FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Ortsunabhängige Steuerung hochpräziser Feinmesstechnikaufbauten

Entwicklung einer flexiblen Software-Architektur zum Fernstudium in der Messtechnik

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Wirtschaftsinformatik

eingereicht von

Michael Zawrel
Matrikelnummer 1025319

an der Fakultät für Informatik
der Technischen Universität Wien
Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Gerald Futschek

Wien, 27. Mai 2018

Michael Zawrel	Gerald Futschek

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Location-independent operation of a high-precision metrology setup

Development of a flexible software architecture for metrology research and education from a distance

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Business Informatics

by

Michael Zawrel
Registration Number 1025319

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Gerald Futschek

Vienna, 27th May, 2018

Michael Zawrel	Gerald Futschek

Erklärung zur Verfassung der Arbeit

Michael Zawrel
Autokaderstraße 3-7/48/13, 1210 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 27. Mai 2018

Michael Zawrel

Acknowledgements

At first, I want to thank Dr. Gerald Futschek for his willingness to act as a supervisor for this cross-border and interdisciplinary thesis and his advice in the process. Next, I want to thank Dr. Jorge Bauer, Dr. Numan Durakbasa and their team for providing the expertise in metrology and control theory. Thank you Nacho and Ezequiel for your support during the early stages in Argentina and Austria. I further want to thank Julia for the moral support and the great coffee during most of the writing stage. Special thanks goes to Ruth and the men at JJ Harlow's Irish Pub in Roscommon, Ireland, for proofreading and contemplating the meaning of some sources. A thank you to every reader, who makes my efforts writing this thesis worthwhile.

Finally, a big thank you to my parents Helmut and Monika for everything they have done for me up to this point in my life and for empowering me to follow my educational path.

Kurzfassung

Die moderne Messtechnik erfordert eigens errichtete Laboratorien, kontrollierte Umgebungsbedigungen und hochpräzise Maschinen. Für zahlreiche Bildungs- und Forschungseinrichtungen ist die teure Ausstattung ohne Partner nicht mehr leistbar. In dieser Diplomarbeit wird eine Software-Systemarchitektur zur ortsunabhängigen Ansteuerung von *Koordinatenmessmaschinen* (CMMs) per Internet vorgestellt. Ziel ist, die Bildung von Kooperationen zur Anschaffung von Messtechnikmaschinen zwischen Institutionen voranzutreiben, die bisher, aufgrund ihrer geographischen Distanz und daraus resultierenden Reisekosten, von einem geteilten Investment nicht profitieren konnten. Die präsentierte Referenz-Architektur kann als Vorlage für die Schaffung einer neuen Gruppe an flexiblen Systemen zur Distanzsteuerung von CMMs dienen. Sie nützt eine Web-Applikation als Benutzerschnittstelle und unterstützt sowohl physische als auch simulierte Experimente. Die zentralen Komponenten zur Kommunikation mit der CMM (*Acquisition Server*) und zur Entgegennahme von Benutzerkommandos (*Laboratory Server*) ermöglichen parallele Bedienung mehrerer Maschinen und die Durchführung komplexer Operationen und Berechnungen. Die Sichtkontrolle von entfernten physischen Maschinen erfolgt per Webcam, simulierte Maschinen werden per Remote-Desktop überwacht. Um ungewollte Kollisionen von Abtastkopf und Messobjekt, ausgelöst durch die Latenz in der Signalübertragung via Internet, zu vermeiden, wurden einige Änderungen an den Steuerungsmechanismen im Vergleich zu konventioneller, lokaler CMM-Steuerung vorgenommen. Mithilfe einer prototypischen Implementierung der Architektur (*"Manejo"*) konnte gezeigt werden, dass die adaptierte Steuerung bei Distanzsteuerung via Internet dieselbe Messgenauigkeit ermöglicht wie aus dem lokalen Netzwerk der CMM. Eine weitere Erkenntnis dieser Arbeit ist, dass sich die Latenz der Internetsteuerung maßgeblich auf den Messvorgang, und in der Folge negativ auf dessen durchschnittliche Dauer auswirkt. Die vorliegende Arbeit soll einen Beitrag zur Etablierung der Feinmesstechnik via Internet leisten und zukünftige Forschung zu ihren Charakteristiken sowie ihren Auswirkungen auf die Ausbildung neuer Messtechnikerinnen und Messtechniker vereinfachen.

Abstract

Metrology is performed in specially designed laboratories using highly precise machinery. Many institutions are no longer able to shoulder the costs of acquiring expensive equipment alone. In this thesis, a system architecture for the location-independent operation of *coordinate measuring machines* (CMMs) via the internet is introduced. It aims to encourage collaboration between education and research institutions that are located far apart from each other. The reference architecture presented herein can be used as a starting point for the creation of a new type of CMM remote control systems. It uses a web-application as an interface for the operator to control either physical or simulated remote laboratories. A centralized control server, consisting of an *Acquisition Server (AS)* component and a *Laboratory Server (LS)* component is able to govern multiple machines and perform complex operations and calculations. The CMM is observed via webcam at physical remote experiments and a remote access tool or embedded simulation window at simulated remote experiments. Some changes in the control mechanisms have been made compared to conventional local CMM operation with regard to the inconstant latency and less reliable command transmission via the internet. Using the prototypical implementation *Manejo*, it was possible to show that the control system can achieve equivalent measurement accuracy within a local network and via the internet. Furthermore, it was discovered that internet control increases the mean duration of the probing process of a measuring object. The system can be used to promote new discoveries in the field of remote metrology, like the long-term effects of using remote or simulated laboratories in metrology education on the learning outcome of students.

Contents

CHAPTER 1

Introduction

1.1 Motivation & Problem Statement

Metrology is the science of measurement. It includes both theoretical and practical aspects of measurement [BIP18]. Early standardization attempts led to the creation of the metric system and subsequently the *International System of Units*. Metrology gear revolutionized production ever since. Typical modern applications are research in bionics and quality control in industry.

Metrology requires complex and sophisticated machinery. The equipment for highly precise measurement setups such as that used at the "Department of Interchangeable Manufacturing and Industrial Metrology / Production Metrology and Quality" (AUM)[1] at the TU Wien can be worth millions. For that reason, many institutions establish collaborations and share the investment costs of all types of high-precision measuring machines. The TU Wien has an ongoing cooperation with the "Universidad Tecnológica Nacional de Buenos Aires" in Argentina.

Research in metrology is conducted either manually on the machine or over directly connected computers containing software for the specific machine. Control input directly addresses the actuators of a machine, which can either perform a shift (translational displacement) or rotation. The controls can be combined into more complex behavior such as spherical scanning or recording Computer Numerical Control (CNC) programs for later re-use. In many cases, measuring occurs tactile, that means by approaching a work piece until the sensing head slightly touches it. Stronger contact is called a *collision* and can lead to damaging the sensitive scanning head. The challenges posed by the tactile measuring process are the scope of this thesis.

As measurement setups cannot be relocated easily, researchers have to travel extensively for utilizing the machinery, which leads to even higher costs. It would be desirable to

[1] Austauschbau und Messtechnik, TU Wien - http://aum.ift.tuwien.ac.at/.

remotely operate metrology setups for avoiding traveling expenses and thus making the investment economically more attractive. Furthermore, remote controlling would increase the number of potential research partners. Geographical distances would turn into an advantage, since the time difference would enable utilization around the clock. As it stands presently, most laboratories are closed at night.

However, remote operating raises some noteworthy questions about reliability, security and fail-safety, since delays within the network connection or other disturbing factors could lead to damage of the precious equipment.

1.2 Research Questions & Expected Result

Research in remote metrology spans a spectrum ranging from the educational effects of conducting remote and simulated metrology experiments, to the technical feasibility of solutions, as well as implementation strategies and the effect that the control mode has on the outcome of measurement procedures. This thesis focuses on the control strategy and metrological results. In the course of the thesis, a solution for location-independent operation of high-precision metrology machines will be developed to answer the following research questions:

Can remote and simulated laboratories be successfully used for operating highly-precise metrology equipment via the internet?

