UNIVERSITY OF GOTHENBURG

Evaluation of Face Recognition APIs and Libraries

B.Sc. Software Engineering and Management Thesis

Philip Masek
Magnus Thulin

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
Gothenburg, Sweden, June 2014

Evaluation of Face Recognition APIs and Libraries

P. Masek
M. Thulin
[NAME B. FAMILYNAME,]

Examiner: M. Ericsson

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Gothenburg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Gothenburg, Sweden June 2014

Evaluation of Face Recognition APIs & Libraries

Magnus Thulin
B.Sc. Software Engineering
Gothenburg University
Gothenburg, Sweden
Email: magnus2lin@gmail.com

Philip Masek
B.Sc. Software Engineering
Gothenburg University
Gothenburg, Sweden
Email: philip.masek@gmail.com

***Abstract*—After years of research, the commercialization of face recognition technology is apparent with the emergence of several face recognition libraries and APIs. Organizations and developers are faced with identifying critical success factors when selecting a face-recognition API or library to be used within the development of a product. This study aims to (1) understand which quality characteristics derived from the ISO/IEC 25010 standard are important for an organization adopting the technology and (2) evaluate two client-side libraries and two cloud based APIs according to the quality characteristics identified. Data was extracted by interviewing a company investigating face recognition technology for software-reuse and an experiment was carried out to evaluate the chosen software by extracting metrics from the ISO 9126 standard. An organization adopting face recognition technology prioritised reliability, functional suitability and maintainability as the most important. The experiment concluded the chosen cloud-based APIs were more computationally accurate than the client-side libraries. However, the data collected concluded that the chosen client-side libraries have less failure density than cloud-based APIs.**

I. Introduction

Face recognition is fast becoming a familiar feature across software applications. After years of research, widespread commercialization of the technology is apparent where face recognition is being increasingly integrated into consumer products [1]. The commercialization of the technology has led to the availability of several Application Programming Interfaces (APIs) and software libraries. Cloud based face recognition APIs offer easy to use server-side solutions for monthly subscriptions. Moreover, client-side libraries are available to integrate face recognition technology into custom software products. Face recognition spans several complex disciplines such as pattern recognition and computer vision [2]. Organizations and developers can integrate face recognition software through reuse. However, with the emergence of several face-recognition APIs and libraries, organizations are faced with identifying critical success factors when selecting a face-recognition API or library to be used within the development of a product.

A. Problem Domain & Motivation

To identify critical success factors when adopting software for reuse an evaluation of software quality should be carried out. Software can be evaluated with respect to different aspects such as functionality and usability, namely, quality requirements (QR) or quality characteristics [3]. Quality characteristics may have a number of sub-characteristics, such as learnability is a sub-characteristic of usability. In order to carry out an effective evaluation the most important quality requirements specific to the field will be captured. The most important quality characteristics sub-characteristics will then be assessed by applying software evaluation metrics. This study aims to address the following research questions identified in Table III.

TABLE I
Research Questions

RQ1:	What software quality characteristics for face-recognition libraries and APIs are important to an organisation adopting the technology?
RQ2:	How do face recognition libraries and APIs fulfil the quality requirements in regards to the industrial use case?

B. Research Process & Data Collection

This paper presents the results of an empirical study that includes data collected through in-depth interviews and data resulted by applying an experiment. Personal interviews with various stakeholders at a company were conducted to gain insight as to which quality characteristics the organization prioritize over when adopting face-recognition software. Two open-source libraries and two commercial face-recognition APIs were evaluated in an experiment. Software evaluation metrics were extracted from an ISO standard and applied in an experiment to evaluate how each API and library fulfils one sub-characteristic of quality requirements identified by the case company. The resulting data from the experiment will assist in recommending the case company an API that values the quality requirements defined in Research Question 1.

C. Contribution

There are several methodologies and datasets that exist for evaluating face recognition (e.g FERET [4], FRVT [5], LFW [6]). In current research, the evaluation typically assesses the accuracy of identifying and recognizing faces in different circumstances such as pose, illumination and expression. Since quality requirements play a central role when developing a successful product [7], the study will contribute by presenting a method to evaluate the quality of face-recognition software and present the results of the evaluation.

D. Scope

In this paper we perform a lean software evaluation of face-recognition APIs and libraries. The quality requirements are idenfitied by the industrial use case. This study will evaluate the most popular face-recognition APIs and libraries in current research, as evaluating all of them are out of scope for this paper. Similarly, sub-characteristics of the three most important quality requirements will be assessed. Although an assessment of all the sub-characteristics would be necessary to encompass a complete evaluation of the software, this study aims as a preliminary study for future work.

E. Structure of the Paper

The remainder of the paper is organized as follows: In Section 2 the background and related work is presented. The research methodology is described in Section 3 and the deviations from the experiment plan in Section 4. Section 5 introduces the results and analysis, section 6 the research threats to validity, where the discussion and conclusions are presented in Section 7 and Section 8.

II. Background & Related Work

In this section we outline face-recognition technology, current research on the evaluation of face-recognition software, quality requirements in software engineering and software evaluations metrics.

A. Face Recognition Technology

Overview: The input to a face recognition system is always an image or a video stream. The output is the identification or verification of one or more subjects that appear in the frame. Some approaches define face recognition systems as depicted in Fig. 1 where face detection and feature extraction may run simultaneously.

Face detection is defined as finding the location of a human face within the scene of an image or video stream. Face detection can be used in several application domains such as face tracking, pose estimation and human computing interaction systems. Feature extraction consists of finding relevant facial features from a detected face. The extracted data results in face regions, variations, angles or measure which may or may not be human relevant, such as eye spacing [8]. Feature extraction has other application areas such as gender recognition, emotion recognition and gaze estimation. Face recognition attempts to recognize a detected subject by comparing the detected face against a database of *known* faces. The system can report back the identity of the person if it has been found. The process typically involves a comparison model, a classification algorithm and an accuracy measure [8].

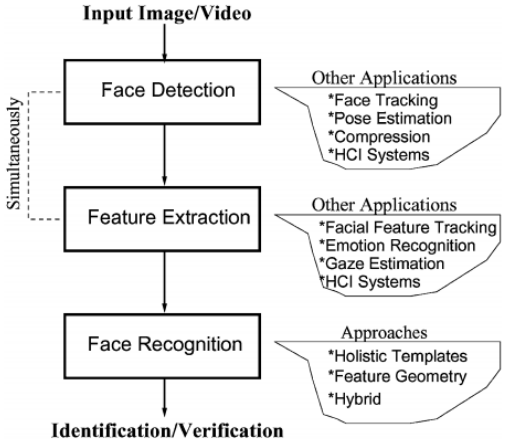

Fig. 1. The configuration of a generic face recognition system [2].

