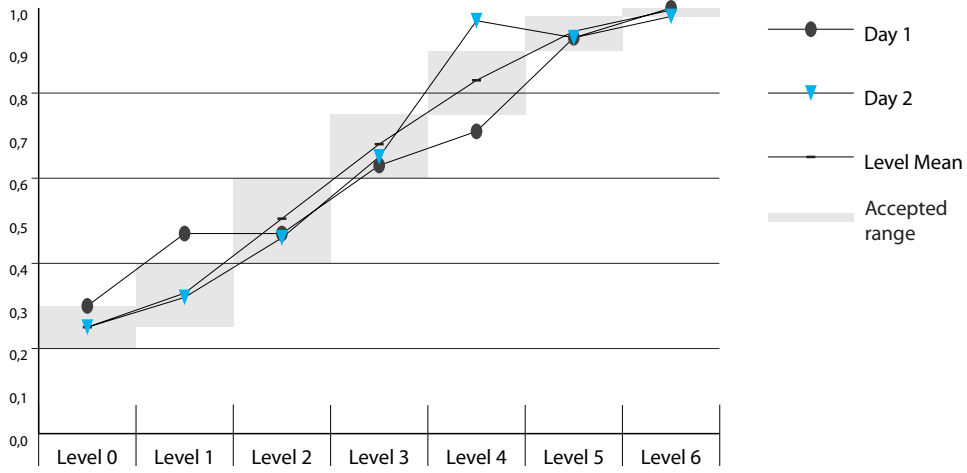


Figure 17 Participant VII - Calibration graph

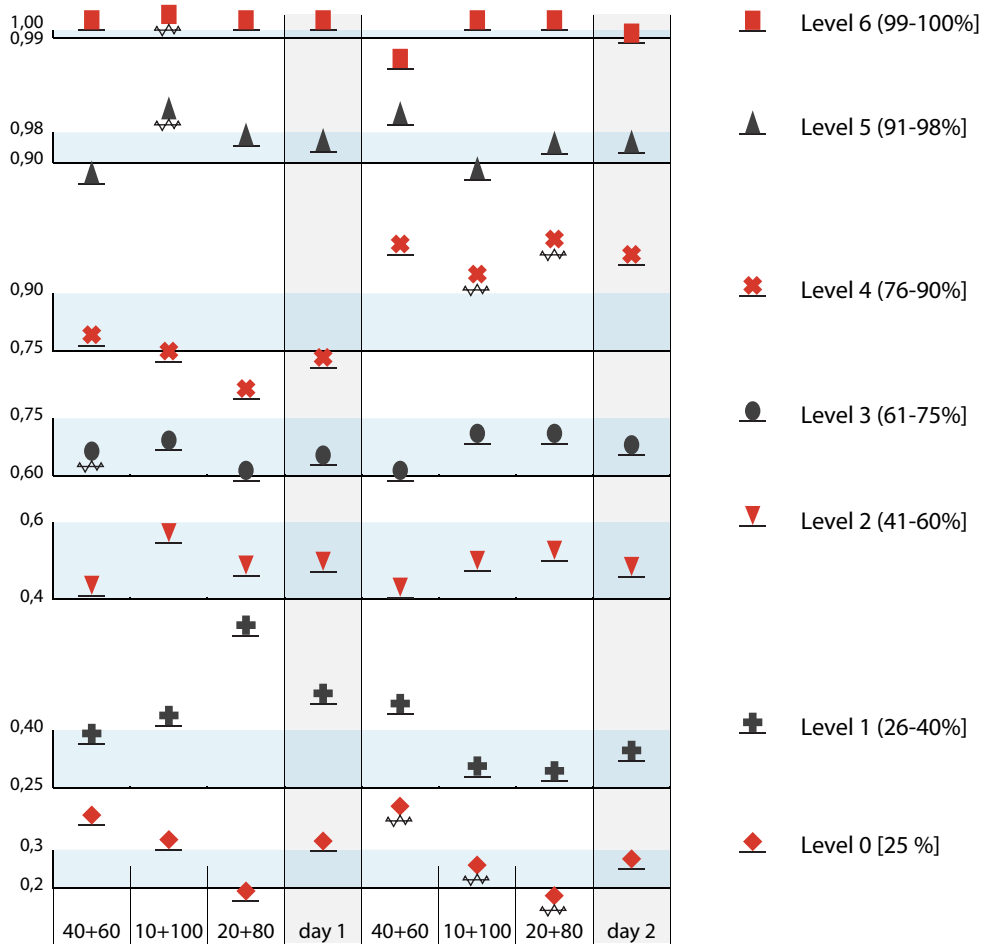


Figure 18 Participant VII - Learning graph

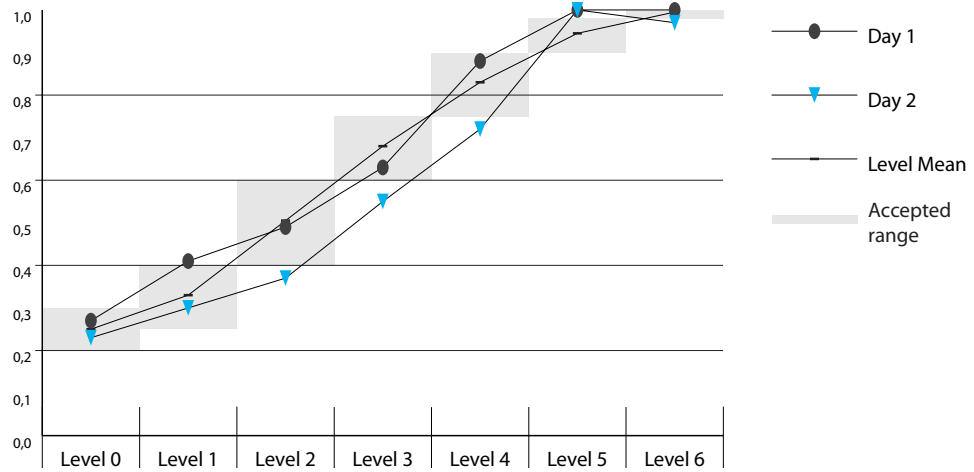


Figure 19 Participant VIII - Calibration graph

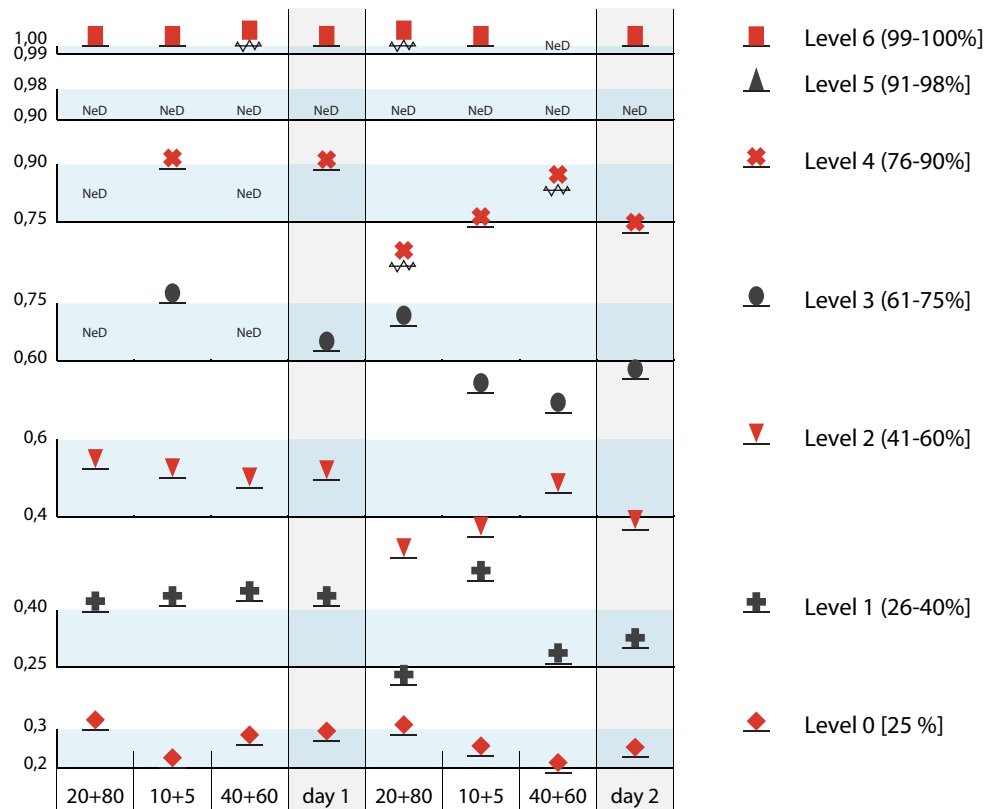


Figure 20 Participant VIII - Learning graph

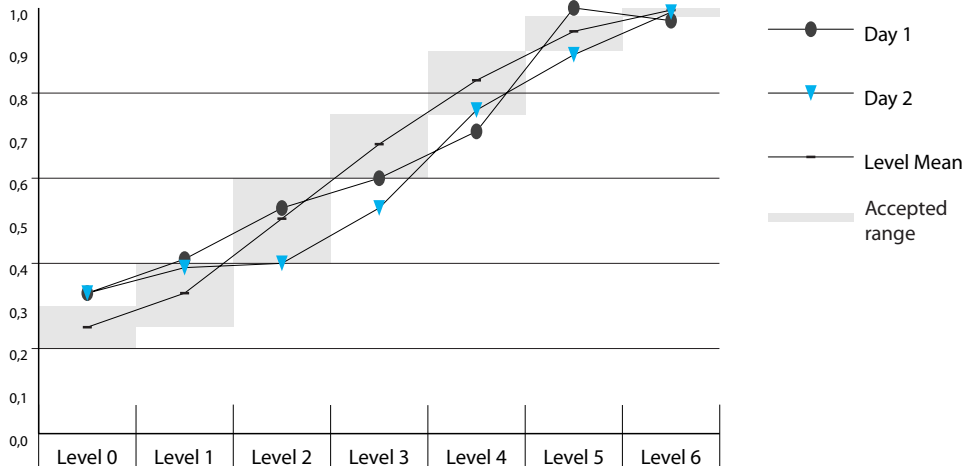


Figure 21 Participant IX – Calibration graph

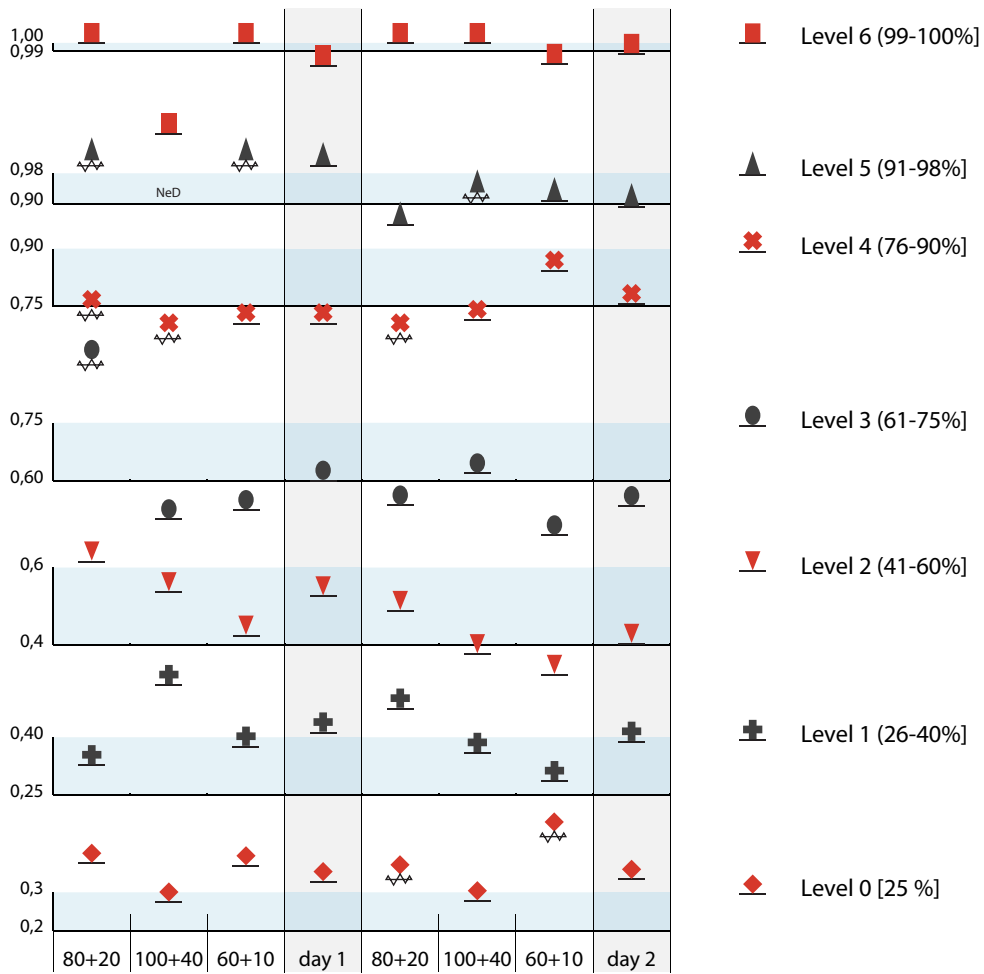


Figure 22 Participant IX – Learning graph

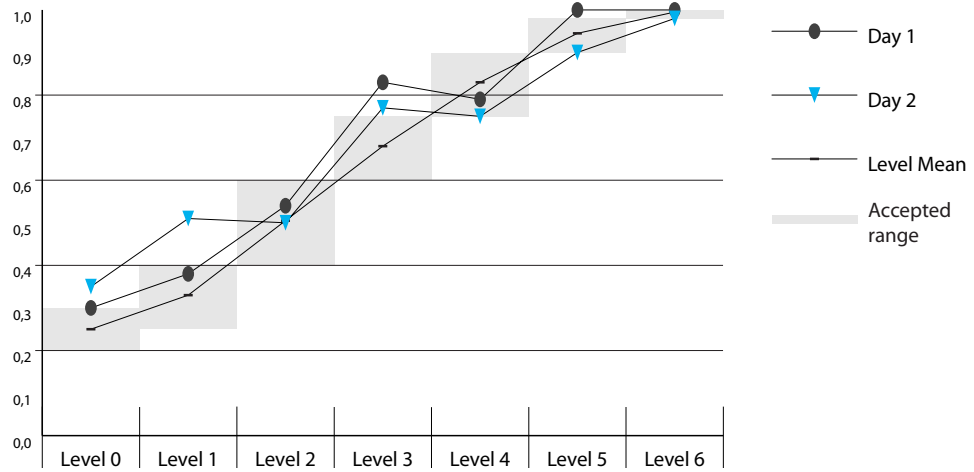


Figure 23 Participant X - Calibration graph

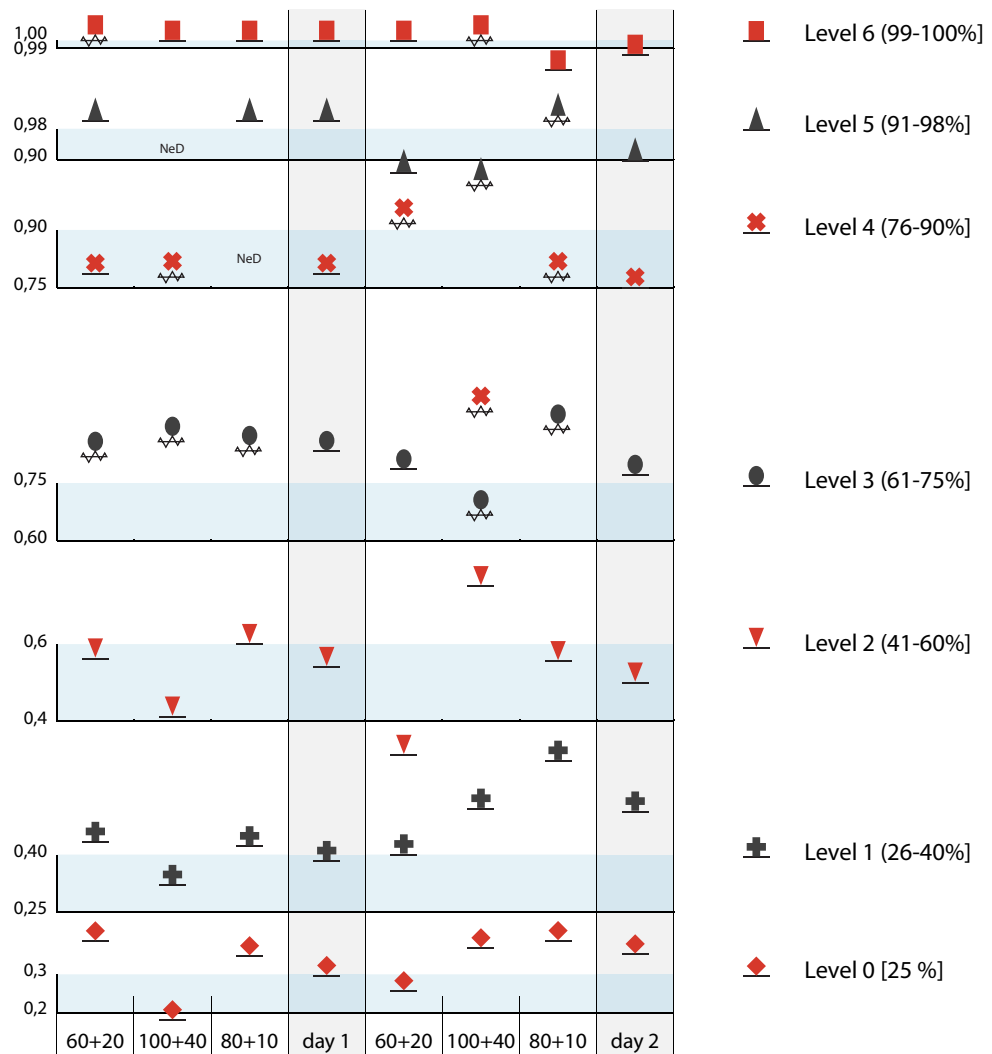


Figure 24 Participant X - Learning graph

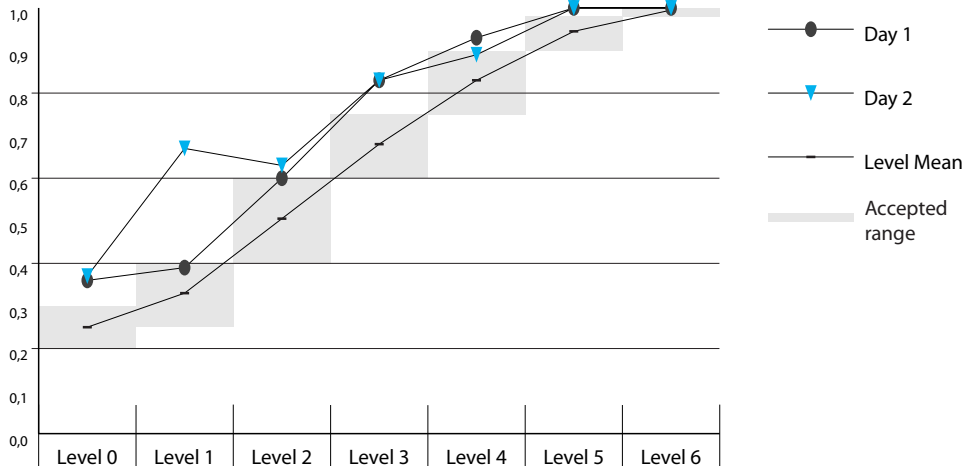


Figure 25 Participant XI – Calibration graph

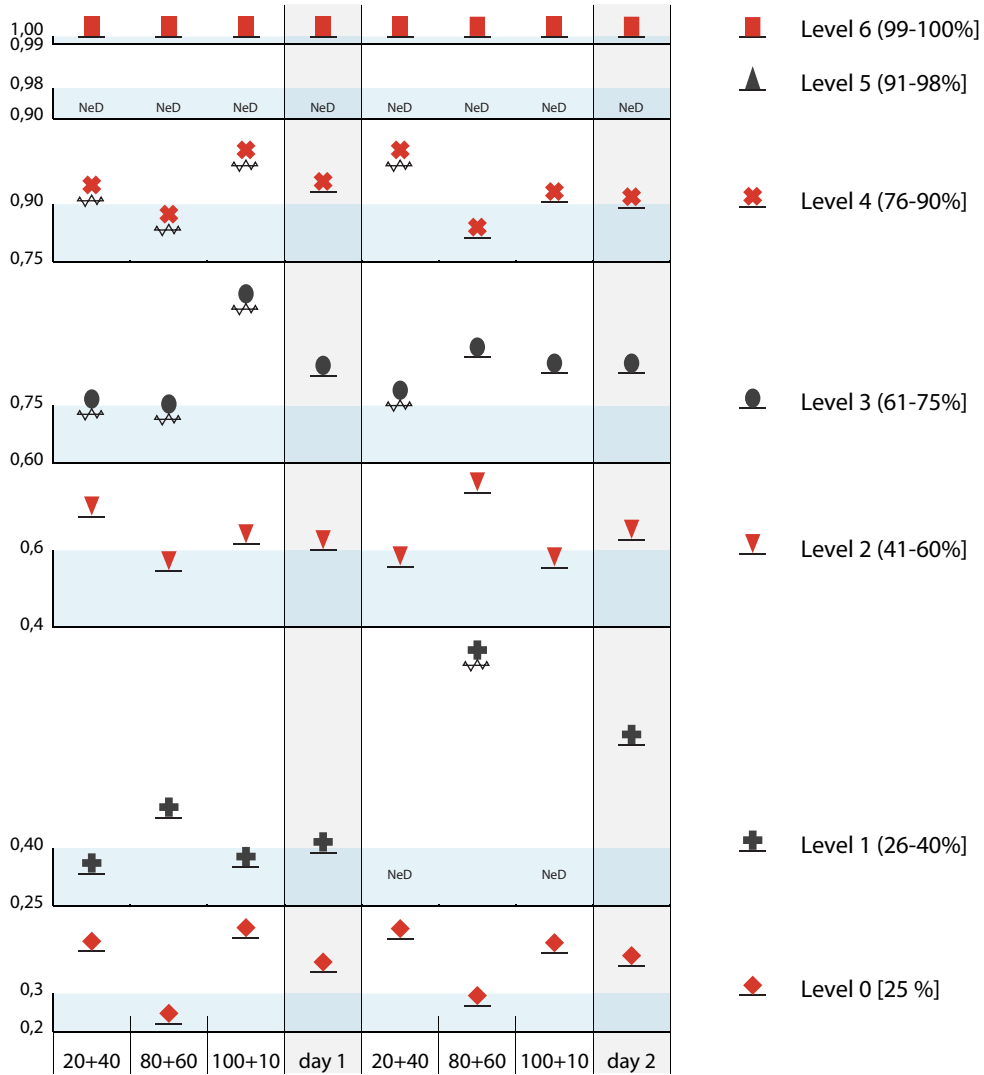


Figure 26 Participant XI – Learning graph

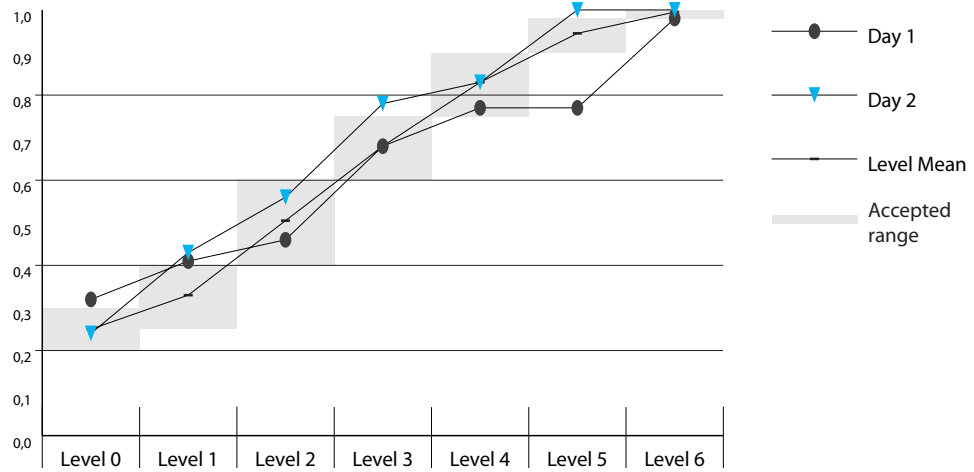


Figure 27 Participant XII - Calibration graph

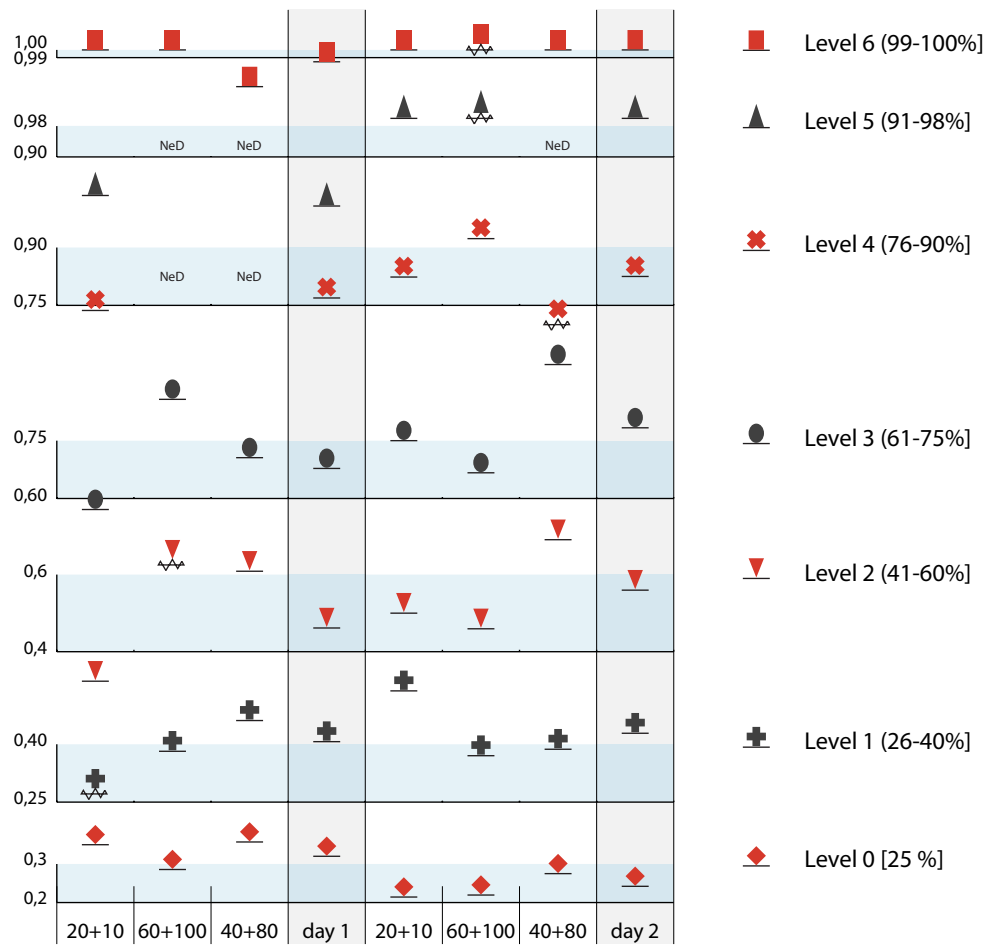


Figure 28 Participant XII - Learning graph

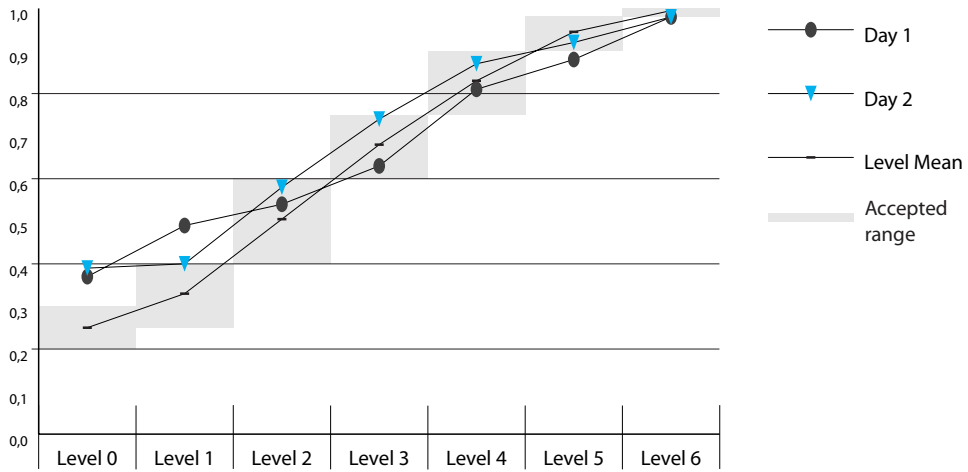


Figure 29 Participant XIII - Calibration graph

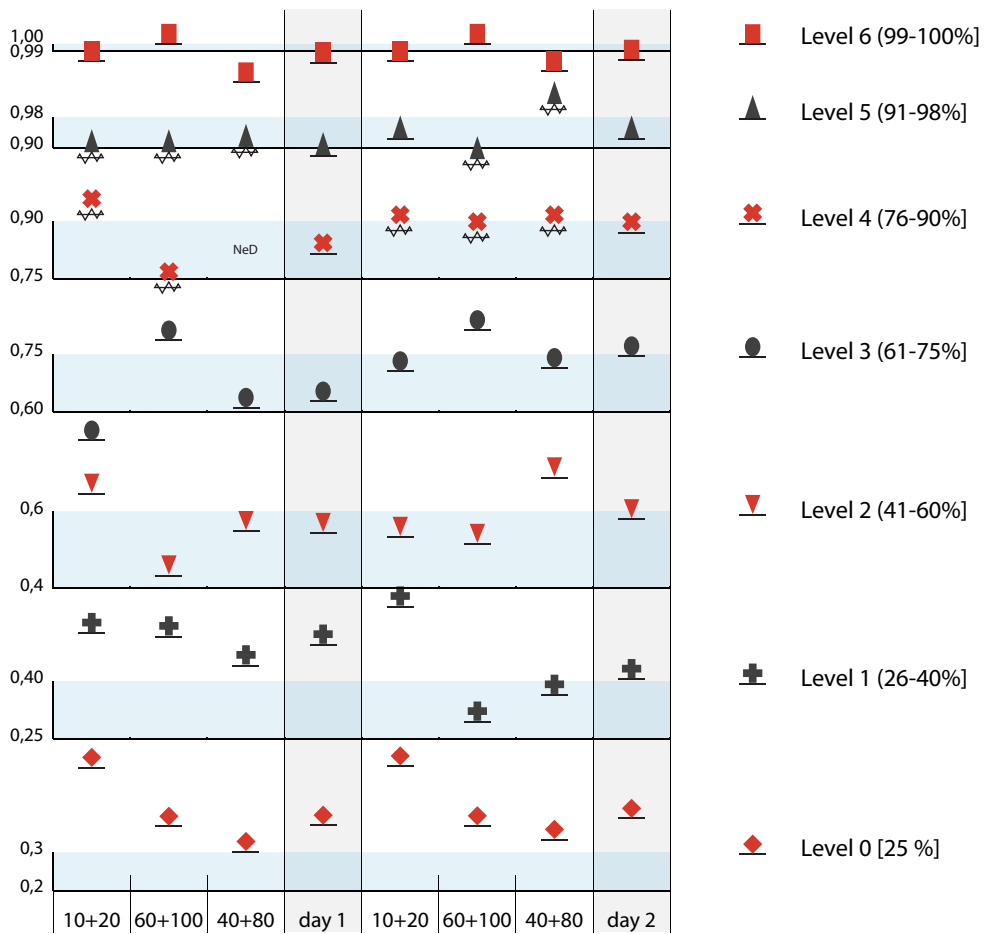


Figure 30 Participant XIII - Learning graph

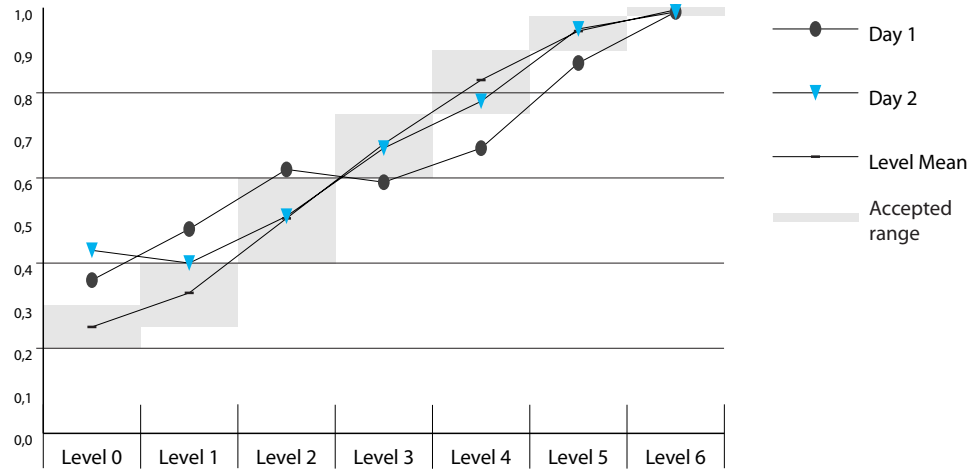


Figure 31 Participant XIV - Calibration graph

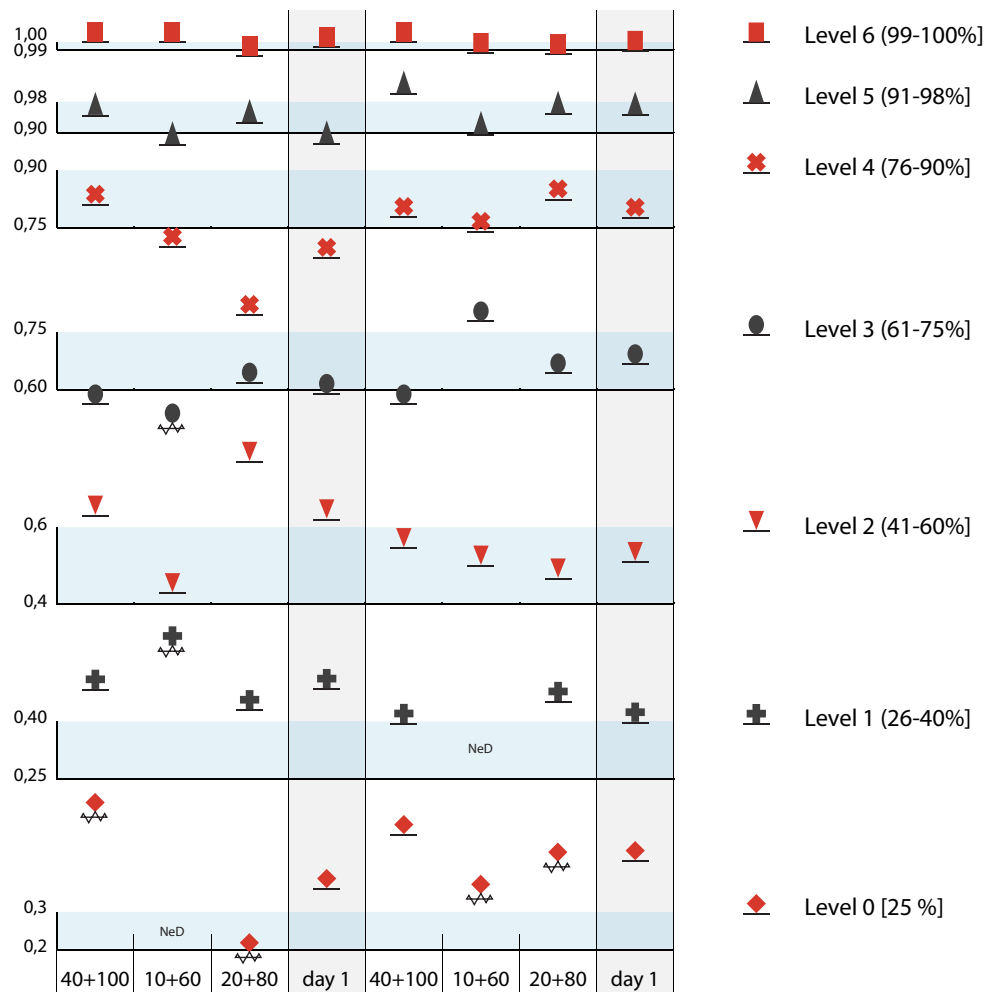


Figure 32 Participant XIV - Learning graph

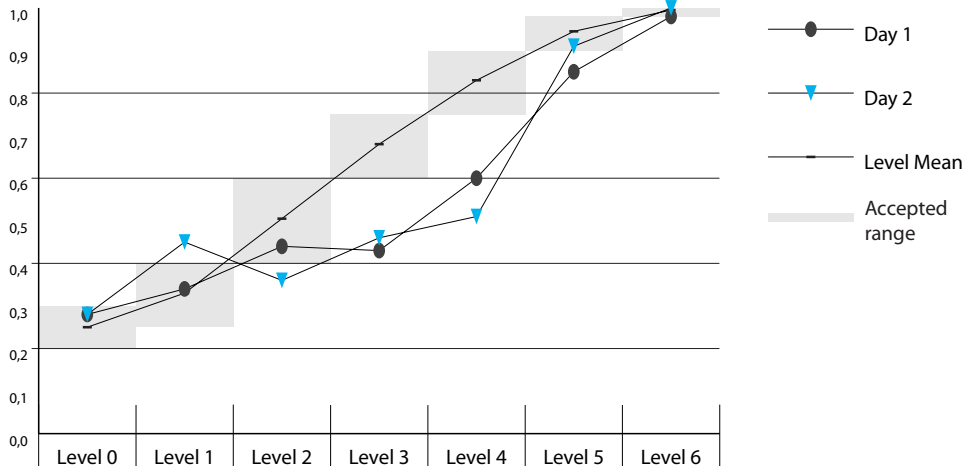


Figure 33 Participant XV – Calibration graph

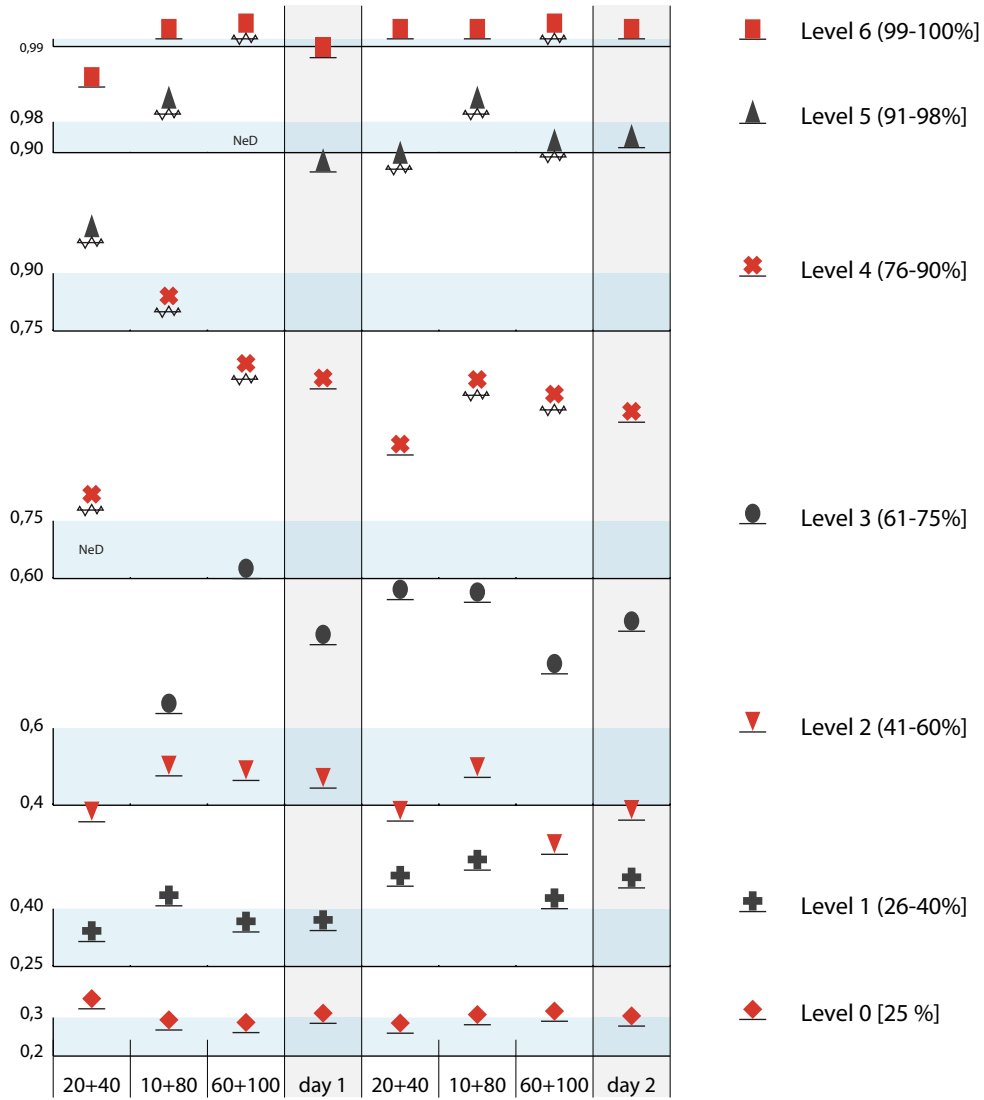


Figure 34 Participant XV – Learning graph

3.3.1.1 Summary of the calibration learning ability

Of the 15 participants in the experiment 6 people achieved a clearly better calibration on day 2, as an overall assessment of their performance. 7 people had a positive adjustment at some level towards the ideal, but not a very clear direction or the results are ambiguous. 2 subjects didn't show any clear signs of a positive adjustment; they either stayed the same or ended up with a worse calibration (even if there has been a positive move on some of the levels the overall impression is of non adjustment). 3 were relatively adequate on day one, keeping this tendency stable on day two.

The ability to stay at the same calibration level, when this was adequate in the first place, is considered a positive adjustment. The rationale for this is threefold. (1) Keeping your calibration at the same level indicates no regression back to the mean, i.e. the performance was not an over-achievement due to e.g. cheating or a sharpened concentration because of experiment stress. (2) It seemingly is more difficult to keep up behavior not caused by conscious choices when made aware of them and told to keep them up. As an example, consider thinking about the way you walk. Usually this is a very instinctive movement, but when being forced to think: "first the right leg, then the left leg, then the right leg", you'll walk in a very jagged manor and perhaps even stumble in your own legs. (3) Not having an achievable goal to work towards makes the thought process difficult to control. As the nature of the experiment, with its constant feedback, emphasizes the use of a conscious behavioral strategy the mind gets activated. So when first called to the task of calibration learning, it will not be dismissed by adequacy and simply turn of consciousness again.