Q1: *How should a software-system for operating CMMs be designed to allow for the same accuracy at both remote local and internet control?*

Q2: *How does CMM control via the internet perform in terms of duration of operation compared to remote control via local network?*

Remote control in precision metrology is in its early stages and it is still uncertain for which applications it will become effective and efficient enough to win recognition. Educational institutions, research institutions and industrial companies have different requirements for effectiveness and efficiency. Answers to the selected research questions could serve as an indicator for the future development of the field and point to aspects of remote controlling metrology equipment that need to be researched more extensively.

The expected outcome of this thesis is an analysis of the additional challenges imposed by remote controlling metrology machines. The focus is on the safety of the equipment and ease of use.

An integral part of this analysis will be to find a system architecture that allows for the control of the actuators of metrology machines in order to examine the surface of work pieces, independently of the actual data input device and physical location of the operator. It should be designed to be easily expanded to different types of measuring machines. A software component will be programmed, converting user input into controls for the

machine. Functionality will be proven via a specific implementation of the architecture for operating a Coordinate Measuring Machine (CMM) with haptic sampling. This implementation will be used to answer the research questions and will further be available to support the research community in answering additional questions and developing new remote control solutions for metrology machinery.

By implementing the proposed safety measures the software should compensate for problems resulting from the transmission of the control input over networks without guaranteed quality of service that are not observed when the machine is directly connected to a computer.

For remotely operating the machines, a graphical user interface will be created. The control experience for the operator should be similar to the previous experience of local control, but may be adapted to different control mechanisms.

Due to the limits of a thesis, it is not a target nor possible to implement all the functions of typical control software in metrology, but to implement basic functionality and allow for easy extension. The implementation should allow for performing an interactive point probe on a sample workpiece.

The location-independent operation should enable better utilization of available machines, and more fruitful cooperation between research institutions and/or businesses.

1.3 Methodological Approach

The methodological approach comprises five phases:

- The first phase is the analysis of state-of-the-art technology. The functional and nonfunctional requirements of a control framework for metrology machines are surveyed by discussions with experts from the National Technical University (UTN) in Buenos Aires. Additional information about how present metrology control software satisfies those requirements is also collected.

- In the second phase, the challenges of controlling real-time systems via unreliable connections are researched from the literature and recommendations for action collected. Insights gained are considered in the architecture design in phase four.

- In the third phase, a CMM is modeled within a simulation software with the necessary grade of detail to control the single actuators. This simulation model is used for identifying additional needs and verifying the specified requirements. An investigation of potential solutions is conducted, and the progress of the implementation is observed with the help of this model. It is used for testing during all stages of development.

  A further advantage of using a model is the need for abstraction which directs the focus on the relevant parts of the system. With the chosen approach, different

configurations and extreme conditions can be tested, that could not be tested in the real world due to a lack of flexibility of physical machines, higher risk of damaging expensive equipment and opportunity costs of machine downtime. Machines that are currently not in the inventory of accessible metrology labs can even be simulated, which helps developing a more open software architecture.

- In phase four, the system architecture is designed considering the findings of earlier phases, and a prototype created for further research. The protoype also serves as a verification of the design to answer research question Q1.
- In phase five, the prototype is used to conduct local and remote measuring experiments on the simulated CMM. The results are then compared to assess the performance of the remote control, answer research question Q2 and then evaluate the thesis.

1.4 Structure of the Work

The thesis is structured as follows. This introductory chapter is followed by an elaboration of the basics of modern metrology laboratories, machines and measuring practices in Chapter 2. Chapter 3 provides an overview on Networked Control Systems (NCSs) in research and education and what is required to positively influence learning outcomes. Building on those theoretical foundations and practical experiences, in Chapter 4 the collected information is used to design a software-system architecture. It is demonstrated on a sample implementation in Chapter 5 how the design recommendations can be put in practice to operate a CMM in a simulated laboratory. In Chapter 6, internet delay patterns and their potential influence on remote real-time operation are discussed and their factual impact probed by performing local and remote experiments on the CMM control system. The thesis is concluded by a discussion about the main findings and future work in Chapter 7.

CHAPTER 2

Production Metrology in Practice

2.1 Limits of Metrological Reproducibility

Industrial Production Metrology is a fundamental tool to gain information and knowledge in all phases of the life-cycle of a product to help link the separate processes. [KPS+05] It is the subdivision of metrology which is concerned with controlling industrial manufacturing, testing technical products with high accuracy and improving the quality of various products. [ORDH95]

Progress in natural sciences and technology in the past decades have led to significant enhancements in production metrology. The precision of measurements has improved by several magnitudes and a multitude of metrology machinery was developed, providing unprecedented insights and allowing for more precise manufacturing. All of these advances have, in return, inspired engineers to design increasingly complex elements and miniaturization of components for reducing material costs, weight and energy loss.

As a result, accuracy requirements nowadays are often beyond what is measurable under unregulated, ambient conditions. In order to overcome this obstacle, specifically devised laboratory conditions must be established prior to a measuring procedure. External factors with the potential to distort results must be eliminated as much as possible. Several standards have been established in the industry to ensure reproducibility. In the following, some of the efforts to meet those regulations are explained.

2.1.1 Measurement Deviations

No measuring instrument is one hundred percent accurate. A measured value can therefore not be referred to as True Value but instead only as Correct Value with a particular uncertainty. Measurement Error as defined in DIN 94:1994 is ”the measurement result minus the true value of the measurand”, which means the deviation from the true value of the measurand. It is an error component individual to each measurement and therefore it

is not reproducible or predictable. The measurement error depends on various influential factors and is naturally unknown. [DIN94]

Another performance figure of a measurement procedure is the Measurement Uncertainty. ISO14253-1 [ISO13b] states that no measurement is complete without a declaration of its respective measurement uncertainty. Measurement uncertainty is a relatively new criteria in comparison to the measurement error. It distinguishes between a systematic error component, dependent on the measurement instrument and a random error component of the measurement. The closeness of the mean value of measurement of a measuring instrument to the true value of the measuring object is called Accuracy of the instrument. The systematic error is the mean deviation from the true value and is often referred to as the Bias of the instrument. Precision of an instrument is a figure for the repeatability or reproducibility of results and it describes to which degree multiple measurements on the same instrument lead to the same results. [JCG12]

Measurement error and measurement uncertainty are connected to a particular measurement. A characteristic number to evaluate and classify the performance of a measuring device is the Limit of Error. The Maximum Permissible Measurement Error or Limit of Error is the extreme value of measurement error, with respect to a known reference quantity value, permitted by specifications or regulations for a given measurement, measuring instrument, or measuring system. [JCG12]

The multitude of factors distorting measurement results and leading to measurement errors and uncertainty can be categorized into five groups: Human, material, method, environment and machine. For each measurement procedure the relevance and impact of those factors differs. Each factor can be of systematic or coincidental impact. Known systematic error factors might partially be avoided or corrected, which decreases the measurement uncertainty.

Human error can be reduced by extensively training and keeping the personnel focused. The measurement method must be chosen to allow for the actual evaluation of the measured object. The material of the measured object can be checked to avoid errors through unexpected physical behavior caused by material impurity. Inaccuracy of the measurement machine should be detected at regular inspections and can be lessened by re-calibration. In order to minimize environmental distortion factors, measurements are conducted in specifically devised and constructed measurement rooms.