There are many approaches to face recognition. Yang et al. [9] categorise various methods into four categories, however appearance-based methods have been showing superior performance to others methods [10]. Face recognition software typically uses an appearance-based methods which adopt a certain machine learning algorithm that learn the model of a human face. The algorithm is trained by collecting a large set of face and non-face examples. In answering Research Question 2, the evaluation will assess face detection and feature extraction, excluding face recognition, since this is in-line with the industrial use-case described in section 3.

Evaluation in current Research: There are a number of research papers benchmarking and evaluating the accuracy of face recognition software [1], [11], [12], [13], however there are none evaluating the software quality in particular. There is a wide variety of datasets that are used in research to carry out an evaluation, however there lacks a clear training and testing protocol [11]. Without a clear training and testing protocol, it is difficult to carry out exact comparisons of the results derived from various research papers. In similarity to this paper, B. Becker and E. Ortiz [1] evaluate face recognition supplied by three client side libraries, two cloud-based APIs and two consumer applications. The paper concluded that the cloud-based APIs had significantly worse performance. The paper hypothesizes that factors such as the focus on smaller datasets and the ability to train the algorithm with a small amount of images contributed to the poor performance. They further hypothesize that online APIs are designed to be lightweight, prioritizing speed over

accuracy. The client side libraries performed relatively well compared to the online APIs. This may be apparent due to the fact that the client side libraries are capable of being trained.

The evaluation of face recognition in current research assess the functional requirements. Berntsson Svensson et al. argue that non-functional requirements, namely quality requirements, are equally important to functional requirements. Hence, this paper will evaluate the sub-characteristics of quality in regards to available face recognition software.

In research presented in [1], [11], [12] and [13] we extracted commonly evaluated face-recognition software and compiled the findings in Table II. This table was used when selecting the libraries and APIs to be evaluated in answering Research Question two (RQ2).

TABLE II
FACE RECOGNITION LIBRARIES & APIS

Name	Type	Available
OpenCV	Open Source Library	Yes
OpenBR	Open Source Library	Yes
PittPatt	Open Source Library	No
LambdaLabs	Cloud Based API	Yes
ReKognition	Cloud Based API	Yes
SkyBiometry	Cloud Based API	Yes
Face++	Cloud Based API	Yes

B. The Prioritization of Quality Requirements in Industry

In software engineering there are a number of quality models such as ISO/IEC 9126 (revised to ISO/IEC 25010), McCalls, FURPS, Boehms and Dromeys which each have a variety of features, factors and characteristics of quality [14]. Quality models provide a framework to define, predict and assess software quality during development and post development [15]. Comprehensive specification and evaluation of software quality is a key factor in ensuring value to stakeholders [16]. This can be achieved by defining quality characteristics that are associated with stakeholders' goals and objectives for a system.

In a study presented by Berntsson Svensson et al. [7], the researchers investigate the prioritization of quality requirements at 11 companies. The prioritization of the quality characteristics were derived by semi-structured interviews and the quality characteristics were based on Lauesens' comparison of ISO9126 and McCalls quality factors [17].The three most important quality characteristics resulted by the study were usability, performance and reliability. The prioritization of usability and performance are in line with the findings in Berntsson Svensson et al. [18]. However, the results from a similar study in [19] were quite different from the findings in [7] and [18]. B. Philips et al. believe the results may be different due to cultural differences in organizations [19]. B. Philips et al. further suggest that even though clear definitions of quality characteristics are formulated by ISO standards, participants in the interview interpret them differently in their company context [19].

C. Software Quality Metrics

Software metrics and quality models play a critical role in the measurement of software quality. By reviewing the quality models presented by Samadhiya et al. [14] and Berander et al. [20] the ISO 9126 standard supports all of the perspectives of quality and supports both bottom-up and top-down approaches [14]. The ISO/IEC 9126 has been revised to ISO/IEC 25010, however it does incorporate the same software quality characteristics with some amendments [16]. The ISO/IEC 25010 standard includes 8 quality characteristics with corresponding sub-characteristics as displayed in Fig. 2. The quality characteristics were extracted from the ISO/IEC 25010 standard and applied in answering Research Question 1 (RQ1). The subsequent metrics that follow the sub-characteristics were used in the experiment in answering Research Question 2 (RQ2).

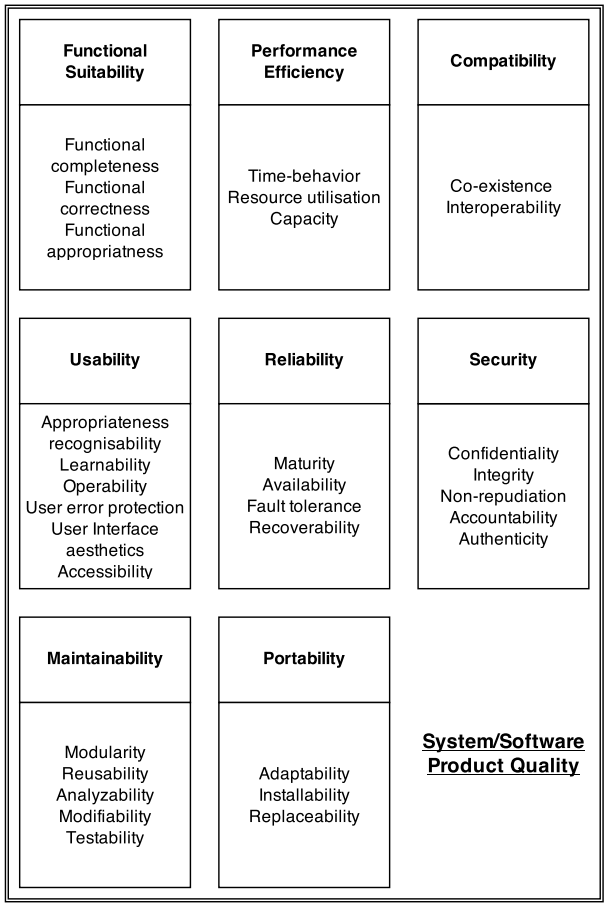

Fig. 2. ISO/IEC 25010 Product Quality Model [16].

III. METHODOLOGY

The study was conducted in cooperation with a company. The company develops and sells software and

hardware equipment for counting people. They are part of a growing business intelligence sector that specializes in visitor traffic information systems for stores, shopping centres and public places. Visitor traffic information systems improve organizational performance by analysing foot-traffic, identifying trends and to use as historical data. The company is the leading provider of visitor traffic information systems in Scandinavia. Outside of Scandinavia, the company has over 11,000 installations in 40 countries. The compnay is investigating face recognition technology to determine the *age* and *gender* of visitors which will introduce an additional level of detail to their existing visitor traffic information system. The company outsources as much software as possible in order to remain focused on their main business objective of counting people. Adopting face-recognition technology through reuse is very attractive to the company, however they are faced with the problem of effectively evaluating APIs in order to make the right decision.