There is a difference in the degree of improvement between the people only having a single goal to work towards, and the people who needed to apply different strategies on several parts of the uncertainty scale.

I.e. those who e.g. needed to become generally more confident, and those who needed to become both more and less confident on different levels. It may be that the latter individuals just needed several more rounds of question piles with outcome feedback, as they had more adjustment to do before reaching a realistic level.

As expected, the levels that caused the most trouble for the participants were levels 4 and 3, and to some extent level 1. This I believe is to due to the nature of assessing not easily accessible probabilities like the ones on these levels (76-90%, 61-75% and 26-40% respectively). By not easily accessible, I mean that they are not implicit in their meaning in the same way e.g. levels 6 and 2 are. Using level 2 to illustrate this, if you with 100% certainty can exclude two of the alternatives, you have two alternatives left. If left to guess work you will in

the long run get 50% of the answers correct, which is in compliance with this level. “Fifty-fifty” is a notion very deeply internalized in most people, even so deep that they use it when facing highly unlikely events [16]. The difficulty of understanding probability and percentiles will be discussed in more detail later. It is worth highlighting the extremely well calibrated performance on level 6 for all participants. This holds both in total, and for individual piles (with some exceptions, but those are by no means far from the accepted state of a hit rate of 99%-100% correct answers).

3.3.2 Task difficulty and knowledge level

Task difficulty is determined by how difficult it is to get the answer to a question correct. Klayman [15] calls such questions *contrary questions*; i.e. the answer is different (contrary) to what the judge believe the correct answer to be. Hard questions are those that fewer judges are able to get correct. When grouped together, one can determine if one collection of questions is harder than another by looking at the portion of correct answers for many judges on that collection. For the purpose of this analysis I call this the *global task difficulty*. Another way of determining difficulty is by using an assessors own categorization of question difficulty, i.e. the labeled degree of certainty to which the answer given is correct or not. E.g. the answer to a question the assessor deems as “without a doubt” to be correct is easy for that assessor in contrary to an answer that bear the label “no idea”, which is hard. I call this the *internal task difficulty*. This categorization is not necessarily in accordance with actual measured performance, but is rather a view of how judges interpret difficulty according to own perception of ability.

The knowledge level of a participant is here determined by the share of correct answers for that person on a collection of questions. This is by no means a classification of smart and stupid people, but is here used to find the persons that partook in the experiment who are more or less knowledgeable in comparison with each other. This ranking makes it possible to see if there is any correlation between learning ability and knowledge level in this experiment.

In this section the results are organized to check if the factors task difficulty and knowledge level correlate to the ability to learn more realistic uncertainty assessment. This investigation arises from following two hypotheses: (1) the knowledge level of an individual is assumed to be connected with that person’s ability to gain knowledge, i.e. the knowledge level is high because of a good learning ability; (2) if something is difficult to perform it is also harder to master, i.e. the more difficult a task is the more challenging are the learning aspects related to that task. In this experiment a heightened difficulty level in task to be performed, will increase the cognitive tax and thereby making learning uncertainty assessment more challenging.

3.3.2.1 Correlation between knowledge level and learning ability

In the purpose of finding if the knowledge level of a person has a correlation with the ability to learn, the hit rate for each person per pile on day 1, day 2 and both days are presented in table 3. In table 3, the six participants that showed clear signs of realistic calibration learning are highlighted. As these six are found to be low, medium and high in knowledge, i.e. portion correct answers is low, medium and high, it appears that the level of knowledge of an individual does not affect the learning ability in any distinct direction. Thereby, the amount of knowledge possessed by a person initially does not seem to be a prerequisite for possession of uncertainty assessment learning ability.

Participant \ Piles	Day 1						Day 2						Both days					
	10'	20'	40'	60'	80'	100'	10'	20'	40'	60'	80'	100'	10'	20'	40'	60'	80'	100'
XV	0,56	0,40	0,39	0,50	0,36	0,35	0,61	0,50	0,44	0,41	0,46	0,39	0,58	0,45	0,41	0,46	0,41	0,37
VIII	0,59	0,56	0,44	0,49	0,39	0,36	0,49	0,54	0,43	0,39	0,36	0,38	0,54	0,55	0,43	0,44	0,38	0,37
III	0,65	0,66	0,50	0,44	0,50	0,36	0,68	0,65	0,49	0,51	0,49	0,44	0,66	0,66	0,50	0,47	0,49	0,40
X	0,71	0,64	0,46	0,53	0,48	0,36	0,75	0,56	0,61	0,50	0,50	0,45	0,73	0,60	0,54	0,51	0,49	0,41
IX	0,68	0,66	0,58	0,46	0,54	0,41	0,73	0,60	0,51	0,55	0,56	0,54	0,70	0,63	0,54	0,51	0,55	0,48
XII	0,68	0,51	0,58	0,50	0,54	0,44	0,71	0,71	0,68	0,56	0,51	0,48	0,69	0,61	0,63	0,53	0,53	0,46
II	0,71	0,69	0,63	0,50	0,48	0,43	0,70	0,70	0,58	0,59	0,46	0,58	0,71	0,69	0,60	0,54	0,47	0,50
VII	0,70	0,69	0,49	0,54	0,58	0,50	0,71	0,61	0,63	0,63	0,53	0,44	0,70	0,65	0,56	0,58	0,55	0,47
VI	0,75	0,66	0,57	0,56	0,46	0,59	0,80	0,74	0,53	0,54	0,54	0,49	0,78	0,70	0,55	0,55	0,50	0,54
IV	0,76	0,68	0,69	0,66	0,58	0,50	0,81	0,69	0,59	0,52	0,49	0,45	0,79	0,68	0,64	0,59	0,53	0,48
XIII	0,76	0,68	0,61	0,61	0,45	0,51	0,74	0,73	0,64	0,61	0,58	0,53	0,75	0,70	0,63	0,61	0,51	0,52
XI	0,79	0,73	0,64	0,55	0,56	0,55	0,75	0,80	0,61	0,75	0,68	0,73	0,77	0,76	0,63	0,65	0,61	0,64
I	0,91	0,80	0,73	0,65	0,64	0,68	0,81	0,79	0,80	0,78	0,78	0,64	0,86	0,79	0,77	0,71	0,71	0,66
V	0,89	0,83	0,69	0,67	0,75	0,61	0,90	0,84	0,74	0,71	0,76	0,69	0,89	0,83	0,71	0,69	0,76	0,65
XIV	0,88	0,81	0,78	0,74	0,68	0,68	0,85	0,84	0,71	0,77	0,71	0,71	0,86	0,83	0,74	0,75	0,70	0,69
For all	0,73	0,67	0,58	0,56	0,53	0,49	0,74	0,69	0,60	0,59	0,56	0,53	0,73	0,68	0,59	0,57	0,55	0,51

TABLE 3 HIT RATE ON EACH PILE FOR ALL PARTICIPANTS. THE PARTICIPANTS ARE SORTED AFTER THE TOTAL PORTION CORRECT ANSWERS.

Table 3 also confirms that the question piles given to the participant were indeed of different difficulty. It also shows that the difficulty level did follow the money label on the piles. Although for some participants the portion correct is not descending with the increase of difficulty, it is so as a whole. One can therefore consider the money labels on the piles to be in agreement of the degree of difficulty when comparing the question piles with each other. E.g. the 100' pile is the pile that is the most difficult, in that the portion of correct answers for all participants is the lowest.

3.3.2.2 Differences in learning ability on Hard and Easy Questions

Global task difficulty

To see if there is a relation between global task difficulty and learning ability, the hard and easy questions are identified from the money label (supported by table 3). This gives that the 100' and 80' piles are hard, and the 10' and 20' piles are easy - relative to each other.

Figure 35 and 36 visualize performance from day 1 to day 2 for the hard and easy piles in calibration graphs. As it turns out the performance was already quite good on day 1. Four of seven hit rates are inside level boundaries on day 1 and all levels on day 2, for the hard questions. On the easy questions, accepted hit rates are found on the same five levels on both days. Feedback has had an impact on improving the performance for the hard questions' top levels. This effect is not seen on the easy questions, staying virtually identical to day 1 on day 2.

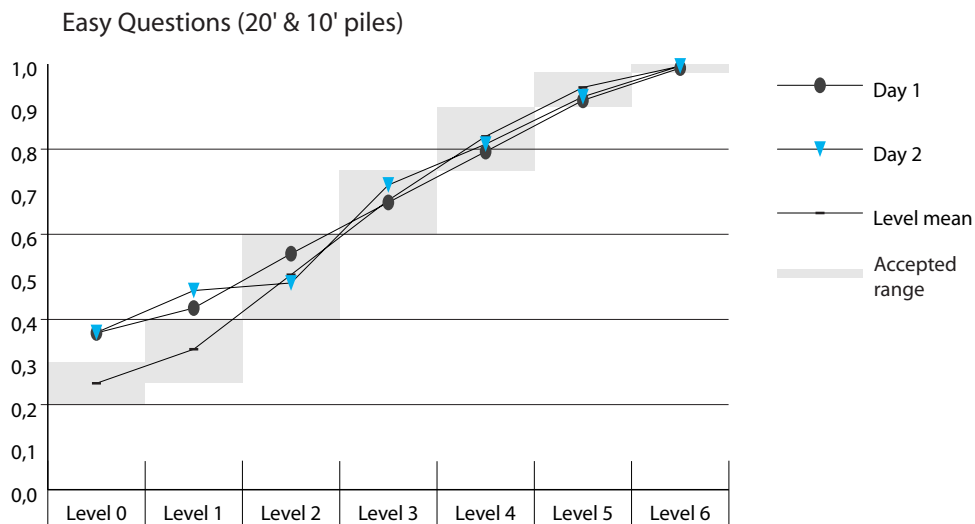


Figure 35

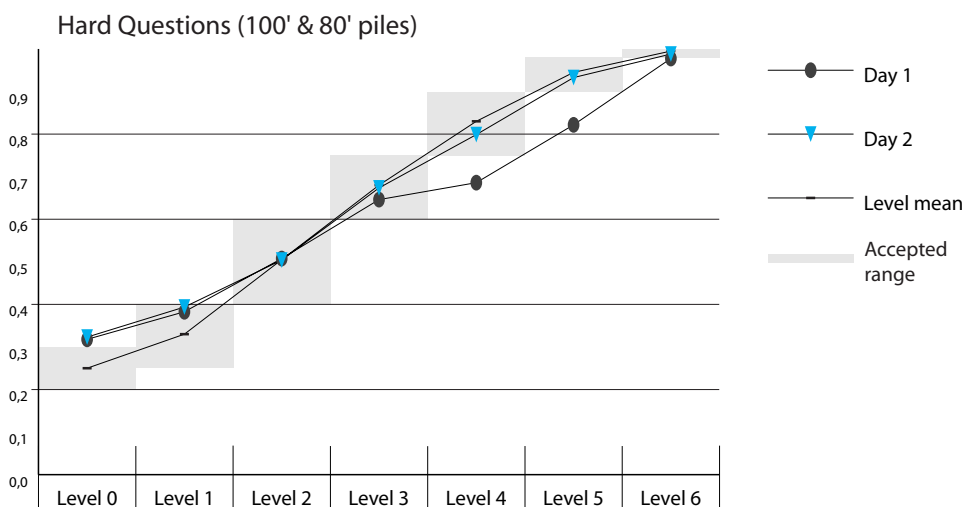


Figure 36

Internal task difficulty

The learning and calibration graphs (figure 5 to 34) were investigated in purpose of investigating the influence of internal task difficulty on learning ability. In this respect the easy questions are the ones a participant has a high degree of confidence in the given answer to be correct, i.e. questions set to level 5 (“really sure”) and level 6 (“without a doubt”). The hard questions are the ones with a low degree of confidence in the given answer, i.e. level 0 (“no idea”) and level 1 (“Some idea”). As a general tendency it was observed more positive adjustments on the internally easy questions than the hard (11 of 15 participants and 6 of 15 participants, respectively). Considering all participants, figure 36, there is a slight overconfidence on internally easy questions which diminishes on day 2; the slight under-confidence on the hard questions does not seem susceptible to the same adjustment.

The individual and group analysis of correlation between internal task difficulty and learning, together, indicate that internally easy questions are more easily adjusted towards realistic uncertainty assessment. Hard questions seem very resistant towards improvement in this learning environment.

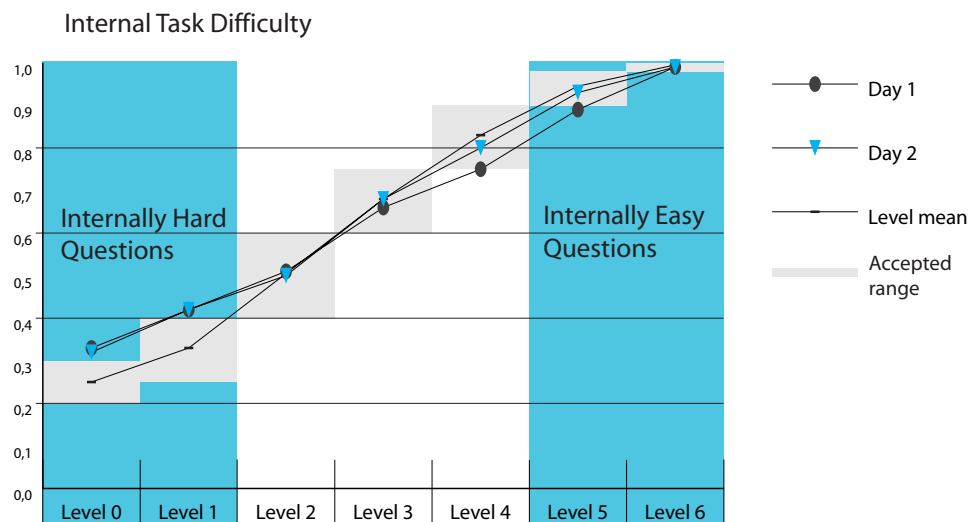


Figure 37 Internal Task Difficulty, all Participants all Questions

Connecting internal and global task difficulty with learning ability

The issue of hard and easy for the individual can also be analyzed by using the hard and easy question piles (global task difficulty) 100' & 80' and 20' & 10' respectively. In general for all levels there is no difference in learning when this is considered, and there is no significant difference in over- or under-confidence relative to either (globally) hard or easy question piles. This indicates that a participants own categorization of difficulty is what indicates what degree of learning is possible, not the “objective” global categorization.

3.3.2.3 Initial Knowledge Level

The top and bottom three participants from table 4 were chosen to represent more or less knowledgeable people respectively. The two groups also turned out to be similar to each other in calibration learning ability performance found earlier.

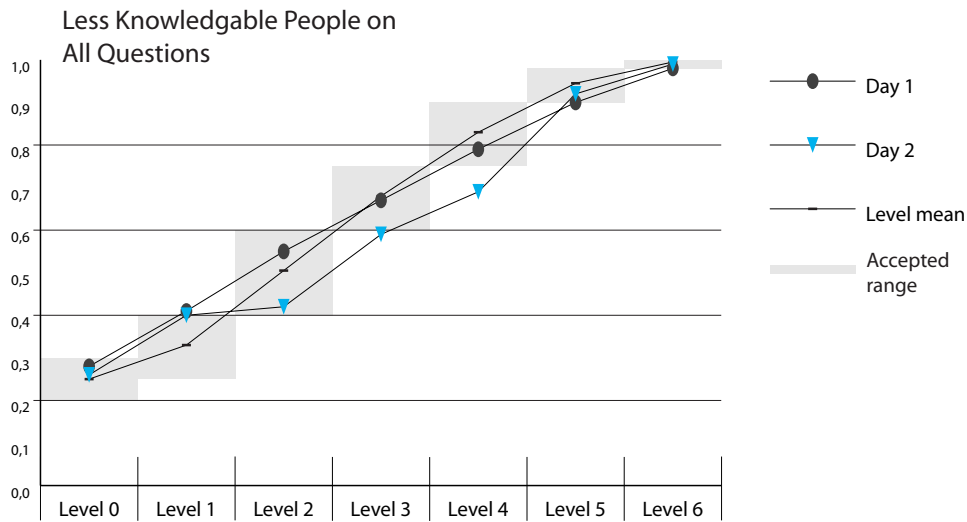


Figure 38

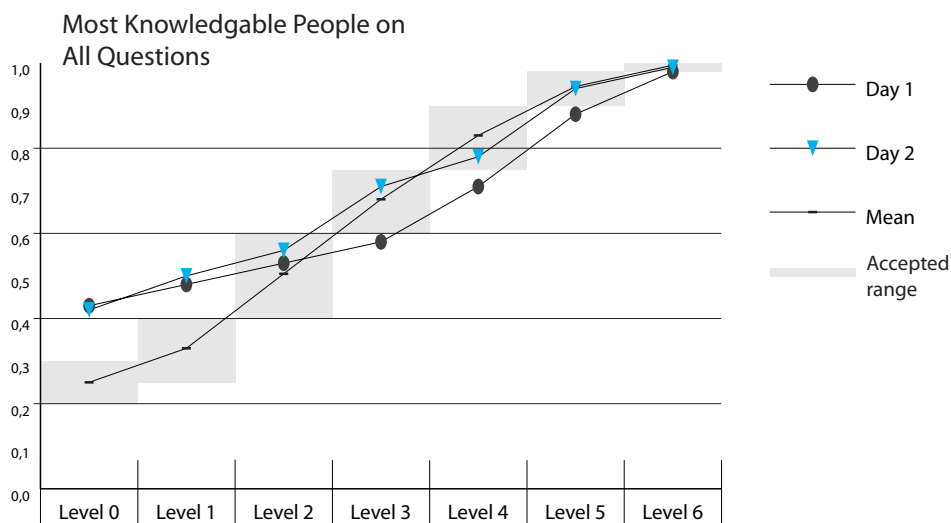


Figure 39

In figure 38 and 39 the most striking difference is that the most knowledgeable people start with a slight overconfidence on levels three through five and move towards the mean and accepted rate, while the least knowledgeable people start with an overall perfect calibration but move away from the mean towards overconfidence on levels 2 through 4 on day 2. As the shift is so small, and the performance is relatively good in both divisions on both days, it can just as well be contributed to chance as to any other factor. The least knowledgeable

people are also more realistic on internally hard questions than the most knowledgeable people. These levels do not improve for either group.

When checking for differences in regards to task difficulty (easy/hard versus most/least knowledgeable people), most knowledgeable people show a large overconfidence on globally hard questions when these were considered internally easy. This was changed to under-confidence on day 2 (see appendix B). This can imply that the most knowledgeable people do have a better learning ability on hard questions than they have on the easy questions. Therefore more knowledgeable people could have a higher ability of learning globally difficult questions, if they consider these internally easy. The informal impression of the participants is that more knowledgeable people have higher confidence in own abilities, Klayman et al. [15] also finds this. The difference here is that they do not display any major difference in **overconfidence** compared with the other participants who adjusted after feedback. The other three divisions of knowledge versus difficulty did not show any significant correlation with learning ability.

3.4 Discussion of Results

In many other studies of calibration ability, the main conclusion has been overconfidence [29]. In this light the most surprising finding in this experiment is how well calibrated most of the participants were initially, as well as the improvement seen in almost all participants after feedback. Lichtenstein and Fischhoff [3] concluded that people's calibration abilities are at best fair, overconfidence being the dominant reason for this, i.e. people tend to think they know more than they do. They arrived at results indicating that even on tasks where people could be considered experts, they didn't do remarkably better than those who weren't. Looking only at the easiest accessible confidence levels 6, 5, 2 and 0, people did not stray far. The sharpest contrary to the findings of Lichtenstein and Fischhoff, is the consistent well calibrated results for all participants, all piles, and both days on level 6. Indicating that people do know what they know when they are without a doubt sure. Difference in the tasks used here, apposed to tasks used in Lichtenstein and Fischhoff's study, is believed to have contributed to this difference; i.e. the motivation to do well was presumably stronger in this study due to the questions used. In contrary to other studies of confidence intervals [15], level 5 (90-98%) did not display the gross overconfidence usually seen at 90% confidence. Here the problem areas were identified as level 3 (60-75%) and 4 (75-90%), which is similar to what Lichtenstein and Fischhoff found.

How the participants learned to adjust their uncertainty assessment of the task, can be seen in their use of the probabilities. The change in distribution of number of questions at the different levels on day 1 compared with day 2, indicates strategies of learning by the participants. In this study the reshuffle of the question distribution is equivalent with the change in probability usage. In [18], the concern is probability forecasting of an incident not yet taken place (football match outcome). They found: “*In contrast to the control group, subjects responded to calibration feedback and training by decreasing the number of different probabilities used [...] and by increasing their usage of lower probabilities and decreasing their usage of higher probabilities*”. Although Benson’s study differs in task to be performed, the strategy used by the participants indicates behavioral change induced by calibration feedback; the same type of feedback given to the participants in this study. Here, the strategies are however not uniform for all participants. The calibration adjustment made by most of the participants in this study, was to significantly change the number of questions on a level. In most cases this had a positive effect on assessed uncertainty (level chosen) compared with real uncertainty (hit rate). The change in sample size on the levels from day to day, showed no common pattern among the participants. Some participants increased the use of the lower uncertainty levels, like in Benson’s experiment; some increased the use of the upper uncertainty levels; some made the sample sizes more similar to each other etc. Few of the participants had approximately the same sample size on each of the levels from day to day, i.e. nearly all of them had some kind of shift in levels used on day 2 of the experiment. Although no general shift tendencies are shown, there are strategy inclinations. I.e. when a participant is experiencing trouble accurately assessing a level to the real uncertainty of that level, the sample size has a tendency to go down; this is most frequent on levels 3 and 4. If there is an increase in confidence due to e.g. a well calibrated result on day one, level 0 and/or level 1 has a drop in sample size, with an equal increase for levels 1 and/or 2. Some participants has a shift in sample size that follow the uncertainty adjustment needed to get better calibrated, i.e. if the participant needs to become less confident the shift is downwards, if the participant needs to become more confident the shift is upwards. This is in accordance with the coaching advice given to the participants, also indicating the motivation and willingness to do well. A conclusion of these findings may therefore be that when trying to train calibration ability, this is in part done by finding ones own distribution of internal task difficulty.

Another striking feature in the findings is the almost non existence of the hard-easy effect. The hard-easy effect is when calibration goes systematically from over- to underconfidence as task difficulty decreases. There are some features in the design of this experiment that differs from other studies that must be mentioned before continuing the discussion of the hard-easy effect.

The calibration graphs display the participants own distribution of uncertainty over seven predefined levels; i.e. their probabilities was not grouped together into meaningful intervals in hindsight to create these graphs, as done in e.g. [3]. The uncertainty levels are intervals, i.e. a level has a range in where the hit rate can lie to be accepted. This gives the curve a legitimate visual difference from the ideal curve, i.e. even if the performance curve moves from over the mean to under as we move from level 0 to level 6 this shift cannot be deemed an hard-easy effect if inside the accepted range. Bearing this in mind, there is no hard-easy effect in the majority of participants' performance, or when only considering global task difficulty, or when dividing into more or less knowledgeable people. Why then, is the hard-easy effect absent here, when it's so often found in other studies? Suntak et al. [30] discuss four causes for the hard-easy effect: (1) It is due to decision bias, e.g. lack of attention to the quality or weight of evidence; (2) response criteria, i.e. the lack of information about task difficulty and what the best partition is, prevents sufficient change when task difficulty changes; (3) the biased selection of stimuli and cues in calibration experiments; (4) error in the judgment process either in the formation of an internal degree of belief, or in the process of generating a response. After conducting a literature review on the hard-easy effect, Juslin et al. [29] find that it is often regarded as *the* finding in calibration research. They argue that the hard-easy effect is approaching dogmatism. They state that the effect is caused by the lack of testing for, and discussion over, one or all of the following methodological problems: scale-end effects, linear dependency and regression effects. The effect is therefore subject to naive empiricism, i.e. the uncritical acceptance of empirical observation, and is near eliminated when controlled for scale-end and linear dependency. A preliminary belief is that the lack of hard-easy effect in this experiment is due to two major factors. First, the participants were informed of the difficulty of the questions by the money label on the piles, and therefore could be confident that the degree of difficulty was the same for that pile. This is also supported by the results in section 3.3.2.1: the portion correct answers are similar for all participants, and that the hit rate decreased as the difficulty increased. Second, the participants were given feedback on performance tailored to task difficulty. The suggested explanation is supported by both Juslin et al. and Suntak et al. The issue is however not very thoroughly analyzed, as it surpasses the scope of this thesis, and further analysis may uncover other factors contributing to the results.

Another view of the good performance observed, both initially and after feedback, is the motivation to do well. The educational institutions, as well as being a political resolution, emphasize the importance of heightening the general knowledge level in our society. As well as being on the curriculum from kindergarten to high school - with perhaps the exception of gossip-news

and sports facts - facts about geography, history, literature, basic science, world politics etc. are all stressed as important knowledge to possess. This is supported by a statement made by one of the participants: "I know I should know this". It is therefore a belief that the nature of the questions used in this experiment induce a stronger motivational effect than questions and tasks used in e.g. [3, 15, 31], both to get the answer right and to assess the uncertainty level associated to it. Another motivational factor to do well was the personal follow-up each participant received. It was the same person that coached and helped them on their performance a total of 11 times, as well as being easily accessible to answer questions. Personal follow-up is recommended from several sources, e.g. [32], to motivate people to do better on given tasks.

Regarding the interpretations of the results, there is large room for ambiguity. How should one view improvement at some level, but worsened calibration at others? There is a human behavioral tendency that can issue enlightenment. When focusing, we can only focus on one thing at a time [28]. The advantage of this is the undivided attention to the task at hand; the disadvantage is, of course, the total neglect of everything else. When a participant's focus has been applied to a problem area or a preferred area of improvement other areas are left hanging. If no improvement is visible in a given area, it may be because the needed focus for improvement has not been issued. This can explain why participants with multiple problem areas either show slower overall improvement or improvement only in some areas while other are worsened.

There are three major environmental design factors that more than others contribute to the learning environment: the nature of the task, the nature of the feedback and the availability of the probabilities.

3.4.1 The nature of the task

The task carried out in this experiment is similar to many experiments on uncertainty calibration done in a cognition context. In these experiments however the questions used has generally two alternatives, or they use scales of values, e.g. "how many calories are in a liter of milk?". However similar related in difficulty, the questions used are independent of each other, i.e. knowing the answer the one question does not help you in finding the answer to another. In this study there were four alternatives to choose from for each question. As for the nature and wording of the questions, here they are taken from the board game version of the world wide popular game show: "Who wants to be a millionaire?"TM. Questions in most other studies have been created by the experimenter/researcher in charge, e.g. [31]. The critique previously directed towards the questions used in these kinds of experiments is that they are too difficult [15], mainly due to the fact that over average educated people (researchers) made them. By using questions made by a

third party that had entirely different motivations when creating them, this experiment has obtained a great deal of distance and objectivity to them. One can only speculate, as there had been no contact with the editorial staff at TV2, but from the rules of the game one can expect the questions labeled with the same money amount to be of a similar difficulty; the questions labeled with a higher money amount are more difficult for the majority of judges; there is a constant increase in difficulty level as the money amount increases gradually through the 15 different piles from 1.000kr to 2.000.000kr – implying a easy level from 1.000kr to 10.000kr, a intermediate level from 10.000kr to 100.000kr and expert level from 100.000 to 2.000.000kr. The game gives the impression “anyone can become a millionaire”; one could therefore expect that no special qualification, other than perhaps a general education or an interest in the world around, is necessary to do well. This also implies that everybody can be deemed an expert at this.

The real task in this study, was to assess the uncertainty associated with the answer to these questions. As the participants can be assumed experts at answering trivia questions, the cognitive load is reduced. This, then, frees more energy and capacity to concentrate on the real task of learning better uncertainty assessment.

3.4.2 The nature of the feedback

The feedback planned, and intentionally given, to the participants was an overview of their performance, i.e. their hit rate on the different uncertainty levels after the completion of a question pile. Each question answered by the participants was also subject to feedback, as they immediately were informed if they had answered correct or not. The rationale for letting the participants themselves check the answer immediately after answering, in spite of the possibility of cheating, was to not lose motivation due to boredom. The personal tailored feedback was also supported by oral coaching, helping the participants better understand their performances and finding ways of learning to improve. All this has given the effect of a very aggressive feedback cycle throughout the experiment. All in all, the feedback given to the participants was the *outcome feedback* of each question answered, the *calibration performance feedback* given by the statistical overview of their hit rate after each pile and after day 1, and a light *process feedback* by giving informal interpretation and coaching orally. As feedback is the alpha and omega of learning [32], the feedback given here contributed to a very favorable learning environment. Lichtenstein and Fischhoff’s study on training for calibration [31], also found improvement in calibration to come quickly, when training and feedback was given.

3.4.3 The availability of the probabilities

The understanding of percentiles and probabilities is not trivial to most people [28, 33, 34]. The creation of the natural language translations to the uncertainty levels was a direct design move exactly because of this. Natural language description is exposed to subjective interpretation. Percentiles, in contrary, are unambiguous in ranking. E.g. does “fairly” imply more certainty than “pretty” or vice versa, when there is no doubt that 75% is higher than 60%. The description of the uncertainty levels were frequently and passionately debated during the first introduction to the experiment, both between the participants and I, and amongst themselves. This in turn contributed to a heightened understanding of the task at hand, and the meaning of “hit rate” and “uncertainty level”. The burden of cognitive load is lightened when the number of “uncertainty labels” to choose from is reduced to the seven predefined levels. In other experiments on judgment, e.g. general-question, stock prediction, weather forecasting, a high overweight of them give the participants full freedom to choose probability (percentage), e.g. [3, 15, 18, 31]. Observations in these experiments is that participants i.a. only use a few “favorite” percentages, reduces the percentage given from two digit to one digit after a few rounds of feedback, have a higher use of boundary percentages (like 0.5 and 1.0, in two alternatives questions). This implies that the burden of choosing an appropriate percentage to denote their uncertainty is just as high as finding a “ball park” figure of it. There are too many uncertainty percentages to choose from when given the possibility to use the whole number range. Participants are perhaps distracted by the triviality of stating their probability one or two hundreds higher or lower – which has no actual importance. The researchers also groups the probabilities together in what they feel are meaningful intervals when conducting their analysis of the results; leaving the participants effort of being very detailed in their uncertainty statements a waist. By letting the participants choose from a predefined list of intervals, the cognitive load is freed and can be used to better “get in touch” with actual uncertainty (perhaps with a bi-effect of reducing general statistical angst among the participants in the future). There are strong incentives that the framing of the probabilities in this experiment is a major contributing factor to the high degree of learning among the participants.

3.4.4 Potential limitations

The experiment process and design had known hazards before the experiment was executed. These were cheating, boredom, and indifference. Measures were taken to minimize the potential negative impact they would have on the validity of the results.

3.4.4.1 Cheating

By giving the participants the freedom to, themselves, check the correctness of their answer, one cannot rule out the possibility of cheating; also, considering the fact that the participants were in competition for scratch tickets. The condition to receive these scratch tickets was however to be one of the 10 persons that best adjusted towards the ideal, the total correct answers was only to serve as a tie breaker in cases of doubt in ranking. If in fact cheating occurred I believe it would have affected the levels 3 and 4 the most. These were the levels that most people had the hardest time distinguishing between as well. When chatting casually with the participants, the actuality of their actions surfaced. The ones, who admitted to trying to keep score over their hit-rate, explained that they quickly lost interest in this. Instead they focused on getting as many of the questions correct, rather than trying to remember seven different hit-rates which were in continuous evolution. It therefore seems that the value and status of having the most correct answers surpassed the achievement of being the best at approximating the uncertainty levels to the ideal. A reason for this can be the accessibility of having a concrete number of correct answers, which meant the same for everyone. Apposed to, the invisible and non comparative skill of calibration improvement, which wasn't as easy to compete with (with the one sitting next to you). As they were expected to answer about 960 different questions, keeping track of their hit rate (in memory only) would have been an astonishing commitment just to win 10 scratch tickets. To sum up, if cheating did occur, it was quickly stifled due to the large amount of questions to be answered. Also the cognitive dissonance [34] was too great for most people to bother with it, i.e. the potential reward was too small to outweigh having to become a cheater for it.

3.4.4.2 Boredom and indifference

Another threat to the validity of the results is that the vast amount of questions to be answered could lead to boredom and indifference. As this was a known factor, measures were taken to prevent the likelihood of this happening. As the questions are taken from a well known and popular game show, they in themselves are believed to contribute to a positive interest. This was confirmed by the level of commitment expressed by several of the participants during the experiment. Curiosity induced by the questions (what the correct answer was), and the opportunity to use, and shine on, otherwise uncalled-for knowledge (like sports facts, and royalty news), were frequently stated; as well as statements like: "This is fun!", makes it safe to presume the task in itself didn't generate too much boredom and indifference. A bigger threat would be the amount of questions to be answered, taking about 8 hours over two days to plough through. To prevent boredom and indifference due to this, the participants could take as many and as long breaks as they wanted. They were, however, urged to take them in-between, rather than in the middle,

of a pile of questions. A last point concerning indifference is that most of the participants were friends or fellow students; i.e. all of them were very akin, and appreciated the possibility to contribute to the cand.scient. thesis research. As to boredom influencing the results, the continued improvement seen in many (almost all) of the participants on the last two piles on day two is a strong indication of the contrary.

3.5 Summary and Conclusion

The research questions in this experiment are a refinement to a particular situation based on the general research question RQ_I: *“Given a favorable learning environment, can people learn to better calibrate uncertainty estimates when provided feedback on performance?”* Based on the RQ_I, the research question specific for this experiment is are:

1. Given the favorable learning environment outlined, how good are people at assessing uncertainty initially?
2. Given the favorable learning environment outlined, are people able to learn to be better calibrated if initially being over- or under-confident?
3. Does the level of difficulty on a group of questions affect the possibility of learning to be better calibrated?
4. Does the personal level of knowledge affect the possibility of learning to be better calibrated?

To investigate the research question, participants answered general knowledge questions. The questions were taken from the board game “who wants to be a millionaire?™”, each question has four alternatives. Participants assessed uncertainty on the chosen answer to a question. The questions were grouped into piles with similar difficulty, provided by the game by a money label. Uncertainty was given by choosing from a predefined list of prediction intervals (seven in total). Ideally the actual frequency of correct answers was to fall within the uncertainty level, chosen by the participant, on questions assessed to that level. The correct answer to a question, after alternative and pertaining uncertainty was given, was checked by the participant herself. Participants were given calibration feedback after finishing a pile of questions, there was in total 12 question piles to be answered (each containing about 80 questions). Informal oral coaching was given in connection with the calibration feedback. A summary of performance on day 1, both formal and informal, before the starting day 2 of the experiment was handed out on paper. The calibration ability of the participants was good initially, and a good calibration learning ability was found in 13 of the 15 participants. The task difficulty and knowledge level were found to have minimal impact on the learning ability.

The cognitive tax of the participants was reduced since they can be considered experts at the task (answering general knowledge questions). There were used three kinds of feedback: outcome, performance and process. These were given frequently and were personalized. The uncertainty levels used was thoroughly explained, and given natural language descriptions. This may have contributed to heightening the understanding of the concept of probability, as well as possibly reducing probability angst. All this contributed to a favorable learning environment. In conclusion, as there was a large degree of learning observed, given favorable conditions people have the ability to learn more realistic uncertainty assessment.

In regards to task difficulty, this was separated into two perspectives: *global* and *internal*. The former gives the difficulty of a task by how many are able to perform it correctly (the share of correct answers for many judges). The latter, determines task difficulty as assessed by the individual. Globally hard questions had an observed overconfidence on the top levels, except level 6, on day 1; this was, however, totally eliminated on day 2. Globally easy questions had slight underconfidence on the two lowest levels both days, otherwise perfectly calibrated. This may indicate that hard questions, although initially too optimistically assessed, can be improved when given feedback. Easy questions are initially well calibrated, but the lower levels seem to be harder to adjust to a realistic level even after feedback is given. In regards to internal task difficulty, the results indicate that internally easy questions are more easily adjusted, whereas internally hard questions are harder to adjust.

The results do not indicate that there is any significant difference in the ability to learn when level of knowledge is considered. The results imply that the most knowledgeable people are initially the most overconfident, but they have the ability to adjust given feedback.

It must be noted that there was a difficulty of drawing clear conclusions. This was mainly due to the good initial calibration of the majority of participants. However, since the majority also improved their initial calibration, it can be concluded that given the favorable learning environment learning uncertainty assessment is possible.

If the improvements seen really are internalized, i.e. the participants actually learned to better their uncertainty assessment, as opposed to just displaying a willingness to adjust to feedback, is unknown. To find this one would have to get a hold of the same participants for a new round of question answering. If they still have the same calibration as they had at the end of the experiment, only then could one assume that they had **learned** to be more realistic.

4 Experiment no. 2

- The Software Development Experiment (SDE)

4.1 Motivation of study and study design

In this experiment we take the leap into the world of software engineering, with the challenges and special circumstances that this field brings with it. As described in section 1.1, software professionals have previously shown poor uncertainty assessment skills on most likely effort estimation in software projects. It is therefore important to find out if it is possible to improve uncertainty assessment skills, as this will help in reducing cost overrun in the software industry.

Although significantly different in subject and task to be performed (trivia questions vs. programming), GQE contributed to the design of this experiment. The most important contribution was the indication that learning more realistic uncertainty assessment, using relevant outcome feedback, is possible. In addition it was believed that giving the participants a predefined list of probabilities to choose from was one of the factors contributing to the well adjusted calibration results from GQE. In addition to the natural language description of the level of uncertainty, the probabilities were chosen so that they presumably were easily accessible to the participants. This was done by choosing probabilities that can easily be translated into fractions, as well as being symmetrical visually, see table 4. The rationale for this is that fractions are better understood intuitively than some percentages, and therefore they implicitly have a natural language description. It is also a belief that fractions are more used than percentage when assessing uncertainty in everyday life, and therefore easier to understand and use. If in fact the developers saw the pattern described is unknown.

%	\sim fraction	Natural language description of the probabilities (translated from Norwegian)
0	0	Time used will be never within this interval
5	1/20	Highly unlikely that the time used will be within this interval
20	1/5	Fairly unlikely that the time used will be within this interval
35	1/3	Not very often will time used be within this interval
50	1/2	Half of the time the time used will be within this interval
65	2/3	Relatively often will time used be within this interval
80	4/5	Fairly likely that the time used will be within this interval
95	19/20	Highly likely that the time used will be within this interval
100	1	Time used will always be within this interval

TABLE 4 UNCERTAINTY LEVELS USED IN SDE

The natural language description of the probabilities was only stated in the handed out instruction manual, not as in GQE where it was displayed with every question. This was done to tone down the experimental feel of the environment. The aim here was to create a favorable learning environment, not necessarily a sterile experiment setting. The rationale being it would be more damaging with a sterile setting, as this potentially hides or represses natural behavior.

The use of feedback was restricted to relevant outcome feedback. As experiment no 1 had established that when using experts at the task at hand, this induced a good calibration learning effect on the uncertainty assessment of that task. Therefore the interest in this experiment shifted to see **how** they eventually learned in a favorable environment. Not to spoil personal strategy evolution, no coaching was given other than the initial briefing, and instruction manual, about the experiment (see Appendix D). There was however no restriction on answering questions of all sorts throughout the experiment.

4.2 Design of study

4.2.1 Research questions

In this study both the general research questions, RQ1 and RQ2, are investigated in a specific setting. This shift is in both learning and in required skill. The shift in learning is from **whether** learning is possible to **how** learning is achieved; the shift in skill from skills potentially possessed by **everybody** to the skills of the **software developer**. Therefore a refinement of the general research questions to suit the special circumstances of software development is a needed. The level of programming development skill seems

to be a poor indicator of ability to assess realistically the uncertainty of most likely effort, as found in e.g. [6]. There is a poor learning from experience, i.e. amount of experience has no correlation with ability to assess realistic uncertainty of effort estimates. Therefore, a better understanding of the conditions for learning from experience in the context of software effort estimations is needed. This study aims to address these issues specifically by investigating:

- How much do programmers improve their assessment of the uncertainty of estimates of most likely effort on the basis of outcome-related feedback?
- Is there a relation between the learning strategies for improving uncertainty assessment used by the programmers and their ability to learn from feedback?

4.2.3 Measures

There are no standard measures of uncertainty assessment performance. Jørgensen argues in [2] that one should differentiate between people's ability to assess the *average level of uncertainty* of a set of tasks and the *relative difference* in uncertainty between different tasks. Using this as inspiration, the following definitions and measures are used in this study:

T = The set of n development tasks

$ActEff_j$ = Actual effort required to complete Task j

$EstML_j$ = Estimated most likely effort of Task j

MRE_j = Magnitude of relative estimation error of task j

$$= |ActEff_j - EstML_j| / ActEff_j$$

REE_j = Relative estimation error of task j

$$= ActEff_j - EstML_j / ActEff_j$$

$Int1_j$ = [90% of $EstML_j$; 110% of $EstML_j$]

$Int2_j$ = [60% of $EstML_j$; 150% of $EstML_j$]

$Int3_j$ = [50% of $EstML_j$; 200% of $EstML_j$]

The widths, i.e., the percentages, were chosen to reflect a narrow effort interval ($Int1$), a medium-wide effort interval ($Int2$), and a wide effort interval ($Int3$). There is applied more than one interval to enable analyses of possible differences in learning effects related to width of interval.

$Conf1(Int1_j)$ = The developer's assessed probability (confidence) of including $ActEff_j$ in $Int1_j$

$Conf2(Int2_j)$ = The developer's assessed probability (confidence) of including $ActEff_j$ in $Int2_j$

$Conf3(Int3_j)$ = The developer's assessed probability (confidence) of including $ActEff_j$ in $Int3_j$

$AvConfLev1(T)$ = Average value of $Conf1(Int1_j)$ for tasks $j=1..n$
 $AvConfLev2(T)$ = Average value of $Conf2(Int2_j)$ for tasks $j=1..n$
 $AvConfLev3(T)$ = Average value of $Conf3(Int3_j)$ for tasks $j=1..n$

$HitRateInt1(T)$ = Proportion of $Int1_j$ -intervals that includes $ActEff_j$ for tasks $j=1..n$
 $HitRateInt2(T)$ = Proportion of $Int2_j$ -intervals that includes $ActEff_j$ for tasks $j=1..n$
 $HitRateInt3(T)$ = Proportion of $Int3_j$ -intervals that includes $ActEff_j$ for tasks $j=1..n$

Applying these definitions the ability to assess the average level of uncertainty is defined as:

$Overconfidence(Int1,T) = AvConfLev1(T) - HitRateInt1(T)$
 $Overconfidence(Int2,T) = AvConfLev2(T) - HitRateInt2(T)$
 $Overconfidence(Int3,T) = AvConfLev3(T) - HitRateInt3(T)$

The measure is termed “Overconfidence”, because a positive value indicates overconfidence in the accuracy of the estimate of most likely effort. Consider the following example: Assume that an estimator estimates and assesses the estimation uncertainty of a set of tasks (T). On average, the estimator believes that there is a 50% chance of including the actual effort in $Int1$ for the set of tasks 1..n. The estimator’s average confidence level ($AvConfLev1(T)$) is then 50%. The proportion of actual effort values included in $Int1$ is, on the other hand, only 30%, i.e., the $HitRateInt1(T)$ is 30%. Then the level of overconfidence is calculated as the difference between average confidence and inclusion rate of $Int1$ of the set of tasks in T, i.e., $Overconfidence(Int1,T) = 50\% - 30\% = 20\%$.