2.1.2 Measurement Rooms

The structural organization of a measurement room ensures that most disturbing influences will be reduced and kept constant. The most important environmental influences are:

- temperature (thermal conduction, convection and radiation)
- vibrations

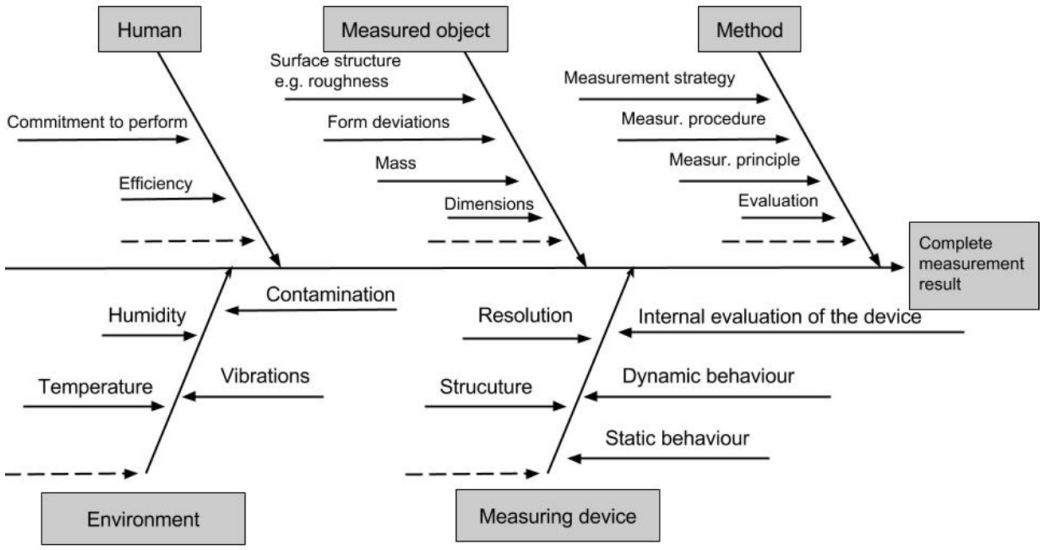


Figure 2.1: Cause and effect diagram of production metrology. (from [Tü15])

- humidity
- pollution

Air pressure and air speed are also factors in some measurements. In order for the measurement results to be accepted by experts, the environmental conditions during the moment of measuring need to be documented. [WS00]

The stability properties of the aforementioned factors define the categorization of a measurement room. Rooms with better environmental constancy may be used for more precise and critical metrology tasks, but are more costly to build, maintain and operate.

The reference temperature of length metrology is 20 °C. For example, precision measuring rooms of category 1 need to maintain this temperature and allow for temperature variations of only 0.4 K within seven days. Temperature and humidity can be controlled via an air conditioning system. In order to keep the amount of air particles below the necessary thresholds, special air filter systems are inevitable. Furthermore, human operators are only permitted in a limited number and for the shortest possible duration, due to body heat emission. Operators need to follow a protocol upon entering a clean measurement room and wear protective clothing to avoid outside contamination. The basis for planning and dimensioning the air conditioning and filtering system results from an exact analysis of all expected interferences.

One challenge that designers of measuring rooms face, is the trade-off between size and efficiency. Bigger measuring rooms with higher air volume are less susceptible to temperature rise through operators and surpassing the air-particle limits. Free space for additional machines provides opportunities for future expansion of activities. On the

Description	Grade class	Basic Tempera-ture	Temperature variations (K) during 15 min 60 min 4 hr 12 std over 24 std 7 day	Temper-aturegra dient (K/m)	Humidi-ty variation (%) within 30 - 60	Nadir acceleration (m/s^2) under 10Hz over 70 Hz
Precision measuring room	1	Reference temperatur e	0,2 0,2 0,2 0,2 0,4 0,4	0,1	10	0,02 0,2
Fine measuring room	2	According to definition	0,4 0,4 0,6 0,8 0,8 1,0	0,2	20	0,04 0,3
Standard measuring room	3	According to definition	- 1,0 1,5 - 2,0 2,0	0,5	20	0,04 0,3
Measuring room close to production	4	According to definition	- 2,0 3,0 - 3,0 4,0	1,0	30	0,06 0,6

Figure 2.2: Characteristic Values for Different Classes of Measuring Rooms following VDI 2627. (from [DIN95])

other hand, cost factors increase and it is more difficult to ensure uniform ventilation in bigger rooms. There is a limit to the efficacy of air conditioning systems in measuring rooms. If they produce accelerated air speeds, the measurement could be influenced.

The suitability of a planned location for a metrology laboratory or measurement room for quality control in high-precision manufacturing needs to be investigated extensively before the location can be approved. Important here is a ground-analysis on vibrations. Once approved, it must then be ascertained whether a flexibly stored foundation is required for the building. A thick base should be built so that it does not have contact to the remaining foundation. This prevents a transfer of building oscillations on the installed measurement devices. The base plate and the walls of the room need to be thoroughly insulated to avoid thermal conduction. [WS00]

These systems necessary to ensure stable environmental conditions make rooms for highly precise measurement vastly cost-intensive. Therefore, it is crucial to properly plan ahead, as alteration of the room structure can be financially demanding.

2.1.3 Traceability and Calibration

Workpieces and their components must fulfill a set of requirements regarding their geometry. In a globalized economy, where components are produced by a number of suppliers across the world and re-production and re-shipping is often not feasible, manufacturers must be able to rely on the quality of delivered goods. At assembly, all the pieces have to fit together. Out of this necessity, the Geometrical Product Specification (GPS) has been developed. It includes requirements of

- Dimensional Tolerance:

  Tolerances of step, dimensions, distances etc.

  Tolerance of size (linear and angular)

- Geometrical Tolerance:

  Form

  Orientation

  Location

  Run-out

- Tolerances on surface texture parameters:

  Roughness

  Waviness

By now, a whole family of GPS standards which are all closely related has been established in the industry. They contain rules of specification regarding global manufacturing principles and processes, as well as characteristics of utilized machine elements. Geometrical product specification and verification standards cover various steps of the production life cycle such as design, product development, manufacturing, quality assurance and metrology. [DOAS01] After initially creating multiple competing standards in this field, the International Organization for Standardization (ISO) published an internationally acknowledged document for the specification of geometrical tolerances and deviations, the ISO 1101.

Another international standard, specified in its current version in ASME Y14.5-2009 (for the USA) and in ISO 1101 for the rest of the world, is Geometrical Dimensioning and Tolerancing (GD&T). It is a language of symbols that is used to accurately describe geometrical requirements in mechanical engineering drawings. [ISO13a]

The efforts on reducing external influences on a measurement result are made to improve the conformity of a product to its specification, in other words, to improve the production quality. Efforts put into standardization and documentation are necessary to ensure that a product is of a certain quality. In order to justify the trust placed in a manufactured product, all of the the quality assurance steps must be carried out and documented, in compliance with the standards. This feature is called *Metrological Traceability.*

According to [JCG12], metrological traceability is defined as ”property of a measurement result whereby the result can be related to a reference through a documented unbroken chain of calibrations, each contributing to the measurement uncertainty”.

NOTES: A ”reference” can be a definition of a measurement unit through its practical realization, or a measurement procedure including the measurement unit for a non-ordinal quantity, or a measurement standard. Metrological traceability of a measurement result

does not ensure that the measurement uncertainty is adequate for a given purpose or that there is an absence of mistakes. The International Laboratory Accreditation Cooperation (ILAC) considers the elements for confirming metrological traceability to be (1) an unbroken metrological traceability chain to an international measurement standard or a national measurement standard, (2) a documented measurement uncertainty, (3) a documented measurement procedure, (4) accredited technical competence, (5) metrological traceability to the SI, and (6) calibration intervals (see ILAC P-10:2002). Metrological traceability requires an established calibration hierarchy.

This definition of metrological traceability obligates that all measurement devices utilized in a chain of measurements need to be calibrated regularly. Calibration is the "operation that, under specified conditions, in a first step, establishes a relation between the quantity values with measurement uncertainties provided by measurement standards and corresponding indications with associated measurement uncertainties and, in a second step, uses this information to establish a relation for obtaining a measurement result from an indication".