A. Interviews

To answer research question one, a qualitative research approach was carried out, namely, in-depth semi-structured interviews [21]. Qualitative research aims to investigate and understand phenomena [22]. A qualitative research approach is useful when interviewing individuals in order to capture their motivation and understanding of a particular phenomena [22]. Since there was a potential for receiving a wide range of rich and diverse data, a semi structured approach was used. Semi-structured interviews ensure data is collected from predefined topics but also allows the interviewer to probe deeper when required. We chose to do interviews instead of a large scale survey, since quality characteristics can be tailored to any subject domain and the understanding of the quality characteristics can differ amongst individuals. Therefore, we found it important to be present when eliciting data to make it possible to elaborate on the purpose of the study and our objectives. Several times we found we had to take extra time in explaining the quality characteristics extracted from the ISO/IEC 25010 standard, especially when we were faced with interviewing employees from different departments, such as technical and non-technical interviewees. The decision to interview people with different roles was taken to give an overall perspective of the quality prioritization of the company as a full entity, rather than a focused opinion of the technical personnel as suggested by Wong [23]. The researchers did not have an influence on the selection of interviewees, instead the researchers interviewed all of the employees at the case company. Fifteen employees at the company participated in the interview. Two groups derived from the fifteen employees, one group of ten employees from the development department, and the remaining five from the administrative department.

Development team:	Head of development System Architect Back-end developers Font-end developers System administrator Image analyst Face recognition developer

The second group being the administrative team, with people working with customer relations and support.

Administrative team:	CEO/Business Developer Support Manager Sales Manager Customer supports

1) Data collection: Two interviewers and one interviewee attended all of the interviews that were performed on sight at the case company. During the interview, the purpose of the study, the control questions and the $100 Test were presented to the interviewee. The interview was structured in two parts. Firstly, control questions were asked to allow the interviewee to familiarize themselves with the topic in order to limit the potential for answers to drift from the subject of software within the field of image processing and analysis. This was apparent while we ran pilot tests of the interview. Interviewees would typically drift from the subject area and begin answering questions from the perspective of the company or from their past experiences. In second part of the interview, interviewees were presented with a list of quality characteristics derived from the ISO/IEC 25010 [16] standard accompanied with suitable explanations. The interviewees were instructed to prioritize the quality characteristics according to a virtual $100. The $100 Test is a cumulative voting technique, where the highest total reflects the most important requirement [24]. Following the $100 Test, the interviewee was instructed to motivate their prioritization. The interviews varied between 15 to 20 minutes are were recorded with an audio device.

2) Data analysis: The data analysis started after conducting the interviews and was split into two phases. The analysis was firstly done by transcribing the audio recordings to be able to revise the answers and validate the data. The transcriptions provided a qualitative result from the interviews that was later compared and used for validating the $100 Test results. After transcribing the interviews, a phase of organizing and processing the results from the $100 Test began. Firstly the answers from each interviewee was put together into a table to see the total amount spent on a quality characteristics sorted by teams. This was later transferred to a percentage based result. The formula for calculating the quality characteristics priority percentage is:

$$QA\ weight\ \% = \frac{QA\ spending}{total\ team\ budget}$$

To gather the overall opinion of the company, both teams' results were combined to form a table where the result was calculated by using the same formula stated above.

B. Experiment

To answer Research Question two (RQ2), an experiment was carried out following the guidelines presented by Shull et al. [25].

1) Goals: The goal of the experiment is to evaluate face recognition libraries and APIs with respect to the quality characteristics identified in Research Question one (RQ1). Namely, metrics derived from sub-characteristics of the most important quality characteristics will give an indication on how each face recognition system fulfils the sub-characteristic. The experiment will firstly establish if there is a difference in quality between client-side libraries and cloud-based APIs and then uncover the individual differences between the chosen experimental units.

TABLE III
Experiment Goals

Goal 1	Analyse the computational accuracy of the face recognition libraries and APIs in respect to the number of inaccurate computations encountered by the user within a certain operation time.
Goal 2	Analyse the failure density of the face recognition libraries and APIs in respect to the number of failures that are detected during a defined trial period.
Goal 3	Analyse the change cycle efficiency of the face recognition libraries in respect to how a users problem can be solved within an acceptable time scale.

2) Experimental Units: The most popular face recognition libraries and APIs in current research are presented in Table II. In this paper we have selected to evaluate two open-source client-side libraries and two commercial cloud-based APIs.

OpenCV (version 2.4.5) is an open-source computer vision library used extensively in companies, research groups and by governmental bodies. OpenCV released its own face recognition module (cv::FaceRecognizer), however it uses older algorithms and is not as powerful as newly developed algorithms [1]. The face recognition module supports built in functions for training the algorithms.

OpenBR (version 0.5.0) is a open biometrics framework that supports the development of face recognition algorithms and reproducible evaluations. OpenBR is currently in a beta version. The algorithms used in OpenBR are described in Klontz et al [11] and OpenBR does support functions for algorithm training.

Face++ (www.faceplusplus.com) is a cloud based API that offers face detection, face analysis and face recognition. Face++ has received first place in the FDDB [26], 300-W challenge [27] and LFW [28] dataset and benchmarking tests.

ReKognition (www.rekognition.com) is a cloud based API that performs face, scene and concept recognition. It is unknown as to which algorithm the API uses and supports limited training of the algorithm.

3) Experimental Material: The experimental materials consist of sample test data and algorithm training data. The sample test data is used as input to the face recognition libraries and APIs, whereas the training data is used to train the algorithms.

Sample Test Data: Labelled Faces in the Wild (LFW) [6] is a databased consisting of face images of people published by the University of Massachusetts. The experiment used LFW as test-data and training material. The dataset was chosen for the experiment since the sample images are images of faces from unconstrained environments, which is in line with the use case of the company. Images of faces in unconstrained environments ensure there are no restrictions over environmental conditions such as scale, pose, lighting, focus, resolution, facial expression etc.

The database consists of 13,233 images. Of the 13,233 images there are 5,749 people. The database does not include metadata describing the gender of the subject in the image, however the name of the person is supplied. Therefore, the database was modified to be suitable for the experiment. Firstly, the gender of each image was determined by using a service that converts a persons name to a suitable gender by supplying a confidence level. Once the images were classified into male and female, the images were sorted accordingly. Only images with a gender confidence level of at least 90% were used. Images with a less confidence rate or an unknown gender were discarded. The resulting database after manipulating LFW is displayed in Fig. 3.

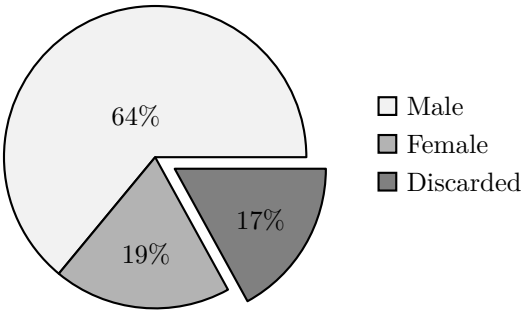

Fig. 3. The sample test database after the manipulation of the LFW database.