The measures of ability to assess relative difference of uncertainty between different tasks are defined as follows:

$RelUncAbility(Int1,T)$ = correlation between $Conf1(Int1_j)$ and MRE_j , for $j=1..n$
 $RelUncAbility(Int2,T)$ = correlation between $Conf2(Int2_j)$ and MRE_j , for $j=1..n$
 $RelUncAbility(Int3,T)$ = correlation between $Conf3(Int3_j)$ and MRE_j , for $j=1..n$

These measures are based on the assumption that there should be a correlation between confidence in the accuracy of the estimate of most likely effort (Conf) and the estimation error (MRE). When the confidence in the accuracy of the estimate of most likely effort is low, we would expect a high MRE, i.e. we expect these measures to give high negative values if the estimator is skilled at assessing the relative difference in effort estimation uncertainty between different tasks.

4.2.4 Subjects, Tasks and Material

At the University of Oslo there was advertised for highly skilled Java programmers. Based on their CV's and short interviews, the (believed) best five programmers were selected. All of them had extensive programming experience, having programmed thousands of lines of code in leisure, educational and industry contexts.

The tasks used in the experiment were all taken from beginners books on Java, programming courses at the Department of Informatics, and the Java Sun's home page. In all there were 18 tasks, relatively small in size, pilot tested to be from 30 min to 4-5 hours in work-hours for an experienced programmer. The nature of the tasks is of typical student assignment; the participants, all being students, can therefore be considered experts at this kind of tasks. Most tasks required a GUI or another type of visual solution, the rest were text-based. Although similar in complexity and type, none of the tasks were interdependent on each other. They also had significantly different solution demands, requiring the use of different techniques and problem solving strategies. The sequence of tasks was similar for all participants (see section 4.2.5 for more details), there was no obvious increase in task difficulty and complexity that would influence the results.

See Appendix D for task descriptions.

The experiment was held at the Department of Informatics' "Abel" computer lab. The lab has UNIX work stations with a Java environment installed. All the participants had their own user accounts and were familiar with the computer lab and the programming environment. The programming was conducted using a text editor; most of the participants used 'Emacs' which they had customized to their own preference.

4.2.5 The experiment Process

The experiment took place over a period of about two weeks. Each participant participated for five work-days. The first day of the experiment all the participants were briefed about the experiment. During which it was emphasized that the purpose of the experiment was to study how well they were able to improve their effort uncertainty assessment when given feedback. The participants had received the instruction manual a couple of days earlier. The task description, i.e. the requirements description, was handed out one at the time. The participant read the text, briefly analyzed the problem, and estimated most likely effort (EstML). The estimated effort was given as an input to a web page, which calculated the different effort intervals Int1([90%, 110%] of EstML), Int2([60%, 150%] of EstML) and Int3([50%, 200%] of EstML) in work-hours. The participants were then required to assess the probability of their actual effort being inside these intervals.

When this was done, they proceeded with solving the task; when finished, time spent was registered. Figure 40 illustrates this process for one example task.

An example of a task run through

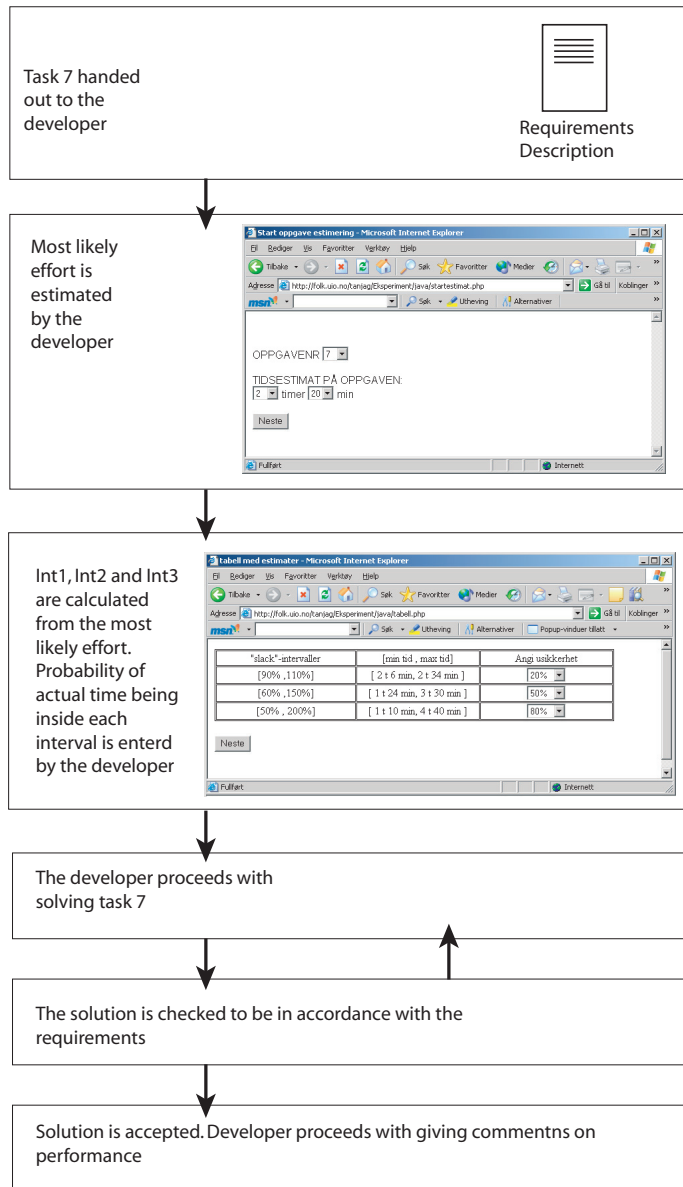


Figure 40 SDE process

During work on a task the participants could take as many brakes as they wished, brake-time is not included in the actual effort time. When they believed themselves finished with a task their solution was tested based on the requirements of the task. If the solution was not accepted, e.g. due to bugs or missing functionality, they were asked to change or correct the program. When a satisfactory solution was obtained, they were presented with their actual effort, their most likely estimated effort and the calculated effort intervals

with assessed uncertainty. The participants commented on their actual effort in comparison to the effort intervals, and other remarks they deemed relevant to their work effort. A new task was handed out when they were finished with their comments. For practical reasons, i.e. to ensure that a task would be finished during the same workday, the succession of tasks differs slightly for each of the participants. This makes the comparison between the participants more difficult. However, due to tasks being similar and independent of each other, it does not affect the analysis of the ability to learn to make uncertainty assessment and of learning strategies to any great extent. The developers finished between 14 and 18 tasks during the five work-days.

After the development phase of the experiment was completed, the participants were interviewed about their learning strategies and other issues that highlighted their learning development.

4.3 Results

4.3.1 Development and estimation skill

As previously stated, the sequence of tasks was for practical reasons slightly different for each of the participants. As not all of them finished all 18 tasks, a subset of 12 tasks completed in almost the same succession are used to assess the developer's development and estimation skill. In table 5 the total effort (TotEff) in work-hours used to complete the tasks, the median estimation error in % of actual effort (Median MRE), and the average time in minutes spent estimating a task (AvEstTime), are shown.

Developer	A	B	C	D	E
TotEff	16:48	19:36	27:08	22:49	24:49
Median MRE	34 %	18 %	35 %	37 %	22 %
AvEstTime	6,8	5,3	4,8	5,4	n.a

TABLE 5 COMPARISON OF SKILL ON THE 12 TASKS COMPLETED BY ALL DEVELOPERS

As seen in table 5 the time spent on the subset of similar tasks imply that the developers are equally skilled programmers. This suggests that differences in uncertainty assessment are probably not caused by differences in programming skill. The Median MRE shows some difference in measured estimation skill, developers B and E showing better estimation accuracy. Interestingly these are the two developers who showed the least improvement in their uncertainty assessment performance, see section 4.3.3. Analysis of

whether estimated most likely effort was generally over or under confident (REE_j) revealed no general tendency for either. Analysis did not uncover indications of learning more accurate most likely effort estimation. Indicating the tasks given, are independent of each other. The average time spent on deriving most likely effort and uncertainty is very similar for all of the developers. In the interviews it was found that all developers used expert judgment, sometimes supported by decomposition of the task into sub-tasks, to estimate most likely use of effort. Expert judgment is the estimation method most commonly applied in the industry as well [8].

4.3.2 Ability to assess relative difference in uncertainty

Developer	Int1	Int2	Int3
A	0.19	0.24	0.20
B	0.11	-0.07	-0.31
C	-0.33	-0.21	0.18
D	0.16	0.11	n.a.*
E	0.48	0.42	0.21

* All confidence levels where 100%

TABLE 6 CORRELATION BETWEEN CONFIDENCE AND MRE (ALL TASKS)

Applying the RecUncAbility measure on all developers and all tasks completed by them, table 6 shows the correlation between confidence level and estimation accuracy. In the case of negative correlation this implies that the developer is good at distinguishing between high and low effort uncertainty tasks. A positive correlation means that it is more typical that a high uncertainty task is considered to be a low uncertainty task, and vice versa.

As seen in table 6, most of the correlations are positive and/or low, suggesting a poor ability to assess the relative difference in uncertainty between the tasks. Developers B and C were the only ones able, to some extent, to separate high uncertainty from low uncertainty tasks. In contrary to the expected, the similarities of the tasks could be a reason for this poor performance. In keeping the resemblance in complexity and likeness in assignment, it was believed that this would help the developers to more clearly see when high and low uncertainty measures were needed by learning from experience. Jørgensen [2] conducted an industrial study of 70 real-life software projects where the situation induced more heterogeneous tasks. In this study it was found a correlation of -0.26 for Int3 between confidence level and estimation error.

To see if there was any learning from experience only the correlation for the ten last tasks is calculated in table 8. If there was substantial learning, this would give more negative correlations and higher negative values.

Developer	Int1	Int2	Int3
A	0.04	-0.10	0.11
B	-0.25	-0.02	-0.45
C	0.11	-0.11	0.25
D	-0.17	-0.05	n.a.*
E	0.47	0.49	0.19

* All confidence levels where 100%

TABLE 7 CORRELATION BETWEEN CONFIDENCE AND MRE (LAST 10 TASKS)

The data in Table 7 suggest that some of the developers, i.e. developers A, B and D, did improve their performance, but not by very much. The improvements may also be due to random variation. All in all these findings indicates that the developers were poorly skilled at separating high and low uncertainty task and this skill improved only slightly, at best.

Interestingly, the correlation between the time spent on the estimation and the uncertainty assessment of a task and the estimation error (MRE) was better than, or just as good as, the correlations in Table 6 and 7. This means that the variance in time spent on the estimation work provided just as good an indicator of the variance of the uncertainty of use of effort, as the developers' own uncertainty assessments. This further supports the poor ability of the developers to assess the relative difference in effort uncertainty between tasks in the previous analysis.

4.3.3 The ability to assess the average level of uncertainty

To analyze the ability to assess uncertainty and the improvement from outcome feedback, "learning graphs" were created for all five developers. The learning graphs visualize the developer's ability to use the outcome feedback to adjust their average level of effort uncertainty assessment to the real uncertainty level. The graphs are in the chronological order the developers solved the tasks, showing evolvement over time. The learning graph was derived as follows:

- 1) Compare assessed uncertainty (confidence) and actual uncertainty (hit rate) for the first five tasks (Comparison Point 1).
The degree of overconfidence at Comparison Point 1 is calculated as $Overconfidence(Int1,T)$, $Overconfidence(Int2,T)$, and $Overconfidence(Int3,T)$, for $T=\{Task1...Task5\}$.
- 2) The comparison at Comparison Point 2 is conducted similarly, but now for $T = \{Task2...Task6\}$.
- 3) Etc.

The rationale for including only the five last tasks in the set of tasks (T) is that it turned out to be a useful number of tasks to study the learning effect. Including fewer tasks would lead to more random variation in the learning

curve due to difference in task complexity, and including more tasks may have hidden some of the learning progress information. Figure 41-45 show the learning graphs for the developers A, B, C, D and E.

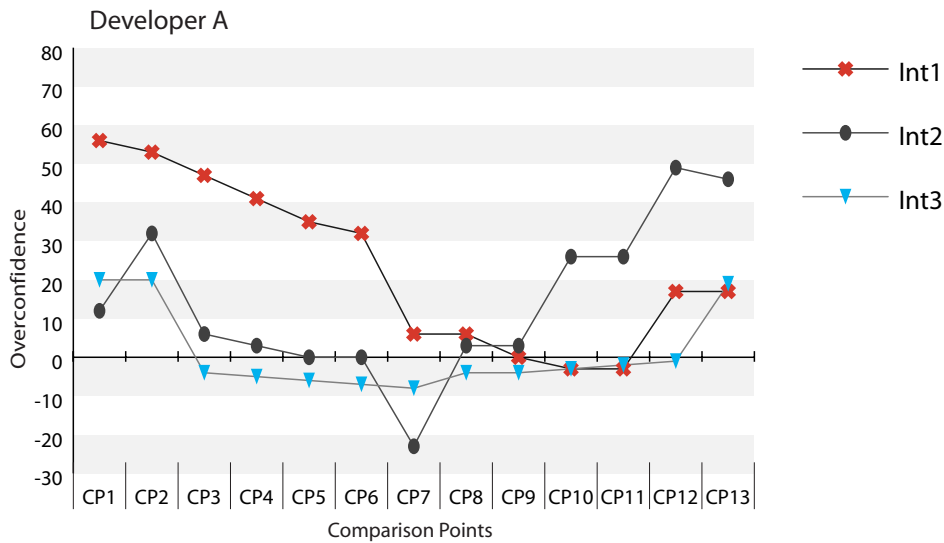


Figure 41 Developer A

Developer A starts out with a very high overconfidence on Int1, and medium high overconfidence on Int2 and Int3. Int2 and Int3 get stabilized around 0 by CP3. Int3 stays at a minor under-confidence level the rest of the experiment, with a relapse up to the initial overconfidence level at the last comparison point. At CP7 there is a drop in confidence for Int2 before it steadily increases towards the end. Int1 gradually loses overconfidence, and is very well asses on CP9-CP11, before increasing at the two last points. Notice the drop between CP6 and CP7. In the interview the developer uttered his understanding of the effort uncertainty assessment becoming less abstract around this point.

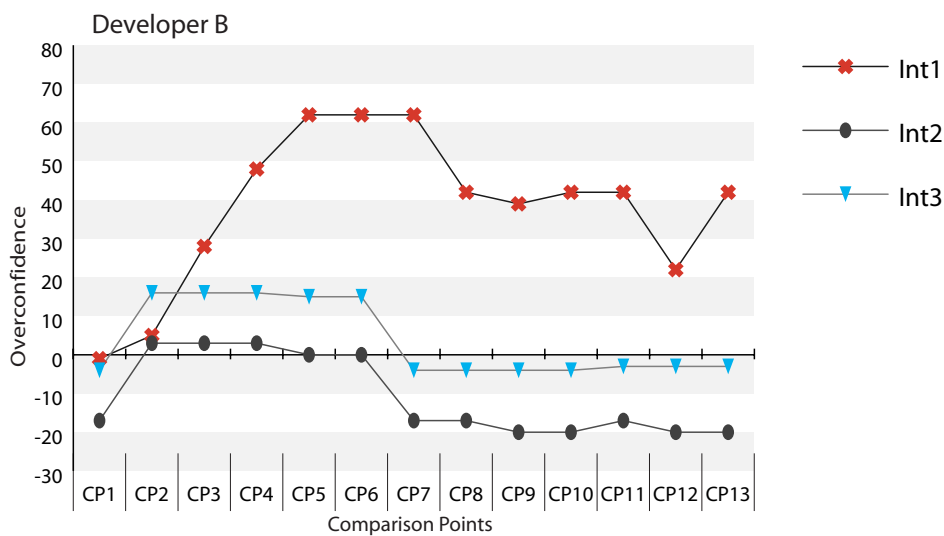


Figure 42 Developer B

Developer B starts at a well calibrated level, Int2 being mildly under-confident. Int2 is adjusted close to 0% before regressing back to the initial under-confidence level. Notice the very parallel appearance of Int2 and Int3, as well as the realistic level for all comparison points for Int3 from CP7. Int1 moves to a very high overconfidence, and there is little improvement to find. Developer B had the most accurate effort estimates and had a much higher proportion of actual effort values inside the Int2 interval compared with most of the other developers.

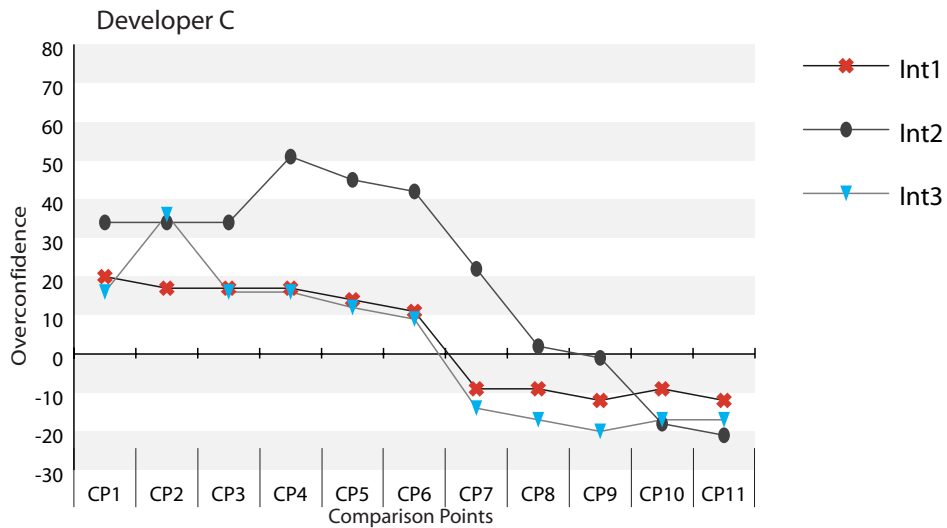


Figure 43 Developer C

Developer C started out overconfident, and then moved to a under confidence for all levels. The degree of under-confidence is however not severe. There are clear signs of learning for all intervals.

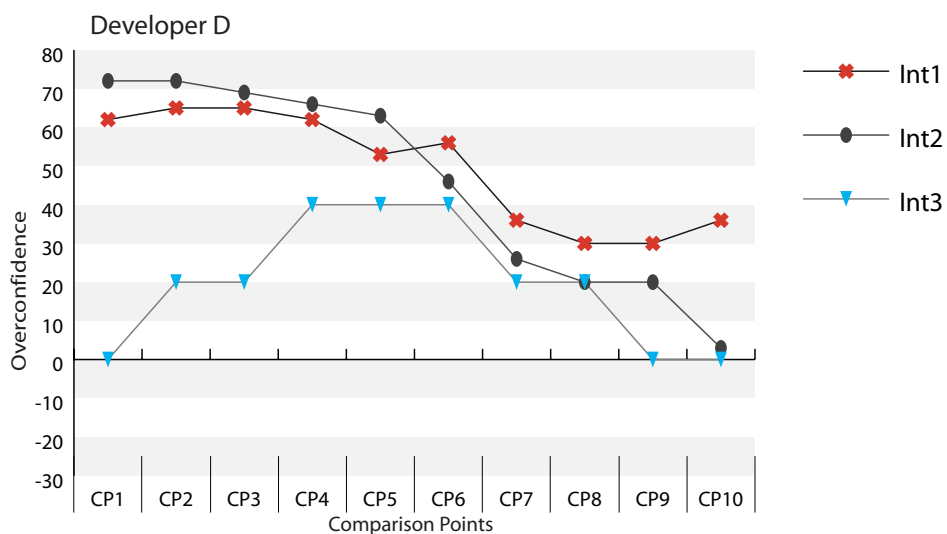


Figure 44 Developer D

Developer D has a very high overconfidence for Int1 and Int2 up to CP6. Int3 starts at a realistic level and moves towards the same level of overconfidence as the other intervals, it reclines back to a realistic state along with Int2. Int1, however, although moving downwards, stops at a relatively high degree of overconfidence. In the interview, the developer said that during the course of the experiment, he had started to regard the Int1 level as a utopian confidence interval, more of a goal to strive towards rather than actually assess realistically. There are signs of learning for Int2 and Int3.

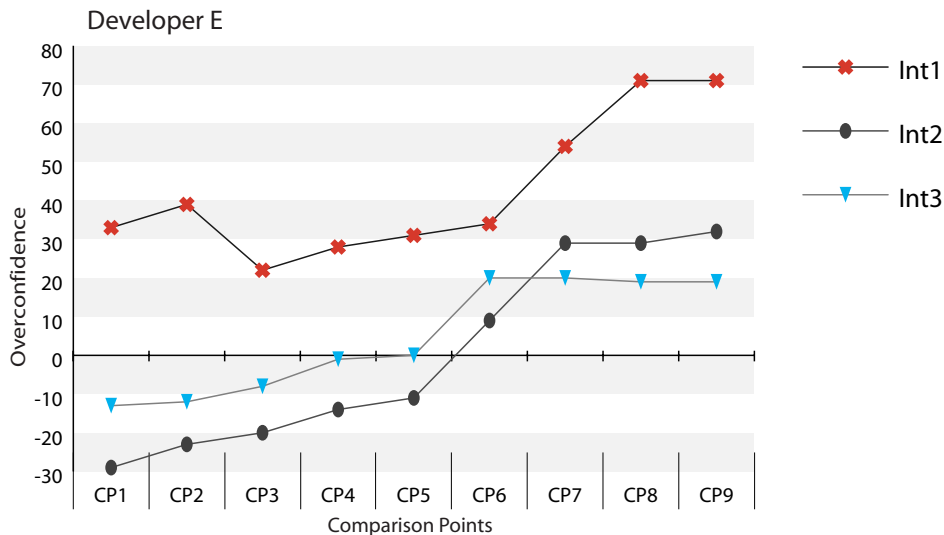


Figure 45 Developer E

Developer E starts with assessing Int2 and Int3 with some degree of underconfidence, Int1 is highly overconfident. All the levels move towards an even higher overconfidence than originally assessed, Int1 as high as 70%. There are no clear signs of learning.

There is a great variety in the developers' ability to learn uncertainty assessment of development tasks. Developers E and B display little sign of learning; developers A, C and D exhibit modest learning from experience.

4.3.4 The uncertainty assessment strategies

The strategies for uncertainty assessment and learning described in this section are based on (1) the comments provided by the developers after each task completion, (2) the interviews with the developers when all tasks were completed, and (3) a comparison of the comments and interview with actual performance. Based on the experiences in [1, 2, 4-12, 24] the software development estimation effort uncertainty assessments are divided into three main strategies:

S1 (Hit rate-based strategy): Uncertainty assessments through comparison of previous hit rates with current confidence levels. For example, if only 30% of previous actual effort was inside the IntI effort interval, then the confidence of the next IntI effort interval should not deviate too much from 30%.

S2 (Analogy-based strategy): Uncertainty assessment through recall of a small set of similar tasks (typically 1-2 tasks) and use of the estimation error of those tasks to set the confidence levels. For example, if the estimation error of the two most similar tasks was about 30%, it is likely that this level of estimation error will occur on the current task as well.

S3 (Intuition-based strategy): Uncertainty assessment without any explicitly formulated strategy, i.e., an intuition-based uncertainty assessment where mainly the properties of the current task is evaluated.

The comments, given by the developers indicating strategy after each task, correspond well with the confidence levels chosen. In the interviews, however, there were discrepancies with some of their descriptions of uncertainty assessment strategies compared with the two other information sources. All sources are valid, but the interviews hold strong indication of biases due to its nature as an information source. There is a strong inclination to believe that statements made during the interviews are affected by how the developers themselves wanted to have conducted the uncertainty assessments in hindsight. The discrepancies are believed to be due to memory and hindsight biases [34]. When questioned if they formed any strategies during their development false memories of doing this could appear. This may be to satisfy internal hindsight revelations that this was something they did, or should have done, or thought they did, when they actually didn't.

4.3.4.1 Main uncertainty assessment strategies

Developer	Source: Task comments + analyses of chosen confidence levels	Source: Interviews
A	S1	S1
B	S3	S2 + S3
C	S2	S1 + S2
D	S2	S2
E	S3	S1 + S2

TABLE 8 MAIN UNCERTAINTY ASSESSMENT STRATEGIES

Developers A and D's descriptions of their use of uncertainty assessment strategy in the interviews, matched the strategies derived from the task comments and the analysis of the chosen confidence levels. The other developers had deviations in pronounced strategy and observed strategy. Developer E in particular described his strategy very differently from observed performance, e.g. there was no use of SI in the learning graph in figure 45. Regardless of this difference, it is possible that developer E tried to apply SI, but was unable to apply the feedback properly or used the feedback in a biased manner. If this was the case however, there should have been some evidence of the use of SI in the task comments. In short, there are reasons to believe that, in general, the strategy categorization based on the task comments and analysis of chosen confidence levels is more realistic than the ones based on the interviews. As the difference between interviews and the other sources show, the task of extracting uncertainty assessment strategies is difficult. To increase the validity of the analysis there may be need for other means of extracting strategies employed, e.g. the use of think-aloud-protocols.

4.3.4.2 The general affecting the particular

The developers were also asked about their decision making strategies *in general*, i.e. whether they were mainly analytical or intuitive by nature. Developers A, D, and E perceived themselves as dominantly analytical, while developers B and C perceived themselves as dominantly intuitive decision-makers. Developers A and D stated that they switched to more analytical strategies, after experiencing their intuition to be frequently very biased. These comments may be interpreted as providing evidence of an ability to be analytic on a meta-level, i.e., to be analytic about choice of decision strategy, which may be a good indicator of learning ability. Developers A and D were also the two of the developers that displayed learning in the experiment. The third developer that displayed learning was developer C. Developer C perceived himself as an intuition-based decision-maker, he also stated that he was aware of his tendency to be overoptimistic in his intuition-based decisions and always tried to compensate for this. This is supported by his comments being clearly more analytical than those of developers B and E. Although this does not show that developer C applies analytical strategies, it shows at least a mature level of reflection about his own biases. This suggests another type of meta-learning ability, i.e., the ability to identify and compensate for one's own biases. In [35], Wilson identifies the awareness of own biases as one of four key steps to correct a mental bias.

Summing up, the results suggest that it's hard to provide a simple model that predicts when a developer will be able to improve uncertainty assessments based on outcome feedback. The results do however show a possible connection between the use of analytical strategies and reflection about

properties of a strategy to indicate a good learning ability.

4.3.5 Perceived skill level, the build of (unjust) confidence

Skill level both in programming abilities and in estimated most likely effort uncertainty assessment, contributes to the build of confidence internally in the developer. All the developers expressed a high confidence in own abilities as programmers, some more modest than others. All the developers expressed satisfaction over their assessment of uncertainty on the effort intervals. Only developer A clearly expressed a bettered understanding of the uncertainty percentages during the course of the experiment. He stated to feeling the confidence intervals and uncertainty percentages were very abstract on the 5-7 first tasks, before really getting the notion of what frequency meant in this context. The others more or less repeated what was stated during the initial briefing and in the instruction manual. If they really understood what the notion of probability (frequency) meant in this context is hard to say. This is further backed by the indecisiveness most of them uttered when asked follow-up questions on the issue.

During the interview the developers was confronted with their actual hit rate, which in most cases did not show an expected adjustment in accordance with past performance. A tidal wave of explanations making the calculated hit rate of lesser importance and accuracy usually followed. What was to be reckoned a hit in an interval had a wider definition for the developers then that used in the analysis. The boundaries were victim of a “fuzzyfication”, i.e. the developers added a subjective slack to them. Actual time outside a interval by about five minutes was usually counted as a hit. If the task was small, i.e. most likely effort estimate about 1 hour, they justified this by the narrowness of the intervals and a hit could just as likely be due to chance as skill. If the task was big, i.e. most likely effort estimate about 4 hours, they justified it by the wideness of the interval; “what’s five minutes compared with 2 hours”. This “ball park” time frame was added to the boundaries by all of them in a kind of semi-conscious state. This only worked to increase the number of hits, i.e. they never regarded a time just inside the boundaries as a miss. Even when this was pointed out to them, they didn’t see anything biased about it. All participants stated that the estimation of most likely effort and getting as close to it as possible was perceived as their main objective in the experiment, and the uncertainty assessment of the intervals was a sub-objective. The notion that lowering the uncertainty level to better calibrate hit rate on the different intervals would accomplish the same as an actual hit, surfaced late in the experiment and never for some of them. This behavior, of counting more hits than actually occurred, contributed to an unjust increase in confidence as the perceived number of hits was higher than it really was.

Crude estimation overruns was also handled in a less than beneficial way to stimulate learning of realistic uncertainty assessment. When shown such overruns, all of them admitted to not consider them when assessing uncertainty at later tasks. Whether the miss was caused by a small bug, misapprehension of the programming task etc. didn't seem to matter. The miss was considered an abnormality in their performance, and they therefore didn't take it into account, the common attitude was that only normative behavior was to be considered. Albeit there were several of these kinds of misses, some more severe than others, this attitude didn't modify – even in the interview setting. Developer A was the only one who reflected, during the interview, that this actually was a bit dim attitude. He recognized that large misses will always occur, you just don't know where and when, and the easiest way to “handle them” was to decrease certainty of estimated effort. The causes of the misses are easily identified in hindsight, and therefore perhaps become so obvious to the developer that the perceived risk of it happening again diminishes dramatically. The hindsight bias, where cause and effect is clearer and more obvious after the fact, is a very common phenomenon, e.g. [28, 34]. It contributed strongly to the unjust build of confidence. Another human behavior bias can also shed some light on the developer's behavior. If outcome is positive it is contributed to own skill, while negative outcome is explained by uncontrollable external factors [28]; this effect also contributed to the unjust build of confidence.

None of the developers wrote down any kind of hit rate score that they updated as more information was available (as more tasks were completed). Some wrote down time spend and/or estimated effort and/or uncertainty levels, however none of them used this historical data actively. They all relied on their memory when eventually recalling past performance. As none of them checked their memory with reality, as well as having an initial biased view of reality, memory bias was rampant among the developers. This behavior also contributed to a further increase in confidence of their perceived uncertainty assessment skill. As confidence in skill increased, memory was further biased in their favor, crating a spiral of unjust confidence build.

4.4 Discussion of Results

In this experiment the learning ability of the software developer was the center of attention, together with how they learn uncertainty assessment of effort estimates. The research questions were:

- How much do programmers improve their assessment of the uncertainty of estimates of most likely effort on the basis of outcome-related feedback?
- What is the relation between the learning strategies for improving uncertainty assessment used by the programmers and their ability to learn from feedback?

In regards to how much developers improve uncertainty assessment on the basis of outcome feedback, this was not as much as expected. In other studies [1, 2] it has been found that software professionals strongly underestimate the uncertainty connected with software development effort estimation; e.g. a 90% certainty in estimated actual effort gives a typical hit rate between 60-70%. This overconfidence is not limited to software development, studies in other domains report similar findings [15, 16, 19]. Further, there seems to be no improvement of estimation skill as a result of on-the-job experience, this is also present in most domains [36]. Two reported reasons for the poor learning from experience are lack of relevant feedback and lack of immediate feedback [22], it is therefore believed that learning may improve as better framing of feedback is provided. The learning environment in this study was set up with this in mind, i.e. the solving of many similar tasks and immediate feedback. The relevance of the feedback was ensured by similarity of the tasks to be solved. The timeliness of the feedback was ensured by providing the feedback immediately after the completion of a task, and just before the uncertainty of the effort estimation of a new task was provided. The hindsight bias was reduced by the short duration (less than 5 hours) of the task. An important rationale for these design decision is that if poor learning is found in a learning friendly environment, even poorer learning would be expected in a more realistic (learning-unfriendly) situation. That is to say, it was studied necessary, and not sufficient, conditions for learning. Nevertheless, the developers struggled with learning uncertainty assessments from experience and two out of five developers may not have learned at all.

The estimation strategy most frequently stated used was decomposition of the task (bottom-up). This was often supported or replaced by e.g. “the educated guess”. In [8] potential problems, with the use of decomposition, are identified as only being effective for accurate estimates if the uncertainty of the whole task is high. Further weaknesses of bottom-up are that it’s easy to forget activities, underestimate unexpected events, it’s dependent on a developer

with proper experience, and it doesn't encourage history-based criticism of the estimate. Considering the tasks used in the experiment are beginner tasks, which rarely fall under high uncertainty, the use of decomposition is unfortunate. On the other side, one can argue that the high level of detailed effort estimation demanded, i.e. man-minute effort, as well as the small size of the task lead to higher uncertainty. However, the poor calibration of uncertainty seen in the experiment does not suggest that decomposition was especially successful. When focus is mainly on the decomposed parts and their uniqueness in the task context, focus is not on learning uncertainty assessment. This, then, perhaps lead to the use of an intuition-based strategy of uncertainty assessment that hampered learning because i.a. past experience wasn't considered.

The uncertainty assessment strategy of the two developers with the poorest learning was found to be mainly intuition based, as well as showing a lower level of reflection over their strategy than the others. This suggests that for learning to emerge, an explicit strategy that incorporates past experience is necessary. Improvement should then be visible as more data becomes available. The strategies used by developers with better learning may have been:

S1: Adjustment of confidence levels based on hit rate of previous (reasonable similar) tasks. As the number of completed tasks increases, the accuracy of the uncertainty assessment improves. This learning strategy is similar to the learning in the formal uncertainty assessment model sketched in [10].

S2: Recall of the most similar tasks and use of the uncertainty (estimation error) of those tasks to determine the confidence of the current task. As the set of similar tasks increases, the accuracy of the uncertainty assessments improves.

Other domains have also uncovered the lack of improvement when using intuition-based strategies, e.g. [20]. The results do not however imply that there will be learning if the uncertainty strategies S1 or S2 are applied. The developers that applied S1 and/or S2 didn't display impressing improvement in this study. The results does however indicate that necessary condition for learning uncertainty assessment are (1) the use of explicit strategies that improve as more data becomes available; and (2) non-reliance on intuition-based strategies. The learning friendly environment created to study strategy is artificial. It is however a belief that if poor learning is found in a learning friendly environment, one should expect poorer learning in a more realistic setting. The creation of an artificial environment is therefore considered a meaningful way of studying learning. It is therefore the weaknesses in the design of this study one will have to identify to isolate reasons for the poor learning observed, before concluding software developers, themselves, to be poor uncertainty assessment learners.

Although the Java tasks used in the study was considered easy for an experience programmer, they still seem to demand relative amounts of concentration to get done. The tasks being easy, and of beginner character, did not seem to lower a “pride in work” attitude. Therefore, due the mere demand of effort in solving a programming task, focus may have shifted towards this rather than learning uncertainty assessment over estimated effort. This is supported by statements recorded during the interviews with the participants. It therefore seems plausible that for learning of uncertainty to occur frequent reminders that this is the goal, and not to excel in programming, is needed.

The nature of the feedback was strictly of a statistical nature, i.e. the developers were given facts about actual time spent and their initial estimate of time and uncertainty. In hindsight it seems that this feedback was framed so that it could be interpreted by the developer in his favor, thereby biasing how much he needed to adjust to reach a realistic uncertainty level, here perhaps resulting in a less than latent adjustment. In this regard, the framing and availability of the probabilities used also comes into play. Through the written comments and statement from the interview, there surfaced discrepancies towards how frequency should be used practically. The developers with the best learning, were also the ones with a reflective attitude towards this problem as well having a realistic outlook, i.e. the uncertainty levels didn’t represent other kinds of goals or meaning, like wishful-thinking, as uttered by some of the others. To aid the learning as best as possible, both the framing of feedback and probability needs to be carefully considered as it can potentially both hamper and support the learning process.

4.5 Summary and Conclusion

This experiment wanted to better understand: (1) how much programmers improve their assessment of the uncertainty of estimates of most likely effort on the basis of outcome-related feedback; and (2) what the relation is between the learning strategies for improving uncertainty assessment used by the programmers and their ability to learn from feedback.

To achieve this, an experiment in a “learning friendly” environment was set up using experts student software developers solving between 14 and 18 small (less than 5 work-hours) programming tasks. The effort estimation uncertainty ability and progress of the five developers were studied by giving feedback after each task.

The conclusion from this study is that some developers have the ability to improve their uncertainty assessment in the learning friendly setting used in the experiment. However not all of the developers showed this tendency, two of them displayed very little or no improvement (learning). They did not seem to be able to use their outcome feedback to improve their performance. These developers used in large part an intuition-based strategy; the other developers displayed better and/or more visible improvement when using an explicit strategy. Environmental influences, like the framing of feedback and the (lack of) availability of the probabilities, lead to an unfortunate increase in unjust confidence of preceded skill of uncertainty assessment. This may also have contributed to the poor learning observed. A hypothesis is therefore that when learning uncertainty assessment from experience by using outcome feedback, a *necessary* but not sufficient condition is the use of an *explicit* uncertainty assessment strategy that improves with more feedback.

An implication of the findings is that one cannot expect uncertainty assessments to improve when they are dominantly intuition-based. The hypothesis is in accordance with the poor effort uncertainty assessment performance and improvement in real software projects [1], and studies from other domains, which show that overconfidence in estimates is difficult to avoid when applying intuition-based strategies [15].

5 Industry and Research Implications

The aim of this study was a focus on learning, stated in the general research questions as:

RQ1: Given a favourable learning environment, can people learn to better calibrate uncertainty estimates when provided with feedback on performance?

RQ2: Is there a relationship between a participant's learning strategy and the amount of learning observed?

Implications of the results found in the experiments held to investigate the research questions are listed below.

1. For learning to occur the learning process may need to be aided by explicitly stated learning strategies. This also incorporates the notion of keeping focus on learning throughout the learning period like e.g. reminding people that learning better uncertainty assessment is the main focus, in contrast to i.a. excelling on the task to be solved.
2. There must be given special attention to the framing of the probability measures used to state uncertainty over effort. This implies that it is important to check for adequate understanding of the concept of probability and uncertainty, give proper explanations of these terms, and issue reminders of the agreed upon definitions at regular intervals during the time of learning. In this respect, it also seems beneficial to support mathematical probability definitions with natural language descriptions, and oral consensus through debate of these definitions.
3. Feedback should be given in such a way that: (1) several kinds of feedback is used and issued frequently, as a minimum it should be given at naturally occurring places; (2) the possibility of subjective interpretations on performance is avoided as much as possible;

- (3) it can be directly transferable as input to future uncertainty assessments, i.e. framing of the appearance to visually match the uncertainty assessment process to come and/or history based tendencies are made clear.
4. Neither task skill level nor degree of initial knowledge seems to have any influence on the ability to learn uncertainty. I.e. the ability to “know how” and learning in respect to uncertainty does not seem to have a relation with one another. This indicates that there are different qualities and learning strategies that are effective for learning the skill of “know how” versus learning “how uncertain is”. The design and framing of learning environment, and feedback, should therefore reflect how learning uncertainty assessment is best obtained when this is the purpose.

In the following, the above identified implications are discussed in separate sections respectively.

5.1 Implication no 1: Aid the Learning Process

The difference in learning in the two experiments is substantial. In GQE 13 of 15 participants showed observable learning; in SDE poor learning was observed by 3 of 5 developers, the remaining two may not have learned at all. There was a dissimilarity in focus, whether learning is possible versus how learning is achieved, contributing to this difference in degree of learning. Nevertheless, measures were taken to ensure similar conditions for learning uncertainty assessment in both experiments. The nature of the task will, however, ultimately control how feedback can be framed and how strategies eventually evolve.

When comparing the tasks used in the two experiments, the gap in task difficulty is what stands out as the dominant difference. This is due to a relative higher complexity of solving a programming task compared with answering general knowledge questions. The latter is a one dimensional task, and the difficulty lies in determining if one knows the answer or not. Programming tasks involve many factors that need to be considered when embarking on a possible solution; the solution must also be validated to be in accordance with requirements. The increase in elements that need to be handled does not seem to be easier to control or more trivial even though the programmer is highly skilled, and the tasks used seem small and simple from the outside. In regards to the amount of tasks solved, this is substantially

higher in GQE than in SDE. Small sample size in SDE, 18 programming tasks versus 960 questions, could therefore be a factor in difference in learning. The programming task, naturally, takes longer to complete than answering a question. Feedback of success or failure therefore also takes longer to receive from the point when a preferred solution is identified. This difference in time from conception of a solution to finished “product” is a stress factor in SDE compared with GQE. All participants in both experiments have previously been identified as experts. Shanteau [37] identifies task characteristics as an crucial factor of experts to perform competently. To support learning it therefore seems that the introduction of explicit learning strategies could be beneficial. It may reduce taxing factors if the characteristics of the task to be solved is of substantial complexity and or has a longer than immediate success or failure response.

Motivation and commitment to do well is a key part of improving performance [7, 32, 38]. The drive behind motivation may arise from different sources, thereby affecting i.a. the degree of learning and the build of the learning. [7] Presents several empirically validated findings that are useful for the tailoring of motivational mechanisms during uncertainty assessment in software development. Some of these guidelines are believed to have validity in general, and GQE is therefore also scrutinized accordingly. In compliance with these guidelines motivation build was directed was towards process and not outcome, and expectations towards outcome (both task performance and uncertainty assessment) were not expressed. Focus on process may have been clearer in GQE than in SDE, as these participants were reminded of this through feedback sessions. Although stated in the initial SDE briefing, there are indications (which surfaced during the interviews) that this may have been unclear or interpreted differently by the software developers. In disagreement with the guidelines [7], external incentives to do well was used in GQE, i.e. scratch tickets were given to the ones who showed best improvement, to reduce indifference and boredom. Such incentives were not used in SDE. The participants in GQE showed no indication of conflicting goals, as found in SDE. The discrepancy between performance, comments made and statements in the interview, in SDE, indicates goal conflict. Their motivational focus was not undivided towards the learning of uncertainty, but rather divided amongst several conflicting goals: e.g. hitting most likely effort dead on, being “good” research objects, finishing ahead of time to maximize pay, pride in work, or distractions caused by e.g. personal issues not related to the experiment. As the developers were left to their own devices after the briefing, they were not inclined to make any kind of stance to defend their use of uncertainty as recommended in the guidelines. To follow this guideline was not in compliance with design of the study, since how developers learn was the focus. But by not doing this, there could have existed lack of proper motivation

and thereby lack of learning. All in all it seems that the motivational build to improve was stronger and/or better for the participants in GQE than in SDE. This could in turn indicate that proper motivational build towards keeping focus on learning is very important to aid the learning process. What necessary motivational cues may be in regards to **learning** realistic uncertainty assessment needs further exploration.