NOTES: A calibration may be expressed by a statement, calibration function, calibration diagram, calibration curve, or calibration table. In some cases, it may consist of an additive or multiplicative correction of the indication with associated measurement uncertainty. Often, the first step alone of the above definition is perceived as being a calibration. A comparison between two measurement standards may be viewed as a calibration if the comparison is used to check and, if necessary, correct the quantity value and measurement uncertainty attributed to one of the measurement standards. [JCG12]

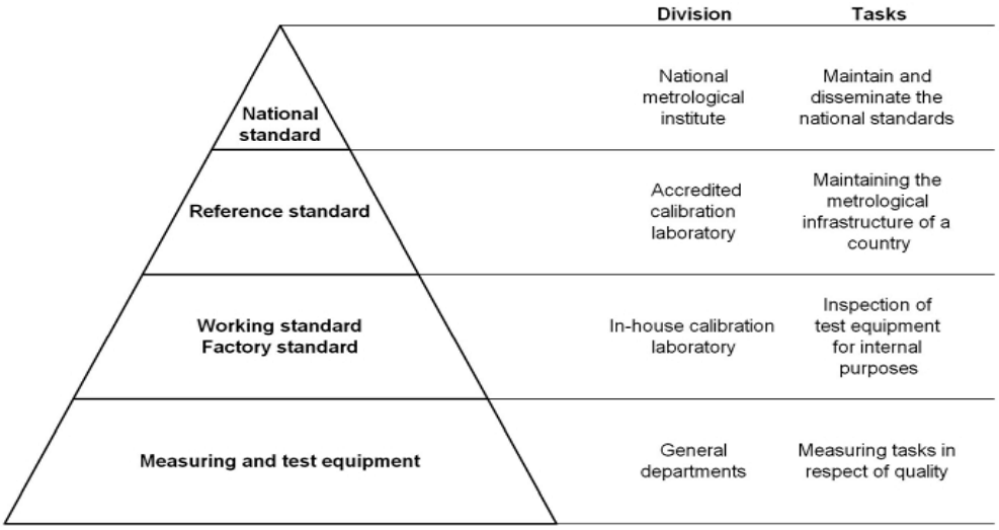


Figure 2.3: Levels of a Calibration Hierarchy. (from [Tü15])

As already mentioned, metrological traceability requires an established calibration hierarchy. That is a "sequence of calibrations from a reference to the final measuring

system, where the outcome of each calibration depends on the outcome of the previous calibration". Only in this way can the conformity of an end-user measuring device be ascertained.

NOTES: The elements of a calibration hierarchy are one or more measurement standards and measuring systems operated according to measurement procedures. Measurement uncertainty necessarily increases along the sequence of calibrations. [JCG12]

The means for the assessment of conformity of a manufactured product comprise of actions taken by all actors along the supply chain. Figure 2.4 depicts the conformity assessment tools defined by the ISO for first parties (suppliers), seconds parties (customers, regulators, trade organizations), and third parties (bodies independent from both suppliers and customers).

Tools for conformity assessment	First party Supplier, user	Second party Customers, trade associations, regulators	Third party Bodies independent from 1st and 2nd parties	ISO standards
Supplier's declaration	x			ISO/IEC 17050
Calibration, testing	x	x	x	ISO/IEC 17025
Inspection	x	x	x	ISO/IEC 17020
Certification			x	ISO 17021 ISO Guide 65

Figure 2.4: Standards of conformity assessment tools. (from [CSS11])

2.2 Dimensional Metrology

Metrology offers tools that allow for many physical characteristics of an object to be tested. Nine areas of metrology can be distinguished, which are split into branches. Each of these branches has its own field of research and engineering with respective machinery and processes.

In the following, we focus on dimensional metrology which is a branch of the area length metrology. In its most basic form, dimensional metrology can be thought of as the determination of length, angles, and other geometric relationships. However, the dimensional measurement process in industry is more complex than just analyzing the dimensions and tolerances of manufactured components. The product design specifications must be taken into account in the planning of the measurement process; the measurement process must be executed to obtain appropriate measurement data; the data must be analyzed and the results reported in a way that accepts/rejects the component and provides feed back to the manufacturing process that produced the component. [ZXK+11]

Optical methods have always been important for dimensional metrology. Telescopes, microscopes, the utilization of light interference and the invention of gas lasers have been important milestones in interferometry. More recently, the rapid growth of computational power and the wide application of opto-electronic components in consumer products has

Metrology area	Branch	CMCs
Acoustics, ultrasound, vibrations	Sound in air; sound in water; vibration	955
Electricity and magnetism	DC voltage, current, and resistance; impedance up to the megahertz range; AC voltage, current, and power; high voltage and current; other DC and low-frequency measurements; electric and magnetic fields; radiofrequency measurements	6586
Length	Laser; dimensional metrology	1164
Mass and related quantities	Mass; density; pressure; force; torque, viscosity, hardness and gravity; fluid flow	2609
Photometry and radiometry	Photometry; properties of detectors and sources; spectral properties; color; fiber optics	1044
Amount of substance	List of 16 amount-of-substance categories	4558
Ionizing radiation	Dosimetry; radioactivity; neutron measurements	3983
Thermometry	Temperature; humidity; thermophysical quantities	1393
Time and frequency	Time scale difference; frequency; time interval	586

Figure 2.5: Metrology areas and their branches, together with the numbers of metrological calibration and measurement capabilities (CMCs) as of 2010. (from ([CSS11])

increased the speed of development of optical metrology. Many components developed for consumer goods are now used for metrology purposes, a development which has drastically reduced the costs for a number of applications. Compared to some mechanical systems, optical methods can acquire more data in less time and have the possibility to measure without touching the part. These characteristics make optical methods especially useful for in-process measurement. Optical length metrology can be classified into geometrical optics, wave optics and quantum optics. [SNRPK02]

Limit Size	Measuring techniques
> 10µm	CMM, mechanical and pneumatic comparators, optical systems
10µm - 1µm	CMM, fine mechanical comparators, optical and electric comparators spin resonance
1µm - 100nm	CMM, electromagnetic and electrostatic comparator, optical interferometer, phase microscopes, dark field microscopes
100nm - 10nm	CMM, laser interferometers, roughness measuring devices, fluorescence microscope
10nm - 1nm 1nm - 0.1nm	Laser confocal microscope, X-ray microanalyzer, SEM, SPM (STM, AFM), electron and X-ray diffraction system

Figure 2.6: Measuring Techniques per Admissible Error Limit Size. (from [DOBB12])

Despite those optical methods being used and researched constantly, coordinate metrology is commonly used as reference method for calibration in dimensional metrology. As exemplification, in [KL16] the authors use tactile coordinate metrology for calibrating and evaluating their novel X-ray computed tomography (CT) measuring concept. The advantages of coordinate metrology include well-established traceability hierarchies and high reliability. Coordinate Measuring Machines can currently be deployed when the admissible limit of error is at minimum in the 10 nanometer range (see [figure: Measuring Techniques per Admissible Error Limit Size]).

2.2.1 Coordinate Measuring Machines (CMMs)

Coordinate measuring machines are of paramount importance within the mechanical metrology systems. They facilitate extremely accurate measurement and versatility in comparison to single-purpose manual measuring instruments. Recent years have seen an increase in flexibility, accuracy and speed which led to a quick ascent of the diffusion rate of CMMs in the airplane business. Before the introduction of CMMs, with conventional metrology tools, test pieces had to be aligned manually for each instrument or even measurement; measurements had to be compared with material measures (i.e. gauge blocks or kinematic standards) and the size, form, location and orientation was measured with different machines.

Coordinate Measuring Machines are machines that give physical representations of a three-dimensional rectilinear Cartesian coordinate system. Mechanically, a CMM comprises of three fundamental parts:

- The mechanical arrangement - with the axes referred onto the three coordinate systems.
- The displacement transducers in the axis directions, which allow any point in the measurement volume of the CMM to be covered by the measurements using a spatial reference point on the probe head.
- The probe head to probe the workpieces in all spatial directions.

Coordinate Measuring Machines gather various points (so-called point cloud data) of a surface and conduct calculations for determining the geometrical characteristics of that surface. [Tü15]

A critical part of modern CMMs is their software. Since the machine only probes a specified number of points, the remaining surface has to be constructed via interpolation on a computer. Some information about the surface is therefore required before starting to probe a geometry part. Increasing the number of points probed increases the time needed for probing an object but typically leads to more precise results. The CMM then estimates the indicated geometry and recognizes deviations and distortions. If present, specifications such as GD&T data are used to describe the intended, flawless geometry. In that case, the CMM then compares its measured results with the specification and calculates the deviations and violations of tolerances.

Depending on the type of control system a CMM uses, they may be categorized into one of the following groups. Each category with a higher number possesses a more comprehensive control unit and illustrates the development of CMMs over time.

- Category 1: Manually driven coordinate measuring machines
- Category 2: Motorized coordinate measuring with automatic probing process

- Category 3: Computer controlled coordinate measuring machines
- Category 4: Coordinate measuring machines linked with CAx systems

In manually driven CMMs (category 1), the control unit historically consisted of just a simple display to visualize the measured coordinates of a point. They require excellent mechanical properties in order to deliver precise measurement results. Environmental conditions during the measurement are usually not observed by such machines, nor do they perform complex calculations.

Category 2 machines are usually already connected to a computer that collects the measurement data, calculates geometries and prints a protocol. Before probing each surface, the human operators select the type of geometry that will be measured, using a physical control block that is mounted onto the machines body. Examples of such geometries are planes, cylinders or edges. The axes of the machine are controlled via this control block as well.