The sample test database was finalized by filtering out unique names and evenly balancing the number of sample images. This resulted in a test data pool of 499 unique images of female faces and 501 unique images of male faces. The resulting training data and test data extracted from the LFW database is shown in Fig. 4.

Algorithmic Training Data: The training data images was extracted from LFW and excluded from the sample

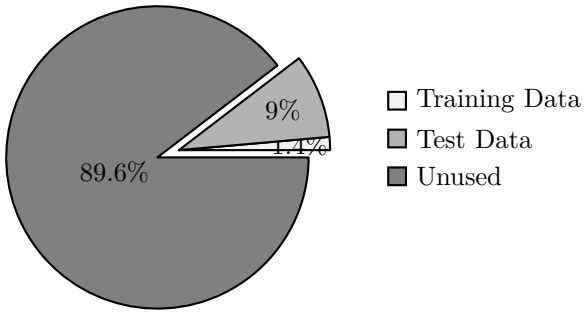

Fig. 4. The overall data used from the LFW database.

test dataset. The training data was created by cropping out the face, applying a histogram equalization to compensate for uneven illuminations on the face and resizing the image into 52x52 pixels. The training dataset consists of 160 images, 80 being female faces and 80 being male faces.

Tasks: The two open-source face recognition libraries were trained by applying the training dataset. Built-in functions in both libraries were used to train the algorithms and no modifications to the source code was made. The two client side libraries and two cloud based APIs were tasked with, (1) locate the face in the image and (2) determine the gender of the detected face. If no face is detected the gender classification cannot be performed.

4) Hypotheses, Parameters and Variables: Three types of variables are defined for the experiment, *independent*, *parameter* and *dependent* variables.

Independent Variable: The independent variable is face recognition software. The face recognition software is grouped respectively as cloud-based APIs and client-side libraries.

Parameters: The parameters subject to the experiment is the training data. The results of the experiment will be particular to the condition of training the client-side libraries [29] as well as the data used for the training procedure. The functionality of the cloud-based APIs cannot be modified in any way, whereas the client-side libraries are open-source with available access to the source code. However, no modification to the source code was made.

Dependent Variable: The results from Research Questions one (RQ1) identify *functional suitability*, *reliability* and *maintainability* as the three most important quality characteristics the organization prioritize over. Quality evaluation metrics for evaluating the sub-characteristics of three most important quality characteristics were attempted to be extracted from the ISO/IEC 25010 standard, however the standard has not yet released a revised version of the quality metrics. Therefore, the evaluation metrics in the experiment were extracted from ISO/IEC 9126-2:2003 [30].

The three dependent variables measured are the sub-characteristics of the quality requirements identified in Research Question 1. They are functional correctness, maturity and changeability. Functional correctness is a sub-characteristic of functional suitability, maturity is a sub-characteristic of reliability and changeability is a sub-characteristic of maintainability. Changeability has been removed as a sub-characteristic from the revised ISO/IEC 25010 standard. However, since the experiment is evaluating both open-source and cloud-based APIs the access to source code is limited. Therefore, an exception was made in adopting the changeability metric as this was the only metric within *maintainability* the experiment could attempt to carry out. In Table IV the dependent variables and following metrics are presented.

Hypotheses: The dependent variables are analysed to evaluate the hypotheses of the experiment. The null and alternative hypotheses [31] are state below. The null hypothesis is denoted H_{0i_j}, and the corresponding alternative hypothesis is denoted H_{1i_j}. The i corresponds to the goal identifier and the j is a counter for cases where more than one hypothesis is formulated per goal [25].

- H_{01_1} — There is no difference in *functional accuracy* between client-side libraries and cloud-based APIs for gender classification..
- H_{11_1} — There is a difference in *functional accuracy* between client-side libraries and cloud-based APIs for gender classification.
- H_{01_2} — There is no difference in *functional accuracy* between client-side libraries and cloud-based APIs for face detection.
- H_{11_2} — There is a difference in *functional accuracy* between client-side libraries and cloud-based APIs for face detection.
- H_{02} — There is no difference in *failure density* between client-side libraries and cloud-based APIs.
- H_{12} — There is a difference in *failure density* between client-side libraries and cloud-based APIs.
- H_{03} — There is no difference in *change cycle efficiency* between client-side libraries.
- H_{13} — There is a difference in *change cycle efficiency* between client-side libraries.

5) Design: A Quasi experiment design is used since random assignment is not applicable in this case. The experimental units, being face-recognition software, are formally categorized as client-side libraries and cloud-based APIs. A Quasi experiment is useful when random assignment to treatment groups is impossible [32]. The experiment is not aimed to establish a cause and effect relationship, but rather an observation on how the software under evaluation perform.

6) Procedure: A self developed application named FacE[1] was used for performing the test case in Table V. The self developed application schedules the test case (TC1) and assigns a randomized set of 20 images to the experimental units. The test case was run 50 times which

[1]http://github.com/Pletron/facE

TABLE IV
DEPENDENT VARIABLES

Dependent Variable	ISO 9126 Characteristic	Metric Name	Purpose of the Metric	Method of Application	Measurement Formula	Interpretation of Measured Value
Accuracy Metric	Functionality	Computational accuracy	How often does the end users encounter inaccurate results?	Record the number of inaccurate computations based on specifications.	$X = \frac{A}{T}$ A=Number of inaccurate computations encountered by users. T=Operation time.	$0 \leq X$ The closer to 0 the better.
Maturity Metric	Reliability	Failure density against a test case.	How many failures were detected during defined trial period?	Count the number of detected failures and performed test cases.	$X = \frac{A}{B}$ A = Number of detected failures. B = Number of performed test cases.	$0 \leq X$ It depends on stage of testing. However, at the later stage, the closer to 0 better.
Changeability Metric	Maintainability	Change cycle efficiency	Can the user's problem be solved to his satisfaction within an acceptable time scale?	Monitor interaction between user and supplier. Record the time taken from initial user's request to resolution of problem.	Average time = $T_{av} = \frac{Tu}{N}$ $Tu = T_{rc} - T_{sn}$ $T_{sn} =$ Time at which the user finished to send request for maintenance to supplier with problem report. $T_{rc} =$ Time at which user received the revised version release (or status report). N=Number of revised versions	$0 \leq T_{av}$ The shorter the better unless the number of revised versions is large.

The metrics are derived from ISO 9126-2:2003. [30]

resulted in 1000 unique images that were applied to each experimental unit.