The learning of uncertainty assessments in both experiments are driven by the goal of realism. In the software engineering context this isn't as straight forward as one would think or wish for in a research of learning context. Jørgensen et al. [1] unravel a double meaning connected with effort estimation and the pertaining uncertainty assessment. Developers believed that by creating more realistic wider prediction intervals it would make them look less skilled in effort estimation and software development. It turned out that they were not paranoid; project managers actually do evaluate prediction intervals in this way. Wilson [35] would classify this as a mental contamination. Even though this is unwanted, and has been discovered by the judge in question, ridding behaviour of this influence isn't necessarily unwanted. A reason for this is that continued acceptance, and execution, of this biased behaviour may hold higher rewards on a personal value scale. Here, this would be to e.g. achieve a favourable view from the boss or perhaps to achieve easier communication with e.g. colleagues. Although realistic in regards to uncertainty, very wide intervals may hold low practical value; i.e. determining that the first transatlantic flight took place somewhere between the year 1900 and 1970, gives little useful information although one can be 90% certain that this is true. The implication of this is that if research in trying to see if true realism is possible, this should be strongly communicated with subjects of learning. This would, in turn, add to the artificiality of the research if this is not what the real world wants to achieve. As depicted here, the reason for not wanting to strive towards realism is outside influence. The use of explicit learning strategies could therefore also serve as a political tool in fighting e.g. the boss's prejudice, as it would be something that was objective and thereby not mirror skill. The resistance towards realism could also be internal, as found during the interviews in SDE. One developer stated that to actually get actual effort inside the smallest interval (int1) was something to strive after, not necessarily accomplish. This further supports the need for shifting focus and motivation towards that learning realistic uncertainty learning is desired, perhaps best done by continues reminders and unambiguous communication.

The findings in SDE point towards the use of explicitly stated strategies, like S1 and S2, may lead to better learning than the use of intuition based strategies. Avoiding intuition based strategies would therefore be beneficial. Such strategies seem to surface more frequently when incentives to use other

approaches are absent. Both continuous reminders of the learning focus, and inviting to use explicit strategies could have a positive effect on learning.

In sum, aiding the learning process by providing and sustaining an atmosphere of learning presence throughout the learning period, done either by the use of explicitly stated learning strategies or frequent reminders that learning of uncertainty is the focus, could contribute to increase in learning.

5.2 Implication no 2: Avoid Probability Angst and Arrogance

The understanding of probability is often hindered by angst and arrogance, and as long as understanding is absent or difficult learning ability suffers. Bernstein nails the crux of probably understanding (on p. 48 in [39]): *“Probability has always carried [a] double meaning, one looking into the future, the other interpreting the past, one concerned with our opinion, the other with what we actually know.”* The difficulty of distinguishing between the two meanings of probability was seen at the start of both experiments. The participants on GQE were keener on discussing this issue than the software developers on SDE. All participants in both experiments were briefed and given the opportunity to question and discuss the meaning of probability. The discussion continued throughout GQE during coaching sessions, when feedback of the frequency distribution was given. This gave GQE participants reason and time to ponder and ripe over the dual meaning, whereas the software developers were left to their own devices and an instruction manual. It is possible that the consensus arrived at during the briefing in SDE regressed back to the “gut feeling view”. Statements in the interview of the developers support this assumption, as there is a correlation between the people using the frequency approach, i.e. the ones not mainly relying on intuition based strategies, and learning ability. The low number of developers studied puts a reservation on this connection. It, however, is supported by the conclusion made in [5]. The notion of using past experience was hard to come by for most of the software developers. That the probabilities should reflect frequency of hits in the confidence intervals given similar tasks many times in the long run was believed understood. The use of past performance was, however, limited to what they could remember, as none of them wrote down easily applicable accumulated performance feedback for themselves. The ones who kept some kind of record of their performance didn't use it actively during the estimation process on a new task. In contrast, the subjects on GQE was frequently reminded of and coached with experiences on past performance. The “out-of-sight-out-of-mind” effect seen in SDE is a well known phenomenon, also reported in e.g. [4, 35] , and described in e.g. [28] and [34]; it alas contributed to

sustaining an arrogant attitude to the practical use of frequency (probability).

In SDE there was observed a bias in perceived success rate. The developers were more lenient as to what counted as success compared with what was statistically accepted. Einhorn writes in [17] that if unaided memory for coding, storing, and retrieving outcome feedback contributes to a illusion of validity of this feedback. Therefore, when the developers were left to their own devices, they frequently used outcome feedback (given by unaided memory) that gave a biased impression of their performance. By their use of statistically historically un-corrected information, and the subjective completion characteristics of programming tasks, they never had the same conditions to support more realistic calibration as the subjects in GQE. This arrogance towards the use of probability was not so strong, and for most participants completely absent, in GQE. One could speculate that a reason for this difference in behaviour is that the software developers had more to prove in regards to skill. The developers were especially picked because of their skill level; they perhaps transferred a high skill level to be a requirement in all areas in the experiment even though it was made clear it was not. Not displaying lack of understanding over probabilities could therefore be a function of not wanting to show limited abilities on an overall level. In comparison, the participants in GQE had nothing to prove, as they were picked on the basis of being human beings. Therefore, it was “allowed” to have difficulties in understanding probabilities.

Also due to the challenges of subjective success interpretations, the software developers were more prone to fall victim for biases, e.g. memory bias or regarding risk of failure as lower if oneself is in control of the task, as described in [26, 32]. How this should be handled is influenced by more than the straight forward statistical interpretation that was so easily accessible in GQE. Should one (a) use the interpretation of the estimator in charge, or (b) stress the educational/learning of a “correct” definition. The first suggestion would lead to huge organizational communications difficulties, described in e.g. [10]. It is however actually one of the present strategy in the industry today [6]. But taking into consideration the industry’s continual request to “solve the estimation problem” [24], more needs to be done. Therefore, stressing better guidance of the definition does seem to be the preferred path. However it is no “silver bullet” as this trail is littered with other factors contorting the “clarity” of statistics, like i.a. previously discussed conflicting goals.

The angst and arrogance towards probability and uncertainty resulted in overconfidence, due to an unjust build of confidence in skill, in SDE. Taylor [39] gives a perspective on the overconfidence issue, so often deemed unreasonable and unwanted by e.g. decision and behavioural science, as well

as here. He points to several studies where the people who were prone to be the most realistic are also the ones with moderate depressions and/or low self-esteem. The self-serving interpretations, unrealistically positive self-serving evaluations, exaggerated perceptions of control and unrealistic optimism can be seen as survival mechanisms in order to maintain mental-health in an otherwise harsh world. He argues that the previously established fact of realism as a characteristic of a healthy soul is not as obvious and clean cut as believed. It braces us to deal with i.a. negative feedback, motivation to master a task, and the ability to be happy and content. This view of overconfidence implies that not only may it be unfertile to “kill it”, but it may also be impossible. It would therefore be a much more effective and healthy strategy to use explicitly objectively guided learning strategies to achieve realism, rather than intuition based strategies as they could be a subject of the above mentioned survival instincts.

To avoid the unfortunate attitude towards probability and uncertainty a first step seems to acknowledge that people may not have a casual relationship to uncertainty [33]. Further it is no doubt that proper understanding is a crucial key to achieve learning in the use of these concepts. To ease the transition towards acceptance (externally and internally), together with proper explanations of these terms, this can be achieved with open and including debate. It is also seems important to issue reminders on the agreed upon definition, serving also to check that understanding hasn't diminished or changed.

5.3 Implication no 3: Frame Feedback carefully

Feedback was given in both experiments, the nature and use of the feedback is however different. The feedback given in GQE was of a rather aggressive nature, employing outcome feedback, performance feedback and to some extent processes feedback. As the motivation of SDE was to see how, rather than if, improvement in uncertainty assessment occurs; the coaching aspect of the feedback cycle was left out. There was also a more one-sided outcome feedback issued, at the expense of clear performance feedback. The developers were not given any cue as to how they could arrive at the certainty percentages for the confidence intervals. As already discussed in section 5.2, this contributed to unfortunate behaviour in regards to achieving learning. Feedback was also given at naturally occurring places in SDE, i.e. immediately after the completion of a task, as in GQE.

The major difference in task completion verification influences feedback by

making it liable to more subjective interpretation in SDE contra GQE. As illustrated by the developers' biased tendency to leave out extreme cases of over-run. If feedback is not bombastic in its clarity on learning performance, it can easily be distorted to suit the participant's perceptions of own learning progress. This indicates that as task complexity increases, the need for additional learning support other than outcome feedback gets more pressing; as it may lower ability to deduct relevant information without guidance.

Increase in confidence is anticipated to be greater when performing a difficult task, e.g. programming, successfully than an easy task, e.g. answering general-knowledge questions. One can therefore assume that the build of confidence, unjust or otherwise, in SDE occurred more rapidly and forcefully. This is supported both by the developers' written comments and statements in the interviews. On GQE, build of confidence was, in comparison to numbers of tasks completed, relatively slower. It took hundreds of questions answered and continuous coaching to increase the confidence of those who showed need for improvement, both trivia wise and calibration wise. It is therefore a thought that when increase in confidence is very rapid, it is an increased danger of it being exaggerated in accordance to actual performance. This bias is further enhanced when there is no "outsider" putting a damper on the confidence-build. Issuing personal help in interpreting feedback may decrease possibility of subjective interpretations of feedback, and thereby reducing hampering of learning. If this is not practical or possible, the framing of feedback becomes exceedingly important.

The visual presentation of the feedback given in the first experiment was of a design that matched how uncertainty was to be given in the next task, i.e. next question and next question pile. The visual presentation in SDE, although containing all available information on performance on the task just completed, was not directly visually similar to how uncertainty was to be given on future tasks. This seems to have contributed to some of the information becoming "lost in transition". The participants on GQE were given a historical summary of day 1 performance before starting day 2. Such an accumulated presentation of past performance with tendencies was not issued to the programmers at any point. As stated earlier, many of them thought of wanting such a summary, but actually putting this wish into action had too many mental barriers. In the interviews they stated that if accumulated performance feedback was issued together with outcome feedback on the task just finished, this could have had an impact on their learning strategy. Further research will have to be done to eventually verify if this actually would lead to a difference in behaviour.

The experiences from the two experiments, indicates that framing of feedback

need to be done very carefully, also supported by [32]. If done properly, i.e. diversely, frequently, unambiguous and transferable to future use, it could be the most essential factor in achieving learning. The danger is that if framing is unfortunate, e.g. violating the suggestions listed above, it could lead to poor or no learning. In a worst case scenario, such biased learning could be damaging in regards to use in real world settings as unwanted confidence in effort uncertainty is allowed to continue.

5.4 Implication no 4: Acknowledge the difference in learning “know how” versus “how uncertain is”

In GQE it was found that level of knowledge, found by portion overall correct answered questions, did not seem to be connected with the ability to learn uncertainty assessment. The software developers used in SDE was found to be of equal skill level in regards to programming abilities, as some of them possibly did not learn at all, skill level does not seem connected with ability to learn uncertainty. Lichtenstein et al. [3] also didn't find a dependence between knowledge level and calibration ability (realism in uncertainty assessment). Ericsson et al. [40] writes *“individualized training activities especially designed by a coach or teacher to improve specific aspects of an individual's performance through repetition and successive refinement”* as fundamental for learning progress. As they are fundamentally different skills, it may be that learning to **know how** to do something requires different learning strategies than learning **how uncertain** something is. The implication of this is that learning uncertainty assessment needs a tailored learning environment that may not be similar to how one would design learning e.g. to acquire a programming skill. The design and framing of a learning environment and feedback should reflect the special needs of learning uncertainty assessment. What these special needs may be needs to be further investigation.

6 Summary and Conclusions

The theme of this thesis is uncertainty assessment on estimated effort. Two studies (experiments) were conducted under artificial conditions in the purpose of investigating issues concerning learning realistic uncertainty assessment. More specifically, whether this is possible given favorable learning conditions, and how learning eventually occurs.

The first experiment (GQE) used general knowledge questions to investigate the possibility that people possessed the ability to improve uncertainty assessment when given outcome feedback. Participants answered questions with four given alternatives. For each answer they considered how probable it was that their answer was correct. After answering a pile of questions with the same difficulty, six question piles on each of the two experiment days, they received feedback on performance. They were given a summary over day 1 performance, before embarking on day 2 of the experiment. It was found that the participants were quite adequate initially, and 13 of 15 participants had a positive adjustment towards realism on day 2. There was not found any significant connection between a participant's knowledge level or the global difficulty of the questions used and the ability to learn uncertainty assessment. There were indications of questions assessed as easy by an individual were also easier to adjust towards realistic uncertainty assessment. Internally difficult questions were on the other hand very robust towards positive adjustment for nearly all participants; this was independent of the questions being hard or easy globally. Initial knowledge level had no impact on learning ability.

The second experiment (SDE) had a software development context. It investigated how much improvement, given outcome feedback, software developers made in assessing the uncertainty of estimated most likely time effort. The personal learning strategies of the developers were also investigated. Simple Java tasks and expert programmers were used. The estimated most likely effort for a task was used as basis for calculating three confidence intervals. The widths of the intervals were chosen to reflect a narrow,

a medium-wide, and a wide effort interval. For each of these intervals the developers were to assess the probability of actual time used on a task being inside. They were given feedback immediately after task completion. Three developers, using explicitly stated strategies, displayed poor learning. The two remaining developers, who mainly used an intuition based strategy, did not display any observable learning. There was revealed environmental factors that caused an unfortunate influence on learning and behavior, i.e. factors that may have sustained an unjustified overconfidence in own uncertainty assessment skill, and thereby could have hampered learning.

On the basis of the studies conducted the following conclusions and implications were reached: The learning process may need to be aided by explicitly stated learning strategies. This also incorporates the notion that focus on learning must be sustained in some way throughout the learning period. This can e.g. be done by frequent reminders of the goal of the learning session. There must be given special attention to the framing of the probability measures used to state uncertainty over effort. This implies that it is important to check for adequate understanding of the concept of probability and uncertainty. Oral consensus through debate of these definitions seems beneficial in checking for understanding. The probability terms should be properly explained; in this respect, it also seems favourable to support mathematical probability definitions with natural language description. Reminders of the agreed upon probability definitions issued at regular intervals during learning help increase understanding and prevent misconceptions appearing during time of learning. Feedback should be given in such a way that: (1) several kinds of feedback is used and issued frequently, as a minimum it should be given at naturally occurring places; (2) the possibility of subjective interpretations on performance is avoided as much as possible; (3) it can be directly transferable as input to future uncertainty assessments, i.e. framing of the appearance to visually match the uncertainty assessment process to come and/or history based tendencies are made clear. The design and framing of the learning environment and feedback given should reflect the special challenges in learning acquisition of uncertainty assessment, as this seems very different from learning other kinds of skill.

6.1 Further Work

Design weaknesses and the implications found in the experiments will be used as input in the design of an eventual next experiment. This next experiment is planned to be conducted in a real-world context, and circumstances and special needs specific for such a setting will have to be taken into consideration when using experiences made here.

Appendix

Appendix A		
Calibration learning ability in the GQE, results for each participant		97
Appendix B		
Figures and data used in analyzing Task Difficulty and Knowledge level in GQE		129
Appendix C		
Experiment 1 Participant Manual		143
Appendix D		
Experiment 2 Participant Manual		155
Appendix E		
Java Development Tasks used in Experiment no 2		163
Appendix F Article		169

Appendix A

- Calibration learning ability in the GQE, results for each participant.

In the following the analysis of each participant's performance in the GKQ-experiment is given.

Definitions relating to the use of the uncertainty levels
(restated from section 3.2.2):

${}_{\text{up}}\text{Lev}_n$ = upper limit Level n

${}_{\text{lo}}\text{Lev}_n$ = lower limit Level n

$Q\text{Lev}_n$ = the number of questions with assessed uncertainty of Lev_n .

HitLev_n = the number of correct answers with assessed uncertainty of Lev_n .

$\text{HitRateLev}_n = \text{HitLev}_n / Q\text{stLev}_n = \text{ratio of portion correct on a } \text{Lev}_n = \text{hit rate}$

Participant I

TABLE A-1

Piles n	20' &	100' &	10' &	Day 1, total	20' &	100' &	10' &	Day 2, total	
	80'	60'	40'		80'	60'	40'		
QLev _n	0	3	10	6	19	10	10	1	21
	1	5	9	1	15	13	13	8	34
	2	4	9	15	28	9	9	4	22
	3	2	4	3	9	17	7	11	35
	4	10	7	6	23	10	11	5	26
	5	16	13	11	40	22	19	21	62
	6	75	54	91	220	44	44	79	167
HitLev _n	0	6	24	14	44	24	28	8	60
	1	14	19	4	37	26	24	12	62
	2	13	21	29	63	13	12	5	30
	3	11	8	3	22	18	16	17	51
	4	20	12	7	39	12	13	12	37
	5	19	18	13	50	23	22	27	72
	6	77	58	92	227	44	45	79	168
HitRateLev _n (HitLev _n / QLev _n)	0	0,50	0,42	0,43	0,43	0,42	0,36	0,13	0,35
	1	0,36	0,47	0,25	0,41	0,50	0,54	0,67	0,55
	2	0,31	0,43	0,52	0,44	0,69	0,75	0,80	0,73
	3	0,18	0,50	1,00	0,41	0,94	0,44	0,65	0,69
	4	0,50	0,58	0,86	0,59	0,83	0,85	0,42	0,70
	5	0,84	0,72	0,85	0,80	0,96	0,86	0,78	0,86
	6	0,97	0,93	0,99	0,97	1,00	0,98	1,00	0,99

Starting at level 6 (99-100%), there is an improvement from day 1 to day 2. She has reduced the number of questions on this level on day two, as well as only missing a single question the hole day (in the 100+60 piles), which is a good improvement for this level as she missed 7 questions on day 1. Note that she had 227 questions on level 6 on day 1 and 168 on day 2, which is way beyond what most of the others have on this level. This indicates a very large amount of knowledge in the field of trivia and I also would expect a very justified confidence in own ability to get these kinds of questions correct.

A learning effect on level 5 (91-98%) is not so clear, although it looks like she's trying. As the number of questions in this category increases on day 2, it is reasonable to assume that she is trying to use this level more actively for the questions she would have put on the top level the day before. The number of misses is the same on both days. On day one they are kind of equally distributed but on day two they are escalating towards the end of the day (perhaps getting tired); also indicating that she is trying to grasp the

difference between the top two levels.

Level 4 (76-90%) improves towards the end of day one, and is within the level boundaries on the two first points on day 2 before dropping significantly on the last piles. This level has on the whole a small share of the questions, and this should be taken into account. Looking at the overall improvement from day to day, there is movement towards level boundaries and the number of questions are (almost) the same on each day.

Level 3 (61-75 %) has the lowest number of questions on day one, in fact leaving all the points on this level questionable (with perhaps exception of the total for day1 and day2). On day two the amount of questions has more than doubled, and it looks like some of the questions from level 2 have found their way to this level as the numbers of questions on level 2 has decreased proportionately. At the end of the day she gets the point within the boundaries, after first being way over and then way under - indicating she is trying to adjust.

On level 2 (41-60%) she is pretty good on day one, getting the two last points within the boundaries. On day two she is consequently under confident, but the share of questions is cut in half and the last point has only a sample of 5 questions - so there should not be laid too much into this level. It is interesting to note that both the level above and below experience an increased share of questions on day two. It can therefore be suspected that the questions moved from level 2 to levels 1 and 3. This may indicate that she concentrated more on these two levels than level 2.

Level 1 has little indication of improvement or adjustment in the right direction but an increased under confidence on day two. The points for the 10+40 piles are low. By only looking at the two first points for each day the share of questions even increases on day two, but with no lucky effect as the under confidence persists at this level too.

On level 0, where the "no idea" rule should be applied, there also shows up a under confident indication. The 100+60 point on day two (36%), is the one nearest 25%, and the point with the largest share of questions (28). I do believe she tried to apply the rule as best she could; it is difficult to say if she could have adjusted better or the numbers are true random small numbers and would have adjusted themselves with time and more questions.

Looking at level 0,1,2 as a whole there is under confidence on all of the levels, and looking at levels 4,5,6 (disregarding level 3 because of the low share of questions) there is overconfidence. She more easily complies to kill her overconfidence, and therefore adjust towards the boundaries of these levels, than to become more confident on the lower levels. The impression of her overall personality is reflected in this behavior. She has high confidence in the things she knows and low confidence when she becomes unsure. As well as she easily lets herself be shot down when challenged on something she knows. I can theorize that she took the feedback of being too overconfident on the upper level and applied it on an overall basis even though the lower levels weren't that

bad on day one (especially as they became worse on day 2). All this indicating that she became less confident on day 2 on the whole; this is also supported by her parallel shift in the adjustment graph from day 1 to day 2 (figure 2).

Participant II

TABEL A-2

Piles n	60' &	20' &	100' &	Day 1,	60' &	20' &	100' &	Day 2,	
	10'	80'	40'	total	10'	80'	40'	total	
QLev _n	0	1	7	5	13	10	10	8	28
	1	12	14	26	52	19	18	18	55
	2	18	17	9	44	19	19	15	53
	3	13	11	6	30	16	15	11	42
	4	14	10	11	35	12	10	19	41
	5	10	2	5	17	3	4	5	12
	6	29	32	22	83	24	16	16	56
HitLev _n	0	10	26	16	52	24	38	40	102
	1	41	43	60	144	39	38	41	118
	2	37	30	28	95	32	33	22	87
	3	16	16	13	45	24	15	13	52
	4	17	10	16	43	14	13	23	50
	5	10	3	5	18	3	5	5	13
	6	29	32	22	83	24	17	16	57
HitRateLev _n (HitLev _n / QLev _n)	0	0,10	0,27	0,31	0,25	0,42	0,26	0,20	0,27
	1	0,29	0,33	0,43	0,36	0,49	0,47	0,44	0,47
	2	0,49	0,57	0,32	0,46	0,59	0,58	0,68	0,61
	3	0,81	0,69	0,46	0,67	0,67	1,00	0,85	0,81
	4	0,82	1,00	0,69	0,81	0,86	0,77	0,83	0,82
	5	1,00	0,67	1,00	0,94	1,00	0,80	1,00	0,92
	6	1,00	1,00	1,00	1,00	1,00	0,94	1,00	0,98

Level 6 (99-100%) is within the boundaries all the way except for one point, day 2 piles 20+80. Checking the numeral values I see that there is only the one miss, and the reason this has such a visible effect is due to the relative small sample (17 questions) at this point. This reduces the severity of this point being just below the boundaries (98.2 %) down to nil. The data also confirms this, as there only was the one miss among 87 q the first day and 57 the second day when the miss occurred. This level has stayed stable at the accepted level. Level 5 (90-98%) has a very low sample quanta, total 13 the first day and 18 the second, leaving us with no reliable points the hole of day 2. However there is only on miss on day one and one miss on day two, which is in line with the instructions (definition) for this level, albeit few misses but bearing a small

sample.

On level 4 (76-90%) it starts off within the limits, then goes above (noting that this point only has 10 q) and then below again. This could be random or feedback adjustment. Totally for day 1 it was right at the middle of the level. On day two all the points are within level boundaries.

The three top levels, 6, 5 and, 4, start out good and end good. The participant has managed to keep the assessments relatively stable throughout the experiment, indicating that she has good control over these levels of confidence.

The rest of the levels show in contrast more desultory observations. Level 3 (61-75 %) starts out a bit under confident, and then hits the level dead on before sinking fairly low underneath the limit. It should be noted that the last point only has 13 q (the two preceding points had 16 each), so by looking at the number of misses (3, 5 and 7) they too are increasing steadily throughout the day. The total for the day however gives the feedback of a well adjusted confidence level. This might explain the why it went so badly the next day. It starts out with a hit; this is also the point with the largest sample data (24 questions). The next two points are well over the limits for this level; however the samples are small, 15 and 13. The number of misses on day 2 is 8, 0 and, 2, also point at a too stern adjustment from day one. This could also be random, but as the total for day two has more sample questions than day one (52 to 45) it's not that likely; especially since we can see the same tendencies on the rest of the lower levels.

Level 2's, the fifty-fifty category, first point is 49%, it then increases a bit, but is still within the boundaries. The last point of the day drops below the lower limit – but again summing up to be within the limits of the level. Day two also gives two stable first points and then the last over the top point, summing up the day to be just outside the limits (61%).

At level 1 day 2 is a stably under confident, already starting on the last point of day one.

The “no idea” level, level 0, is ok the first day, but then gets a too high percentile on the first point of day two. This level has the clearest sign of adjustment towards the level of the levels 0 through 3. The number of questions at this level also doubles from day one to day two; the two last points having the largest sample data. As this is the level that is supposed to mirror true randomness, it does look like this is what is happening when the sample size grows. This then, indicates that the participant is on the right path at this level.

As this is myself I am free to self interpret my own behavior, it seems I may have experienced a regression towards the mean on day two. This is one of two theories I have for why it went like this. The other is related to the stress caused by actually being the only participant in my own experiment to hit

within all of the levels after day one. As I started the experiment after all of the participants had finished day one, and some of them even day two, I had looked at and sorted their data as well as being the one who gave them the oral feedback interpretations of their results, as well as being deep into the research problem and questions. As I undoubtedly sharpened my senses when embarking with the question piles, both feeling excited to find out how I would do as well as feeling a bit of pressure to do well. When looking at the summary of day one, I was surprised over the results. So when starting day two I told myself to relax as I seemed to have a good control over my uncertainty levels. This could have triggered the regression back to my mean if the results from day one actually were an extraordinary achievement for me; that I actually am under confident at the lower levels. And therefore I did not get a true feedback of my abilities and had nothing to guide me towards learning a better assessment of my uncertainty. The other possible effect would trigger the casually known effect of: when you focus too much on something you are doing, you mess it up. Like trying to think about how you walk when you're walking (right foot left foot right foot left foot) you start walking more slowly and more jagged than you otherwise would. As well as the lack of a goal to work towards (to under confident or to overconfident), just trying to stay the same I believe is much more difficult to grasp when it is the first time you're doing something. This effect is also seen in some of the other participants.

Participant III

TABLE A-3

Piles n	60' &	10' &	100' &	Day 1, total	60' &	10' &	100' &	Day 2, total	
	80'	40'	20'		80'	40'	20'		
QLev _n	0	31	5	15	51	6	6	21	33
	1	16	31	29	76	24	21	10	55
	2	10	19	17	46	23	22	18	63
	3	8	14	4	26	9	13	9	31
	4	3	5	6	14	10	14	5	29
	5	2	4	3	9	2	5	4	11
	6	5	15	8	28	6	13	21	40
HitLev _n	0	96	25	53	174	37	28	58	123
	1	34	65	62	161	51	44	26	121
	2	11	30	22	63	40	39	36	115
	3	9	16	5	30	12	16	10	38
	4	3	5	7	15	13	16	5	34
	5	2	4	3	9	2	5	5	12
	6	5	16	8	29	6	13	21	40
HitRateLev _n (HitLev _n / QLev _n)	0	0,32	0,20	0,28	0,29	0,16	0,21	0,36	0,27
	1	0,47	0,48	0,47	0,47	0,47	0,48	0,38	0,45
	2	0,91	0,63	0,77	0,73	0,58	0,56	0,50	0,55
	3	0,89	0,88	0,80	0,87	0,75	0,81	0,90	0,82
	4	1,00	1,00	0,86	0,93	0,77	0,88	1,00	0,85
	5	1,00	1,00	1,00	1,00	1,00	1,00	0,80	0,92
	6	1,00	0,94	1,00	0,97	1,00	1,00	1,00	1,00

On the top level participant III had one miss on day one, and non on day 2. She also increased the number of questions on this level on day 2. Level five (90-98%), however, has pore representation, but has a slight increase in sample size (and improvement) that all in all gets her within the boundaries on day 2. On level 4 there is a doubling of questions on day to, from 15 to 34. Although the two first points on day 1 have only 3 and 5 questions; the total for the whole day is 15 questions with one miss giving us a hit rate of 93%. It is possible that more misses would occur with more questions committed to this level, adjusting the hit rate within the limits; this is in fact what we se on day 2. The two first point of the day have about 15 samples and are nicely within the boundaries (perhaps a bit close to the boarders on the low side and the other on the high side) giving us a point of the day that is very close to the middle of the level. One can still wonder if this adjustment is caused by the doubling of the sample size, or if she is adjusting nicely to feedback. Looking at the three top levels all in one gives the impression that she in fact is adjusting,

perhaps helped by the sample size increasing on day two on all top levels. As she displayed a bit under confident behavior on these levels on day one (both orally and in the statistically), we can assume the adjustment is caused by her trying to be more confident on the questions she places in these levels. We also observe a general under confidence of all levels on day one.

Level 3 is the only level that does not seem to want to get better, or said in another way: she does not seem to be able to grasp how to adjust this particular level.

Level 2, the fifty-fifty level, there is a very nice adjustment on day two. The sample size dramatically increasing (63 to 115) can be a contribution to this. However it seems that there has been a good “sleep on it” effect on this level. Level 0 and 1 has by far the largest sample, but this does not seem to affect the hit rate significantly, leaving me wanting to use this as support of actual adjustment on the top three levels and level 2, rather than contributing the approximation towards the ideal to larger sample size.

Level 1 only shows a sign of any adjustment at all on the final point of day 2 after being freakishly stable up to this point. This point is has the lowest sample size, and it is not unthinkable that it would not have broken the line with more questions.

Level 0 fluctuates from over to under to just on the boarder on day one – making the point of the day within the boundaries. On day two they are under then just on the lower limit and then well over, also giving the total of the day to a point within the boundaries. It looks like she has difficulties with applying more confidence on these levels, which can be understandable as their purpose is to house the “no idea” and “some idea” questions; especially as she displays a very overall underconfident behavior.

It is difficult to see by the day 1 to day 2 adjustment (figure 5) if the participant has a parallel shift adjustment to feedback or a better approximation to the ideal, since her day one graph is very parallel to the ideal graph. But as the day 2 graph isn't as smooth as the day one, I believe this to point in the direction of conscious adjustment towards the ideal. Keeping in mind that the “job” on day two was to become more confident on all the levels, and this of course demands a parallel shift in itself. As well as the adjustments seen in the gliding graph level by level, I believe the participant was on her way to become in control of her uncertainty levels. Being so bold as to state that generally underconfident people need more time and resources to adjust than more confident people, she just needed a bit more time and encouragement to correct the levels outside the limits on day 2.

Participant IV

TABLE A-4

Piles n	Day 1,			Day 1, total	Day 2,			Day 2, total	
	40' & 10'	80' & 20'	100' & 60'		40' & 10'	80' & 20'	100' & 60'		
QLev _n	0	2	8	10	20	1	3	0	4
	1	6	6	6	18	7	10	15	32
	2	17	13	10	40	22	18	24	64
	3	16	25	19	60	20	18	20	58
	4	17	16	23	56	22	15	8	45
	5	21	17	19	57	27	19	8	54
	6	37	15	6	58	13	12	3	28
HitLev _n	0	6	23	27	56	2	5	2	9
	1	14	14	19	47	25	32	40	97
	2	34	30	26	90	43	38	64	145
	3	28	35	26	89	23	36	32	91
	4	18	19	32	69	25	19	11	55
	5	23	23	23	69	29	19	9	57
	6	37	16	7	60	13	12	3	28
HitRateLev _n (HitLev _n / QLev _n)	0	0,33	0,35	0,37	0,36	0,50	0,60	0,00	0,44
	1	0,43	0,43	0,32	0,38	0,28	0,31	0,38	0,33
	2	0,50	0,43	0,38	0,44	0,51	0,47	0,38	0,44
	3	0,57	0,71	0,73	0,67	0,87	0,50	0,63	0,64
	4	0,94	0,84	0,72	0,81	0,88	0,79	0,73	0,82
	5	0,91	0,74	0,83	0,83	0,93	1,00	0,89	0,95
	6	1,00	0,94	0,86	0,97	1,00	1,00	1,00	1,00

On Level 6 the two last points on day one are below the limit, the number of misses also support a “too confident” conclusion. There are no misses at all on day two and halved sample size. This indicates she has taken into account which questions she actually is “really really” sure on, and perhaps moved the others down a level. (If I don’t remember completely wrong, it’s what I advised her to do).

Level 5 is also displays over confidence on the first day, although the first point is within the limits it’s just at the boarder. The second point is outside, and the last point of the day displays a movement back towards the level limit. The second day we start within the boundaries, but perhaps spooked by being overconfident on day one the second point is all most underconfident, but then the last point moves just under the limit (this points has 1 miss of 9 q and therefore accepted as being inside the limit).

Level 4 starts above the boundaries, then moves within an accepted rate, but continues to move below the level on the last point. The second day the

first point is within, but close to the upper limit, and the second also on the inside, but close to the lower limit. The third point on day two is outside, but it contains only 11 questions and has three misses and must therefore be considered inside.

The top three levels experience a drop in sample size; perhaps she is moving the questions more towards the level they belong in. Showing a better evaluation of where the uncertainty belongs. I think this is the strongest indication of adjustment from Marina on the top levels.

Level 3 is also a bit overconfident to start with, the two last points are within the level acceptance rates. Day two starts with a very high under confidence, and then has an equally extreme overconfidence on the second point, before just sneaking the last point within the lower limits of the level. Even if this level was ok assessed on day one, she did manage to adjust towards the end of day two. This can perhaps be explained by the questions from the above three levels being pushed down to this level. As this level has a constant sample, questions from this level are perhaps also being pushed further down.

Level 2 is within the boundaries on day one except for the last point; the same pattern is seen on day two. But even though they are outside they don't deviate too much, and as these occurrences appear on the end of both days they can be the result of tiredness as well as anything else.

Level 1 starts with a slight under confidence, and then gets adjusted to an accepted rate on the last point. On day 2 all the points are within the levels limits.

Level 0 experiences a dramatic drop in sample size, none of the points of day two can be used and the sum of the day (9 q) can't be trusted either (since this is supposed to be the random category this number is too small). It seems that the participant has taken into account the under confidence seen on day one in this level, and moved most of the questions a level up; a strategy that gives a positive response on the points for level 1.

Participant IV, it seems, has managed to keep her cool when finding out she already is relatively well adjusted, and even improved the levels that were outside the limits on day one. She is also showing insight in her own abilities by distributing the questions better along the uncertainty scale on the second day, moving them towards level 2.

Participant V

TABLE A-5

Piles n	40' &	20' &	10' &	Day 1,	40' &	20' &	10' &	Day 2,	
	100'	80'	60'	total	100'	80'	60'	total	
QLev _n	0	16	14	3	33	13	15	5	33
	1	18	9	8	35	11	10	5	26
	2	11	9	8	28	19	8	6	33
	3	14	10	5	29	10	19	14	43
	4	9	29	19	57	15	15	13	43
	5	9	25	26	60	19	19	37	75
	6	27	30	57	114	27	42	49	118
HitLev _n	0	39	23	12	74	26	29	13	68
	1	33	16	17	66	22	15	11	48
	2	15	19	17	51	35	16	13	64
	3	20	15	10	45	14	23	18	55
	4	16	31	22	69	17	16	19	52
	5	10	26	27	63	19	19	37	75
	6	27	30	57	114	27	42	49	118
HitRateLev _n (HitLev _n / QLev _n)	0	0,41	0,61	0,25	0,45	0,50	0,52	0,38	0,49
	1	0,55	0,56	0,47	0,53	0,50	0,67	0,45	0,54
	2	0,73	0,47	0,47	0,55	0,54	0,50	0,46	0,52
	3	0,70	0,67	0,50	0,64	0,71	0,83	0,78	0,78
	4	0,56	0,94	0,86	0,83	0,88	0,94	0,68	0,83
	5	0,90	0,96	0,96	0,95	1,00	1,00	1,00	1,00
	6	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00

A difficult case to analyze, as all his levels fluctuates a great deal and not necessarily towards expected rates. The danger of seeing patterns where there are none are imminent!

The top level is as is shall on both days.

Level 5's (98-90%) first point is just below the limit, but as this is caused by one miss in 10 it's ok. The rest of the points on day 1 are also nicely within the boundaries. The whole of day 2 there are no misses, therefore all the points are outside this level. As this level represents "to be very sure but with a mild reservation" one can't say he didn't adjust, he just incidentally got all the answers correct.

Level 4 starts with being below and then over and then within the limits.

Participant V is seemingly adjusting throughout day one. On day two he starts with a hit then he goes over, and then he goes under – but all in all this gives an accepted point for the day.

On Level 3, the two first points are within accepted ranges, but the last (having

only 10 q and 5 misses) is below. Then, on day two he starts within the limits, before he hits over and then adjusts downwards but not enough to get inside. Level 2 is the only one he manages to adjust into after getting the first point outside, and then keep it stably inside for the rest of the day and the whole of day two.

On level 1 the points actually start fluctuating more towards the end of the experiment; but all the points are stably under confident. The same goes for level 0.

Perhaps since the total feedback from day one said that he was quite adequate, he suddenly lost his foothold on day two and started to mess it up. But as he was within the same levels on day two (except level 3) one can't really say that either. It would be more correct to say that he did not proceed to adjust himself as the message he got was that he didn't need to.

One could also say that he is over-adjusting, and he's not managing to stabilize his effort.

Or is he not trying at all, and all the points are displaying random behavior more than a conscious adjustment strategy? But as it is not in the nature of randomness to fluctuate in this way (this looks more like a "man-made" randomness), one can't say he isn't trying to adjust, he's just not managing it very well. Perhaps it was easiest for him to understand "with out a doubt sure" and "fifty-fifty" than the other levels. But a final blow towards a conclusion of non-learning/adjustment is that all his points for the day (except level 4) are almost at exactly the same point on day 2 as day 1. Because of the large fluctuation of the points, I am inclined to trust the day to day graph as it shows no movement of the graph. In addition, the points don't show me a general willingness to fluctuate towards the level they should be in, but rather fortuitous behavior.

Participant VI

TABLE A-6

Piles n	40' &	60' &	80' &	Day 1, total	40' &	60' &	80' &	Day 2, total	
	100'	10'	20'		100'	10'	20'		
QLev _n	0	14	10	7	31	12	5	11	28
	1	16	13	11	40	12	11	11	34
	2	15	9	12	36	14	12	12	38
	3	7	9	12	28	1	12	13	26
	4	8	8	9	25	7	10	13	30
	5	16	17	9	42	11	15	6	32
	6	18	40	30	88	24	42	36	102
HitLev _n	0	35	20	28	83	54	27	32	113
	1	34	26	33	93	30	25	26	81
	2	33	23	26	82	25	24	29	78
	3	9	15	17	41	5	16	15	36
	4	14	16	15	45	9	11	15	35
	5	18	20	11	49	13	15	7	35
	6	19	41	30	90	24	42	36	102
HitRateLev _n (HitLev _n / QLev _n)	0	0,40	0,50	0,25	0,37	0,22	0,19	0,34	0,25
	1	0,47	0,50	0,33	0,43	0,40	0,44	0,42	0,42
	2	0,45	0,39	0,46	0,44	0,56	0,50	0,41	0,49
	3	0,78	0,60	0,71	0,68	0,20	0,75	0,87	0,72
	4	0,57	0,50	0,60	0,56	0,78	0,91	0,87	0,86
	5	0,89	0,85	0,82	0,86	0,85	1,00	0,86	0,91
	6	0,95	0,98	1,00	0,98	1,00	1,00	1,00	1,00

The first day participant VI was under confident on the lower levels, and overconfident on the top levels; levels two and three was relatively adequate. The degree of over and under confidents is however not very high except the fourth level. The sample size on each of the levels stays relatively the same on day 2 as it was on day one. Only the top and bottom levels have an increase in sample size, the rest has a reduction but no more than 15 questions on any one level.

Level six starts with the first point having one miss in 19 questions, the second point also has only one miss but the sample size here is 41. The last point has no misses. So one can say there has been an adjustment throughout the day, this also continues the whole of day 2 with no misses at all.

Level five does not quite visualize the adjustment that happens throughout the experiment as well the numeric values. On day one it was 2 miss in 18, 3 miss in 20 and 2 miss in 11. Day 2 has 2 misses in 13, 0 misses in 15 and 1 miss in 7. So although the number of misses goes down on day two, and moves her

graph closer to the ideal, the sample size also goes down. I therefore can not say definitely that there has been a positive adjustment towards the ideal, or no change on this level.

Level 4 is too overconfident on day 1. On day two there is a change over night and the points are very much closer to the acceptance rates of the level. The first point has 2 misses in 7, then 2 in 11 and 2 in 15, which also points in the direction of positive adjustment. Although the sample size has dropped, at this level there is more room for error than on level 5 (where the drop in sample size made it difficult to vote in favor of a better adjustment).

Level 3 starts with a point just outside the upper rim, but has few questions. She then gets a point just on the lower limit, and the last point of the day is inside the boundaries. The two last points are a bit misleading visually, as the second point has 4 misses of 16 and the third has 2 misses of 15 indicating an improvement towards the ideal rate. The point of day two is therefore more accurate as it has more data in it; this point is within the limits.

Level 2 fluctuates a bit throughout the experiment, but only the second point on day one is outside (and only just with a 39% hit rate). There can be seen a positive adjustment towards the middle value of the level towards the next point for almost every point.

Level 1 starts with under confidence on the two first points, the last point smack at the middle value. The second day all the points are stably just outside the boundary.

Level 0 has a similar day one as level 1. On day two, the two first points are at the lower border, first just on the inside and then just on the outside. Then there is an over compensation putting the last point over the upper limit, but not as high as the highest point on day one.

Looking at the adjustment from day one to day two in the day to day graph, the problem levels on day one has had an adjustment towards the ideal on day two. Not all the levels can conclusively be interpreted as adjustment towards the ideal, but these are low in sample size. There were no real problem levels, needing extra attention except level 4. The feedback from day 1 to the participant was therefore that she could consider herself rather adequate on an overall basis, as well as the tendency to be overconfident on the top levels and under confident on the lower. In summary she adjusted very nicely to the feedback given about her abilities.

Participant VII

TABLE A-7

Piles n	20' &	100' &	10' &	Day 1, total	20' &	100' &	10' &	Day 2, total
	80'	60'	40'		80'	60'	40'	
0	12	9	3	24	3	2	1	6
1	20	16	29	65	15	12	12	39
2	9	12	12	33	19	18	23	60
QLev _n	3	5	12	27	10	13	13	36
4	13	13	10	36	14	10	12	36
5	11	12	17	40	20	18	12	50
6	12	23	20	55	19	20	18	57
0	33	30	18	81	8	9	7	24
1	55	39	45	139	34	43	45	122
2	22	22	26	70	47	38	46	131
HitLev _n	3	8	18	43	17	19	19	55
4	17	18	16	51	14	11	12	37
5	13	12	18	43	20	21	13	54
6	12	23	20	55	20	20	18	58
0	0,36	0,30	0,17	0,30	0,38	0,22	0,14	0,25
1	0,36	0,41	0,64	0,47	0,44	0,28	0,27	0,32
HitRateLev _n	2	0,41	0,55	0,46	0,47	0,47	0,50	0,46
(HitLev _n / QLev _n)	3	0,63	0,67	0,59	0,63	0,59	0,68	0,65
4	0,76	0,72	0,63	0,71	1,00	0,91	1,00	0,97
5	0,85	1,00	0,94	0,93	1,00	0,86	0,92	0,93
6	1,00	1,00	1,00	1,00	0,95	1,00	1,00	0,98

On the top level participant VII has two misses on the second point of day two, after having a perfect strike on day one.