The control software of category 3 systems allows for a higher grade of automation by recording probe programs that may be executed an arbitrary number of times. This is especially useful when the dimensional characteristics of multiple units of the same object have to be probed. Category 3 machines represented a dramatic improvement in probing time in batch-production settings.

The introduction of category 4 systems occurred as part of an integration of the whole manufacturing process from design to quality control. Three-dimensional models of production goods, corresponding to standardized data exchange formats, can be imported into the machines memory and probe sequences automatically performed. In this manner, deviations from desired shapes estimated by the machine using the measured points of the measuring object can be determined, as well as deviations from the actual design. The essential advantage of the integration of model designs of objects is that they have already been created in the design phase of the manufacturing process, which saves time and rules out significant sources of errors. The translation of the computer model into a probe program for CMMs that completely tests all discriminable surfaces is a crucial step in the process.

However, this categorization represents the historic development of CMMs and does not necessarily make a statement about their usage modes. A modern control system might just as well allow for usage patterns of category 1, i.e. for manual shifting of axis. Hence, the categories can also be seen as modes of operation for a CMM control software.

2.3 Operative Mechanisms of Coordinate Measuring Machines

Coordinate Measuring Machines are contact probe systems. This means that a sensing head has mechanical contact with the measuring object in order to get a measurand.

The quality of a probe depends significantly on the quality of the sensing head. Tips of sensing heads are generally spherical and are produced in various diameters, each serving another application. As material for sensing head tips, ruby is used due to its hardness, homogeneity and durability. Ruby spheres can be produced as an almost ideal sphere with a deviation of only 0.25 micrometers. The other part of a sensing head is a stylus that carries the ruby tip and connects it to the suspension on the CMM. Multiple sensing heads can be mounted to modern CMMs at the same time in a variety of configurations, allowing for reaching points that are located on the probing system-opposing side of the measurement object. When multiple sensing heads are mounted, the operator has to indicate to the machine which of them will be touching the measurement object at the next contact. It is advised to mount the minimum possible number of sensing heads for a probe and the shortest possible stylus to minimize measurement errors. On contact, the probe system of a CMM sends an electronic signal to the control unit. Therefore, contact probe systems are also called electro-mechanical probe systems.

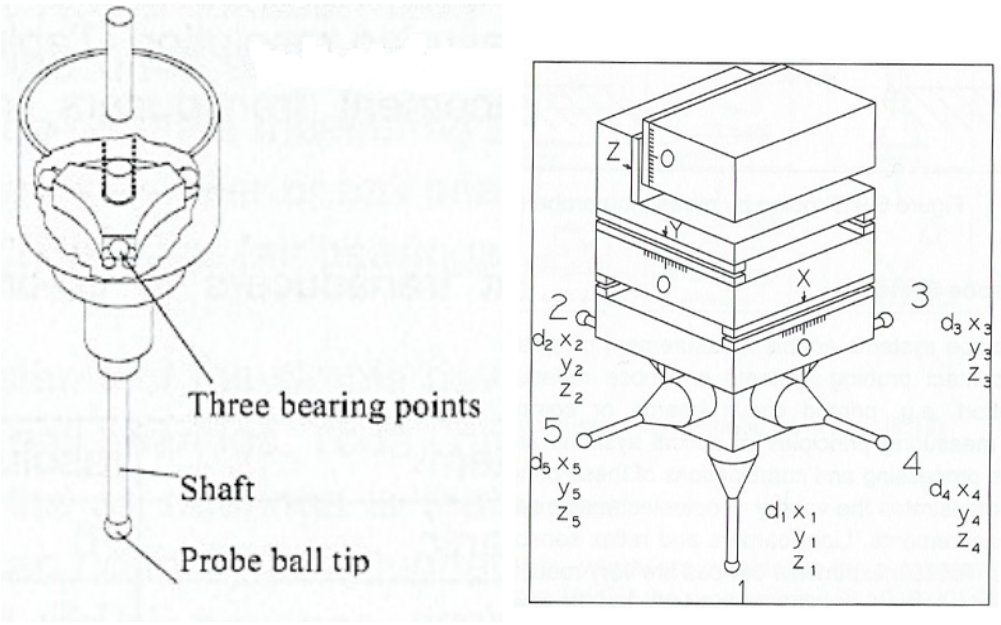
2.3.1 Types of contact probe systems

Amongst the contact probe systems, there are two types that can be distinguished:

- Touch trigger systems
- Measuring systems

Touch trigger systems only send one single electronic signal at contact. The sensing head of touch trigger systems is typically mounted elastically so that it has six spatial degrees of freedom. This ensures that the sensing head does not break immediately when excessive force is exerted but it can swivel out up to a certain angle. The touch signal always contains the sensing head position on each of the three machine axes at contact. Depending on the generation of the probe system, it may additionally encode the direction from where the contact occurs, and can even encode the displacement of the sensing head compared to its neutral position.

Measuring probe systems facilitate the scanning of a workpiece surface with continuous time or distance dependent data transfer to the control unit. More than 250 measurands per second can be obtained that way. Naturally this requires significantly more complex electronic control and typically a different mechanical construction is used. Three Cartesian length-measuring systems, carrying spring parallelograms, form the body of the probing system. Displacement of a parallelogram induces an analog signal providing information about force and direction vectors. The applications of measuring probe systems are manifold and they also economize the probing of free form surfaces. Due to the much more complex mechanics, electronics and software requirements, transition from touch trigger probe systems to measuring probe systems is very costly and this step often poses a financial challenge to businesses and universities. [Dur03]



(a) Touch trigger probe system. (b) Measuring probe system.

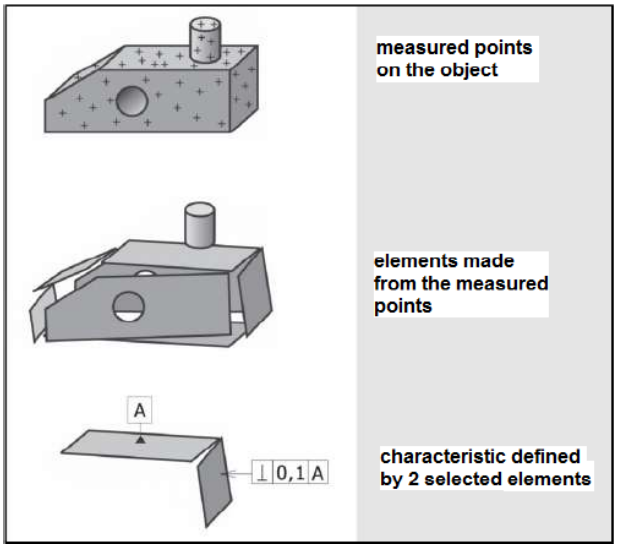
Figure 2.7: Mechanical principle of touch trigger and measuring probe systems. (from [Dur03])

2.3.2 Measurement process with CMMs

Despite the advances in manufacturing technologies, for many reasons goods are still assembled from a number of standard geometries. For probing, goods are virtually decomposed into those geometries, which are gauged successively. Each geometry has its characteristic minimal amount of measuring points to be sufficiently defined. In this case, an ideal geometry is assumed and no deviation from this ideal shape can be calculated. Additional points may be measured in order to get a more accurate understanding of the actual properties of the test object. The CMM software then calculates an estimator for geometry properties as well as standard deviation and other statistical indices. It might even compare the measurement to the product specifications and judge about compliance with manufacturing error limits.

Figure 2.8a depicts how a complex element is composed of standard geometries like planes and cylinders. Before probing such a geometry, the CMM must be informed as to which shape will be measured, so that it can perform the correct calculations. Figure 2.8b shows how many points have to be measured at least, for approximating a geometry of a respective type.

The measurement of one point in the three-dimensional space often requires more than just one probing step. Such a case is measuring the position of a corner, where the position of each axis is probed separately to get a precise result. In the case of the line



(a) Decomposition of complex object.

element	minimal number of points for defining	examples
point	1	
line	2	
plane	3	
circle	3	
sphere	4	
cylinder	5	
cone	6	
torus	7	

(b) Standard geometries.