TABLE V
TEST CASE (TC1) PROCEDURE

1. Randomly select 20 images from the database.
2. Flag the 20 images to avoid testing the same image twice.
3. In sequence, select an image from the set of 20 images and send it to all the experimental units.
4. Capture the results from the experimental units.
5. Record test data such as processing time and any errors that may occur.
6. Store the results and test data in a database.

To achieve goal one and two of the experiment, FaceE was run on a virtual machine in VirtualBox running Ubuntu Raring Ringtail (13.04), 4 gigabytes of memory, an Intel Core i7 processor and with graphics acceleration enabled. The data collected in each execution of the test case (TC1) was (1) the execution time for each experimental unit, (2) the computed gender, (3) the location of the detected face and (4) error messages such as time-outs or critical failure. The computed gender and the location of the face was then translated into *success* or *failure* depending on the ground truth value, as further explained in Table VI. To achieve goal three of the experiment, the data was collected be means of investigating bug-reports on each systems' bug-tracker and applying the dependent variable metric. The time of open and close for all issues posted on each issue tracker respectively was collected, regardless of the priority or risk.

7) Analysis Procedure: A statistical analysis procedure was carried out in the experiment. The data collected was applied to the metrics associated to each dependent variable where the results from the metrics are on a ratio scale. An implementation of the categorized nominal Pearson's chi-squared test generated the results for the hypothesis testing. Pearson's chi-squared test is a statistical hypothesis test that is applied to categorized data [33]. As the study categorizes the face recognition and the gender recognition as failed or successful, the chi-test suited the research goals. The conventional .05 significance level was adopted throughout each hypothesis test [33].

IV. DEVIATIONS FROM THE EXPERIMENT PLAN

In analysing the maintainability of both client-side libraries and cloud-based APIs, it was noticed that the cloud-based APIs do not expose a public issue tracking system, which was needed during the data collection process. Therefore, goal three and the corresponding hypotheses were altered to only analyse the maintainability of client-side libraries, which do expose a public issue tracker.

V. RESULTS AND ANALYSIS

This section provides the analysis of the data extracted from the interviews and from the experiment to answer the research questions.

A. Case company's quality prioritization

In analysing Research Question 1 (RQ1), this section presents the most important quality characteristics the company prioritise over when adopting face recognition technology for reuse. In Figure 5 we can see the top ranked quality characteristics that the development team and the administrative team prioritize over. The development team prioritizes reliability as the most important quality characteristic, followed by functional suitability and maintainability. Moreover, the administrative team prioritize the same quality characteristics as the development team, however reliability is presented as the most important, followed maintainability and functional suitability.

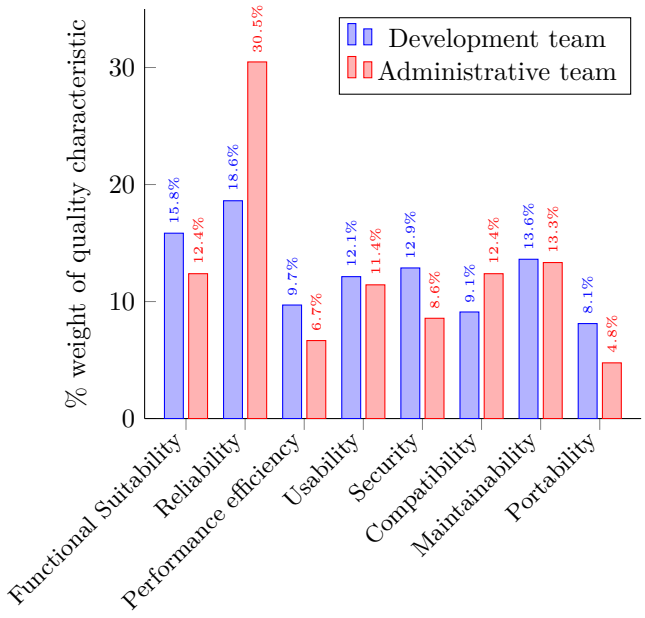

Fig. 5. Chart illustrating the prioritization of quality characteristics from the perspective of the development team and the administrative team

In Figure 6, the total prioritization of quality characteristics for the case company is presented. Reliability has been prioritized the highest amongst both groups. Functional suitability follows as the second most important quality characteristic for the company, where maintainability is not far behind.

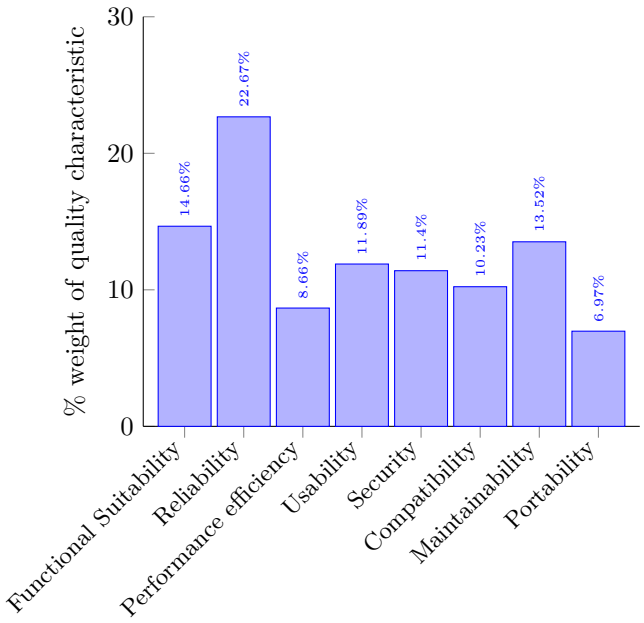

Fig. 6. Chart illustrating the prioritization of the whole company

TABLE VI
EXPERIMENT DATA COLLECTION DESCRIPTION

(1)Successful gender estimations	A correct estimation is based on face detection that is not a false positive and that the estimated gender is equal to the image's gender ground truth.
(2)Failed gender estimations	If the detected face is a false positive or the gender estimated is not equal to the image's ground truth, then it is counted as a failed gender estimation.
(3)Successful face detection	A successful detection is a non false-positive face, where the number of successful face detection is calculated by taking the amount of recognized faces subtracted by the amount of failed face recognitions.
(4)Failed face detection	A failed face detection is when there is no face detected or that the face detected is a false-positive.
(5)Successful API calls	A successful API call is when there is a response to the request that is equal to either no detected faces or a detected faces.
(6)Failed API calls	A failed API call is if the response to the request is equal to a timeout, crash or unexpected behaviour.
(7)Average bug fix time	Average bug fix time was extracted by applying the formula for the maintainability metric explained in Table IV.

B. The Software Quality of Face Recognition APIs and Libraries

In analysing Research Question 2 (RQ2), this section presents the results from the hypotheses tests for goal one, two and three of the experiment. The data collected from each experimental unit is shown in Table VII where the processing of each data attribute is described in Table VI.