Level 5 starts out with 2 misses in 13 questions, which places the point outside the accepted rate. The next point has no misses, and therefore lands over. The last point is within the boundaries, 1 miss in 18 questions. In total for the day, the point is within the limits. On day two, the first point is outside the upper limit, the next point is outside the lower limit, the last point of day two at an accepted rate; the total for day two is within the limits. The deviations were not so severe in the first place on this level, as indicated by the number of misses compared to sample size. She manages to keep herself on a stable adequate assessment at this level.

Level 4 steadily deteriorates throughout the day; sample size is almost the same for all points. On day two, there seems to be overcompensation for the first day, all of the points are steadily under confident. The sample size decreases with 14 questions on day two.

Level 3 is relatively stable with two points just outside the lower limits; there was no need for heavy adjustment here on day one.

Level 2 experienced a dramatic increase in data – 61 more questions on day two. This does not affect the stability from day one to day two either negatively or positively; if anything it confirms it by giving us a larger sample size.

Level 1 has a drop in sample size from day one to day two, but as it is large on both days this has no real impact on our conclusions for this level. The first point starts inside the level borders, the next one just slightly outside, and then a mad increase in ratio on the last point; giving the total for the day a point outside. The first point on day two is outside the upper rim, but not so much as the last point of day one. The two last points are nicely within the parameters. This, I believe, does show good adjustment ability on the feedback that shows extreme violations.

Level 0 starts outside the upper limit and stops outside the lower limit. The same tendency is seen on day two but here the sample size much lower (54 questions less).

In summary participant VII was quite adequate on day one. There was a tendency to be a bit overconfident on the top levels and a bit underconfident on the two lower levels; leaving levels 2 and 3 with passing ratios (almost) all the way through the experiment. On day two the underconfidence on the lower levels is diminished. Level 4 seems to be the problem level, where the tendency has turned out to be much more underconfident than it was overconfident on day one.

Participant VIII

TABLE A-8

Piles n	40' &	20' &	60' &	Day 1, total	40' &	20' &	60' &	Day 2, total	
	100'	10'	80'		100'	10'	80'		
QLev _n	0	17	4	7	28	6	3	6	15
	1	28	20	32	80	10	18	13	41
	2	11	17	20	48	12	15	18	45
	3	1	12	2	15	18	15	14	47
	4	4	16	3	23	7	14	5	26
	5	0	2	0	2	0	3	0	3
	6	3	21	6	30	11	14	4	29
HitLev _n	0	57	20	27	104	21	13	32	66
	1	71	49	76	196	49	38	50	137
	2	21	34	42	97	41	43	39	123
	3	4	16	4	24	26	29	30	85
	4	4	18	4	26	11	19	6	36
	5	0	2	0	2	0	3	0	3
	6	3	21	6	30	11	15	4	30
HitRateLev _n (HitLev _n / QLev _n)	0	0,30	0,20	0,26	0,27	0,29	0,23	0,19	0,23
	1	0,39	0,41	0,42	0,41	0,20	0,47	0,26	0,30
	2	0,52	0,50	0,48	0,49	0,29	0,35	0,46	0,37
	3	0,25	0,75	0,50	0,63	0,69	0,52	0,47	0,55
	4	1,00	0,89	0,75	0,88	0,64	0,74	0,83	0,72
	5	NaN	1,00	NaN	1,00	NaN	1,00	NaN	1,00
	6	1,00	1,00	1,00	1,00	1,00	0,93	1,00	0,97

The sample size for level 4 increases with 10 questions on day two. In total, the point for day one on level 4 is within the boundaries but close to the upper limit. On day 2 two of the samples have few questions in them but the total for the day gives us a point below the lower limit.

The sample size for level 3 also has an increase but here it's a more dramatic enlargement going from 24 to 85 questions. There is only one point on day one we can read with any hope of accuracy, and it's just on the upper border. On day two, the points increasingly become more and more overconfident, giving the days point a rate below the accepted.

Level 2 has a very nice and stable line of points on day one. On day two, there is a tripling of sample size, from 42 to 123 questions; presumably taking these from level 1 and 0. The day starts out with a point below the limit, take note of that the sample size in this one point is as large as the whole of day one. The second points moves slightly towards the accepted ratios, and the last point of the day is within the level boundaries. The total point of the day is therefore

somewhat misleading.

On level 1 she is stably just outside the upper limit the whole day. On day two it starts below, then moves above, and then downwards again (this time just inside). It looks like she's trying to adjust her under confidence from the first day, and eventually manages to assess better at the end of the day.

Level 0 is nice and stable in the proximity of the ideal for the entire experiment.

The participant for some reason becomes overall more overconfident on day two, i.e. a parallel shift on day two compared to day one. Perhaps her confidence was low before embarking on the experiment, when she got her feedback of not being as bad as she might have thought; her confidence perhaps got a boost after day one. Or perhaps we see a regression back to her mean on day two, as she intensified her effort on day one knowing she usually didn't do so well in trivia games. There is a decrease of number of correct answers from 47% on day one to 43% on day two that may support this.

The four top levels has a small portion of the questions on day one, except for levels 6 and 5 (that stay the same) there is a tendency to push questions upwards on day two. This is perhaps due to the increase in confidence. This has been seen in other candidates when this strategy is applied, it shakes the ground for the two first points before getting the last point of the day in closer proximity than the first point.

Participant IX

TABLE A-9

Piles n	80' &	100' &	60' &	Day 1, total	80' &	100' &	60' &	Day 2, total	
	20'	40'	10'		20'	40'	10'		
QLev _n	0	9	14	14	37	4	5	4	13
	1	17	23	9	49	18	14	6	38
	2	16	15	11	42	19	15	9	43
	3	9	7	11	27	15	13	11	39
	4	8	4	12	24	8	10	16	34
	5	9	1	11	21	11	11	20	42
	6	28	15	23	66	18	16	36	70
HitLev _n	0	24	51	38	113	12	18	9	39
	1	52	43	24	119	38	39	21	98
	2	26	28	26	80	39	40	28	107
	3	10	14	21	45	28	21	24	73
	4	11	6	17	34	12	14	19	45
	5	9	1	11	21	13	12	22	47
	6	28	17	23	68	18	16	37	71
HitRateLev _n (HitLev _n / QLev _n)	0	0,38	0,27	0,37	0,33	0,33	0,28	0,44	0,33
	1	0,33	0,53	0,38	0,41	0,47	0,36	0,29	0,39
	2	0,62	0,54	0,42	0,53	0,49	0,38	0,32	0,40
	3	0,90	0,50	0,52	0,60	0,54	0,62	0,46	0,53
	4	0,73	0,67	0,71	0,71	0,67	0,71	0,84	0,76
	5	1,00	1,00	1,00	1,00	0,85	0,92	0,91	0,89
	6	1,00	0,88	1,00	0,97	1,00	1,00	0,97	0,99

A 2 miss in 17 questions on the top level has a very dramatic visual effect on the first day. On day two, the only point with a deviation has only one miss in 37 questions; we can definitely say that there has been a improvement from day to day at this level.

Level 5 has a small sample size on day one; it's doubled on day two. We can see a slight over night adjustment at this level.

Level 4, note low sample size on the two first points on day one and the first point on day two, is bobbing just below the accepted rate before getting a lift over the limit on the last point of day two. This last point is also the point with the largest sample size at this level.

Level 3 is at a very under confident point staring off, before zooming below the accepted hit rate limit. He stays there, before hitting an all time low at the last point. Day two has an increase from 45 to 73 in sample size. The overconfidence persists.

Level 2 stars just outside level limits, before moving steadily downwards, but

the second and third point of day one is within the boundaries of the level. The second day starts with a point right at the middle value and then moving downwards and outside for the two next points.

Level 1 is more fortunate with the movement of the points. It starts with an accepted point, and then moving high above, and then adjusts down to the accepted ratio again. Day 2 starts outside the upper limit; the two last points are inside although moving downwards.

Level 0 is at a stably under confident level all the way.

Except for level 5 there is an overconfidence tendency on the top levels from level 3 inclusive. There

There seems to be an overall increase in confidence, alas not too favorable in this participant's case. The feedback should have suggested an overall less confidence strategy would have been a better road to take. Perhaps he is a under confident adverse person, and when told he should not be so cocky he responds with becoming even more so.

Participant X

TABLE A-10

Piles n	Day 1,			Day 1, total	Day 2,			Day 2, total	
	60' & 20'	100' & 40'	80' & 10'		60' & 20'	100' & 40'	80' & 10'		
QLev_n	0	17	10	16	43	9	28	24	61
	1	16	17	16	49	14	14	18	46
	2	18	7	15	40	10	15	10	35
	3	9	6	10	25	11	8	8	27
	4	11	7	4	22	11	3	7	21
	5	14	5	15	34	13	5	8	26
	6	8	14	19	41	17	12	25	54
HitLev_n	0	44	55	46	145	35	76	62	173
	1	37	53	38	128	35	27	28	90
	2	32	17	25	74	32	20	18	70
	3	11	7	12	30	14	12	9	35
	4	14	9	5	28	12	7	9	28
	5	14	5	15	34	15	6	8	29
	6	8	14	19	41	17	12	26	55
HitRateLev_n (HitLev_n/ QLev_n)	0	0,39	0,18	0,35	0,30	0,26	0,37	0,39	0,35
	1	0,43	0,32	0,42	0,38	0,40	0,52	0,64	0,51
	2	0,56	0,41	0,60	0,54	0,31	0,75	0,56	0,50
	3	0,82	0,86	0,83	0,83	0,79	0,67	0,89	0,77
	4	0,79	0,78	0,80	0,79	0,92	0,43	0,78	0,75
	5	1,00	1,00	1,00	1,00	0,87	0,83	1,00	0,90
	6	1,00	1,00	1,00	1,00	1,00	1,00	0,96	0,98

There does not seem to be much change in participant X, except a slight parallel shift in a more confident direction on day two.

Starting at the top level, it's at an accepted ratio, the one miss on the last point on day two is acceptable.

On level 5, there is sign of adjustment willingness in the sense that there occurs misses on day two. The sample size is relative small on each point, and the 2 misses on the first point and the one miss on the second, has an exaggerated effect on the hit rate. Therefore, I can conclude that there is a positive adjustment that he dares to get more misses on this level – using it as instructed.

Level 4 has very uncertain points, as almost all of them are small in sample size. Looking at the numeric values, one is inclined to accept the performance on both days as acceptable. It is impossible to decide if there has been a continuous adjustment or no change – as the two days only differ with one miss more on the second day, both having 28 questions.

At level 3 all the points on the first day are stably well over. The second day, there is a slight decline at the first point; the second is within the limits before the last point is at the same hit rate as the first day. Again, there is the problem of the small sample size. There is a slight movement towards the ideal on day two, when considering the number of misses compared to sample size.

On level 2 sample sizes can no longer be an excuse, being about 70 on both days. The first point is close to the upper boundary of the level, then a point close to the lower boundary, and then up onto the upper border again. On day two the first point is below, the second is high above the level, and the last point gets within the boundaries.

On level 1 the points are out, in, and out; the point on the inside being the one with the largest sample. But then on day two, the points take off on an escalation path away from the accepted rates. No adjustment here.

On level 0 the same pattern is seen as on day 1, starting too high, then below, and then too high again. Day two starts ok, but then the two last points are equally too high.

Participant XI

TABLE A-II

Piles n	20' &	80' &	100' &	Day 1, total	20' &	80' &	100' &	Day 2, total	
	40'	60'	10'		40'	60'	10'		
QLev _n	0	18	12	24	54	19	11	15	45
	1	8	11	7	26	3	7	2	12
	2	11	18	16	45	24	30	15	69
	3	8	5	11	24	6	14	15	35
	4	10	5	12	27	9	13	19	41
	5	3	0	0	3	0	0	2	2
	6	52	38	37	127	52	39	50	141
HitLev _n	0	44	54	54	152	43	41	37	121
	1	24	23	20	67	5	8	5	18
	2	16	33	26	75	43	40	27	110
	3	11	7	11	29	8	16	18	42
	4	11	6	12	29	9	16	21	46
	5	3	0	0	3	0	0	2	2
	6	52	38	37	127	52	39	50	141
HitRateLev _n (HitLev _n / QLev _n)	0	0,41	0,22	0,44	0,36	0,44	0,27	0,41	0,37
	1	0,33	0,48	0,35	0,39	0,60	0,88	0,40	0,67
	2	0,69	0,55	0,62	0,60	0,56	0,75	0,56	0,63
	3	0,73	0,71	1,00	0,83	0,75	0,88	0,83	0,83
	4	0,91	0,83	1,00	0,93	1,00	0,81	0,90	0,89
	5	1,00	!	!	1,00	!	!	1,00	1,00
	6	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00

Level 6 is impressive; it has a very high sample size compared to the other participants.

Level 5 can't be assessed due to lack of samples, in total 3 on day one and 2 on day 2.

Levels 4 and 3 have a slight increase in sample size, after encouragement from me to use them more.

Level 4 does end with the two last points being at an accepted level, but there is not much deviation from the ideal from the start.

The two first points on level 3 are within accepted rates; the last point has no misses and is high above the limit. Keeping in mind the small sample size on all points, the net total of the day is however outside the upper border. On day two, the points bob around the same ratio as day one ended on, indicating no adjustment at this level.

Level 2 starts high, and then moves within the accepted parameters. The last point of the day is just outside. The first point of day one is at an ok level, the

second is very high, and the last point is back to the same rate as the start of the day. All this does indicate a general overconfidence that is not dealt with on day two. The sample size is also increased with 35 questions, supporting this view.

Level I has a dramatic cut in sample size from 67 to 18, making all of the points on day two unreliable. There are few other participants that had such a drop in sample size at this level. The first day two of the points are within the limits and one point is above.

Level 0 fluctuates over a rate of about 35%, a bit to high.

Participant XI did seem to have a very negative attitude (both generally and towards the experiment), and gave the impression she was not big on change she didn't support or understand, i.e. she implied that stuff she didn't understand or disagreed with ("why should one want to investigate this") wasn't worth understanding. She made it clear that her opinion was that either you knew something or you didn't, leaving a level division done in the experiment uninteresting. This is reflected in the sample sizes of the levels with "without a doubt", "fifty-fifty" and "no idea" getting in total 68% of the questions on day one and 81% of the questions on day two. These are the levels which best map to her view of the world.

As well as being big on attitude, she was a very confident individual, exhibiting that she knew what she knew. When presented with the notion of the possibility of knowing more than you know you know, or knowing less than you think you know, when I tried to get her to understand the experiment setting better, this was dismissed as crap. She is of course entitled to her own opinion on this! But it does seem that her attitude has colored her willingness to adjust. As other participants seem able to adjust on the more "woolly" levels, she stands alone here. Another theory is that she is unwilling to let go of any of her confidence, this is also seen in other participants unable to adjust.

Participant XII

TABLE A-12

Piles n		20' &	60' &	40' &	Day 1,	20' &	60' &	40' &	Day 2,
		10'	100'	100'	total	10'	100'	100'	total
QLev _n	0	7	18	15	40	3	9	11	23
	1	3	21	24	48	14	10	12	36
	2	11	5	14	30	15	17	29	61
	3	16	12	12	40	15	14	18	47
	4	14	2	4	20	14	12	7	33
	5	16	1	0	17	17	9	2	28
	6	28	16	20	64	36	12	16	64
HitLev _n	0	20	63	42	125	14	41	40	95
	1	11	55	52	118	26	27	31	84
	2	34	8	23	65	30	37	42	109
	3	28	14	17	59	20	21	19	60
	4	19	2	5	26	17	13	10	40
	5	20	2	0	22	17	9	2	28
	6	28	16	21	65	36	12	16	64
HitRateLev _n (HitLev _n / QLev _n)	0	0,35	0,29	0,36	0,32	0,21	0,22	0,28	0,24
	1	0,27	0,38	0,46	0,41	0,54	0,37	0,39	0,43
	2	0,32	0,63	0,61	0,46	0,50	0,46	0,69	0,56
	3	0,57	0,86	0,71	0,68	0,75	0,67	0,95	0,78
	4	0,74	1,00	0,80	0,77	0,82	0,92	0,70	0,83
	5	0,80	0,50	!	0,77	1,00	1,00	1,00	1,00
	6	1,00	1,00	0,95	0,98	1,00	1,00	1,00	1,00

The top level has one miss in 20 on the last point of day one. The points at day 2 are all ok. The sample size is similar on both days (65 and 64).

Level 5 starts with an overconfidence point, but has few questions at the last two points. Perhaps is worth noting that the sample size of the first point is 20, and the last two are 2 and 0. On day 2 has same tendency with 17, 9, and 2 in sample size respectably. Perhaps spooked by the many misses on day one, she uses this level a bit more conservatively than it's supposed to be used. One can theorize that she doesn't quite grasp the difference between this level and the top one, and stops using it instead of trying to get control over it. But there is a willingness to try, as the sample size show, although it seems she gives up on day two as well.

Level 4 starts just outside on day one's first point, and the two consecutive points have to small sample size to be evaluated. The total of the day is inside the level limits. Day 2 starts with a acceptable hit rate. The next point is just outside, having 1 miss in 13. The last point appears below the lower limit, but

as it has 3 misses in 10 I accept it as ok. The total of day two actually moves more towards the ideal than day one.

Level 3 adjusts to an accepted hit rate on the end of day two. The two first points the next experiment day are inside the level boundaries. The last point is however high above, with 1 miss in 19.

Level 2 displays almost the same behavior as level 3, with a good adjustment towards the ideal, and then a last point way over the upper level.

Level 1's points moves from an accepted rate to a under confident last point on day one. Day two has first a under confident point, the last two points within the level limits showing an adjustment on this level.

Level 0 is a bit under confident on day one, but then moves to a nice stable accepted rate on day two.

Participant XII is moving from a general slight under confidence on some levels, towards a slight general better confidence on all levels. As she did not have any major "work to do" after day one, she has stuck with her feel for her uncertainty assessments; except for the last points on levels 3 and 2, but this can be due to tiredness. This is also indicated by the drop in sample size on level 5 and 4 on the last point of day 2 on these levels.

Participant XIII

TABLE A-13

Piles n	10' &	60' &	40' &	Day 1, total	10' &	60' &	40' &	Day 2, total	
	20'	100'	80'		20'	100'	80'		
QLev _n	0	13	18	16	47	11	14	13	38
	1	11	17	11	39	16	10	12	38
	2	20	13	17	50	16	17	22	55
	3	10	11	11	32	12	13	10	35
	4	11	8	3	22	7	6	7	20
	5	7	7	8	22	12	6	6	24
	6	43	17	19	79	43	25	27	95
HitLev _n	0	25	49	53	127	21	38	39	98
	1	21	33	25	79	27	34	33	94
	2	31	30	31	92	30	33	32	95
	3	19	14	18	51	17	16	14	47
	4	12	11	4	27	8	7	8	23
	5	8	8	9	25	13	7	6	26
	6	44	17	20	81	44	25	28	97
HitRateLev _n (HitLev _n / QLev _n)	0	0,52	0,37	0,30	0,37	0,52	0,37	0,33	0,39
	1	0,52	0,52	0,44	0,49	0,59	0,29	0,36	0,40
	2	0,65	0,43	0,55	0,54	0,53	0,52	0,69	0,58
	3	0,53	0,79	0,61	0,63	0,71	0,81	0,71	0,74
	4	0,92	0,73	0,75	0,81	0,88	0,86	0,88	0,87
	5	0,88	0,88	0,89	0,88	0,92	0,86	1,00	0,92
	6	0,98	1,00	0,95	0,98	0,98	1,00	0,96	0,98

Participant XIII does not seem to adjust at the top levels, although this is difficult to determine. The sample size stays relatively stable form day one to day two on both the fifth and sixth level. There are 2 misses at level six on day two, the same as on day one. There is however a slight adjustment on level 5 as the last point has no misses, bringing the total within the boundaries; noting that the goal of this level is not to have all questions correct. If this actually is an adjustment or just randomness that would have become more apparent with a larger sample is hard to say.

The forth level also has a small share of questions, but in total for both of the days the hit rate is within the limits.

Level 3 starts below, then goes over and then adjusts to be immediately inside the boundary. The second day starts ok, and then goes over, before it moves back down the accepted rate. Despite the total of day two is on the boarder, day two has a narrower fluctuation span than day one, leaving me to consider it a positive adjustment.

Level 2 starts outside level limits, before staying nicely within the level limits. All the points have about 30 in sample size, and the last point of day one and the two first on day two are very stable around the level's centre. The last point however moves up to 69%; this can be a result of tiredness or loss of concentration due to boredom or just a random outcome as well as an indication of non learning.

Level 1 has a under confidence the whole of day one, and starts with a high hit rate on day two. It then gets adjusted down to an accepted level on the two last points.

Level 0 has the same tendency on both days to gradually move towards the upper limit of the level from a very high hit rate; but it doesn't quite get there on either of the days. The willingness to adjust is however visible.

The participant did not have any dramatic adjustments that needed attention on day two. The problem levels in her case seem to be the two bottom levels 0 and 1. They both are moving towards accepted levels on the end a day one. But something happens (or doesn't happen) over night, and she seems to start from scratch on day two. The distribution of sample size stays relatively constant on all of the levels, except the lowest level where it goes down with 29 questions; presumably moved mainly up to level 1.

On the overall picture from day to day, she goes from a slight tendency to be overconfident on the top levels and under confident on the lower levels to a general under confidence on day two.

Participant XIV

TABLE A-14

Piles n	40' &	10' &	20' &	Day 1,	40' &	10' &	20' &	Day 2,	
	100'	60'	80'	total	100'	60'	80'	total	
QLev _n	0	6	2	2	9	8	3	5	16
	1	13	7	6	29	11	1	9	21
	2	17	6	10	31	12	8	7	27
	3	9	4	13	23	9	14	9	32
	4	17	14	11	45	14	17	14	45
	5	17	33	25	74	17	17	18	52
	6	40	65	53	156	42	70	62	174
HitLev _n	0	11	5	11	25	16	9	12	37
	1	27	12	14	60	28	5	20	53
	2	27	14	13	50	22	16	15	53
	3	16	8	21	39	16	18	14	48
	4	21	20	21	67	18	23	17	58
	5	18	38	27	85	17	19	19	55
	6	40	65	54	157	42	71	63	176
HitRateLev _n (HitLev _n / QLev _n)	0	0,55	0,40	0,18	0,36	0,50	0,33	0,42	0,43
	1	0,48	0,58	0,43	0,48	0,39	0,20	0,45	0,40
	2	0,63	0,43	0,77	0,62	0,55	0,50	0,47	0,51
	3	0,56	0,50	0,62	0,59	0,56	0,78	0,64	0,67
	4	0,81	0,70	0,52	0,67	0,78	0,74	0,82	0,78
	5	0,94	0,87	0,93	0,87	1,00	0,89	0,95	0,95
	6	1,00	1,00	0,98	0,99	1,00	0,99	0,98	0,99

Participant XIV has a large share of his questions on the top level; it even increases from 157 on day one to 176 on day two. Seen in context with the large sample the one miss on the second point on day two, and the one miss on the third point on day two can be regarded as ok. Due to the large sample the hit rate even gets inside the accepted percentile in spite of the miss.

Level 5 has just a large a drop in sample size as level 6 has an increase. All points are in the vicinity of the level boundaries, the second point on day one and the first point on day two being just outside. The total point of day two ends up at an ok level.

Level 4 has a deteriorating appearance, moving from a center point position all the way down to 52% at the end of the day. On day two, however, he seems to get his act together and the points move very little and have a linear appearance in the vicinity of the level mean.

Level 3 starts by being overconfident on day one, but not very far from the border; the last point of the day gets inside as well. On day two he starts

just below, and then over adjusts before landing in the middle of the level's accepted rates.

Level 2 has an unsteady behavior on the first day, starting outside, moving inside, and the moving even further over then initially. In contrast, on day two there is an impressive steadiness of the points, all of them being inside the boundaries.

Level 1 displays under confidence the entire first day. Although not quite getting on the inside of the limits on day two, the points move closer towards the ideal.

Level 0 has a small share of the questions, a slight increase on day two. Here there seems to be no change, the total of day two is in fact further away than the total point of day one.

Looking at the participant's day to day graph he has narrowed both the over and under confidence on day one and day two, making a very close approximation to the ideal graph on day two.

The problem levels 4 and 2 have almost a magical over night change. As for level 4, I remember talking to him about it; he has a high degree of self-knowledge about what he knows, and the confidence he associates with the uncertainty of his knowledge. He usually is very confident in trivia questions and other fact repeating circumstances, he has a very good memory and remembers vast amounts of detail easily and (seemingly) forever. If one would joke about it, one would say he has autistic tendencies.

So when focusing on the trouble areas, his control over his own knowledge base kicked in giving us the visual effect of almost linear points on day two.

Quoting him freely from my own memory "I know what I know, and the stuff I'm not a 100% sure I know, but more a kind of level 4 confidence, I have to give myself that kind of sturdy confidence anyway. It's a pride thing I guess." And when asked to put away his pride, he adjusted towards, and very close to, the ideal on all the levels except for level 0, and to some degree level 1.

Participant XV

TABLE A-15

Piles n	QLev _n	20' &	10' &	60' &	Day 1,	20' &	10' &	60' &	Day 2,
		40'	80'	100'	total	40'	80'	100'	total
	0	20	19	12	51	7	9	9	25
	1	17	11	18	46	22	21	16	59
	2	5	10	13	28	14	17	12	43
	3	1	2	6	9	6	7	6	19
	4	2	8	5	15	6	7	6	19
	5	4	9	4	17	6	7	8	21
	6	15	15	10	40	14	18	7	39
	0	62	71	46	179	27	32	31	90
	1	54	27	53	134	48	42	40	130
	2	14	21	28	63	39	36	44	119
	3	3	8	10	21	11	13	17	41
	4	7	10	8	25	14	12	11	37
	5	6	9	5	20	7	7	9	23
	6	16	15	10	41	14	18	7	39
	0	0,32	0,27	0,26	0,28	0,26	0,28	0,29	0,28
	1	0,31	0,41	0,34	0,34	0,46	0,50	0,40	0,45
	2	0,36	0,48	0,46	0,44	0,36	0,47	0,27	0,36
	3	0,33	0,25	0,60	0,43	0,55	0,54	0,35	0,46
	4	0,29	0,80	0,63	0,60	0,43	0,58	0,55	0,51
	5	0,67	1,00	0,80	0,85	0,86	1,00	0,89	0,91
	6	0,94	1,00	1,00	0,98	1,00	1,00	1,00	1,00

On the top level participant XV started with one miss, and then all the points are at a 100% hit rate.

Due to the small sample size on level 5, day one is difficult to assess properly. But in total for the day, it's outside the lower limit. On day two there is an improvement, still a small sample, on the total of the day putting the hit rate of the day at an accepted percentile.

On level 4 there is also a small sample to base the analysis on. On day two the points are closer related to each other in hit rate than on day one, but all points are still very overconfident.

On level 3 the problem of sample size is still persistent, the last point of day one is just inside the lower limit. On day two the two first points stay close to the lower limit, but just outside, the last point zooms downwards to a 35% hit rate.

Level 2 is quite ok on day one; the two last points of the day are very close to the ideal. On day two however we start at the same rate as on day one, then

moves towards the centre of the level, but then end up at the lowest hit rate of all at this level.

Level 1's points are close to each other and in the vicinity of the level accepted hit rates, on day two there is kind of a parallel shift upwards – the last point being right at the boarder.

Level 0 is, compared to the other levels, surprisingly linear in appearance all the way through the experiment.

There isn't much adjustment to find in the participant's results. Perhaps there is a slight tendency to stabilize the points, bringing them closer to each other on day two, but without a movement towards the desired percentage; in essence they stay at the same height.

She said she didn't do so well on trivia games, and considered herself to be low on general trivia knowledge. Her low confidence in own abilities in this field is reflected, I believe, in the low share of samples at levels 3-5.

Appendix B

- Figures and data used in analyzing Task Difficulty and Knowledge level in GQE

Table A-16 shows the numerical data for all participants divided on the hard and easy question piles.

TABLE A-16 DATA MATERIAL DIVIDED ON PILE DIFFICULTY,
HIGHLIGHTED % INDICATES THAT THEY ARE AT AN ACCEPTED RATE
(INSIDE LEVEL LIMITS).
Q = NUMBER OF QUESTION,
C.A. = NUMBER OF CORRECT ANSWERS,
% = HIT RATE

Hard Questions, 100' & 80' piles												
Day 1				Day 2				All				
LEVEL	# Q	# C.A.	%	LEVEL	# Q	# C.A.	%	LEVEL	# Q	# C.A.	%	
0	689	219	0,32	0	588	190	0,32	0	1277	409	0,32	
1	656	251	0,38	1	541	213	0,39	1	1197	464	0,39	
2	327	166	0,51	2	496	250	0,50	2	823	416	0,51	
3	178	115	0,65	3	242	163	0,67	3	420	278	0,66	
4	156	107	0,69	4	164	131	0,80	4	320	238	0,74	
5	129	106	0,82	5	119	111	0,93	5	248	217	0,88	
6	267	261	0,98	6	250	247	0,99	6	517	508	0,98	
Total	2402	1225	0,51	Total	2400	1305	0,54	Total	4802	2530	0,53	
Easy Questions, 20' & 10'												
Day 1				Day 2				All				
LEVEL	# Q	# C.A.	%	LEVEL	# Q	# C.A.	%	LEVEL	# Q	# C.A.	%	
0	277	102	0,37	0	241	89	0,37	0	518	191	0,37	
1	382	163	0,43	1	342	160	0,47	1	724	323	0,45	
2	386	214	0,55	2	418	203	0,49	2	804	417	0,52	
3	246	166	0,67	3	278	199	0,72	3	524	365	0,70	
4	243	193	0,79	4	239	194	0,81	4	482	387	0,80	
5	246	225	0,91	5	234	216	0,92	5	480	441	0,92	
6	628	622	0,99	6	649	646	1,00	6	1277	1268	0,99	
Total	2408	1685	0,70	Total	2401	1707	0,71	Total	4809	3392	0,71	

Figures A-1 to A-15 show a participants performance on the easy (10' and 20') and hard (80' and 100') question piles in *learning graphs*.

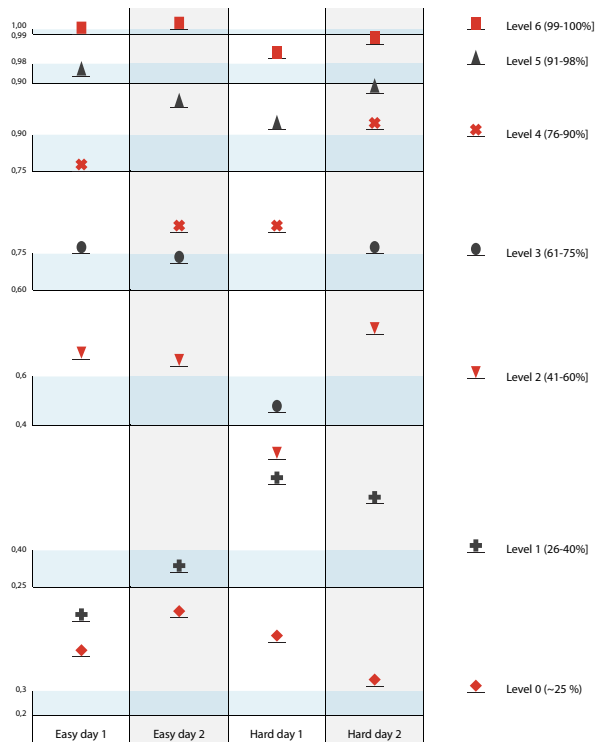


Figure A-1 Performance on globally easy and hard question piles for participant I

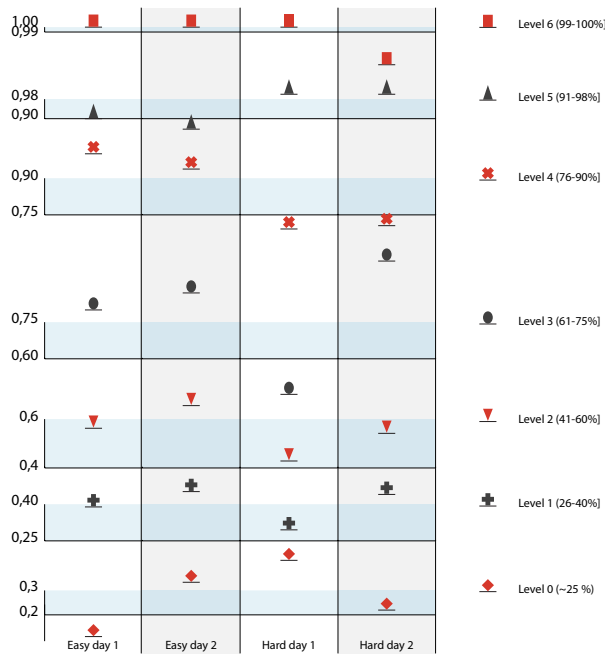


Figure A-2 Performance on globally easy and hard question piles for participant II

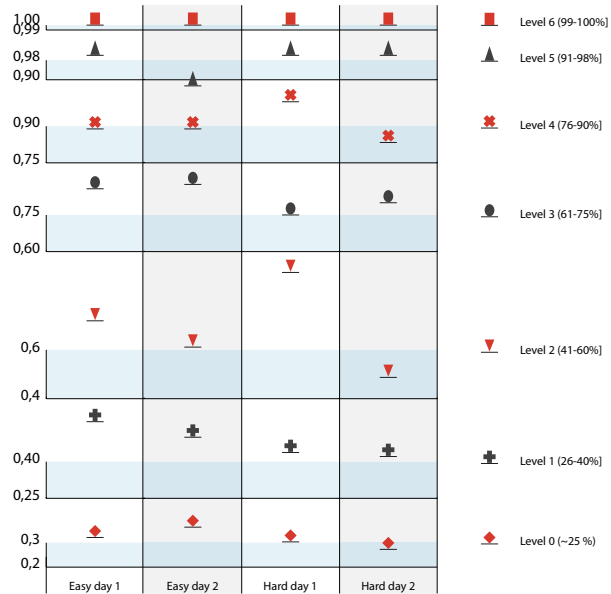


Figure A-3 Performance on globally easy and hard question piles for participant III

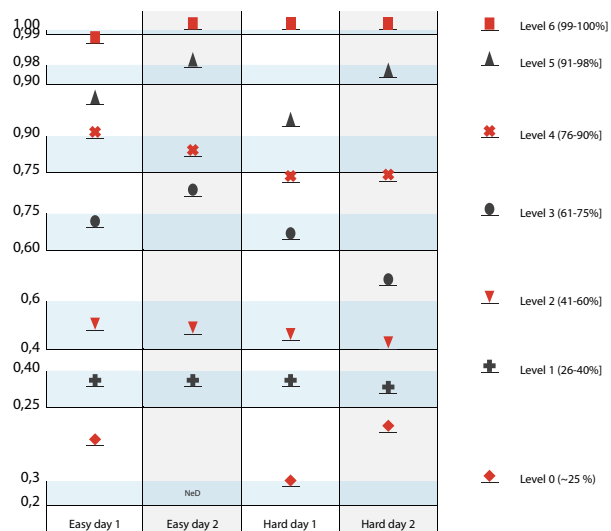


Figure A-4 Performance on globally easy and hard question piles for participant IV

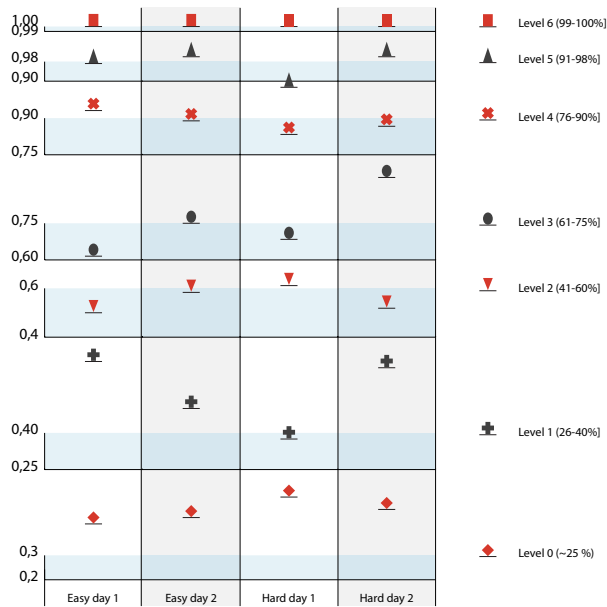


Figure A-5 Performance on globally easy and hard question piles for participant V

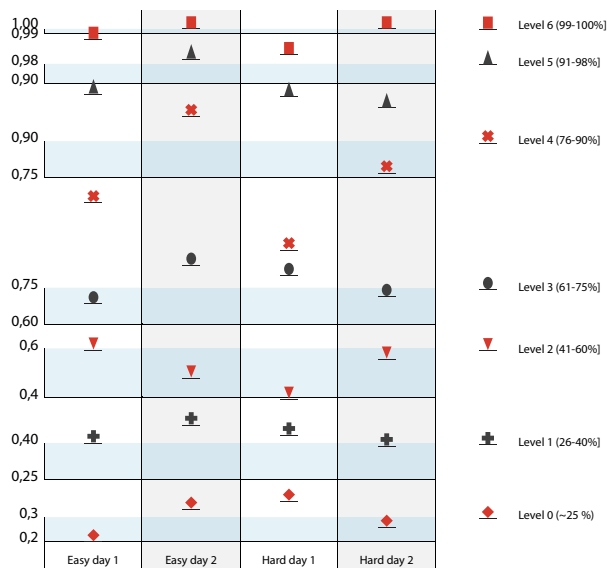


Figure A-6 Performance on globally easy and hard question piles for participant VI

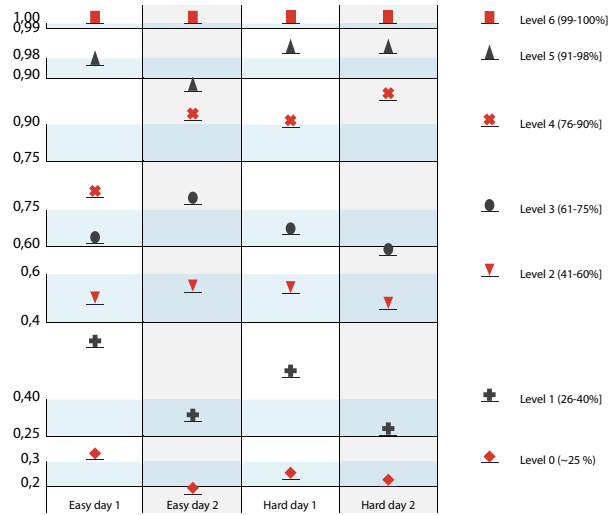


Figure A-7 Performance on globally easy and hard question piles for participant VII

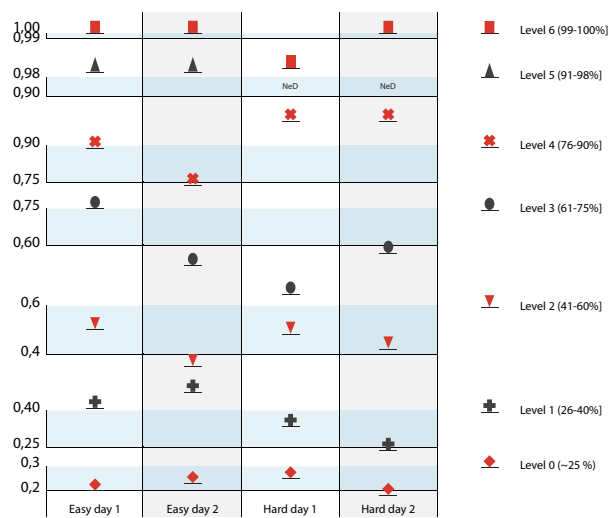


Figure A-8 Performance on globally easy and hard question piles for participant VIII

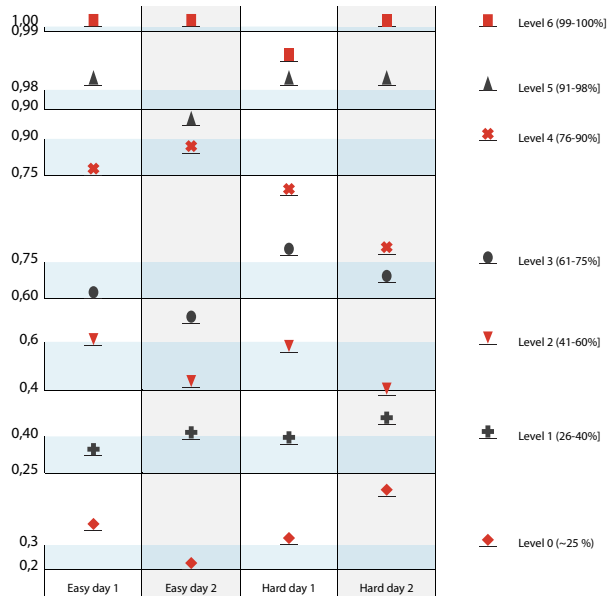


Figure A-9 Performance on globally easy and hard question piles for participant IX

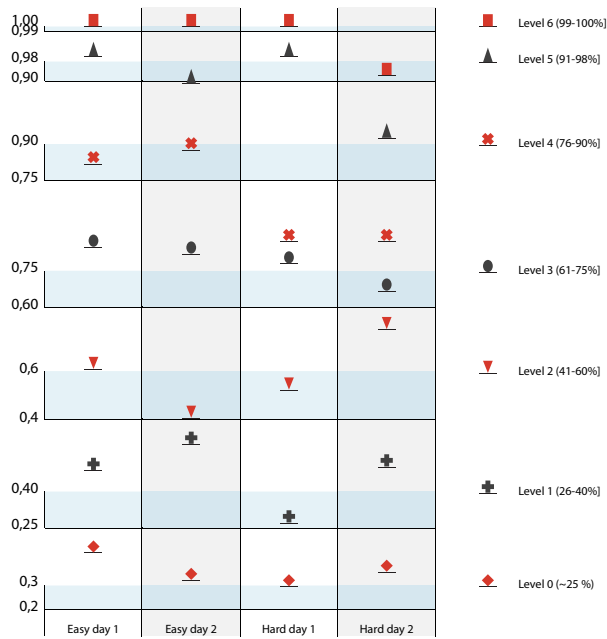


Figure A-10 Performance on globally easy and hard question piles for participant X