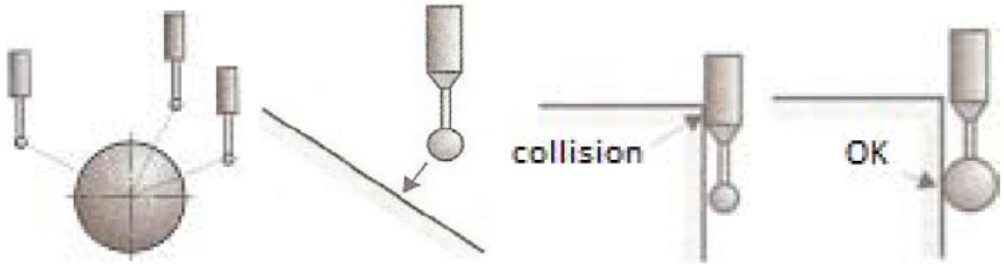
Figure 2.8: Decomposition of objects into standard geometries. (from [FK13])

and plane, only the orientation in space can be measured with two or three probing steps, whereas six probing steps are required to determine the length of the line. The approach taken for measuring a geometry must therefore be made known to the CMM.

Before conducting a probe, a measurement plan must be formulated. This plan contains all operations necessary to thoroughly measure the object and additionally includes the choice of sensing head configuration and which measurement will be performed with which head. Even though modern CMMs are technologically mature enough to make up for some ill-conceived usage by operators, a number of best practice guidelines should be followed to achieve suitable results. To name only a few:

- The measured object must be stably clamped and in a position such that a maximum number of required elements can be probed without requiring a readjustment.
- Measuring points should be evenly located on the measured geometrical element, circular and spherical surfaces measured by diagonally opposing points.
- Measurement points should be approached orthogonally or at least within $\pm 20\,^{\circ}$ from an orthogonal angle, in order to prevent slippage of the sensing head on the measurement object.
- Before accepting a measurand, operators must make sure that the indicated contact has been made by the tip of the sensing head and not the stylus or even a part of the suspension. Such an unintended contact would be called a collision and could lead to damage of the machine and highly erroneous results.

Figure 2.9 provides examples for those guidelines.



(a) Diagonal points at spheres. (b) Approach orthogonally. (c) Mind head collisions. (d) Use larger diameter probes.

Figure 2.9: Best practice guidelines for CMM operators. (from [FK13])

To get an understanding of CMM operation, the measurement process will, in the following, be broken down into coarse, and later detailed sub-procedures. The diagrams used are the result of interviews with experts in the field. Figure 2.10 thereby provides an overview of the complete measurement procedure. It is a merger of the work of Durakbasa [Dur03] and own results gained from expert interviews in the course of this thesis.

Each measurement procedure starts with the planning phase. In this phase, a step-by-step plan for the workpiece probing is created. Additionally, the plan contains all mandatory information about the machine parameters to set and the sensing head configuration. Information about the workpiece can be obtained from external sources such as an import from a CAD program where the object to probe was designed. This simplifies and accelerates the creation of the probing plan and might facilitate the semiautomatic - in the future even automatic - creation of a CNC program for unattended execution of the probing procedure.

Probe automation via a CNC program is useful for reoccurring tasks like quality control in the manufacturing industry, or when machine time is very scarce or expensive compared to the required programming time. Automated probes therefore continuously extend their share in all probes performed. Once created, the automation increases probing speed and reduces errors committed by human operators.

After planning and the potential programming phase, the coordinate measuring machine has to be physically prepared for the probing. This includes establishing the specified sensing head configuration as well as stably clamping the measurement object to the machine in a way, such that all required measurement points can be reached in the specified way by the specified sensing head.

The setup of the workpiece coordinate system is a mandatory step in the beginning of each probe. It sets the exact zero coordinates for all axes and results in more meaningful and legible results. Often, a corner of the measurement object in used as point of origin, and the remaining measurements are denoted as values relative to this origin. After

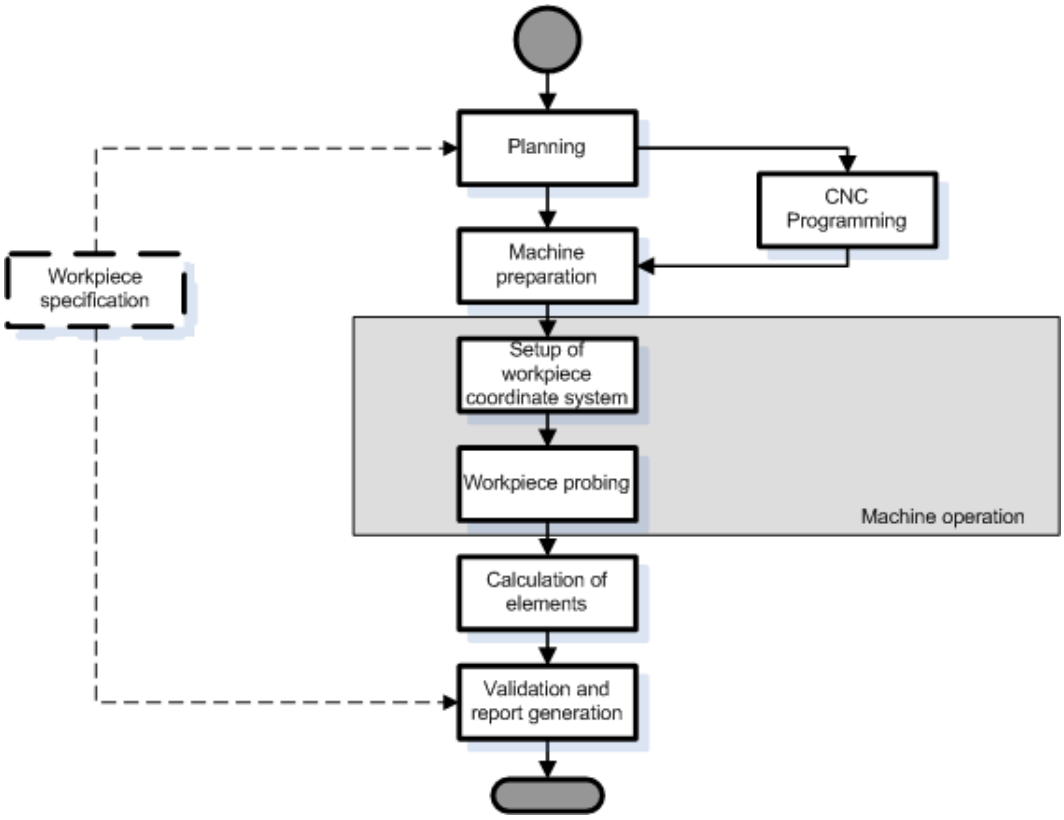


Figure 2.10: Measuring procedure overview.

the setup, the actual probing of the workpiece takes place. During this procedure, the geometries are probed in the specified order and the result is a protocol of executed actions and measured coordinates. The two phases *setup of the workpiece coordinate system* and *workpiece probing* constitute the mechanical activities performed on the CMM. Those will be further examined.

After conducting the probing operations, the results are used to estimate the actual geometries of the measurement object. Geometrical characteristics and deviations from optimal characteristics are calculated. Statistically, assuming no accuracy bias compromises the measurement, the measurement results should, with an increase of measurement points, converge towards the true value. As a consequence, a suitable compromise between the amount of measurements and the probe duration must be found.

As a last step, the measured results and calculated elements are compared to the specifications and production failures pointed out. For this validation, knowledge about the specification is required, which might again originate directly from the workpiece design files to provide an integrated manufacturing and quality control process. Output of the whole measuring process is a detailed report containing the performed actions and gathered results.