TABLE VII
EXPERIMENT DATA COLLECTION

	Client side	**Cloud based**	**Totals**
Successful gender estimations(1)	1327	1767	3094
Failed gender estimations(2)	672	75	747
Total images	**1999**	**1842**	**3841**
Successful face recognitions(3)	1999	1823	3822
Failed face recognitions(4)	0	19	19
Total images	**1999**	**1842**	**3841**
Successful API calls(5)	1999	1928	3927
Failed API calls(6)	0	158	158
Total calls	**1999**	**2086**	**4085**

For hypotheses H_{011}, H_{012}, H_{02}, a Pearson's Chi-square test [34] was applied. To calculate the chi-squared statistical value (χ^2) formulas provided by the Pearson's Chi-square test [34] was used. The critical value $\chi^2_{crit} = 3.841$ was derived by combining the significance value of 0.05 and by having a degree of freedom of 1. If $\chi^2 < 3.841$ would be true the null hypothesis cannot be rejected. The following displayed the results from the statistical test;

- H_{011} With $\chi^2 = 534.1683$ it shows that H_{011} can be rejected by at least 95% accuracy by implying that $\chi^2 < 3.841$ is not true.
- H_{012} With $\chi^2 = 20.7219$ it shows that H_{012} can be rejected by at least 95% accuracy by implying that $\chi^2 < 3.841$ is not true.
- H_{02} With $\chi^2 = 144.1609$ it shows that H_{02} can be rejected by at least 95% accuracy by implying that $\chi^2 < 3.841$ is not true.

For hypothesis H_{03} a two tailed *Mann Whitney U-test* [35] was used due to the uneven distribution of the sample data seen in Fig. 7.

TABLE VIII
BUG FIX DATA

	OpenBR	**OpenCV**
Avg. bug fix time(7)	275h	3312h

The H_0 hypothesis is rejected if the z-score is below or above the $+$-z_{crit} value. This value was calculated by using formulas provided from the Mann Whitney U-test [35]. The z_{crit} is extracted with the chosen (α) significance value. In this case $z_{crit} = (-)1.96$ was extracted at a significance value of 0.05 and a sample size of 21 from OpenBR and 1633 from OpenCV.

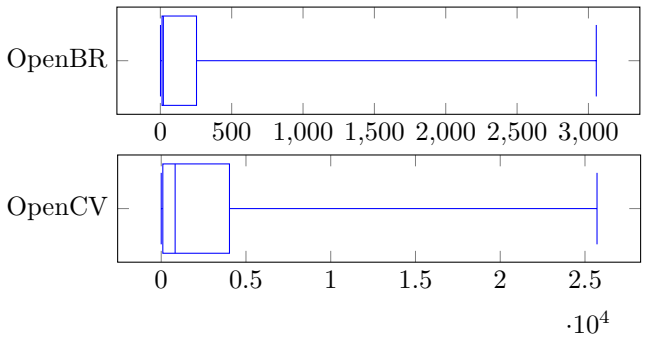

Fig. 7. Box plot normality test

H_{03} With z-score $= -4.340481$ it shows that H_{03} can be rejected by at least 95% accuracy by implying that the z-score value is below the value of z_{crit}.

The results of the hypothesis tests shows that there is a difference between client side software and cloud based software in terms of computational accuracy and failure density. All the hypotheses that addressed these quality sub-characteristics were clearly rejected. Further details are shown in Fig. 8 where the cloud based API ReKonition resulted as the most computationally accurate due to its gender estimation and face recognition success rate. With a 91.4% accuracy it clearly has the lead compared to the most accurate client side software, OpenCV, with its 73.7% accuracy. Fig. 8 reflects a merge between H_{011} and H_{012} where both gender estimation and face recognition have been taken into account.

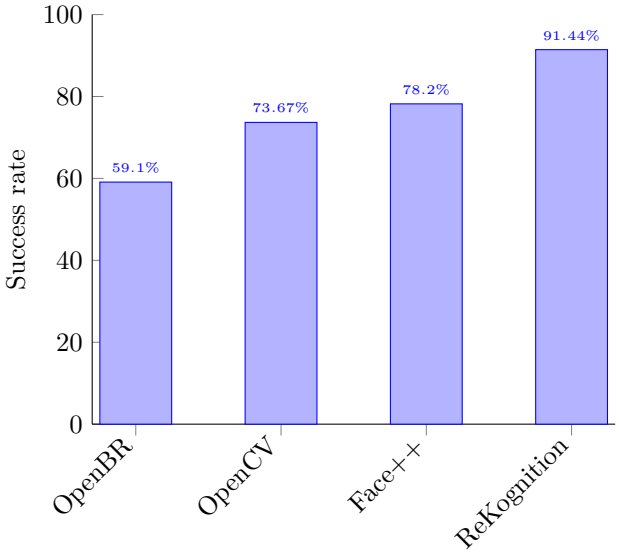

Fig. 8. Chart illustrating success rate for face recognition and gender estimation.

Further analysis in Fig. 9 shows the amount of failures each API encountered throughout the experiment. Failures such as time-outs, a response of server overload, software crashes or unexpected behaviour were taken into account. No API had any failures except for Face++, with a total failure rate of 15.8%. These failures were strictly time-outs or a response of server overload.

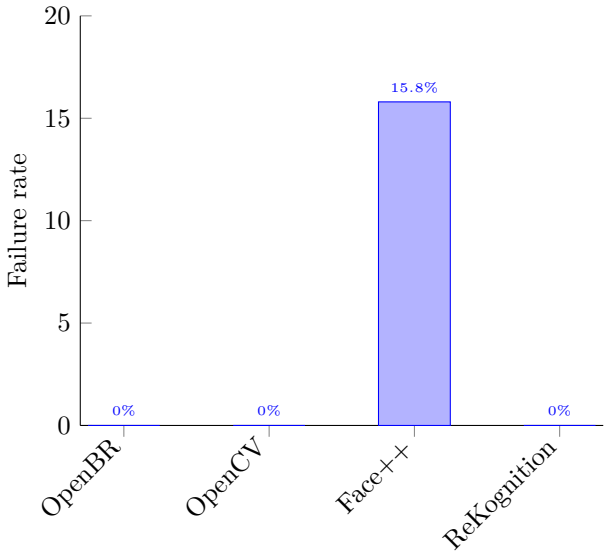

Fig. 9. Chart illustrating response failure rate.

VI. Threats to Validity

In this section, threats to validity to the findings of this research are discussed . The four perspectives of validity and threats are discussed as presented in Shull et al. [25].

A. Construct Validity

Construct validity refers to the degree to which the research and the observations are in relation to theories or constructs. The open-ended semi-structured interview allowed the interviewees to express their own opinions. The qualitative data that was extracted from the $100 Test is subjective to the understanding and interpretation of the interviewees. The $100 test was used instead of numerical assignment since Berntsson Svensson et al. [24] suggests numerical assignment, such as high, medium and low may be subject to interpretation by an interviewee. The dependent variables in the experiment are subject to mono-operation bias, since the FaceE testing application was run over a two day period. Even though the LFW database was modified to be suitable for the experiment, from our own findings the database is widely used and accepted in research. The metrics used in the experiment were derived from a well establish ISO standard.