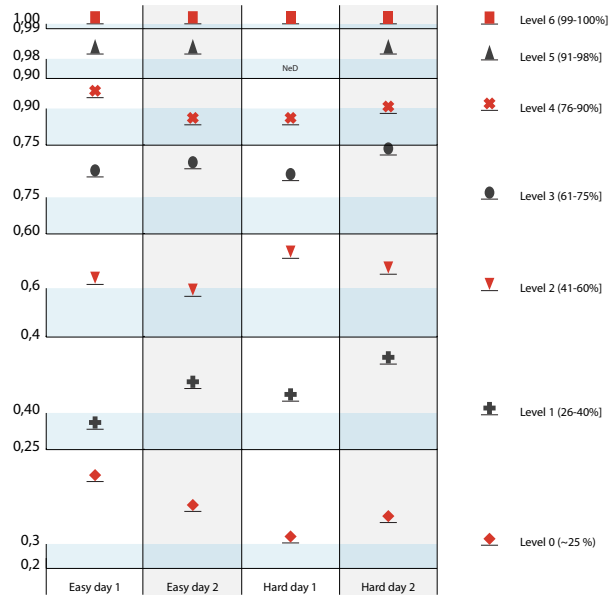


Figure A-11 Performance on globally easy and hard question piles for participant XI

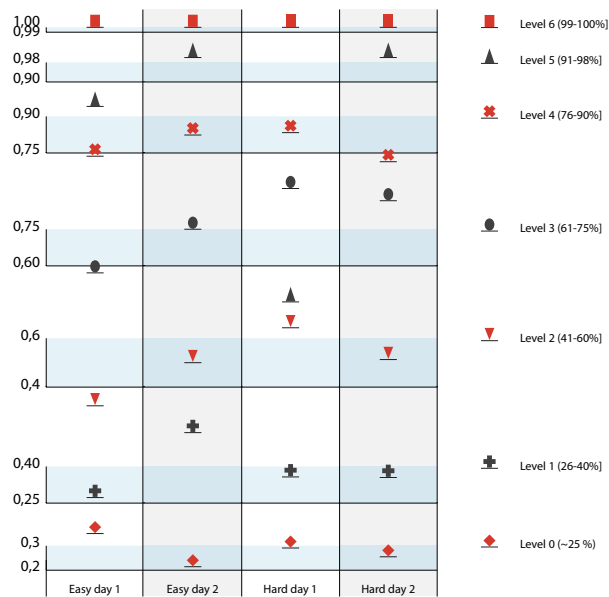


Figure A-12 Performance on globally easy and hard question piles for participant XII

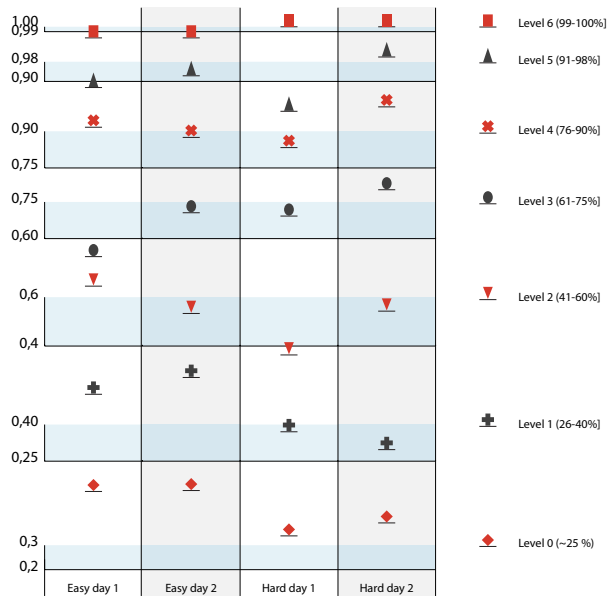


Figure A-13 Performance on globally easy and hard question piles for participant XIII

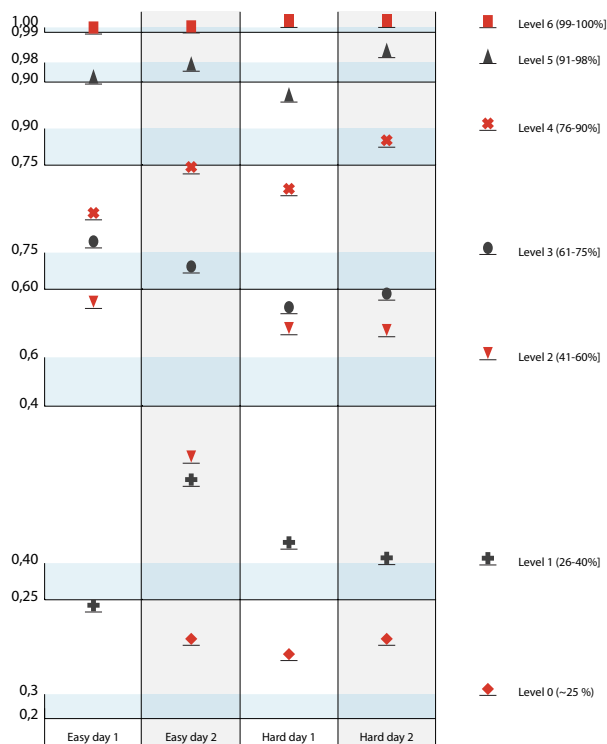


Figure A-14 Performance on globally easy and hard question piles for participant XIV

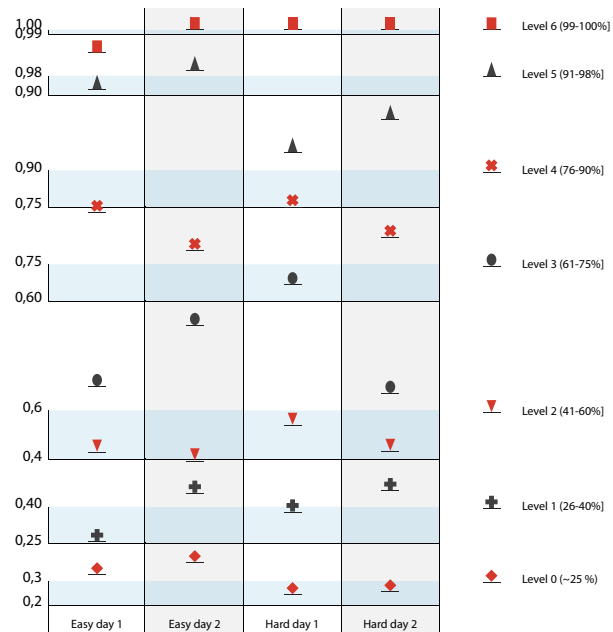


Figure A-15 Performance on globally easy and hard question piles for participant XV {HE anne birgit.ai}

Table A-17 gives a summary of the analysis of each participant's performance when internal task difficulty is considered.

TABLE A-17 INTERNAL TASK DIFFICULTY LEARNING ABILITY SUMMARY,

Y = POSITIVE ADJUSTMENT,

N = NO ADJUSTMENT,

YN = AMBIGUOUS ADJUSTMENT,

U = UNDER-CONFIDENCE,

O = OVERCONFIDENCE,

A = ACCEPTED CALIBRATION LEVEL,

AU = ACCEPTED LEVEL OF UNDER-CONFIDENCE (HIGHER THAN THE MEAN OF THE LEVEL)

AO = ACCEPTED LEVEL OF OVERCONFIDENCE (LOWER THAN THE MAN OF THE LEVEL)

P = PERFECT CALIBRATION (AT THE MEAN OF THE LEVEL)

Participant	Hard internal task difficulty: levels 0 and 1		Easy internal task difficulty: levels 5 and 6	
	Adjustment	Direction (>) of adjustment	Adjustment	Direction (>) of adjustment
I	YN	U > U	Y	O > P
II	N	P > U	Y	P > P
III	YN	AU > AU (no change)	Y	U > P
IV	YN	AU > AU	Y	O > P
V	YN	U > U (no change)	YN	P > AO
VI	Y	U > AU	Y	O > P
VII	Y	U > AU	Y	P > AO
VIII	Y	AU > P	YN	P > P
IX	YN	U > U (no change)	Y	O&U > AO
X	N	AU > U	N	O > U
XI	N	AU > U	N	P > P
XII	Y	AU > AU	Y	O > AU
XIII	Y	U > less U	Y	O > AO
XIV	Y	U > less U	Y	O > P
XV	N	P > AU	Y	O > AO

Table A-18 show the data for all participants, internally hard and easy questions are highlighted.

TABLE A-18

	Day 1				Day 2			
	Level	#Questions	#Correct answers	Hit rate	Level	#Questions	#Correct answers	Hit rate
Hard	0	1534	500	0,33	0	1218	389	0,32
	1	1588	667	0,42	1	1353	566	0,42
	2	1110	569	0,51	2	1436	711	0,50
	3	613	404	0,66	3	809	552	0,68
	4	593	444	0,75	4	613	489	
Easy	5	513	455	0,89	5	531	494	0,93
	6	1268	1249	0,99	6	1244	1234	0,99

Table A-19 shows the two divisions of participants used in analyzing whether knowledge level and learning ability was correlated.

TABLE A-19

PARTICIPANT	TOTAL HIT RATE	GROUP DIVISION IN ANALYZING KNOWLEDGE LEVEL	CLASSIFICATION OF LEARNING ABILITY ON THE CHOSEN PARTICIPANTS FOR KNOWLEDGE LEVEL ANALYSIS (TAKEN FROM THE ANALYSIS OF EACH PARTICIPANT'S CALIBRATION LEARNING ABILITY PRESENTED IN APPENDIX A).
XV	0,45	LEAST KNOWLEDGEABLE PARTICIPANTS	NO CLEAR SIGNS OF POSITIVE ADJUSTMENT
VIII	0,45		SOME POSITIVE ADJUSTMENT
III	0,53		CLEARLY BETTERED PERFORMANCE
I	0,75	MOST KNOWLEDGEABLE PARTICIPANTS	PARTICULARLY AMBIGUOUS RESULTS
V	0,76		SOME POSITIVE ADJUSTMENT
XIV	0,76		CLEARLY BETTERED PERFORMANCE

Figure A-16 to A-19 show calibration graphs for most/least knowledgeable participants on hard/easy question piles.

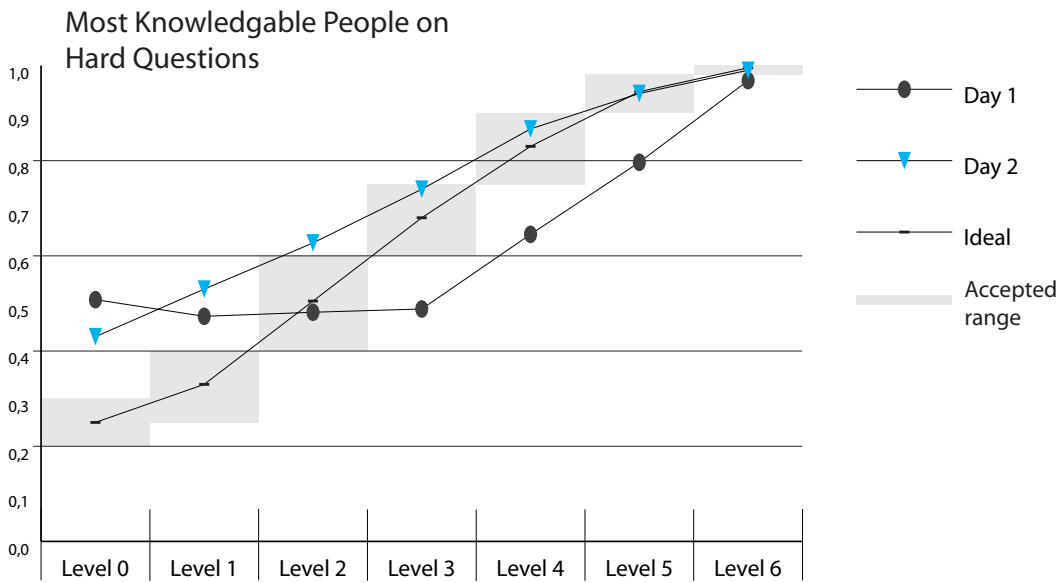


Figure A-16

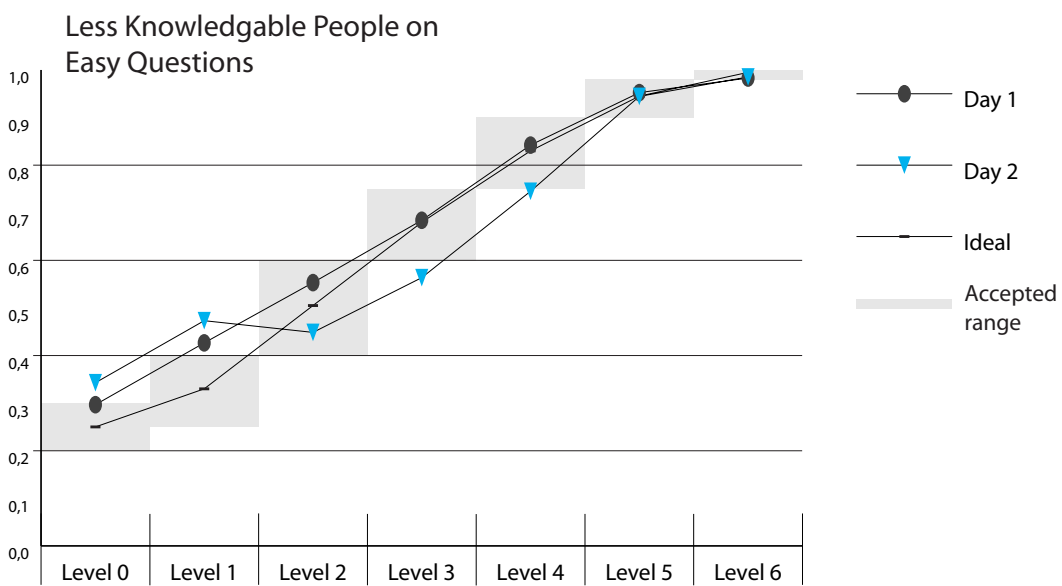


Figure A-17

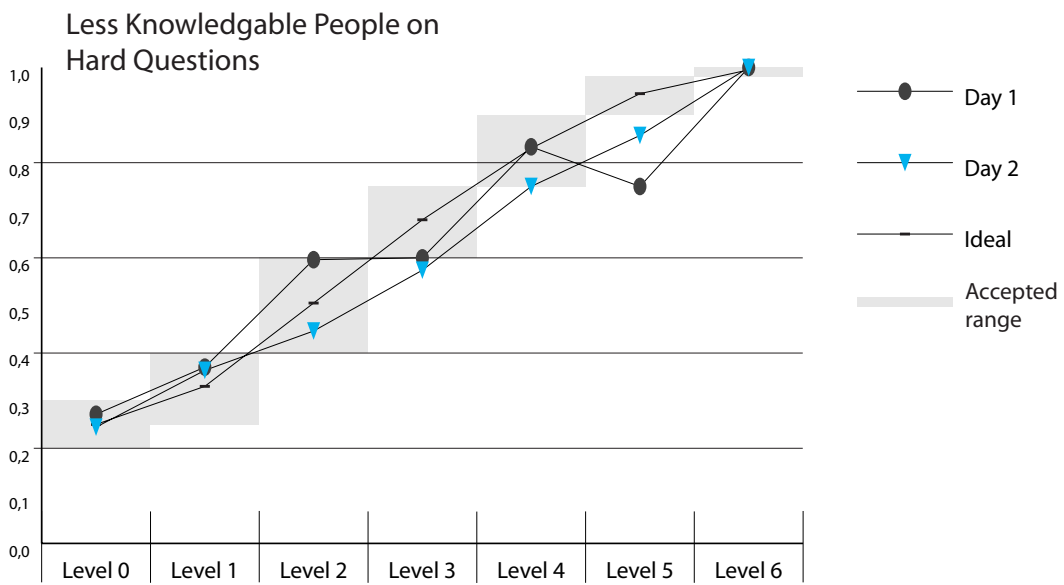


Figure A-18

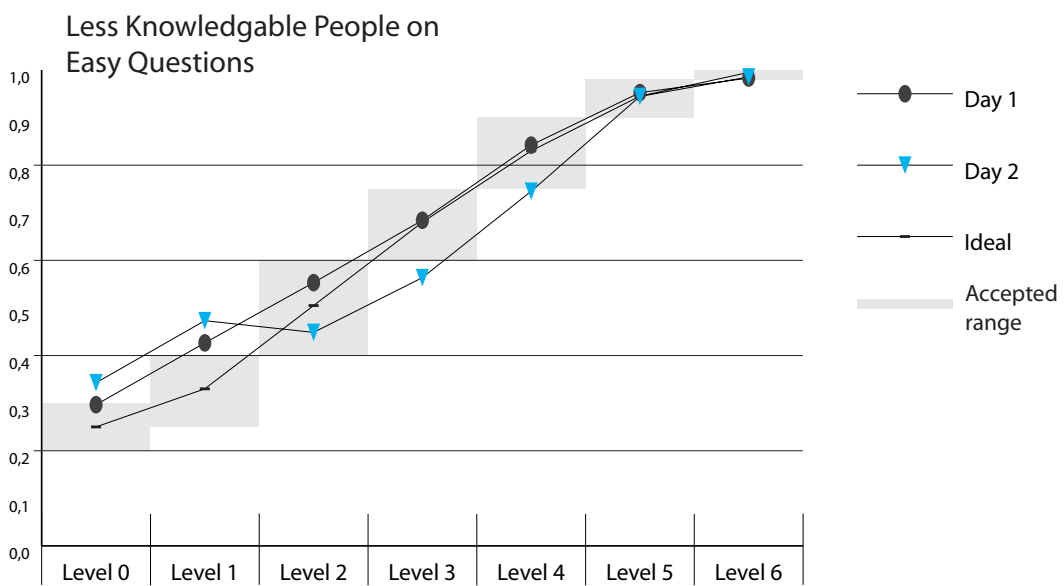


Figure A-19

Table A-20 show the numerical values used in crating figures 17 through 20.

TABLE A-20

Hard Questions, 100' & 80' piles								
	Day 1				Day 2			
	Level	#Questions	#Correct answers	Hit rate	Level	#Questions	#Correct answers	Hit rate
Most knowledgeable participants	0	63	32	0,51	0	93	40	0,43
	1	74	35	0,47	1	83	44	0,53
	2	54	26	0,48	2	59	37	0,63
	3	43	21	0,49	3	50	37	0,74
	4	62	40	0,65	4	45	39	0,87
	5	59	47	0,80	5	51	48	0,94
	6	126	122	0,97	6	99	98	0,99
Hard Questions, 100' & 80' piles								
	Day 1				Day 2			
	Level	#Questions	#Correct answers	Hit rate	Level	#Questions	#Correct answers	Hit rate
Least knowledgeable participants	0	199	54	0,27	0	131	32	0,24
	1	189	70	0,37	1	132	48	0,36
	2	52	31	0,60	2	121	54	0,45
	3	15	9	0,60	3	54	31	0,57
	4	6	5	0,83	4	16	12	0,75
	5	4	3	0,75	5	7	6	0,86
	6	14	14	1,00	6	18	18	1,00
Easy Questions, 20' & 10'								
	Day 1				Day 2			
	Level	#Questions	#Correct answers	Hit rate	Level	#Questions	#Correct answers	Hit rate
Most knowledgeable participants	0	18	7	0,39	0	25	13	0,52
	1	27	11	0,41	1	28	13	0,46
	2	39	25	0,64	2	35	16	0,46
	3	30	21	0,70	3	42	30	0,71
	4	56	43	0,77	4	51	37	0,73
	5	87	81	0,93	5	71	66	0,93
	6	223	221	0,99	6	228	227	1,00
Easy Questions, 20' & 10'								
	Day 1				Day 2			
	Level	#Questions	#Correct answers	Hit rate	Level	#Questions	#Correct answers	Hit rate
Least knowledgeable participants	0	91	27	0,30	0	70	24	0,34
	1	143	61	0,43	1	112	53	0,47
	2	94	52	0,55	2	107	48	0,45
	3	38	26	0,68	3	55	31	0,56
	4	38	32	0,84	4	51	38	0,75
	5	21	20	0,95	5	18	17	0,94
	6	59	58	0,98	6	68	67	0,99

Appendix C

- Experiment no I Participant Manual

The manual given to the GQE participants.



Institutt for Informatikk

Veiledning for Eksperimentdeltakere

—
”Kan sikkerhetsestimering læres?”

Eksperimentet holdes av Tanja Gruschke,
i forbindelse med hovedoppgaven i informatikk.

simula . research laboratory

Info om eksperimentet

Les dette før du starter, lurer du på noe spør Tanja!

Formålet med dette eksperimentet er undersøke om man kan lære å vite hvor sikker man er på at noe er rett. I dette eksperimentet skal vi ta i bruk spørsmål fra brettspillet "Vil du bli millionær" for å se om vi kan finne ut dette.

Du skal gjennom seks runder hver eksperimentdag, i en runde skal du svare på ca 60 spørsmål med ett gitt vanskelighetsnivå (nivåene er 10.000, 20.000, 40.000, 60.000, 80.000, 100.000). Rekkefølgen på nivået er tilfeldig valgt. På eksperimentdag to skal du gjennom de samme vanskelighetsnivåene i den samme rekkefølgen som på dag en.

For hvert spørsmål du svarer på skal du kjenne etter hvor sikker du er på at du har rett. Måten dette måles på er i prosent. Denne prosenten skal speile frekvensen av antall riktige svar etter at mange spørsmål er besvart. For eksempel i de tilfellene hvor du har en følelse av at svaret ditt er "fifty-fifty" (41% - 60%) så skal du ha rett på halvparten av spørsmålene du har markert med denne frekvensen etter en runde. Etter hver runde får du feedback på hvordan du traff med sikkerheten din. Prøv å juster deg etter denne feedbacken, slik at du treffer bedre neste runde. Neste eksperimentdag, når du skal gjennom de samme rundene, er det et håp at du har lært å kjenne "sikkerhetsfølelsen" din bedre.

Det er selvfølgelig en premie for at du skal prøve å gjøre ditt beste. Gevinsten til den som svarer riktig på flest av spørsmålene, OG som klarer å justere seg best i følge feedback er 10 FLAX lodd. Nummer to får fire, og nummer tre får ett. Siden det skal kjøres flere runder av eksperimentet, blir vinnerene kontaktet i ettertid (ca. uke 45).

Takk for at du er med, og lykke til ☺

Instruksjoner

For at du skal få lønn for innsatsen, må du registrere en del info om deg selv på denne siden:

<http://folk.uio.no/tanjag/persinfo.php>

Du får 1000kr for å være med på eksperimentet. Dette er skattefritt hvis du ikke har vært med på, eller skal være med på, andre eksperimenter i regi av Simula Research Laboratory i år. De blir utbetalt 10. november (forhåpentligvis, hvis ikke så kommer de 10 desember).

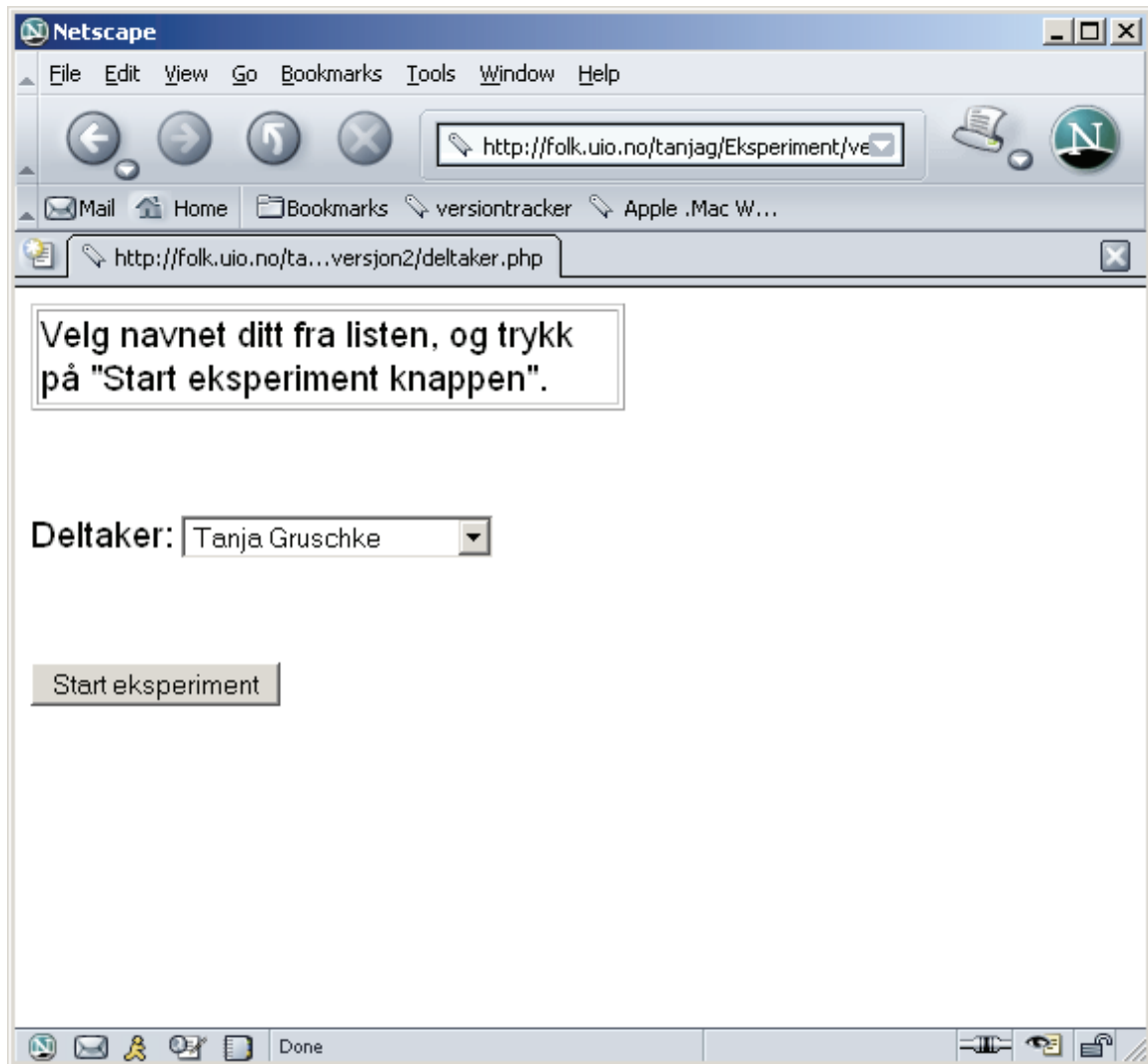
Deretter må du fylle ut Mox-testen (neste side), når du leverer resultatene fra denne får den første spørsmålsbunken, og du kan starte eksperimentet.

Eksperimentet ligger på:

<http://folk.uio.no/tanjag/Eksperiment/versjon2/deltaker.php>

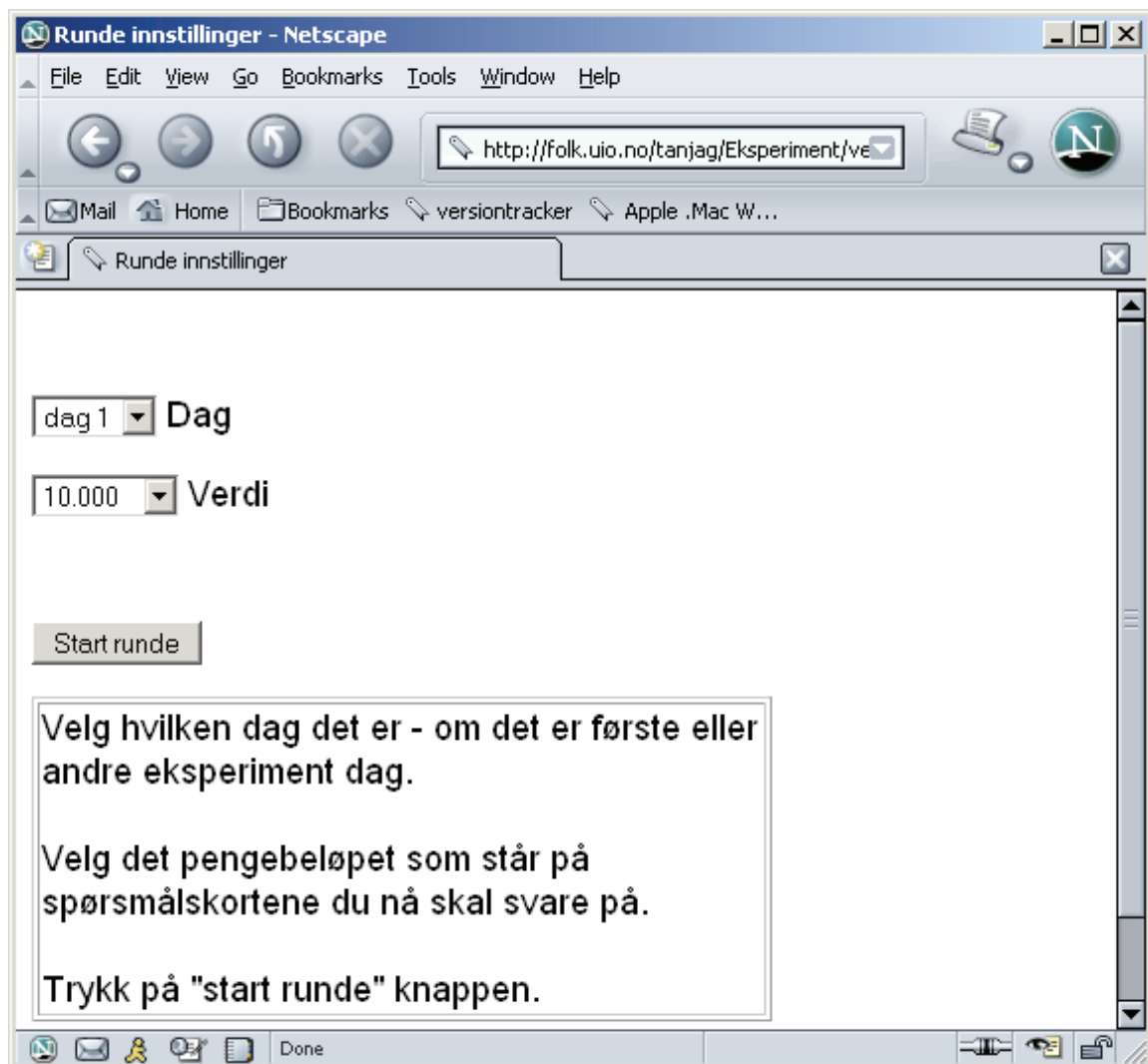
Hvis det skjer noe kan du alltid gå tilbake til denne siden og fortsette der du slapp i spørsmålsbunken din – bare pass på at du fyller de samme valgene på ”Runde innstillinger” siden (se nedenfor for mer detaljert informasjon om denne siden).

Valg av deltaker



Velg navnet ditt fra deltaker rullegardin lisen, og trykk på knappen merket "Start eksperiment". Nå kommer du til:

Runde innstillinger vinduet

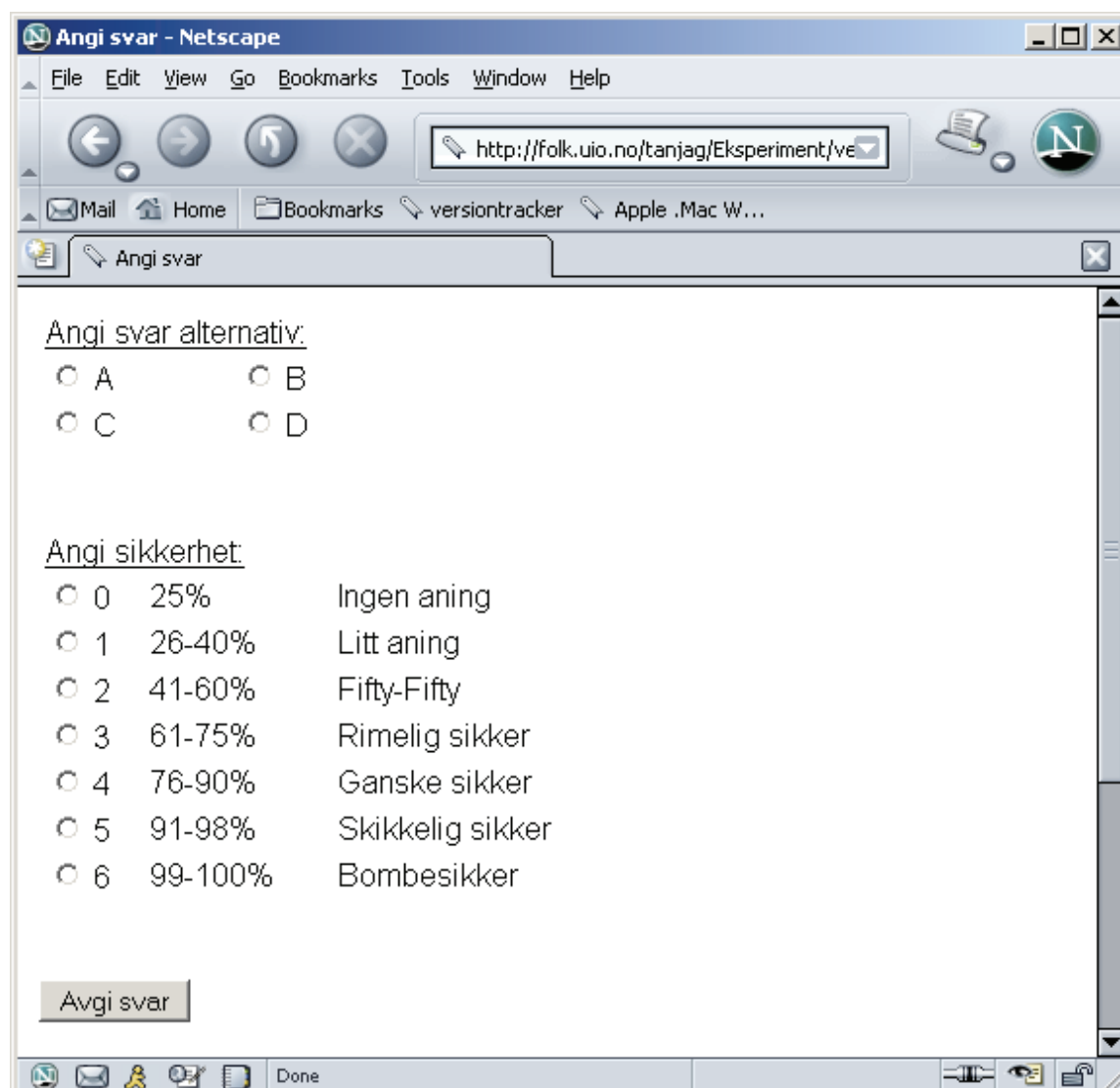


I rullegardin menyen merket med dag, velger du hvilken eksperimentdag det er (det kan hende det kun er et valg, da er det dette du skal bruke).

I rullegardinmenyen merket med verdi, velger du det beløpet som står på spørsmålsbunken du skal til å svare på nå.

Trykk på knappen merket ”**Start runde**” når valgene har blitt gjort. Nå kommer du til:

Avgi svar vinduet

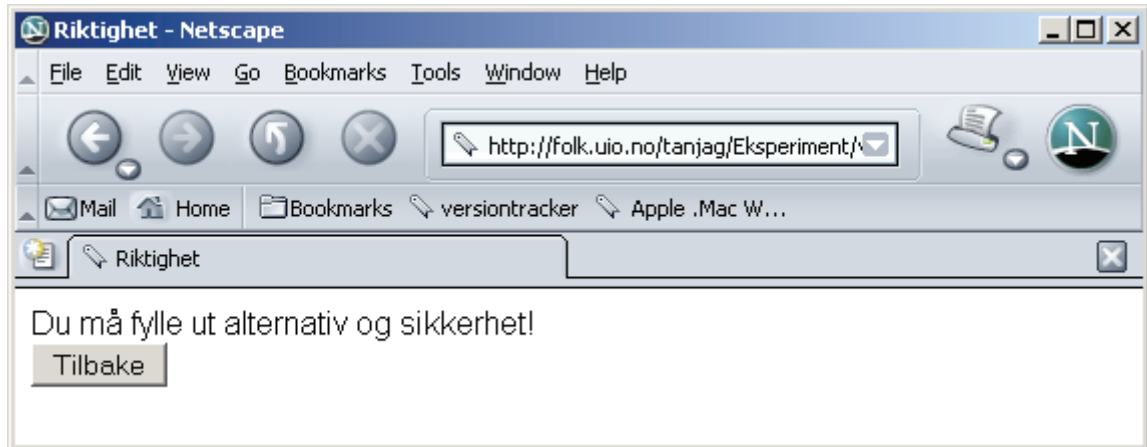


Les spørsmålskortet nå. Du skal kun svare på spørsmålet på den ene siden av kortet (det er et spørsmål på hver side av kortet). Sørg for å legge kortene samme vei når du legger de fra deg igjen, du skal nemlig svare på den andre siden på dag 2 av eksperimentet! Spørsmålskortene er sortert etter det lille tallet i øverste høyre hjørne - etter partall og oddetall.

Så haker du av det alternativet du tror er riktig under "Angi svar alternativ".

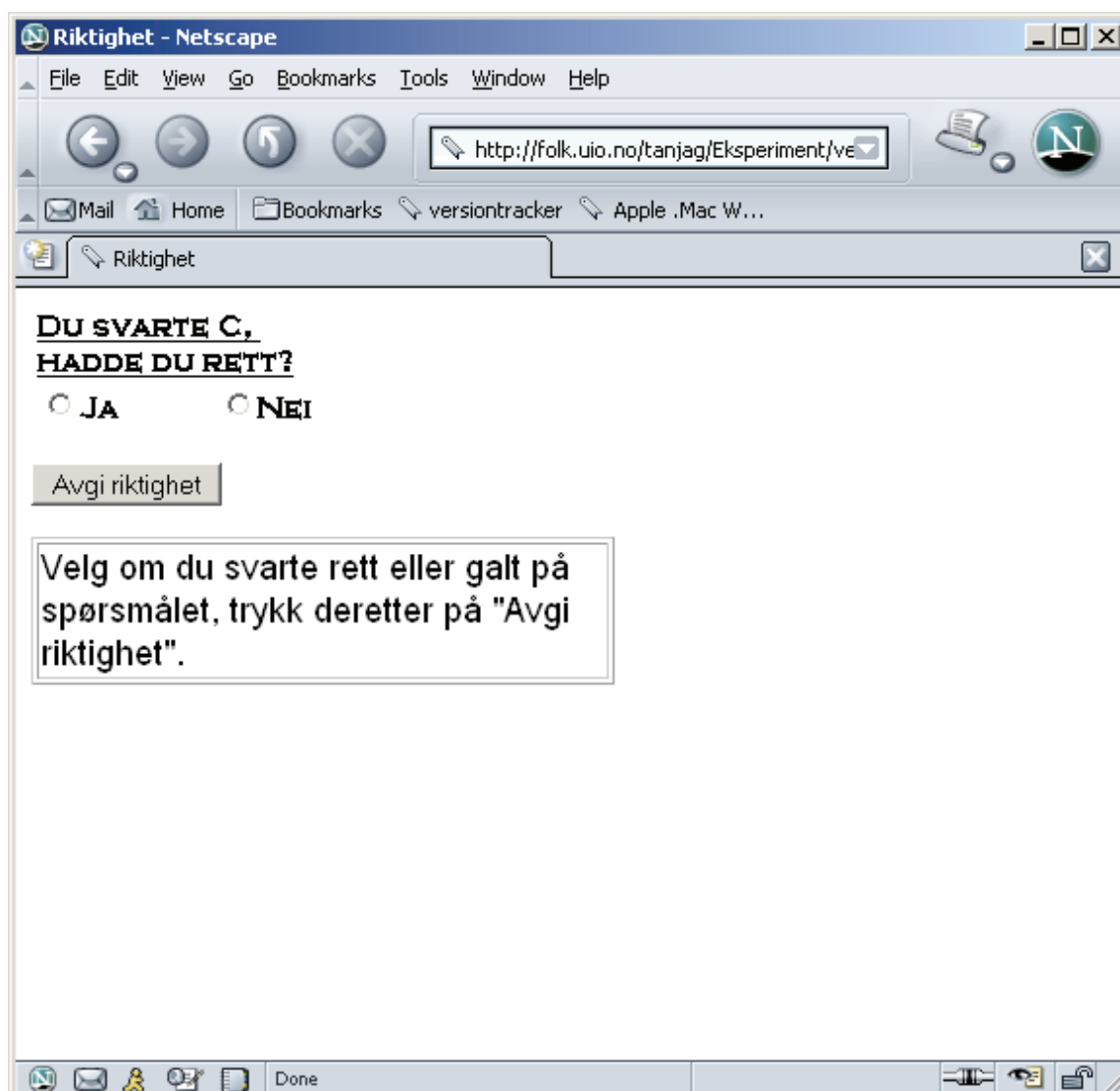
Så skal du huke av hvor sikker du er på at svaret ditt er rett under "Angi sikkerhet".

Når du har valgt både alternativ og sikkerhet trykker du på knappen merket "Avgi svar". Skulle du av en eller annen grunn ikke ha fylt ut både alternativ og sikkerhet, kommer du til et vindu med beskjed om dette:



Trykk da bare på **"tilbake"** knappen og du kommer tilbake til "Avgi svar vinduet". Når all info er fylt ut og du trykker på **"Avgi svar"** knappen kommer du til:

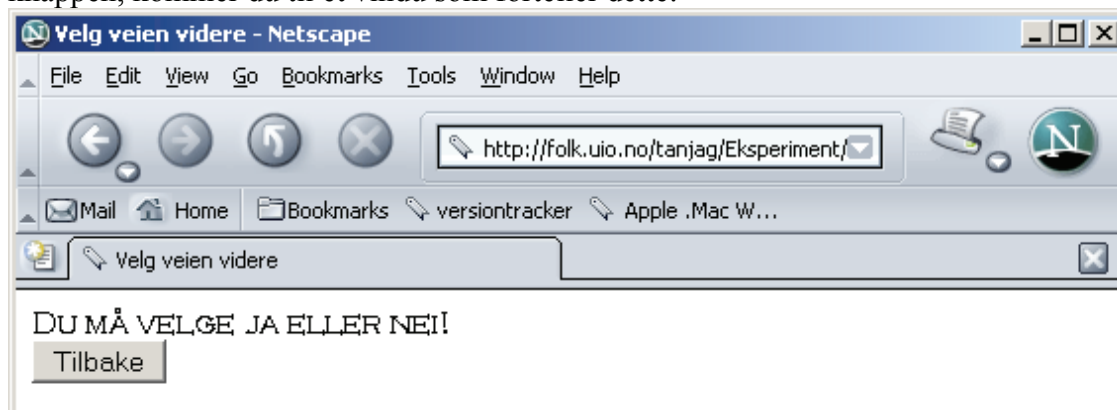
Rett svar vindet



Nå kan du snu spørsmålskortet og se på fasiten på baksiden.