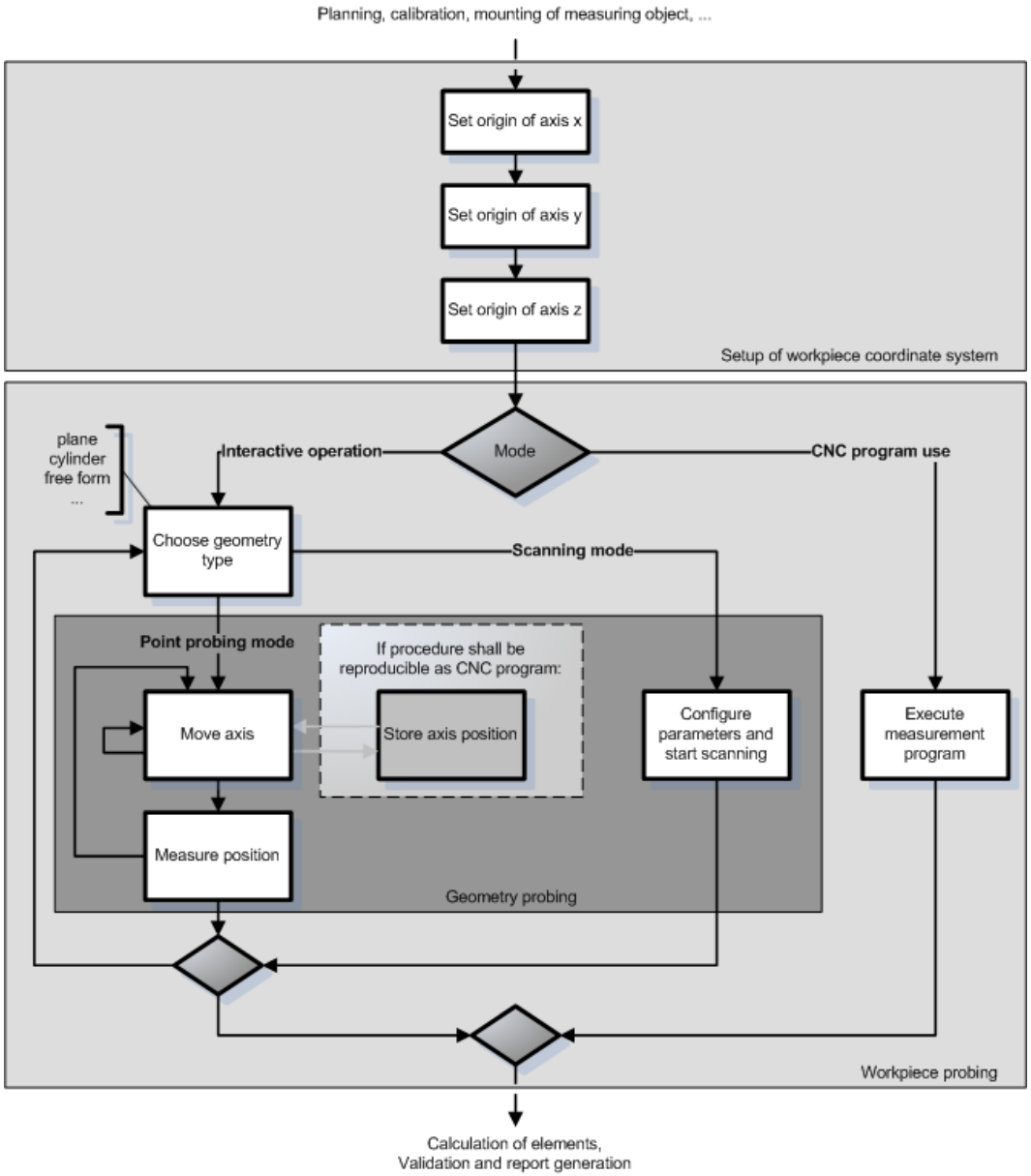


Figure 2.11: Process steps of workpiece measurement.

Next, the probing at the machine is analyzed. Figure 2.11 illustrates this process for both interactive probing by the user and using a CNC program. It immediately becomes clear that the user does not have to interfere with the workpiece probing, as it can be performed automatically. The program steps are usually encoded in terms of relative values from an origin. One action after the other is performed to probe the elements according to the previously defined procedure. Afterwards, the control computer continues with the

next phase - the calculation and estimation of elements.

Automatically determining the position of the point that should serve as origin is more complicated when there is no information about its location in advance. Therefore, the setup of a workpiece coordinate system is an essential step ahead of the probing of every single measurement object, and it is still frequently performed manually. Further automation of the measurement process by using robots to clamp the point of origin of workpieces precisely to a predefined position in the preparation phase, might also eliminate the need for human interaction in this phase.

When probing a measurement object manually, the procedure might depend on the type of probing system in use, or rather the mode of measurement. In point probing mode, machines are operated by moving the axes individually, whereat simultaneous movement of multiple axes is possible. Systems with a scanning mode have the ability to follow an edge or trail on a surface by observing the force and direction vectors on the sensing head and keeping them stable while moving. Manual moving of the axes is therefore not necessary, nor is it required to send an active command for retrieving the sensing head position when a measuring point has been reached. This manual position retrieval is the way to go in point probing mode. Axes movements and active position measuring alternate until the desired number of points has been probed.

A new risk of collision emerges, when the probing process shall be recorded for later reuse. The recording of moves in point probing mode of more than one axis at the same time ("diagonal moves") should be avoided where non-deterministic behavior at execution time can not be ruled out. This is done by storing the machine position after each axis movement, no matter if it is a point necessary for generating the geometry or not.

Regardless of the type of manual measurement mode, the process has to be repeated for each geometry, until the measurement object has been entirely probed and can be completely recreated in the succeeding calculation step.

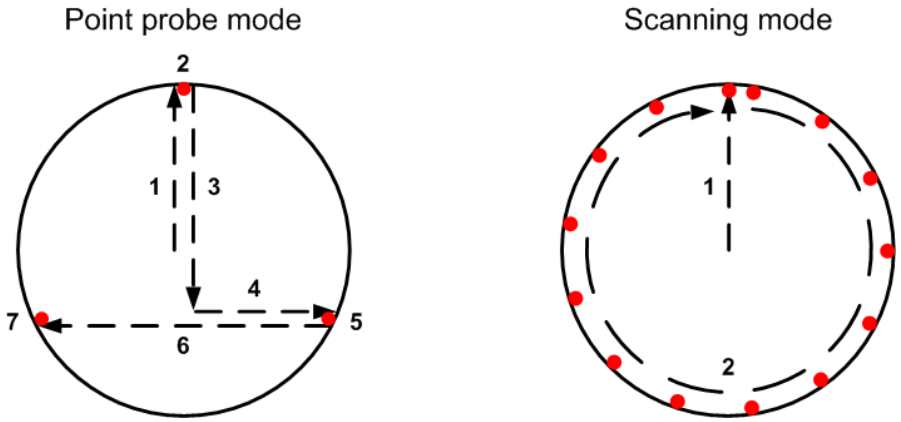


Figure 2.12: Comparison of measurement of a circle from inside, using point probe and scanning mode.

Figure 2.12 shows an example of how the probing of a single geometry is conducted. With the illustrated procedure, the center point of a circle and its diameter can be determined. In point probing mode, the operator has to perform seven actions to measure three points. Those three points represent the minimal possible information required to capture the circle. No conclusion about the roundness or deviation from a perfect circle can be drawn, and the result is highly error-prone. To achieve a more reliable result, the operator could measure more points, whereat each additional point requires two to three extra actions. Accurately determining the manufacturing quality of the circle by manual point probe would most likely not be economically feasible, by automated point probe probably still not. At scanning mode, the operator just has to lead the sensing head to touch the element once, configure the desired amount of measurement points within the circle or the distance between two measurement points and the measuring system probes equidistant points, according to the set parameters. A much higher number of points can be measured in scanning mode and the calculated elements are therefore more accurate. Furthermore, the deviation of each point from its expected location can be calculated and workpieces more reliably validated.

CHAPTER 3

Hands-on, Remote and Virtual Laboratories in Research and Education

3.1 Definition and Comparison of Laboratory Types

Quality control is an indispensable operation in the production industry. In science and engineering education, practical metrology experiments are essential likewise. Much research has been carried out on the exigency for laboratory tasks in those fields. [dZ13] While in the past, hands-on experiments on physical machines were the only tool available, now there are three types of settings for operators.

- Hands-On Labs: Hands-on labs involve a physical investigation process. This type of lab is also termed physical lab in the literature. [APA16] Two characteristics distinguish hands-on from the other two labs:

  All of the equipment required to perform the laboratory tasks is physically set up; and

  the operators who perform the experiment are physically present in the lab.

- Remote Labs: In remote labs experimenters obtain data by controlling geographically detached equipment. Similar to hands-on labs, they require physical space and devices. The distance between the experiment and the experimenter differentiates them from hands-on labs and alters the way operators interact with the device.