B. Internal Validity

Internal validity refers to the extent to which issues may affect the relationship between treatment and outcome. The experiment results derived from the client-side libraries are subject to the treatment in the form of training the algorithms. Since there is no well establish training

data-set and protocol, we used a training dataset that suited the use case. The test data LFW was modified to be suitable for the experiment. The classifying of the gender in each images was not manually verified.

C. External Validity

External validity refers to the degree to which the findings can be generalized beyond the actual study. Since the research strategy for the interviews in Research Question one was qualitative, we did not aim to generalise the findings. Moreover, the results from the experiment are subject to the training and testing material which are in line with the use case of the company. Hence, we did not aim to generalise the findings for the experiment either. The nature of the research would be difficult to replicate by the way the testing and training materials were derived.

D. Conclusion Validity

Conclusion validity refers to the degree to which the conclusions reached in the study are correct. The interviews were conducted at the case company over a two week period. There were no discussion that would influence on the results from the interviewees. Pilot testing the interview was done to avoid poor questions and explanations prior to conducting the interviews. In respect to the experiment, no source code of the experimental units were not manipulated. A pilot test of the FaceE application was run to ensure data was being collected and stored correctly. The pilot test of the FacE was run over a four day period.

VII. Discussion

This study shows that a company looking to adopt face recognition technology has a clear vision of which quality attributes are of importance. When interviewing the company, it was clear that reliability is highly valued throughout the company. A reoccurring statement was that software as well as the hardware needs to be reliable in order to satisfy customer needs. It became obvious that with 11,000 installations in 40 countries, the company cannot afford to have an unreliable system that would require constant support and service. The importance of reliability was further motivated by one interviewee, [reliability is important] *"because it costs so much to recover from errors for a small development department"*. The importance of reliability are in line with the findings in Johansson et al. [36] and in Berntsson Svensson et al. [7], which carried out interviews on the prioritization of quality requirements at a number of companies. By adequately interviewing the company to understand which quality characteristics are the most important, this study could execute tests that would address the quality concerns before suggesting the reuse of software. If the software would meet the expectations of the case company's quality preferences then the reuse of software would be of more interest than developing an in house solution.

The results derived from the experiment may indicate that client-side libraries are more reliable, in terms of failure density explicitly, than cloud-based APIs. The cloud-based APIs had a mean failure rate of 7.9%, however this assumption would not be fair as there was only one cloud-based API that provided a failure rate to the mean. In particular, Face++ failed 158 times during the experiment but performed relatively well in regards to computational accuracy. This is a clear indication of why functional requirements are equally important to quality requirements. Even though Face++ performs relatively well, if the system is not reliable the system is not very usable. During the experiment, we did not expect the client-side libraries to fail. However, for cloud-based APIs there is an additional factor. Failures can occur when communicating over the internet such as packet loss or loss of connectivity. However, in order to ensure our expectations on whether the client-side libraries would fail during operation, the experiment provided valuable data.

The second observation is that the cloud-based APIs had a significantly higher computational accuracy in terms of functional suitability where a greater amount of faces were recognized. This can be a result of a vast amount of training images that the cloud-based software can make use of to finely adjust the algorithms used. Whereas, during the experiment there was no extensive training performed on the client-side libraries.

After analysing the results of experimental units, it was clear that the difference in execution time was significant and that client side libraries provided the results by at least the 10^{th} of the execution time compared to that of the cloud based APIs. This was expected as the cloud based software would be limited due to the network speed which is in line with the study by Becker and Ortiz [1].

Unfortunately, the changeability of the cloud-based APIs could be not analysed. However, the average time for an issue to be resolved for the client-side libraries are relatively slow. OpenCV is a large project that comprises of a big community of developers where issues took long to be closed. Even though OpenBR has a much smaller community compared to OpenCV, they were much faster in closing issues. This may be because OpenBR is focused on face recognition whereas OpenCV is a large computer vision library focusing on several aspects. The two client side libraries are not comparable due to the size of each project, but from the perspective of a user it may be a good indication on how long one should expect a defect to be resolved.

According to the results derived from the experiment, ReKognition is the clear choice due to its low failure density and high success rate in terms of recognizing both face and gender in a picture. In the case of the company, ReKognition would allow a quick to market implementation where minimum development effort would be required. ReKognition did not encounter any failures during the experiment, however if the environment, which

the software would run in, has limited network access a client side library would be a more natural choice. Therefore, OpenCV would be the more suitable choice for this case where it may be more versatile compared to OpenBR with a higher success rate.

VIII. Conclusion

This paper sets out to understand the prioritization of software quality characteristics when adopting face recognition software and to identify how a quality evaluation can be applied. Through interviews with employees of a company investigating face recognition technology, we have found that reliability, functional suitability, and maintainability are the three most important quality characteristics when adopting the technology. Sub-characteristics of the three most important quality characteristics were assessed to evaluate two client side libraries and two cloud based APIs by applying an experiment. The study has found that cloud-based face-recognition software will allow organizations and developers a quick to market implementation. Although cloud-based APIs are highly dependent on network access with a costly subscription, they offer high computational accuracy. The quality of both the client side libraries and the cloud-based APIs may have a significant difference, favouring client side libraries with less probability to fail and better customization possibilities due to access to the source code, whereas cloud-based services ships with fixed functionality. These findings are important contributions to organizations and developers since they uncover the importance of doing a quality evaluation in order to identify critical success factors, as well as providing data on the current software quality of the chosen libraries and APIs. While we have specifically focused on quality evaluation, the evaluation of face recognition software implies that our findings are likely to be of importance to the computer vision community. In terms of future research, we particularly suggest that further investigation of all quality characteristics and sub-characteristics should be carried out with a goal to create a clear training and testing protocol for face recognition software and services.

Acknowledgement

We thank Christian Berger and Richard Berntsson Svensson as well as all the participants that helped in making the data collection possible.

References

[1] B. Becker and E. Ortiz, "Evaluating open-universe face identification on the web," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, June 2013, pp. 904–911.

[2] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: A literature survey," *ACM Comput. Surv.*, vol. 35, no. 4, pp. 399–458, Dec. 2003.

[3] G. Gediga, K.-C. Hamborg, and I. Düntsch, "Evaluation of software systems," in *Encyclopedia of Computer Science and Technology*. Marcel Dekker, 2002, vol. 45, pp. 127 – 153, Also appeared in volume 72 of the *Encyclopedia of Library and Information Science* (2002), 166–192. [Online]. Available: "http://www.cosc.brocku.ca/~duentsch/archive/softeval.pdf"

[4] P. Phillips, H. Moon, P. Rauss, and S. Rizvi, "The feret evaluation methodology for face-recognition algorithms," in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, Jun 1997, pp. 137–143.