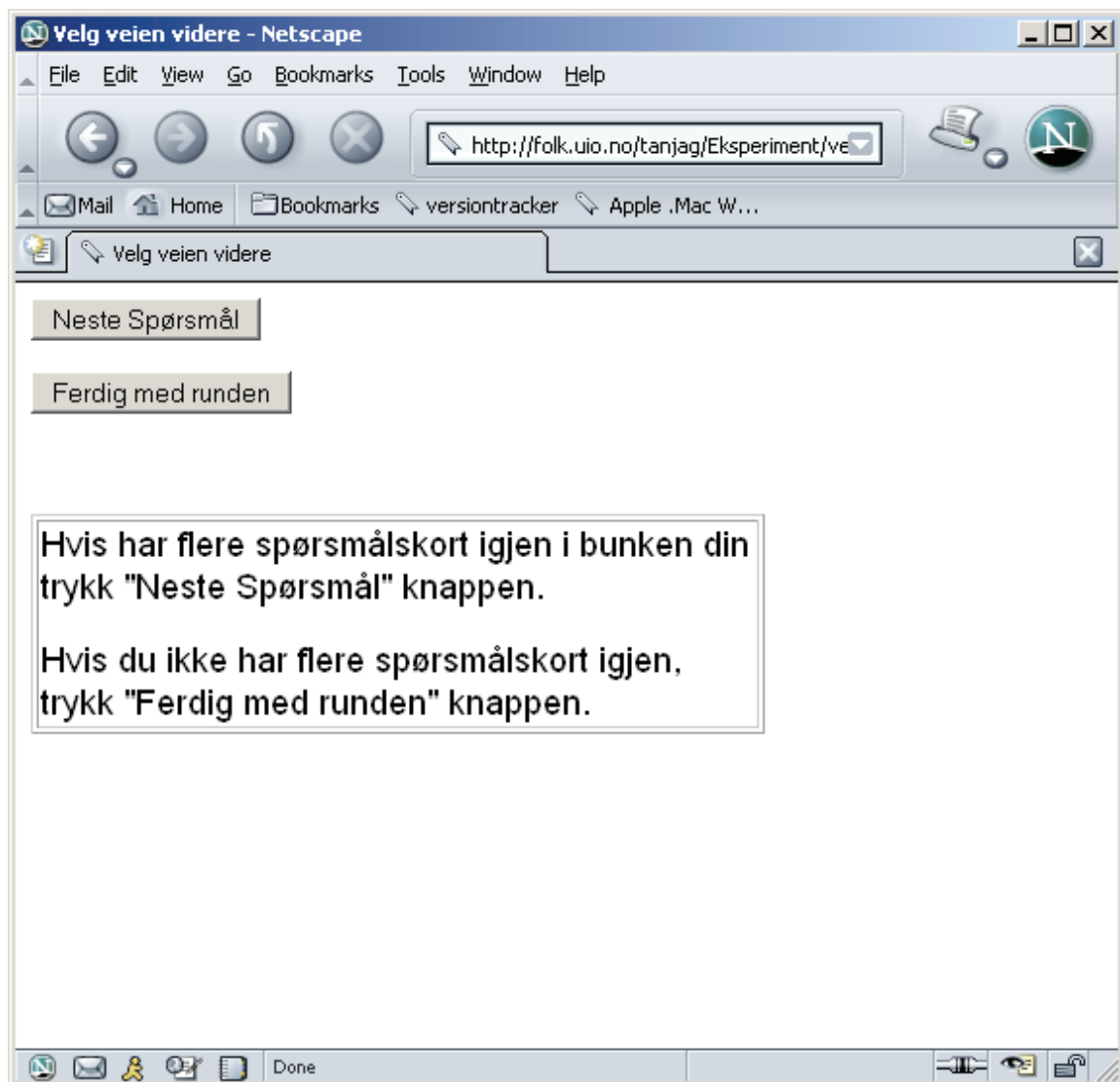
Det står hvilket alternativ du valgte på skjermen, huk av for om du hadde rett(Ja) eller galt(Nei) og trykk på "Avgj riktighet" knappen.

Skulle du av en eller annen grunn ikke ha valgt ja eller nei når du trykker på knappen, kommer du til et vindu som forteller dette:



Trykk da på "tilbake" knappen og du er tilbake i "Rett svar vinduet". Når ja eller nei er valgt og du trykker "Avgi riktighet" knappen kommer du til:

Velg veien videre vinduet



Her er det to knapper. Hvis du har flere spørsmål igjen i spørsmålskortbunken din trykker du "Neste spørsmål" knappen. Da kommer du tilbake til "Avgi svar vinduet", og tar det derfra igjen med det nye spørsmålet.

Hvis du ikke har flere spørsmål igjen å svare på trykker du "Ferdig med runden". Da kommer du til vinduet med feedback, få tak i Tanja så kommer hun og tolker resultatene dine.

Tilbakemelding - Netscape

File Edit View Go Bookmarks Tools Window Help

http://folk.uio.no/tanjag/Eksperiment/ve

Mail Home Bookmarks versiontracker Apple .Mac W...

Tilbakemelding

Her kan du se hvordan du traff med sikkerhets angivelsene dine.

Snakk med Tanja før du går videre!

Sikkerhet	Antall Svar	Antall Riktig	%	
0	25%	0	0	0
1	25-40%	1	1	100
2	41-60%	0	0	0
3	61-75%	0	0	0
4	76-90%	0	0	0
5	91-98%	0	0	0
6	99-100%	0	0	0

Ny Runde

Done

Appendix D

- Experiment no 2 Participant Manual

The manual given to the SDE participants.

[**simula . research laboratory**]

mars 2004

Veiledning for eksperimentdeltakere

"Å lære usikkerhet

– estimering av arbeidstid på programmeringsoppgaver"

Tanja Gruschke
Hovedfag ved
Institutt for Informatikk
Universitet i Oslo



Praktisk informasjon

For å få lønn må du fylle ut en timeliste med nødvendig informasjon og levere skattekort. Timelister er tilgjengelig både på papir og elektroisk, det er bare å si ifra hva du foretrekker. Du får timelønn for deltakelsen. Lønn blir utbetalt den 10ende hver mnd.

Det er det samme hvilke 6 dager i løpet av uke 13 og 14 du deltar på, men det er ønskelig at vi setter opp hvilke dager det er snakk om på forhånd; velg de dagene som passer best for deg.

Du tar selvfølgelig do-, kaffe-, røyke- og spisepauser når du vil; innenfor rimelighetens grenser selvfølgelig!

Formålet med eksperimentet

Hovedformålet med eksperimentet er å undersøke om det er mulig å lære hvor sikker man er på tidsestimater ved hjelp av tilbakemelding. I dette tilfellet blir det å anslå sannsynligheten for å treffe innenfor forskjellige ”slack”- intervaller (se eksempel i tabell nedenfor), som er regnet ut på bakgrunn av deltakerens eget tidsestimat. Deltakeren får se hvordan det gikk med anslått tid og faktisk tidsbruk etter hver oppgave.

”slack”-intervaller	Eks. estimert 1time og 20min
[90 %; 110 %]	[1 t 12 min, 1 t 28 min]
[60 %; 150 %]	[0 t 48 min, 2 t 0 min]
[50 %; 200 %]	[0 t 40 min, 2 t 40 min]

Du har fastsatte prosentpoeng å forholde deg til når du velge sannsynlighet for hvert intervall. Disse er:

Hjelpende beskrivelse av nivået	Frekvens
Helt usannsynlig at tiden vil treffe dette intervallet	0 %
Lite sannsynlig at tiden vil være innenfor dette intervallet	5%
Rimelig sjeldent at tiden vil være innfor dette intervallet	20%
Sjeldent at tiden vil være innenfor dette intervallet	35%
Halvparten av gangene vil jeg treffe dette intervallet	50%
Ganske sannsynlig at tid brukt vil være innenfor dette intervallet	65%
Veldig sannsynlig at jeg vil treffe innenfor dette intervallet	80%
Nesten alltid innenfor intervallet	95%
Alltid	100%

**Det er viktig at du hele tiden tenker over og fokuserer på at du skal lære din egen usikkerhet å kjenne
– IKKE briljere med programmeringsferdighetene dine.**

Gjennom hele eksperimentet er det viktig at du har fokus på at det handler om læring av usikkerhet, og ikke på selve programmeringen. Grunnen til at det blir brukt personer som er dyktige programmerere er nettopp fordi vi ønsker å se bort ifra det

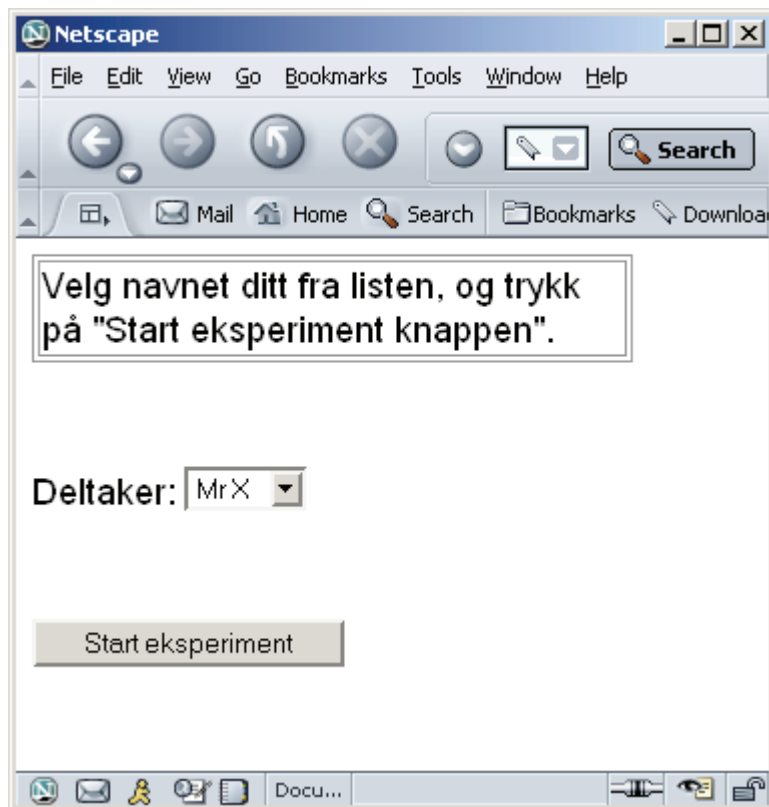
rent programmeringstekniske i dette eksperimentet. Det er derfor et håp at ved å bruke dyktige personer, som har erfaring med programmering, at denne delen vil gå relativt automatisk og dermed ikke påvirke tid brukt på oppgavene.

Gangen i eksperimentet

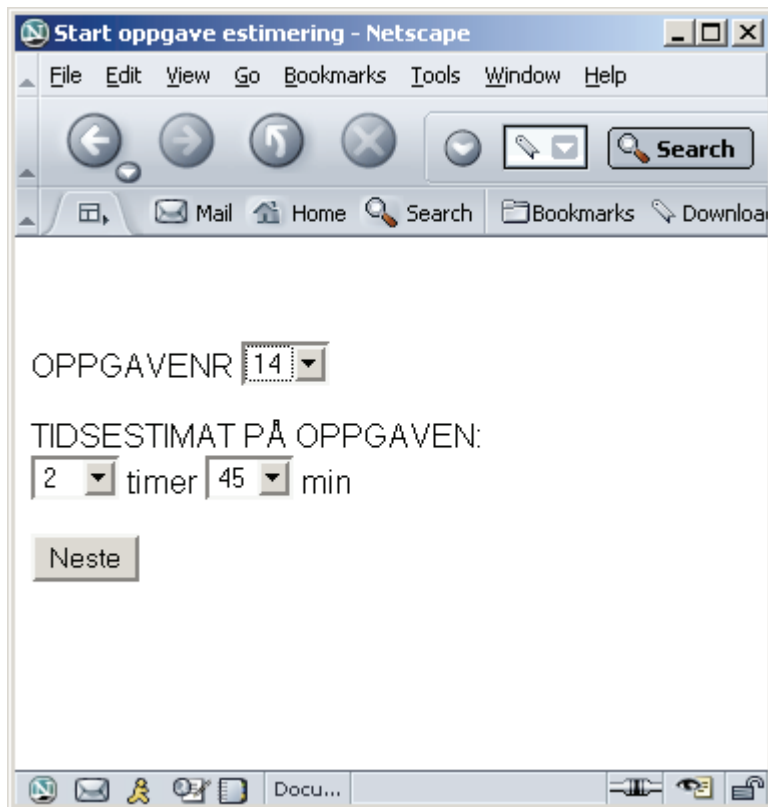
Deltakerne får utdelt en oppgave.

Når du er klart til å starte, gå til

<http://folk.uio.no/tanjag/Eksperiment/java/deltaker.php> finn navnet ditt i listen og trykk på knappen.



På neste side velger du oppgave nr og estimerer arbeidstiden på oppgaven, og trykker på knappen.



Start oppgave estimering - Netscape

File Edit View Go Bookmarks Tools Window Help

← → ↶ ✕

Search

Mail Home Search Bookmarks Download

OPPGAVERN 14

TIDSESTIMAT PÅ OPPGAVEN:

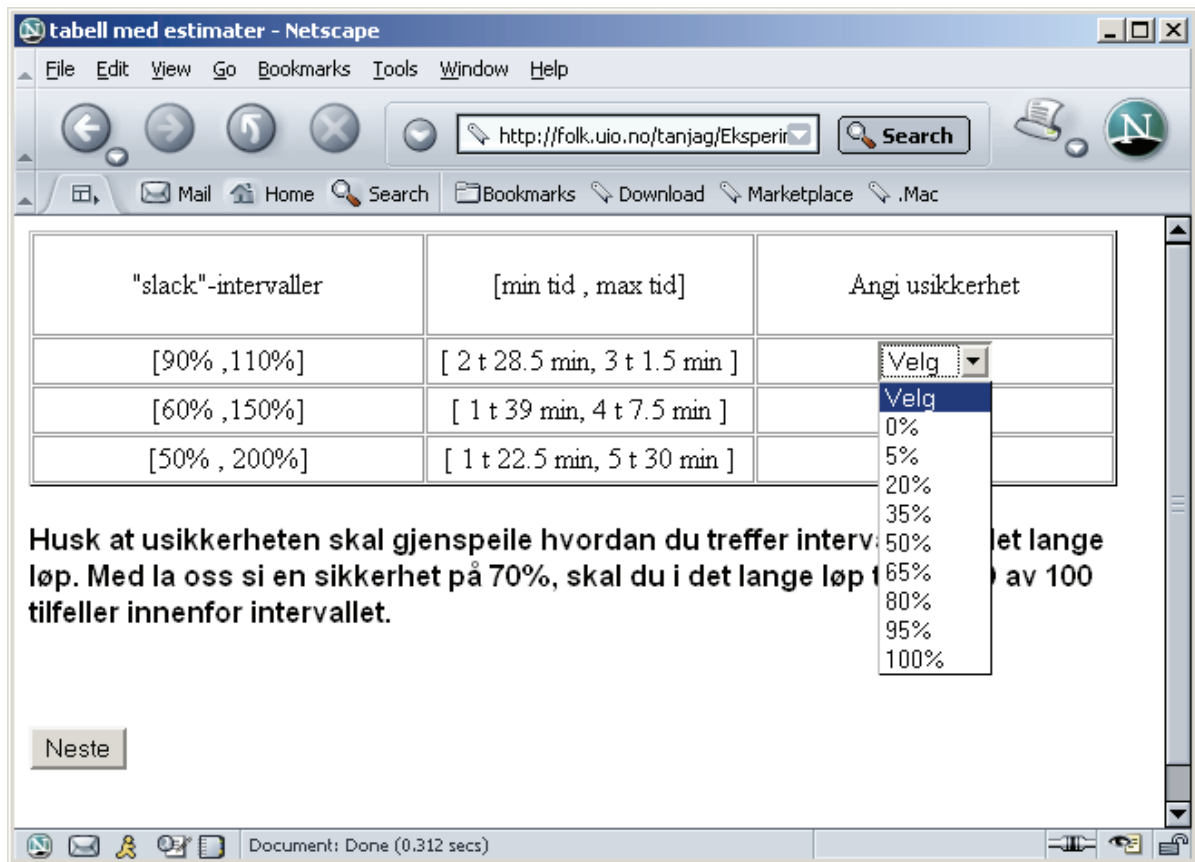
2 timer 45 min

Neste

Docu...

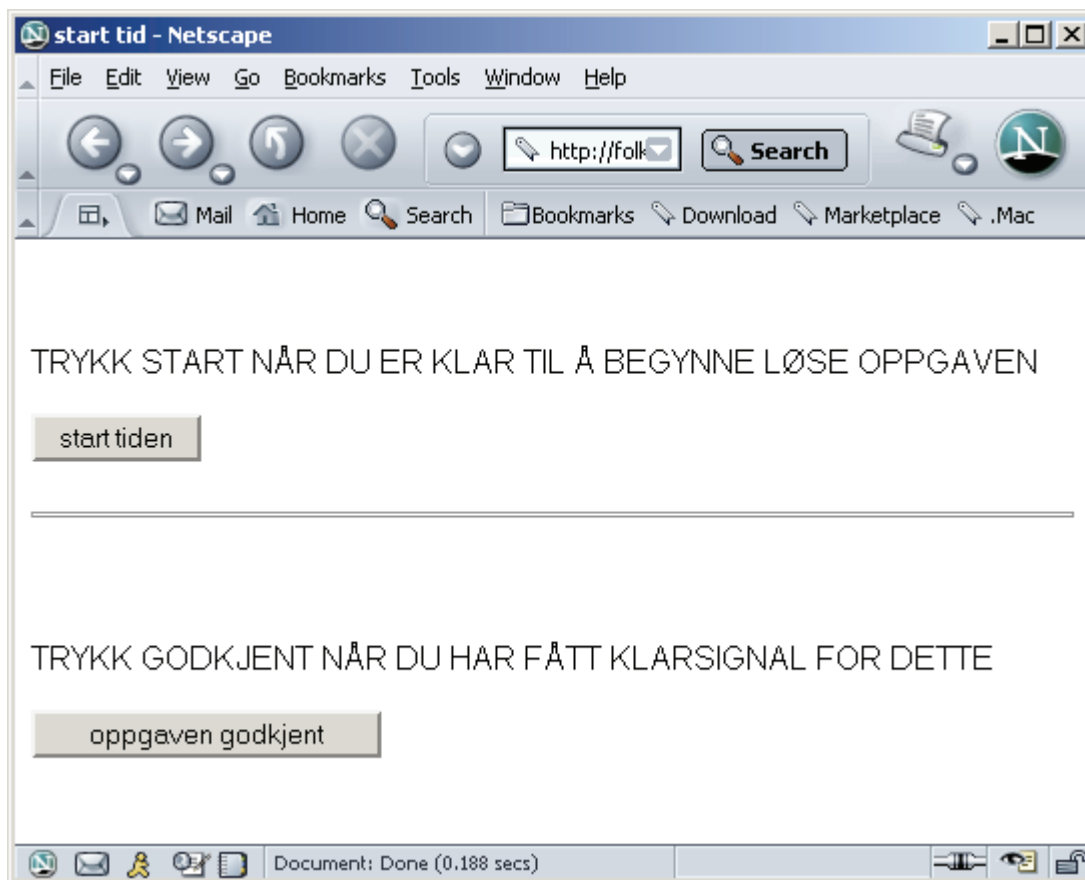
Når du estimerer skal du forsøke å legge så mye arbeid i estimeringen slik at det står i forhold til hvor stor oppgaven er. Forsøk å ha samme nivå av engasjement på alle estimeringsoppgavene gjennom hele eksperimentet.

Nå kommer du til en side som viser en tabell for "slack"-intervallene har blitt regnet ut på bakgrunn av estimatet ditt. For hvert intervall, velger du i rullegardinmenyen hvor sannsynlig du mener det er at du er innenfor intervallet; både det å bruke kortere eller lengre tid enn tiden som står der regnes for å være utenfor. Trykk på knappen når du er ferdig.

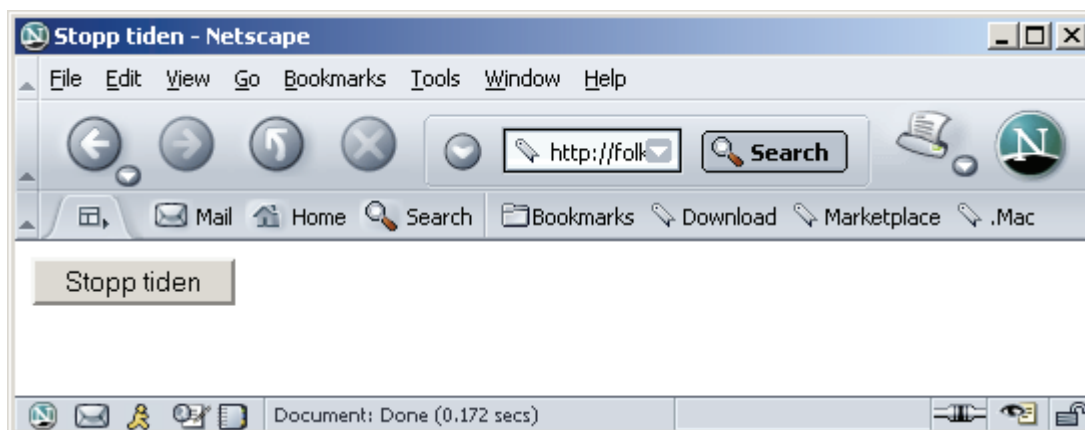


Etter at du har trykket neste skal du skrive opp hvor lang tid du brukte på å estimere på arket "tid brukt på estimering".

På neste side trykker du ”start tiden” knappen når du er klar til å begynne å løse oppgaven.



På neste side trykker du stopp tiden når du mener du er ferdig med oppgaven, eller har tenkt å ta deg en lengre pause.



Når ”stopp tiden” knappen blir trykt kommer du tilbake til ”start tid vinduet”. Hvis du mener du er ferdig med oppgaven må denne godkjennes av Tanja før du trykker ”oppgaven godkjent”. Når du har fått oppgaven godkjent, og trykker knappen, kommer du til et vindu som viser estimatene, de valgte sannsynlighetene og den faktiske tidsbruken din på oppgaven. Skriv noen setninger om hvordan det gikk (se ”Refleksjoner rundt estimering” for mer info om dette), trykk på knappen og du kommer tilbake til deltaker siden og du er klar for neste oppgave.

Om selve oppgaveløsningen

Oppgavene skal løses i Java, og det skal brukes sunne og gode objektorienterte prinsipper i programmeringen.

Det er viktig at det står helt klart for deg hva oppgaven går ut på FØR du begynner å programmere. Spør heller en gang for mye, enn en gang for lite. Spør gjerne underveis hvis det er presiseringer som kommer frem som du lurer på.

Du har lov til å slå opp i alle mulig bøker du vil, og bruke API ressurser på nettet. Det eneste som IKKE er lov er å klippe og lime inn kode som ligger ute, eller som du har skrevet selv (dette inkluderer kode du har skrevet i tidligere oppgaver i dette eksperimentet).

For å få en oppgave godkjent som ferdig, er det et sett av tester som hører til hver oppgave som må gjennomføres, i tillegg til evt. uformell kikk på koden eller andre måter å kontrollere at programmet løser oppgaven tilstrekkelig. Når det gjelder kvaliteten på programmene du skriver, vil denne bli evaluert i ettertid av en tredjepart her på Simula. Det viktigste når det gjelder kvaliteten på koden du skriver, er at du prøver å holde deg på samme nivå i alle oppgavene. Hvis du har løst oppgaver på ”sikker planke” måten, ikke prøv å plutselig lag finurlige geniale løsninger på andre oppgaver – vær konsekvent gjennom hele eksperimentet. Grunnen til dette er at variasjon i programmeringsstrategi, ikke skal være en påvirkning på faktisk tidsbruk og hvor godt du treffer estimatene dine.

Refleksjoner rundt estimering

Når du er ferdig med en oppgave blir du bedt om å skrive noen setninger om hvordan du synes det gikk; her har du noen hjelpespørsmål og pekepinner på hva som er nyttig for å meg å få vite om:

- Den faktiske tidsbruken din kontra hva du estimerte – angi årsaker til at estimatet var nøyaktig/unøyaktig.
- Tenk på at sannsynlighetene du valgte på intervallene, skal reflektere hvor sikker du er på å treffe innenfor disse intervallene i det lange løp. Var noen av sikkerhetsnivåene dine for optimistiske eller pessimistiske? Angi årsaker.
- Tenk gjennom om det er nødvendig å justere sikkerhetsnivåene dine, igjen med tanke på at de skal reflektere frekvensen på hit-raten i det lange løp.

”Debriefing”

Etter at alle oppgavene er gjennomført, eller vi har sluppet opp for tid, skal jeg ha et lengre intervju med hver av deltakerne om forskjellige aspekter rundt eksperimentet. Dette blir holdt når det er mest passende for den enkelte så fort som praktisk mulig etter at eksperimentet er avsluttet (sannsynligvis en uke seinere), og vil sannsynligvis vare mellom 1 til 3 timer.

Appendix E

- Java Development Tasks used in Experiment no 2

In the following the tasks given to the participants in the Java experiment are given.

1. Kundeinfo

Etter å ha studert ved IFI har du startet ditt eget firma, hvor kundene strømmer til. Du føler behov for å lage et system for å holde orden på informasjon om kundene, i første omgang deres navn, adresse og kundenummer. Lag et program som fra brukerens side har følgende oppførelse:

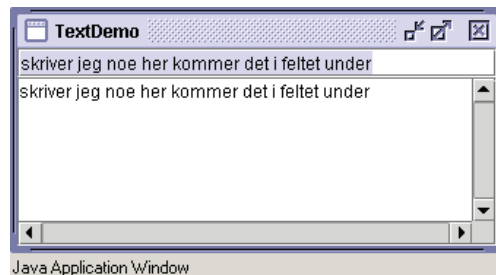
Når programmet starter skal det lese inn informasjon fra fil om tidligere kunder. På en fil (som du selv lager og der du selv definerer filnavn og filformat) ligger kundenummer, navn og adresse. Disse opplysningene skal leses inn fra fila og legges i tre tabeller i Register-klassen.

Programmet skal tilby brukeren å lese inn informasjon om nye kunder. Den informasjonen som leses inn via tastatur skal legges til i tabellene der de tidligere kundene er registrert.

Programmet skal tilby brukeren å skrive ut på skjerm all informasjon om en bestemt kunde (dvs. kundenummer, navn og adresse). Brukeren skal kunne velge mellom å søke på kundenummer eller på navn. Dersom brukeren velger å søke etter et kundenummer, skal programmet spørre brukeren om å oppgi dette nummeret, og all informasjon om kunden med dette kundenummeret skrives ut på skjermen. Dersom det ikke finnes noen kunder med det oppgitte nummeret skal programmet opplyse brukeren om dette. Programmet skal oppføre seg tilsvarende dersom brukeren velger å søke etter en kunde basert på kundens navn.

Når brukeren velger å avslutte, skal all informasjon som ligger i de tre tabellene skrives ut til den samme fila som man hentet inn informasjonen fra. (Dvs. at dersom du la inn flere kunder, skal også disse være med når det skrives ut til fila.) Bruk "object in stream" og "object out stream" til å lese til og fra fil.

2. swing



Bruk Javas swing klasser til å lage et vindu med et felt som man skriver tekst inn i. Når man trykker Enter blir teksten skrevet til feltet under. Når du skriver noe nytt blir dette skrevet over teksten som allerede står i feltet (som flyttes nedover).

3. Histogram

Lag et program som lager et histogram, som gjør det mulig for deg å se på frekvensdistribusjonen av et sett med verdier. Programmet burde lese inn vilkårlige heltall som er mellom 1 og 100 (og inkluderer begge tallene) fra en fil; så produsere et diagram som viser frem det aktuelle histogrammet.

Filer med heltall finner du i mappen "histogram" på <http://folk.uio.no/tanjug/Eksperiment/java/området>.

4. Files

Write a program that prompts the user for a filename and then outputs whether a file of that name exists in the current directory. If it does, it should also output:

Whether the file can be read from or written to

The size of the file

Whether it is a directory
The filename should be given when the program is run; e.g.
>java fileinfo [filename]
Where "fileinfo" is the name of your program, but you can call it what you want.

5. Text adventure game

Background info: Text adventure games are a legacy from a time when computing power was small, when terminal access was commonplace, and when monochrome graphics was "state of the art". Players used imagination, fired by descriptions of old abandoned caves, dungeons, and even futuristic spaceships populated by Vogons.

The art of writing a good text adventure game is a lost one. It's lost; because most players have become so familiar with cartoony VGA graphics that they think anything less would be boring. They've never hunted treasure in the land of Zork, or traveled across the stars with Douglas Adam's Hitchhiker's Guide to the Galaxy. Alas, nor will many have you who read these words.

One of the greatest challenges for a would-be programmer in the early 80's was to write a text adventure game; one that would run inside a 64k memory barrier, and fit on a single 5.25" inch disk for the Apple II. Computer magazines ran ten or fifteen part articles on writing the "ultimate" text adventure game, and programming was fun. Now you can relive those days, and write your very own text-adventure game, and do so in an object-orientated way!

Lag en mini versjon av det slikt spill. Men en helt (deg); to steder du kan dra til (puben og kåken), et par monstre og ting som du finner som er enten våpen eller penger. Du reiser mellom puben og kåken og på veien til en av stedene kan du støte på monstre eller finner ting helt tilfeldig. Ting du kan finne er penger eller våpen. Hvis du har funnet våpen, kan du bruke disse mot monstrene. På puben kan du drikke øl(øl koster penger, de må du finne) eller gå igjen, i kåken kan du sove (avslutte spillet) eller gå ut.

6. Kalkulator

Lag en grafisk kalkulator. Den skal kunne utføre de vanligste matematiske operasjonene – plusse, trekke fra, gange og dele. Den skal vise frem mellomregninger når man bruker de forskjellige operasjonene. Det skal være en knapp for å nullestille kalkulatoren (fjerne mellomregningene fra displayet og vise null) – mellomregningene skal da være borte fra "minnet".

7. Animasjon 2

Write a program to make a graphics image, loaded from a file, grow in size and then shrink on the screen.

8. Calendar

Write a program that displays a calendar for a specified month, using the Date, Calendar and GregorianCalendar classes. Your program receives the month and year from the command line. For example:

```
Java DisplayCalendar 5 1999
```

You also can run the program without the year. In this case, the year is the current year. If you run the program without specifying a month and a year, the month is the current month.

It should look something like this:

March 2004						
Mo	Tu	we	th	fr	sa	su
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

```
>java DisplayCalendar
```

Skal ikke løses grafiskt!

9. Fjernkontroll

Skriv et program som simulerer en video fjernkontrolls interaksjon med en tv. Programmet skal simulere forskjellig vanlige funksjoner som avspilling, stillbilde, opptak og spoling. Fjernkontrollen skal være i et eget vindu (prøv å få layouten til å likne en ordentlig fjernkontroll) og tv-en i et annet vindu. I tv-bildet skal det komme opp hva fjernkontrollen driver med, for eksempel at det står FF når du ”spoler” fort frem eller REC når det ”tas opp” – det er opp til deg hvordan du velger å løse denne biten.

10. Bomring I Uqbar

I Uqbar skal det bygges en bomring rundt byen. Det finnes to slags bomringabonnementer:

Årsabonnement: Fri passering i ett år.

Klippekortabonnement: Gjelder for et visst antall passeringer.

Hver abonnent har et entydig abonnementsnummer, som er et sekssifret heltall (verdi mellom 100000 og 999999).

Informasjon om abonnentene finnes på en fil med følgende format:

- For hvert årsabonnement finnes en linje

```
A <abonnementsnr> <totaltAntallPasseringer> <faktureringsMnd>
```

der A angir at dette er et årsabonnement, totaltAntallPasseringer er et heltall som sier hvor mange bompasseringer som totalt er foretatt av bilen med dette abonnementet siden det ble opprettet, og faktureringsMnd er nummeret på den måneden årsabonnementet må fornyes.

- For hvert klippekortabonnement finnes en linje

```
K <abonnementsnr> <totaltAntallPasseringer> <klippIgjen>
```

der K angir at dette er et klippekort, totaltAntallPasseringer er som for årsabonnement, og klippIgjen er antall ubrukte klipp

PROGRAMMET

Lag et program som først leser abonnementsfilen, og så bygger opp en passende datastruktur. Resten av programmet skal være ordrestyrt, og kunne utføre følgende ordre:

Les passeringsfil

Leser en fil med passeringsdata for en dag, og oppdaterer datastrukturen. Hver linje har følgende format:

```
<abonnementsnr> <antallPasseringer>
```

For klippekort: Husk å redusere antall klipp igjen.

Vis abonnement

Skal be om et abonnementsnummer, og dersom nummeret finnes vise en oversikt over:

totalt antall bompasseringer av den aktuelle bilen

abonnementstype

avhengig av abonnementstype: faktureringsmåned eller antall klipp igjen

Skriv fakturaliste

Skal først spørre om et månedsnummer, og deretter lage en liste over abonnementsnumrene til de årsabonnentene som skal faktureres den gitte måneden, samt de klippekortabonnementene som har færre enn 10 ubrukte klipp igjen.

Beregn gjennomsnitt

Skal beregne og skrive hvor mange bompasseringer hver bil med årsabonnement og hver bil med klippekortabonnement gjennomsnittlig har foretatt. (Programmet skal altså finne og skrive en gjennomsnittsverdi for årsabonnementene og en gjennomsnittsverdi for klippekortabonntene.) I tillegg skal programmet finne og skrive hvor mange ubrukte klipp hver klippekortabonnt gjennomsnittlig har igjen.

11. Nice virus

Lag et program som gjør at det hopper en sau langs bunnen av skjermen. Når musepekeren kommer over sauene og man klikker på den ”fanges den” og den blir borte for alltid.

12. Stoppeklokke

Lag en stoppeklokke, med grafisk brukergrensesnitt, som viser timer, min og sekunder har en start, en stopp og en nullstill knapp. Når du trykker stopp, vises tiden; du kan nå trykke start for å fortsette tidtakingen eller trykke nullstill for å nulle ut klokken.

13. Animation 1

Write a program to load and display a sequence of images from a number of different files, with a 100 millisecond wait between the displays of each of the images.

In the folder <http://folk.uio.no/tanjag/Eksperiment/java/> there are folders with gif images you can use (doggy, tumble and example).

14. Battleships

This is a game normally played by two people using paper and pencil. In this version, a person plays the computer. The computer also records and displays the status of the game. When it is the computer's turn to play, it always plays completely randomly. The game is also slightly simplified.

Two 10 X 10 grids are shown on the screen. The home grid represents an ocean showing where your battleships are. The enemy grid shows where you have fired a shot, but you cannot see the enemy. You don't know where the computer has placed its ships, and the computer doesn't know where yours are. Initially you place 10 battleships somewhere on your grid (all the ships are only 1X1 in size). You do this by clicking the mouse on the squares that you want. The computer also places its own 10 ships somewhere hidden on the target grid. The computer places its ships randomly.

The computer determines randomly who goes first. The computer and you then play in turn.

You “fire” a shot at the enemy (the computer), by clicking on a square on the target grid.

The computer displays the position on the grid, so that you can see where you have fired your shots.

The computer displays whether or not one of its ships has been sunk.

Then it is the computer's turn to fire at you. Although it holds the data on where your ships are it does not use this data in choosing (randomly) where to fire at you. But the computer remembers where it has already fired.

Play continues until on player sinks all the enemy's ships.

15. Pizza applet

Design an order-form applet for a pizzeria. The user makes choices, and the applet displays the price.

The user can choose a pizza size of small (\$7), medium (\$9), large (\$11), or extra large (\$14), and any number of toppings. There is no additional charge for cheese, but all other toppings add \$1 each to the base price. You can choose the toppings that are available, but you must offer at least five different toppings. Examples of toppings are jalapeños, olives, anchovies, mushrooms, tomato, shrimp, and beef. Your applet can use any appropriate components. The applet name is Pizza.

16. Juke box

Lag et program som velger en audio fil ved å bruke en "file dialog box". Ha i vertfall tre knapper som skal kontrollere lyden – "play", "stop" og "loop". Hvis du klikker "play"-knappen spilles audiofilen en gang. Hvis du klikker "loop"-knappen, spilles filen om og om igjen. "Stopp"-knappen gjør at avspillingen stoppes.

Det er lydfiler i "audio" mappen på <http://folk.uio.no/tanjag/Eksperiment/java/>.

17. Teddybjørnspillet

Spillet starter med at du får noen bjørner. Du kan så gi tilbake noen bjørner, men du må følge disse reglene (hvor n er antallet bjørner som du har):

Hvis n er et partall kan du gi tilbake nøyaktig $n/2$ bjørner.

Hvis n er delelig med 3 eller 4, kan du gange de siste to sifrene i n og gi tilbake så mange bjørner. (Det siste sifferet i n er $n\%10$, og det nest siste sifferet er $(n\%100)/10$.)

Hvis n er delelig med 5, kan du gi tilbake nøyaktig 42 bjørner.

Målet med spillet er å ende opp med nøyaktig 42 bjørner.

Eksempel:

Anta at du starter med 250 bjørner. Du kan da gjøre følgende skritt:

Siden 250 er delelig med 5 kan du returnere 42 bjørner. Du har da 208 igjen.

Siden 208 er et partall kan du gi tilbake halvparten, og sitte igjen med 104 bjørner.

Siden 104 er et partall kan du gi tilbake halvparten, og ha igjen 52 bjørner.

Siden 52 er delelig med 4 kan du gange de siste to sifrene (som blir 10), og returnere disse 10 bjørnene.

Du har da igjen 42 bjørner.

Du har nådd målet!

Skriv en rekursiv metode

```
boolean teddy(int n)
```

som returnerer true hvis og bare hvis det er mulig å vinne et teddybjørnspill som starter med n bjørner.

Eksempler:

teddy(250) er true (som vist over)

teddy(42) er true

teddy(84) er true

teddy(53) er false

teddy(41) er false

18. Text animation

Write a program to make some text glide around the screen randomly. Make the text bounce when it encounters the boundary of the window it's in.

Appendix F

-Journal Article

The article “Assessing Uncertainty of Software Development Effort Estimates: The Learning From Outcome Feedback” Submitted to journal of Information and Software Technology, 2005

Assessing Uncertainty of Software Development Effort Estimates: The Learning From Outcome Feedback

Tanja Gruschke¹, Magne Jørgensen²
(1) University of Oslo, (2) Simula Research Laboratory
tanjag@student.matnat.uio.no, magne.jorgensen@simula.no

Abstract

To enable properly sized software project budgets and plans it is important to be able to assess the uncertainty of the estimates of most likely effort required to complete the project. Previous studies show that software professionals tend to be too optimistic about the uncertainty of their effort estimates. This paper reports results on how much, and how, software developers improve their assessments of the uncertainty of their effort estimates when receiving repeated outcome feedback, i.e., feedback about the discrepancy between the estimated most likely effort and the actual effort. We found that a necessary condition for improvement of uncertainty assessments of effort estimates may be the use of explicitly formulated uncertainty assessment strategies. By contrast, intuition-based uncertainty assessment strategies may lead to no or little learning.

1. Introduction

Surveys of software development project effort estimates report that effort estimates are frequently inaccurate. A recent review of software estimation accuracy surveys [1] suggests that the average effort overrun of software projects is 30-40% and that the accuracy of estimates has not improved over the last 10-20 years. Important reasons for the lack of accurate effort estimates were formulated by Alfred M. Pietrasanta at IBM Systems Research Institute as early as 1968: “*Anyone who expects a quick and easy solution to the multi-faceted problem of resource estimation is going to be disappointed. The reason is clear: computer program system development is a complex process; the process itself is poorly understood by its practitioners; the phases and functions which comprise the process are influenced by dozens of ill-defined variables; most of the activities within the process are still primarily human rather than mechanical, and therefore prone to all the subjective factors which affect human performance*” [2]. As we see it, the problems reported by Pietrasanta are just as valid today as in 1968. Some of the problems seem to be inherent in the estimation of software development effort, which suggests that we should expect a high level of effort estimation inaccuracy in future software development projects, i.e., that there is no reason to believe that the “estimation problem” will be solved any time soon.

A consequence of an expected high level of estimation inaccuracy is that project plans, bids and budgets cannot be based on the effort estimate alone. They must be based on a combination of both the estimates of most likely effort or cost, and knowledge about the likely level of inaccuracies of the estimates. That is to say, knowledge about the *uncertainties* of the effort or cost estimates is required. Assume that a project leader estimates that the most likely cost of a project is \$ 100 000. The project leader also recognizes that the estimate of most likely cost is quite uncertain and that it is possible that a use of substantially more resources will be required. He therefore decides to base the budget on estimated most likely cost and a contingency buffer. The contingency buffer is included to increase the likelihood that the budget will not be overrun and the use of it is in accordance with good project management practice [3]. Continuing on our example, the project leader may want to know how likely it is that the project will cost less than \$ 150 000, contingency buffer included. The challenge is then to assess the probability that the project will cost more than \$ 150 000. Not surprisingly, software professionals find such assessments difficult [4]. Unfortunately, as far as we know, there are no alternative proper methods for calculating the size of effort or cost contingency buffers in software projects. In our experience, one cost contingency calculation method frequently applied by the software industry is to set the contingency buffer to a fixed proportion of the project’s estimate of the most likely cost, e.g., to set the contingency buffer, as a rule, as 25% of the most likely cost. This practice is not optimal and leads, in highly uncertain projects, to contingency buffers that are too small.

This paper reports on a study on how, and how much, software developers improve their assessments of effort estimation uncertainty (effort prediction intervals) on the basis of typical “on-the-job” feedback, i.e., from a repeated comparison of assessed uncertainty of estimated most likely effort and actual use of effort. For example, we study the learning strategies in situations where the estimator receives feedback that implies that the estimates are systematically more inaccurate than he assessed them to be. The experiments on learning from outcome feedback are conducted in a learning friendly environment, e.g., many small tasks of similar type, well-defined specifications and with immediate feedback. If learning is absent in such environments, we should not expect much learning in more realistic environments, e.g., large, ill-defined projects with feedback several months from the assessment itself.

The paper is organized as follows: Section 2 briefly describes the terminology related to uncertainty that was used to report the study. Section 3 describes related work motivating our study. Section 4 describes the design of the study. Section 5 reports the results of the study. Section 6 discusses the results. Section 7 concludes.

2. Terminology

The terminology used in contexts of software effort estimation and uncertainty assessment can be confusing. The most important terms are interpreted as follows in this paper:

Effort estimate: Forecast (predictions) of expected effort. Without any further description, the precise meaning of this term may be unclear, e.g., whether ‘estimate’ means the ‘modal’ (‘most likely’), the ‘median’, or, the ‘mean’ value of a distribution of possible effort usage [5]. We therefore try to avoid this term when we need to be precise, e.g., we use the term ‘*estimate of most likely effort*’ when the modal value of the distribution of possible effort usage is meant.

Estimate of most likely effort: The effort value believed to have the greatest chance of being equal (or close to) the actual effort.

Effort Uncertainty: A description of the expected uncertainty in use of effort. The type of description of uncertainty applied in this paper is based on *effort prediction intervals*.

Effort prediction interval: A minimum-maximum interval for effort, with a connected confidence level of including the actual effort value. For example, an estimator may estimate the most likely effort to be 1 000 work-hours and the probability of including the actual effort in the effort interval from 600 to 1 500 work-hours to be 90%. Then, the 90% confidence effort prediction interval is [800; 1 500] work-hours. Effort prediction intervals are used frequently in the planning and budgeting of software projects [4].

Estimation outcome feedback: Information about the discrepancy, if any, between the actual effort (the outcome) and the estimated most likely effort. The information about this discrepancy can be used to improve the accuracy of the assessed level of effort prediction intervals. Estimation outcome feedback is frequently the only type of feedback received in software projects, i.e., there is typically no systematic investigation of reasons for higher or lower uncertainty.

3. Motivation of study and study design

In previous studies [4, 6] we found that software professionals strongly underestimated the uncertainty connected with software development effort estimates. Other studies of software developers, e.g. [7], report similar results. For example, when project leaders are 90% sure that the actual development effort would be included by a minimum-maximum effort interval, the typical inclusion rate (“hit rate”) is only 60-70%. Similar levels of over-confidence in estimation accuracy have been documented in other domains [8-11], so underestimation of uncertainty is not limited to software development effort estimation.

Several studies [12, 13] report poor estimation learning from the results of previous estimates. This lack of improvement of estimation skills as a result of on-the-job experience seems to be present in most domains, according to Hammond [14, p. 278]: “*Yet in nearly every study of experts carried out within the judgment and decision-making approach, experience has been shown to be unrelated to the empirical accuracy of expert judgments*”. Two frequently reported reasons for the fact that estimators are poor at learning from their estimation experience are lack of relevant feedback and lack of immediate feedback [15, 16], i.e., it is believed that learning may improve as better feedback is provided. The level of program development skills seems to be a poor indicator of ability to assess realistically the uncertainty of estimates of most likely effort. In [17], for example, we found that: “*The level of over-confidence was higher in situations where at least one of the team members assessed his/her knowledge to be very high....*” Similar results are reported in [18]. In other words, higher development skill may in some situations result in greater over-confidence. For a comprehensive review of studies on software estimation uncertainty assessments see [19]. As a result of the lack of a correlation between amount of experience in making effort estimations and the ability to assess realistically the uncertainty of effort estimates, (i.e. poor learning from experience) there is a need for better understanding of the conditions for learning from experience in the context of software effort estimations. The study reported in this paper is a step towards that goal.

For the purpose of better understanding of reasons for poor learning and how to improve it, we decided to investigate the relations between uncertainty assessment strategy, feedback, and learning in a software development task-solving context with rather favourable learning conditions. The *relevance* of the feedback was ensured by similarity of the tasks to be solved. The *timeliness* of the feedback was ensured by providing the feedback immediately after the completion of a task and just before the uncertainty of the effort estimation of a new task was provided. The learning bias towards “*hindsight bias*” was reduced by the short duration (less than 5 hours) of the task. An important rationale for these design decisions was that if there was little learning in a situation that was designed specifically to enable learning from feedback, we should not expect even less learning in learning-unfriendly (more realistic) situations. That is to say, we studied necessary, and not sufficient, conditions for learning.

4. Design of study

4.1. Research questions

The two main research questions of this study are:

RQ1: How much do programmers improve their assessments of the uncertainty of estimates of most likely effort on the basis of outcome-related feedback?

RQ2: What is the relation between the learning strategies for improving uncertainty assessments used by the programmers and their ability to learn from feedback?

4.2. Measures

There are no standard measures of uncertainty assessment performance. In an earlier paper [6] we argued that we should differentiate between people's ability to assess the *average level of uncertainty* of a set of tasks and the *relative difference* in uncertainty between different tasks. For the purpose of the study reported in this paper we apply the following definitions and measures:

T = A set of n development tasks

$ActEff_j$ = Actual effort required to complete Task j

$EstML_j$ = Estimated most likely effort of Task j

MRE_j = Magnitude of relative estimation error of task j
 $= |ActEff_j - EstML_j| / ActEff_j$

$Int1_j$ = [90% of $EstML_j$; 110% of $EstML_j$]

$Int2_j$ = [60% of $EstML_j$; 150% of $EstML_j$]

$Int3_j$ = [50% of $EstML_j$; 200% of $EstML_j$]

The widths, i.e., the percentages, were chosen to reflect a narrow effort interval ($Int1$), a medium-wide effort interval ($Int2$), and a wide effort interval ($Int3$). We applied more than one interval to enable analyses of possible differences in learning effects related to width of interval.

$Conf1(Int1_j)$ = The developer's assessed probability (confidence) of including $ActEff_j$ in $Int1_j$

$Conf2(Int2_j)$ = The developer's assessed probability (confidence) of including $ActEff_j$ in $Int2_j$

$Conf3(Int3_j)$ = The developer's assessed probability (confidence) of including $ActEff_j$ in $Int3_j$

$AvConfLev1(T)$ = Average value of $Conf1(Int1_j)$ for tasks $j=1..n$

$AvConfLev2(T)$ = Average value of $Conf2(Int2_j)$ for tasks $j=1..n$

$AvConfLev3(T)$ = Average value of $Conf3(Int3_j)$ for tasks $j=1..n$

$HitRateInt1(T)$ = Proportion of $Int1_j$ -intervals that includes $ActEff_j$ for tasks $j=1..n$

$HitRateInt2(T)$ = Proportion of $Int2_j$ -intervals that includes $ActEff_j$ for tasks $j=1..n$

$HitRateInt3(T)$ = Proportion of $Int3_j$ -intervals that includes $ActEff_j$ for tasks $j=1..n$

Applying these definitions we define the ability to assess the average level of uncertainty as:

$Overconfidence(Int1, T) = AvConfLev1(T) - HitRateInt1(T)$

$Overconfidence(Int2, T) = AvConfLev2(T) - HitRateInt2(T)$

$Overconfidence(Int3, T) = AvConfLev3(T) - HitRateInt3(T)$

We have termed the measure 'Overconfidence', because a positive value indicates overconfidence in the accuracy of the estimate of most likely effort. Consider the following example: Assume that an estimator estimates and assesses the estimation uncertainty of a set of tasks (T). On average, the estimator believes that there is a 50% chance of including the actual effort in $Int1$ for the set of tasks $1..n$. The estimator's average confidence level ($AvConfLev1(T)$) is then 50%. The proportion of actual effort values included in $Int1$ is, on the other hand, only 30%, i.e., the $HitRateInt1(T)$ is 30%. Then the level of overconfidence is calculated as the difference between average confidence and inclusion rate of $Int1$ of the set of tasks in T , i.e., $Overconfidence(Int1, T) = 50\% - 30\% = 20\%$.

Our measures of ability to assess relative difference of uncertainty between different tasks are defined as follows:

$RelUncAbility(Int1, T) = \text{correlation between } Conf1(Int1_j) \text{ and } MRE_j, \text{ for } j=1..n$

$RelUncAbility(Int2, T) = \text{correlation between } Conf2(Int2_j) \text{ and } MRE_j, \text{ for } j=1..n$

$RelUncAbility(Int3, T) = \text{correlation between } Conf3(Int3_j) \text{ and } MRE_j, \text{ for } j=1..n$

These measures are based on the assumption that there should be a correlation between confidence in the accuracy of the estimate of most likely effort (Conf) and the estimation error (MRE). When the confidence in the accuracy of the estimate of most likely effort is low, we would expect a high MRE, i.e., we expect these measures to give high negative values if the estimator is skilled at assessing the relative difference in effort estimation uncertainty between different tasks.

4.3. Subjects, tasks, and, material

At the University of Oslo we advertised for highly skilled Java programmers and selected the five programmers whom we believed to be the best Java-programmers, based on examination of their CVs and interviews. All participants had programmed several thousands lines of code in Java and had some industrial programming experience.

The programming tasks given in the experiment were small, but typical for programming tasks solved by student programmers, i.e., the participants can be considered experts on this type of tasks. There were 18 tasks in all. The tasks were taken from lower grade courses at the department of Informatics at the University of Oslo, beginners' textbooks on Java, and Java Sun's home pages. Most tasks required GUI or other types of visual solutions, others were text-based. The tasks were similar in type and complexity, but there were no inter-dependencies between them. The sequence of tasks was similar for all participants (see Section 4.4 for a more detailed description) and there was no obvious increase or decrease in complexity of the tasks. An example of a task is the following:

Task description for "Battleships": *This is a game normally played by two people using paper and pencil. In this version, a person ("you") plays the computer. The computer also records and displays the status of the game. When it is the computer's turn to play, it always plays completely randomly. The game is also slightly simplified.*

Two 10 X 10 grids are shown on the screen. The home grid represents an ocean that shows where your battleships are. The enemy grid shows where you have fired a shot, but you cannot see the enemy. You do not know where the computer has placed its ships, and the computer does not know where yours are.

Initially you place 10 battleships (all of which are only 1X1 in size), somewhere on your grid. You do this by clicking the mouse on the squares that you want. The computer also places its own 10 ships somewhere hidden on the target grid. The computer places its ships randomly. The computer determines randomly who goes first. You and the computer then play in turn. You "fire" a shot at the enemy (the computer), by clicking on a square on the target grid. The computer displays the position on the grid, so that you can see where you have fired your shots. The computer displays whether or not one of its ships has been sunk. Then it is the computer's turn to fire at you. Although it holds the data on where your ships are it does not use this data when choosing (randomly) where to fire at you. However, the computer remembers where it has already fired. Play continues until one player sinks all the enemy's ships.

The experiment was performed in one of the Department of Informatics' computer labs. The lab has work stations with UNIX and a Java programming environment. All the participants had their own student IT user account and were familiar with the computer lab and programming environment. The programming was done using a text editor (most of them used "Emacs") that the participants had customized to their own needs. The specifications of all the other programming tasks are available by request to the authors.

4.4. The experiment process

The experiment took place over a period of about two weeks. Each participant participated for five work-days. On the first day of the experiment all participants were informed about the experiment and it was emphasized that an important purpose of the experiment was to study how well they were able to improve their effort uncertainty assessments. The participants had received the experiment instructions a couple of days before. The programming tasks, i.e., the requirement specifications, were handed out one at a time. When a programming task was handed out the participant read the text, briefly analyzed the problem, and estimated the most likely effort (EstML). The estimated most likely effort was given as input to a web page, which then calculated the effort intervals Int1 ([90%;110%] of EstML), Int2 ([60%;150%] of EstML), and Int3 ([50%;200%] of EstML) in work-hours. We then asked the developers to assess the probability of including the actual effort (ActEff) in each of the effort intervals Int1, Int2 and Int3. Then, the participants solved the task and registered the actual effort spent.

When working on a task the participants could take as many breaks as they liked, but the breaks were not included in the actual effort. When they believed that they had finished a task, the person in charge of the experiment (one of the authors) would decide whether the program qualified as an adequate solution by testing it. If it was not adequate or had bugs, the participants were asked to change or correct the program. Once the task had been completed satisfactorily, the participants commented on their estimation and uncertainty assessment performance based on a comparison of the actual effort with their estimated most likely effort and uncertainty assessments. When the task and the comments about the estimation and uncertainty assessments were completed, a new task was handed out by the person responsible for the experiment. For practical reasons, i.e., to ensure that a task would be completed within a working day, the sequence of the tasks was slightly different among the participants. This makes the comparison of

estimation performance between the subjects more difficult, but we do not believe that it affected to any great extent our analysis of the ability to learn to make uncertainty assessments and of learning strategies. The software developers solved between 14 and 18 tasks.

When all tasks were completed we interviewed the participants about their learning strategies and other issues relevant for understanding the results.

5. Results

5.1. Development and estimation skill

As described in Section 4, the number and sequence of tasks completed by the developers varied slightly for practical reasons. Hence, to compare the development and estimation performance of the developers, we selected a subset of tasks that was completed by all developers (12 tasks) in almost the same sequence. Table 1 shows the total effort (TotEff) in work-hours needed to complete the tasks in work-hours, the median estimation error in % of actual effort (Median MRE), and the average time in minutes spent estimating a task (AvEstTime).

Table 1. Comparison of skill on the 12 tasks completed by all developers

	A	B	C	D	E
TotEff	16:48	19:36	27:08	22:49	24:49
Median MRE	34 %	18 %	35 %	37 %	22 %
AvEstTime	6,8	5,3	4,8	5,4	n.a.

Table 1 indicates that the developers were similar in development skill, i.e., the total effort (TotEff) spent on the 12 tasks were similar. This suggests that any differences in uncertainty assessments are probably not caused by large differences in development skill. There were some differences regarding estimation skill measured as median MRE, e.g., developers B and E had better estimation accuracy. Interestingly, see Section 5.2, developers B and E were the two developers who showed the least improvement in their uncertainty assessment performance. There are no large differences in average time spent on deriving the estimates of most likely effort and the uncertainty assessments of that estimate. From the interviews we found that all developers applied expert judgment, sometimes supported by decomposition of the task into sub-tasks, to estimate the most likely use of effort. This is the estimation method most commonly applied in the software industry as well [20].

5.2. Ability to assess relative difference in uncertainty

Table 2 shows the correlation between confidence level and estimation accuracy for all developers and all tasks completed by the developers, applying the RecUncAbility measure. Notice that there should be a strong negative correlation if the developers are good at distinguishing between high and low effort uncertainty tasks. A positive correlation means that it is more typical that a high uncertainty task is considered to be a low uncertainty task, and vice versa

Table 2. Correlation between confidence and MRE (all tasks)

Developer	Int1	Int2	Int3
A	0.19	0.24	0.20
B	0.11	-0.07	-0.31
C	-0.33	-0.21	0.18
D	0.16	0.11	n.a.*
E	0.48	0.42	0.21

* All confidence levels were 100%

As can be seen from Table 2, most of the correlations are low and/or positive, which suggests a poor ability to assess the relative difference in uncertainty between the tasks. The only developers who were able, to some extent, to separate high uncertainty from low uncertainty tasks may be developers B and C. A possible reason for this poor ability to separate high and low uncertainty estimates may be that our tasks were relatively similar. In an industrial study of 70 real-life software projects [6], i.e., a situation with more heterogeneous tasks, we found (for Int3) a correlation of -0.26 between confidence level and estimation error. To see if there was any learning from experience

we calculated the correlation for the ten last tasks only (see Table 3). We expect that if there had been substantial learning, there would have been more negative correlations and higher negative values.

Table 3. Correlation between confidence and MRE (last 10 tasks)

Developer	Int1	Int2	Int3
A	0.04	-0.10	0.11
B	-0.25	-0.02	-0.45
C	0.11	-0.11	0.25
D	-0.17	-0.05	n.a.*
E	0.47	0.49	0.19

* All confidence levels where 100%

The data in Table 3 suggest that some of the developers, i.e., developers A, B, and D, did improve their performance, but not by very much. The improvements may also be due to random variation. Overall, we found that the developers were poorly skilled at separating high and low uncertainty task and that the skill improved only slightly, at best.

Interestingly, the correlation between the time spent on the estimation and the uncertainty assessment of a task and the estimation error (MRE) was better, or just as good as, the correlations in Table 1 and 2. This means that the variance in time spent on the estimation work provided just as good an indicator of the variance of the uncertainty of use of effort as did the developers' uncertainty assessments themselves. This further supports the poor ability of the developers to assess the relative difference in effort uncertainty between tasks in the previous analysis.

5.3. The ability to assess the average level of uncertainty

For the purpose of analyzing the ability to assess uncertainty and the improvement from outcome feedback, we created a "learning graph" for all five developers. The learning graph shows how well the developers were able to use the outcome feedback to adjust their assessment of the average level of effort uncertainty to the real uncertainty. The learning graph was derived as follows:

- 1) Compare assessed uncertainty (confidence) and actual uncertainty (hit rate) for the first five tasks (Comparison Point 1). The degree of overconfidence at Comparison Point 1 is calculated as $Overconfidence(Int1, T)$, $Overconfidence(Int2, T)$, and, $Overconfidence(Int3, T)$, for $T = \{Task1 \dots Task5\}$.
- 2) The comparison at Comparison Point 2 is conducted similarly, but now for $T = \{Task2 \dots Task6\}$.
- 3) Etc.

The rationale for including only the five last tasks in the set of tasks (T) is that it turned out to be a useful number of tasks to study the learning effect. Including fewer tasks would lead to more random variation in the learning curve due to difference in task complexity and including more tasks may have hidden some of the learning progress information. Figure 1-5 depicts the "learning graphs" for the developers A, B, C, D and E.

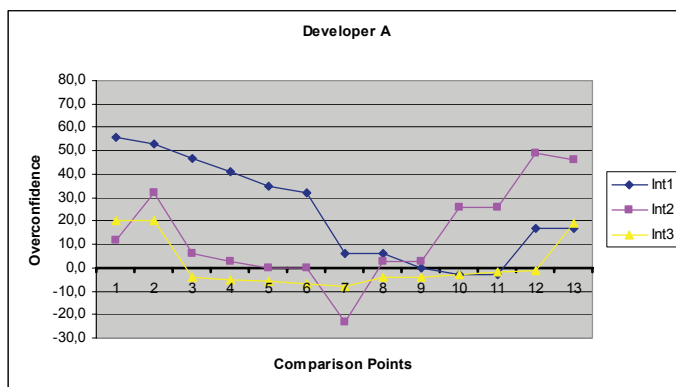


Figure 1. Learning graph of developer A

Initially, developer A was strongly overconfident, especially with respect to Int1 (the narrow effort interval), but then improved for both Int1 and Int3 (the wide interval) and soon reached a good accordance between assessed and actual effort uncertainty, i.e., close to 0% overconfidence. The uncertainty assessments related to Int2 (the medium wide interval) went from overconfident, to underconfident, and then back to overconfident.

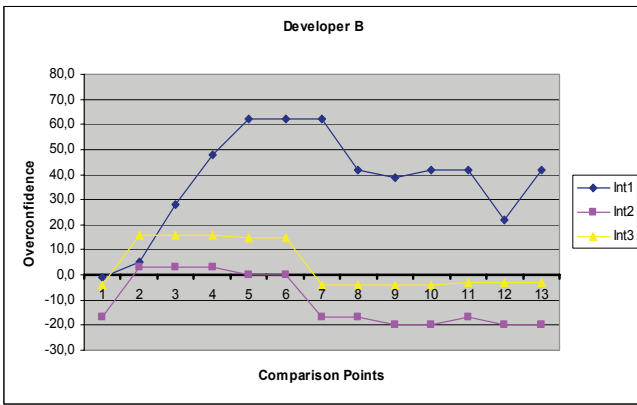


Figure 2. Learning graph of developer B

Developer B had realistic assessments of the probability of including the actual effort in Int2 and Int3 for all comparison points. The developer went, however, from realistic assessment to overconfidence regarding the Int1 effort intervals and there was not much improvement. Notice that developer B had the most accurate effort estimates and had a much higher proportion of actual effort values inside the Int2 interval compared with most of the other developers.

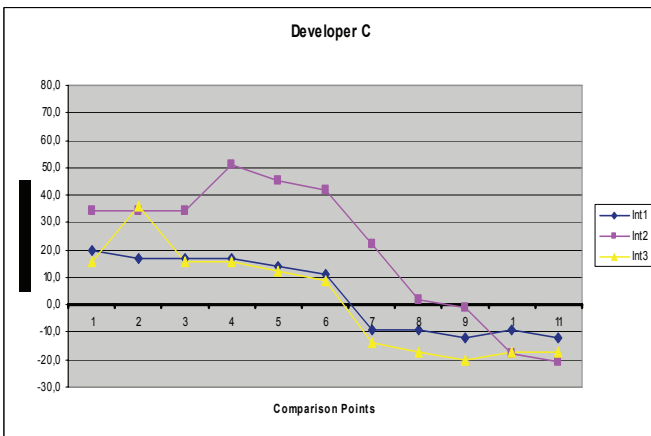


Figure 3. Learning graph of developer C

Developer C went from overconfidence to underconfidence for both Int1, Int2, and Int3. The level of underconfidence is, however, not large and there are clear signs of learning for all intervals.

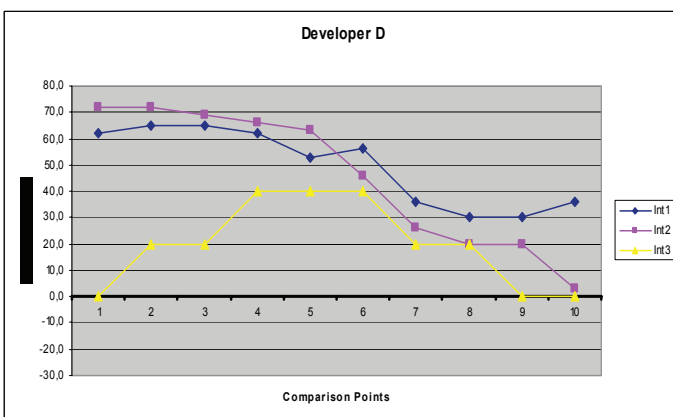


Figure 4. Learning graph of developer D

Developer D started with a high degree of overconfidence of Int1 and Int2, then improved slowly (except for Int3). Developer D had a good correspondence between assessed and actual uncertainty for Int2 and Int3 at the end. Even after completion of all tasks (15 tasks) there was a strong level of overconfidence regarding Int1. However, there are clear signs of (slow) learning.

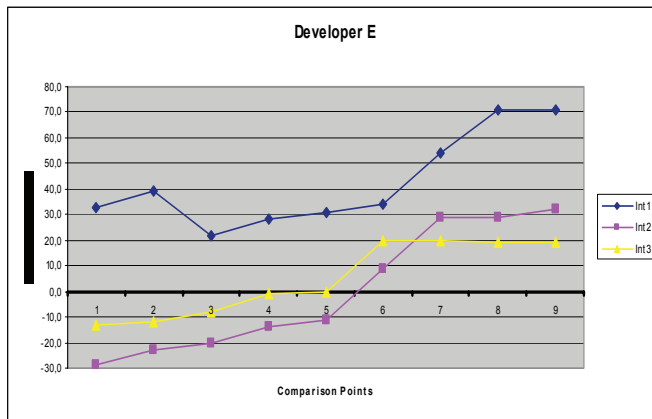


Figure 5. Learning graph of developer E

Developer E went from underconfidence related to Int2 and Int3 to overconfidence. The level of overconfidence related to Int1 increase and at the end is as high as 70%. There are no clear signs of learning.

The developers' ability to learn to assess the uncertainty of the development tasks seems to vary a lot. The performance of developers A, C, and D suggest a learning from experience, while there was little sign of learning for developers B and E.

5.4. The uncertainty assessment strategies

The strategies for uncertainty assessments and learning described in this section are based on (i) the comments provided by the developers after each task completion, (ii) the interviews with the developers when all tasks were completed, and (iii) a comparison of the comments and interview information with the actual performance. The amount of verbal information and analysis is high and the length restriction on this paper allows only a brief summary of selected topics. The analyses were conducted independently by the two authors of this paper. Only a few minor disagreements needed to be resolved through subsequent discussion and further analysis, i.e., there was a high degree of similarity of the results of the independent analyses of the information related to the use of uncertainty assessment strategies.

Based on earlier experience, we categorized software development estimation effort uncertainty assessments into three main strategies:

S1 (Hit rate-based strategy): Uncertainty assessments through comparison of previous hit rates with current confidence levels. For example, if only 30% of previous actual effort was inside the Int1 effort interval, then the confidence of the next Int1 effort interval should not deviate too much from 30%.

S2 (Analogy-based strategy): Uncertainty assessment through recall of a small set of similar tasks (typically 1-2 tasks) and use of the estimation error of those tasks to set the confidence levels. For example, if the estimation error of the two most similar tasks was about 30%, it is likely that this level of estimation error will occur on the current task as well.

S3 (Intuition-based strategy): Uncertainty assessment without any explicitly formulated strategy, i.e., an intuition-based uncertainty assessment where mainly the properties of the current task is evaluated.

We found that the strategies indicated by the developers in the task comments corresponded well with the confidence levels chosen. The developers' descriptions of their uncertainty assessment strategies in the interviews, on the other hand, did not always correspond well with the two other information sources. All sources are valid, but both authors, independently, formed the impression that the interview-based descriptions were sometimes strongly affected by how the developers wanted the authors to believe that they had conducted the uncertainty assessments. Table 3 summarizes our categorization of the strategies applied by the different developers, according to information source.

Table 3. Main uncertainty assessment strategies

Developer	Source: Task comments + analyses of chosen confidence levels	Source: Interviews
A	S1	S1
B	S3	S2 + S3
C	S2	S1 + S2
D	S2	S2

E	S3	S1 + S2
---	----	---------

The developers' own description of their use of uncertainty assessment strategy in the interviews matched the strategies we derived from the task comments and the analysis of the chosen confidence levels for developers A and D. For the other developers there were deviations. In particular, developer E described his strategy very differently from that which we observed, e.g., there was no indication of the use of S1 in the learning graph in Figure 6. It is, nevertheless, possible that developer E tried to apply S1, but was unable to apply the feedback properly or used the feedback in a biased manner. However, if that were the case, there should be some evidence of the use of S1 in the task comments. In short, we think that there are reasons to believe that our strategy categorization based on the task comments and analysis of chosen confidence levels is more realistic than the one based on the interviews. The interviews show, nevertheless, that the task of extracting uncertainty assessment strategies is a difficult task and that we may need sophisticated means to increase the validity of the analysis, e.g., the use of think-aloud-protocols.

We also asked the developers about whether they applied mainly analytical or intuitive strategies when making decisions *in general*. Developers A, D, and E perceived themselves as dominantly analytical, while developers B and C perceived themselves as dominantly intuitive decision-makers. Developers A and D stated that they switched to more analytical strategies, because they had experienced that their intuition was frequently very biased. These comments may be interpreted as providing evidence of an ability to be analytic on a meta-level, i.e., to be analytic about choice of decision strategy, which may be a good indicator of learning ability. Developers A and D were two of the developers that displayed learning in our experiment. The third developer that displayed learning was developer C. Developer C perceived himself as an intuition-based decision-maker, but his uncertainty assessment comments were clearly more analytical than those of developers B and E. Developer C stated in the interviews that he was aware of his tendency to be overoptimistic in his intuition-based decisions and tried to compensate for this. Although this does not show that developer C applies analytical strategies, it shows at least a mature level of reflection about his own biases. This suggests another type of meta-learning ability, i.e., the ability to identify and compensate for one's own biases.

In total, our results suggest that it is not possible to provide a simple model that predicts when a developer will be able to improve uncertainty assessments based on outcome feedback. It is nevertheless possible, from our results, to see likely connections between the use of analytical strategies, reflections about properties of their strategies, and good learning ability.

6. Discussion of results

As discussed in Section 3, assessment of uncertainty of software development effort estimates have been found to be a difficult task and there are reasons to believe that task solving experience and outcome feedback are frequently not sufficient for improvement (learning). In our study we designed the task-solving environment and the feedback in a way that we believed would support learning, i.e., the solving of many similar tasks and immediate feedback. Nevertheless, the developers struggled with learning uncertainty assessments from experience and two out of five developers may not have learned at all.

Examining the strategies of the two developers with the poorest learning, we found that their uncertainty assessment strategies were more intuition-based and they reflected less on their strategies. This suggests that the ability to learn depends on the existence of an explicit uncertainty assessment strategy that improves as more data becomes available. Among the developers there may have been two explicit strategies of that type:

- S1: Adjustment of confidence levels based on hit rate of previous (reasonably similar) tasks. As the number of completed tasks increases, the accuracy of the uncertainty assessments improves. This learning strategy is similar to the learning in the formal uncertainty assessment model we propose in [21].
- S2: Recall of the most similar tasks and use of the uncertainty (estimation error) of those tasks to determine the confidence of the current task. As the set of similar tasks increases, the accuracy of the uncertainty assessments improves.

The lack of improvement in uncertainty assessment when applying intuition-based strategies is in accordance with results from other domains, e.g., the finding that intuition-based assessments may be even more over-confident when a greater amount of information is available [9, 22], and that the bias towards over-confidence is difficult to avoid when processes are intuition-based [23].

Notice that our results do not imply that there actually will be learning from experience in typical software development situations when applying the uncertainty assessment strategies S1 or S2. In [24], for example, we found that many professionals estimation teams had a strong tendency to believe that their effort estimates were accurate, even in situations where they knew that the estimation error in similar occasions typically had been very inaccurate. Our results may, however, demonstrate that *necessary* conditions for learning uncertainty assessments are (i) the use of explicit strategies that improves as more data becomes available, and (ii) non-reliance on intuition-based strategies.

When examining the *threats to validity* of our study one should remember that some of the experimental artificiality was introduced intentionally, to examine learning processes in "learning friendly" environments, i.e.,

environments with many similar tasks and immediate feedback. In other words, if we find poor learning in “learning friendly” environments, we should expect even poorer learning in more realistic environments. This is, we believe, a meaningful way of studying necessary conditions for learning, although not for studying sufficient conditions.

We used student programmers in our study. The students were, however, experts on the tasks to be solved, and their use does not constitute a problematic design issue.

The main problem with our study is, in our opinion, the small number of subjects that we analysed. It is possible that the inclusion of more subjects would reveal that there are, indeed, software developers that are able to improve their uncertainty assessments by applying intuition-based strategies. On the other hand, the results in other studies, as summarized earlier in this section, strongly support our results.

7. Summary and further research

Our study aimed at better answers to the following two questions: 1) How much do programmers improve their assessments of the uncertainty of estimates of most likely effort from outcome-related feedback?, and 2) What is the relation between the strategies employed by the programmers to learn how to improve uncertainty assessments and their ability to learn from feedback?

To answer these questions, we conducted a study of effort estimation uncertainty ability and learning of five developers who solved between 14 and 18 small (less than five work-hours) programming tasks, with feedback after each task.

We conclude that some software developers do have the ability to improve their uncertainty assessments in conditions similar to the one in our experiment, i.e., learning-friendly situations. However, not all developers seem to be able to use the outcome feedback to improve their performance. We hypothesize that a *necessary* (but not sufficient) condition for learning uncertainty assessment from experience (outcome feedback) is the use of an *explicit* uncertainty assessment strategy that improves with more feedback.

An implication of our finding is that we cannot expect uncertainty assessments to improve when they are dominantly intuition-based. Our hypothesis is in accordance with the poor effort uncertainty assessment performance and improvement in real software projects [19], and studies from other domains, that show that overconfidence in estimates is difficult to avoid when applying intuition-based strategies [23].

We therefore recommend that software companies instruct software development effort estimators to use explicit uncertainty assessment strategies, such as the strategies S1 and S2 described in Section 5.

We intend to replicate this study with software professionals and larger tasks. In addition, we intend to conduct experiments in which another developer than the one supposed to complete the task assesses the uncertainty of the effort estimates, and experiments in which the developers are instructed to follow a particular uncertainty assessment instruction.

References

1. Moløkken, K. and M. Jørgensen. *A review of software surveys on software effort estimation*. in *International Symposium on Empirical Software Engineering*. 2003. Rome, Italy: Simula Res. Lab. Lysaker Norway: pp. 223-230.
2. Pietrasanta, A.M. *Current methodological research*. in *ACM National Conference (USA)*. 1968: ACM Press, New York, NY, USA: pp. 341-346.
3. Shenhar, A.J., *One size does not fit all projects: Exploring classical contingency domains*. *Management Science*, 2001. **47**(3): pp. 394-414.
4. 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*, 2002. **70**(1-2): pp. 79-93.
5. Jørgensen, M., *How much does a vacation cost? or What is a software cost estimate?* *Software Engineering Notes*, 2003. **28**(6): p. 5-5.
6. Jørgensen, M., *Realism in assessment of effort estimation uncertainty: It matters how you ask*. *IEEE Transactions on Software Engineering*, 2004. **30**(4): pp. 209-217.
7. Connolly, T. and D. Dean, *Decomposed versus holistic estimates of effort required for software writing tasks*. *Management Science*, 1997. **43**(7): pp. 1029-1045.
8. Klayman, J., et al., *Overconfidence: It depends on how, what and whom you ask*. *Organizational Behaviour and Human Decision Processes*, 1999. **79**(3): pp. 216-247.
9. Oskamp, S., *Overconfidence in case-study judgments*. *Journal of consulting psychology*, 1965. **29**(3): pp. 261 - 265.

10. Vallone, R.P., et al., *Overconfident prediction of future actions and outcomes by self and others*. Journal of Personality and Social Psychology, 1990. **58**(4): pp. 582 - 592.
11. Einhorn, H.J. and R.M. Hogarth, *Confidence in judgment: Persistence of the illusion of validity*. Psychological Review, 1978. **85**(5): pp. 395-416.
12. Lichtenstein, S. and B. Fischhoff, *Do those who know more also know more about how much they know?* Organizational Behaviour and Human Decision Processes., 1977. **20**(2): pp. 159-183.
13. Jørgensen, M. and D.I.K. Sjøberg, *Impact of experience on maintenance skills*. Journal of Software Maintenance and Evolution: Research and practice, 2002. **14**(2): pp. 123-146.
14. Hammond, K.R., *Human judgement and social policy: Irreducible uncertainty, inevitable error, unavoidable injustice*. 1996, New York: Oxford University Press.
15. Shepperd, J.A., J.K. Fernandez, and J.A. Quellette, *Abandoning unrealistic optimism: Performance estimates and the temporal proximity of self-relevant feedback*. Journal of Personality and Social Psychology, 1996. **70**(4): pp. 844-855.
16. Bolger, F. and G. Wright, *Assessing the quality of expert judgment: Issues and analysis*. Decision support systems, 1994. **11**(1): pp. 1-24.
17. Jørgensen, M., *Top-down and bottom-up expert estimation of software development effort*. Information and Software Technology, 2004. **46**(1): p. 3-16.
18. Moløkken, K., *Expert estimation of Web-development effort: Individual biases and group processes (Master Thesis)*, in Department of Informatics. 2002, University of Oslo.
19. Jørgensen, M., *Evidence-Based Guidelines for Assessment of Software Development Cost Uncertainty*. Submitted for publications (an early version can be downloaded from www.simula.no/publication.php).
20. Jørgensen, M., *A review of studies on expert estimation of software development effort*. Journal of Systems and Software, 2004. **70**(1-2): pp. 37-60.
21. Jørgensen, M. and D.I.K. Sjøberg, *An effort prediction interval approach based on the empirical distribution of previous estimation accuracy*. Information and Software Technology, 2003. **45**(3): pp. 123-136.
22. Whitecotton, S.M., D.E. Sanders, and K.B. Norris, *Improving predictive accuracy with a combination of human intuition and mechanical decision aids*. Organizational Behaviour and Human Decision Processes, 1998. **76**(3): pp. 325-348.
23. Wilson, T.D. and N. Brekke, *Mental contamination and mental correction: unwanted influences on judgments and evaluation*. Psychological bulletin, 1994. **116**(1): pp. 117-142.
24. Jørgensen, M. and K. Moløkken. *Eliminating Over-Confidence in Software Development Effort Estimates*, in Conference on Product Focused Software Process Improvement (PROFES). 2004. Japan: LNCS 3009, Springer Verlag: pp. 174-184.

References

Sources referred to in the thesis, as well as sources used in other aspects of the thesis work; experiment design input, ideas for analyses etc.

1. Jørgensen, M., K.H. Teigen, and K. Moløkken, *Better sure than safe? Overconfidence in judgement based software development effort prediction intervals*. Journal of Systems and Software, 2004. **70**(1-2): p. 79-93.
2. 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.
3. Lichtenstein, S. and B. Fischhoff, *Do those who know more also know more about how much they know?* Organizational Behaviour and Human Decision Processes., 1977. **20**(2): p. 159-183.
4. Jørgensen, M. and K. Moløkken. *Eliminating Over-Confidence in Software Development Effort Estimates*. in *Conference on Product Focused Software Process Improvement (PROFES)*. 2004. Japan.
5. Jørgensen, M., *How much does a vacation cost? or What is a software cost estimate?* Software Engineering Notes, 2003. **28**(6): p. 5-5.
6. Jørgensen, M., *Top-down and bottom-up expert estimation of software development effort*. Information and Software Technology, 2004. **46**(1): p. 3-16.
7. Jørgensen, M., *Evidence-Based Guidelines for Assessment of Software Development Cost Uncertainty*. 2004.
8. Jørgensen, M., *A review of studies on expert estimation of software development effort*. Journal of Systems and Software, 2004. **70**(1-2): p. 37-60.
9. Jørgensen, M. and D.I.K. Sjøberg, *Impact of experience on maintenance skills*. Journal of Software Maintenance and Evolution: Research and practice, 2002. **14**(2): p. 123-146.
10. Jørgensen, M. and D.I.K. Sjøberg, *An effort prediction interval approach based on the empirical distribution of previous estimation accuracy*. Information and Software Technology, 2003. **45**(3): p. 123-136.

11. Jørgensen, M. and K.H. Teigen. *Uncertainty Intervals versus Interval Uncertainty: An Alternative Method for Eliciting Effort Prediction Intervals in Software Development Projects*. in *International Conference on Project Management (ProMAC)*. 2002. Singapore.
12. Moløkken, K. and M. Jørgensen. *A review of software surveys on software effort estimation*. in *International Symposium on Empirical Software Engineering*. 2003. Rome, Italy: Simula Res. Lab. Lysaker Norway.
13. Pietrasanta, A.M. *Current methodological research*. in *ACM National Conference (USA)*. 1968: ACM Press, New York, NY, USA.
14. Tversky, A. and D. Kahneman, *Judgment under uncertainty: Heuristics and biases*. *Science*, 1974. **185**: p. 1124-1131.
15. Klayman, J., et al., *Overconfidence: It depends on how, what and whom you ask*. *Organizational Behaviour and Human Decision Processes*, 1999. **79**(3): p. 216-247.
16. Fischhoff, B. and W. Bruine De Bruin, *Fifty-fifty = 50%?* *Journal of Behavioral Decision Making*, 1999. **12**(2): p. 149-163.
17. Einhorn, H.J. and R.M. Hogarth, *Confidence in judgment: Persistence of the illusion of validity*. *Psychological Review*, 1978. **85**(5): p. 395-416.
18. Benson, P.G. and D. Önkal, *The effects of feedback and training on the performance of probability forecasters*. *International Journal of Forecasting*, 1992. **8**: p. 559-573.
19. Bolger, F. and D. Önkal-Atay, *The effects of feedback on judgmental interval predictions*. *International Journal of Forecasting*, 2004. **20**(1): p. 29-39.
20. Pliske, R.M., B. Crandall, and G. Klein, *Competance in Weather Forecasting*, in *Psychological Investigations of Competence in Decision Making*, K. Smith, J. Shanteau, and P. Johnson, Editors. 2004, Cambridge University Press: Cambridge. p. 40-68.
21. Johnson, E.J., *Expertise and decision under uncertainty: Performance and process*, in *The nature of expertise*, M.T.H. Chi, R. Glaser, and M.J. Farr, Editors. 1988, Lawrence Erlbaum: Hillsdale, N. J. p. 209-228.
22. Bolger, F. and G. Wright, *Assessing the quality of expert judgment: Issues and analysis*. *Decision Support Systems*, 1994. **11**(1): p. 1-24.
23. Ericsson, K.A., *The acquisition of Expert Performance: An Introduction to Some of the Issues*, in *The road to excellence*, K.A. Ericsson, Editor. 1996, Lawrence Erlbaum Associates, Publishers: Mahwah, New Jersey. p. 1-50.
24. Jørgensen, M. and K. Moløkken, *How Large Are Software Cost Overruns? Critical Comments on the Standish Group's CHAOS Reports*. Submitted to *Communications of the ACM*, 2004, 2004: p. 8.
25. Armstrong, J.S., *Principles of forecasting : a handbook for researchers and practitioners*, ed. J.S. Armstrong. 2001, Boston: Kluwer Academic Publishers.
26. Gruschke, T. and M. Jørgensen, *Assessing Uncertainty of Software Development Effort Estimates: The Learning From Outcome Feedback*.

- Submitted to journal of Information and Software Technology, 2005, 2004.
27. Jørgensen, M., K.H. Teigen, and K. Moløkken, *Better sure than safe? Overconfidence in judgement based software development effort prediction intervals*. Journal of Systems and Software, 2002. **70**(1-2): p. 79-93.
 28. Myers, D.G., *Intuition - its powers and perils*. 2002: Yale University Press.
 29. Juslin, P., A. Winman, and H. Olsson, *Naive Empiricism and Dogmatism in Confidence Research: A Critical Examination of the Hard-Easy Effect*,. Psychological Review, 2000. **107**(2): p. 384-396.
 30. Suantak, L., F. Bolger, and W.R. Ferrell, *The Hard-Easy Effect in Subjective Probability Calibration*. Organizational Behavior and Human Decision Processes, 1996. **67**(2): p. 201-221.
 31. Lichtenstein, S. and B. Fischhoff, *Training for calibration*. Organizational Behavior and Human Performance, 1980. **26**(2): p. 149-171.
 32. Ericsson, K.A., *The road to excellence*. 1996: Lawrence Erlbaum Associates, Publishers.
 33. Sedlmeier, P., *Improving Statistical Reasoning*. 1999: Lawrence Erlbaum Associates.
 34. Plous, S., *The psychology of judgment and decision making*. 1993: McGraw Hill.
 35. Wilson, T.D. and N. Brekke, *Mental Contamination and Mental Correction: Unwanted Influences on Judgments and Evaluations*. Psychological Bulletin, 1994. **116**(1): p. 117-142.
 36. Hammond, K.R., *Human judgement and social policy: Irreducible uncertainty, inevitable error, unavoidable injustice*. 1996, New York: Oxford University Press.
 37. Shanteau, J., *Competence in experts: The role of task characteristics*. Organizational Behaviour and Human Decision Processes, 1992. **53**(2): p. 252-266.
 38. Abrahamsson, P., *Commitment development in software process improvement: critical misconceptions*. Proceedings of the 23rd International Conference on Software Engineering. ICSE, 2001: p. 71-80.
 39. Taylor, S.E. and J.D. Brown, *Illusion and Well-Being: A Social Psychological Perspective on Mental Health*,. Psychological Bulletin, 1988. **103**(2): p. 193-210.
 40. Ericsson, K.A. and A.C. Lehmann, *Evidence of maximal adaptation to task constraints*, in *Expert and exceptional performance*, K.A. Ericsson, Editor. 1996.