[5] P. Phillips, P. Grother, R. Micheals, D. Blackburn, E. Tabassi, and M. Bone, "Face recognition vendor test 2002," in *Analysis and Modeling of Faces and Gestures, 2003. AMFG 2003. IEEE International Workshop on*, Oct 2003, pp. 44–.

[6] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," University of Massachusetts, Amherst, Tech. Rep. 07-49, October 2007.

[7] R. Berntsson Svensson, T. Gorschek, B. Regnell, R. Torkar, A. Shahrokni, and R. Feldt, "Quality requirements in industrial practice - an extended interview study at eleven companies," *Software Engineering, IEEE Transactions on*, vol. 38, no. 4, pp. 923–935, July 2012.

[8] I. Marqués, "Face recognition algorithms," Tech. Rep., June 2010. [Online]. Available: http://www.ehu.es/ccwintco/uploads/e/eb/PFC-IonMarques.pdf

[9] M.-H. Yang, D. Kriegman, and N. Ahuja, "Detecting faces in images: a survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 1, pp. 34–58, Jan 2002.

[10] C. Zhang and Z. Zhang, "A Survey of Recent Advances in Face detection," Tech. Rep., 2010.

[11] J. Klontz, B. Klare, S. Klum, A. Jain, and M. Burge, "Open source biometric recognition," in *Biometrics: Theory, Applications and Systems (BTAS), 2013 IEEE Sixth International Conference on*, Sept 2013, pp. 1–8.

[12] E. Zhou, H. Fan, Z. Cao, Y. Jiang, and Q. Yin, "Extensive facial landmark localization with coarse-to-fine convolutional network cascade," in *Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on*, Dec 2013, pp. 386–391.

[13] J. Klontz and A. Jain, "A case study of automated face recognition: The boston marathon bombings suspects," *Computer*, vol. 46, no. 11, pp. 91–94, Nov 2013.

[14] D. Samadhiya, S.-H. Wang, and D. Chen, "Quality models: Role and value in software engineering," in *Software Technology and Engineering (ICSTE), 2010 2nd International Conference on*, vol. 1, Oct 2010, pp. V1–320–V1–324.

[15] F. Deissenboeck, E. Juergens, K. Lochmann, and S. Wagner, "Software quality models: Purposes, usage scenarios and requirements," in *Software Quality, 2009. WOSQ '09. ICSE Workshop on*, May 2009, pp. 9–14.

[16] ISO/IEC, "ISO/IEC 25010 - Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models," Tech. Rep., 2010.

[17] S. Lauesen, *Software Requirements: Styles and Techniques*. Addison-Wesley, 2002.

[18] R. Berntsson Svensson, T. Gorschek, and B. Regnell, "Quality requirements in practice: An interview study in requirements engineering for embedded systems," in *Requirements Engineering: Foundation for Software Quality*, ser. Lecture Notes in Computer Science, M. Glinz and P. Heymans, Eds. Springer Berlin Heidelberg, 2009, vol. 5512, pp. 218–232.

[19] L. Phillips, A. Aurum, and R. Berntsson Svensson, "Managing software quality requirements," in *Software Engineering and Advanced Applications (SEAA), 2012 38th EUROMICRO Conference on*, Sept 2012, pp. 349–356.

[20] P. Berander, L. O. Damm, J. Eriksson, T. Gorschek, K. Henningsson, P. Jönsson, S. Kågström, D. Milicic, F. Mårtensson, K. Rönkkö, and P. Tomaszewski, *Software quality attributes and trade-offs*, L. Lundberg, M. Mattsson, and C. Wohlin, Eds. Blekinge Institute of Technology, June 2005. [Online]. Available: http://www.bth.se/tek/besq.nsf/pages/017bd879b7c9165dc12570680047aae2!OpenDocument

[21] T. Wengraf, *Qualitative Research Interviewing: Biographic Narrative and Semi-Structured Methods*. SAGE Publications, 2001.
[22] C. Seaman, "Qualitative methods in empirical studies of software engineering," *Software Engineering, IEEE Transactions on*, vol. 25, no. 4, pp. 557–572, Jul 1999.
[23] B. Wong, "Measurements used in software quality evaluation," 2003.
[24] R. Berntsson Svensson, T. Gorschek, B. Regnell, R. Torkar, A. Shahrokni, R. Feldt, and A. Aurum, "Prioritization of quality requirements: State of practice in eleven companies," in *Requirements Engineering Conference (RE), 2011 19th IEEE International*, Aug 2011, pp. 69–78.
[25] F. Shull, J. Singer, and D. I. Sjøberg, *Guide to Advanced Empirical Software Engineering*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.
[26] V. Jain and E. Learned-Miller, "Fddb: A benchmark for face detection in unconstrained settings," University of Massachusetts, Amherst, Tech. Rep. UM-CS-2010-009, 2010.
[27] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic, "300 faces in-the-wild challenge: The first facial landmark localization challenge," in *Proceedings of IEEE Int'l Conf. on Computer Vision (ICCV-W 2013), 300 Faces in-the-Wild Challenge (300-W)*, Sydney, Australia, December 2013.
[28] H. Fan, Z. Cao, Y. Jiang, Q. Yin, and C. Doudou, "Learning deep face representation," *CoRR*, vol. abs/1403.2802, 2014.
[29] N. Juristo and A. M. Moreno, *Basics of Software Engineering Experimentation*, 1st ed. Springer Publishing Company, Incorporated, 2010.
[30] ISO/IEC, *ISO/IEC 9126. Software engineering – Product quality*. ISO/IEC, 2001.
[31] D. Montgomery, *Design and Analysis of Experiments*, ser. Student solutions manual. John Wiley & Sons, 2008.
[32] N. Juristo and A. Moreno, *Lecture Notes on Empirical Software Engineering*, ser. Series on software engineering and knowledge engineering. World Scientific, 2003.
[33] V. B. Kampenes, T. Dybå, J. E. Hannay, and D. I. K. Sjøberg, "Systematic review: A systematic review of effect size in software engineering experiments," *Inf. Softw. Technol.*, vol. 49, no. 11-12, pp. 1073–1086, Nov. 2007. [Online]. Available: http://dx.doi.org/10.1016/j.infsof.2007.02.015
[34] K. Pearson, "X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling," *Philosophical Magazine Series 5*, vol. 50, no. 302, pp. 157–175, 1900.
[35] D. J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*, 4th ed. Chapman & Hall/CRC, 2007.
[36] E. Johansson, A. Wesslen, L. Bratthall, and M. Host, "The importance of quality requirements in software platform development-a survey," in *System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on*, Jan 2001, pp. 10 pp.–.