- Simulated Labs: Simulated labs are the imitations of real experiments. Programs simulate laboratory environments whereby operators can access and conduct experiments in a virtual space. The laboratory infrastructure is not real either, but

simulated on computers. This lab is also termed *virtual lab*. [APA16] One should be aware though that the term *virtual lab* in the literature is sometimes also used for the entirety of simulated and remote laboratories!

It is not objectively possible to say which of those settings is the best. Naturally they all have their advantages, shortcomings and applications.

Hands-on experiments are seen as too expensive under many circumstances, which is what inspired researchers and engineers to develop remote and simulated labs. Hands-on labs have a high demand for space, instructor time, and experimental infrastructure, all of which are subject to rising costs. Also, due to the limitation of space and resources, hands-on labs have been shown [CSC02] to be unable to meet some of the special needs of disabled operators. On the other hand, hands-on labs allow for perception of the experiment setup with all senses in the most direct way possible and avoid the creation of additional sources of errors. In a way, this makes hands-on labs more reliable than remote labs. [ET09]

Remote labs have the potential to provide affordable real experimental data through sharing experimental devices with a pool of institutions in research, education and industry. The opportunity to operate remote labs from a distance increases the number of times and places experiments can be performed. Remote labs are not bound to regular opening hours on a specific location, especially when the experiment preparation can be done remotely as well. Furthermore, more operators have access to the equipment. All this leads to a higher utilization of laboratory infrastructure and ultimately lower costs. In this way, non-standard machinery becomes more affordable, and the capabilities of the laboratory may be extended, in comparison to a conventional hands-on laboratory. Bigger laboratories are often run more professionally than small ones, and the personnel tend to be more specialized and experienced. These factors compensate for the inferior reliability compared to hands-on labs, caused by the introduction of additional error sources. There have been concerns about the acceptance of remote labs by students, but comparative studies show that students are motivated and willing to work in them. [ET09]

Simulated labs are of an entirely different nature. Reality has to be modeled in the most realistic way possible. They can be used in education, but are not able to probe the production quality of manufactured goods. Realistic simulations may take a large amount of time and expense to develop and still fail to faithfully model reality. Advantages of simulated labs include the absence of costs for operation and maintenance, as well as reduced risk of incidents leading to injuries or damage on the equipment, which is particularly valuable when operators are inexperienced, such as for educational purposes. A simulated laboratory can usually be accessed twenty-four hours a day, seven days a week, and experiments can be repeated numerous times. Multiple users can, in many cases, use the virtual lab simultaneously. Simulations can also be used as a temporary solution until a real laboratory is established.

Alam et al. [AHS14] provide a comprehensive overview of the major properties of hands-on, remote and simulated laboratories (see: Figure 3.1).

Feature	Hands on Laboratory	Simulated Laboratory	Remote Controlled Laboratory
Accessibility	Physical access to labs Advantages: • Realistic data. • Interaction with real equipment. • Open ended experiment. Disadvantages: • Space constraint. • Needs scheduling.	Virtual access via simulation tools Advantages: • Good for concept validation. • No time & space restriction. Disadvantages: • Idealised data. • No interaction with real equipment.	Via web and software Advantages: • No time & space restriction. • Close to realistic data. • Reality feeling. Disadvantages: • Virtual presence in the labs. • Highly dependent on synchronisatio n of hardware, software & internet speed.
Infrastructure	Hardware & software Advantages: • Offer students sense of reality. • Help students work in a team and under academic staff supervision. Disadvantages: • Finite life of hardware. • Needs maintenance. • Vulnerable to damage, theft & misuse.	Simulation software Advantages: • Good for conceptual understanding. • No safety issue. Disadvantages: • Need software update.	Hardware, software, internet Advantages: • Offer students to conduct repeat lab. • Useful if near realistic data is required. Disadvantages: • Finite life of hardware. • Needs software updating. • High speed internet connection.
Pedagogical	Advantages: • Interact with academic staff. • Offer students to collaborate. • Offer students to learn by trial & error. Disadvantages: • Students may not enough time to complete. • Supervision is required. • OH&S issue.	Advantages: • No OH&S issue. • Flexible to use software tools. • Enhancement through animation & virtual reality. Disadvantages: • Academic supervision not available. • No sense of real experiment.	Advantages: • Feeling close to real data. • Suitable for multi-campus & multi institutions program offerings. • Focus on conceptual understanding. Disadvantages: • Need enhancing both social &design skills.
Economical	Expensive capital & maintenance cost	Relatively low cost & no maintenance cost	Cost not clear yet but believed to be in between hands on & virtual labs

Figure 3.1: Advantages and disadvantages of hands-on, simulated and remote laboratories. (from [AHS14])

In order to better understand how laboratories are used in engineering education, the authors of [MN06] developed a four-dimensional skill-goal model. They researched twenty articles for each laboratory type and analyzed how often a development in each skill-group was mentioned as being a goal of using the respective laboratory. The dimensions are

- Conceptual understanding: Extent to which laboratory activities help students understand and solve problems related to key concepts taught in the classroom.
- Design skills: Extent to which laboratory activities increase the ability of a student to solve open-ended problems through the design and construction of new artifacts or processes.
- Social skills: Extent to which students learn how to productively perform engineering-related activities in groups.
- Professional skills: Extent to which students become familiar with the technical skills they will be expected to have when practicing in the profession.

The authors found that obtaining social skills and design skills was not often attributed as a potential learning perspective to remote and simulated labs by the researchers. Design skills have been mentioned as an educational goal within articles about remote laboratories only once.

Alkhaldi et al. [APA16] conducted a literature research and found that certain empirical studies reported no differences in student performance between physical and virtual laboratories. In other studies, virtual laboratories are shown to be advantageous over physical laboratories in acquiring conceptual understanding. Further studies have shown that students who have exposure to both physical and virtual laboratories outperform students who only conduct experiments in physical labs (e.g. [ZOP08]).

3.2 Examples for Remote and Virtual Laboratories

3.2.1 Examples for remote laboratories

Marques et al. [MVCL+14] performed a study to monitor the results of integrating the open remote laboratory system *Virtual Instrument Systems In Reality (VISIR)* into seven different courses all related to higher engineering education. VISIR is an open remote lab, designed for experiments with electrical and electronic circuits. The user interface replicates a physical breadboard, showing all available components and the instrument front panels so that a user can connect the desired circuit and analyze its behavior with several instruments. Previous studies showed the most positive evaluation was for the similarity between the real lab equipment and its remote counterparts. Most users also felt that VISIR should be used as a complement to hands-on labs, and that other courses would also benefit from its use. Figure 3.2 provides an example VISIR solution. The authors wanted to contribute to the scientific debate as to how remote

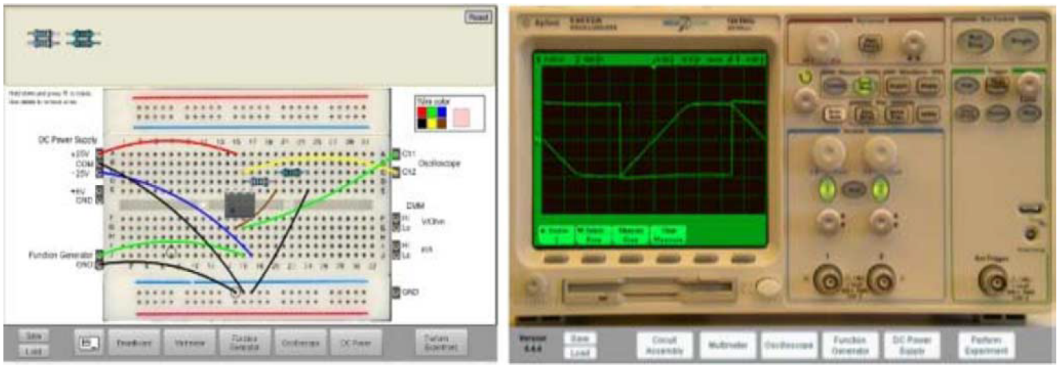


Figure 3.2: Example of VISIR interface windows: (left) circuit mounted on a breadboard; (right) oscilloscope interface. (from [MVCL+14])

labs compare to hands-on labs in terms of learning outcomes. A total number of 1,727 students participated in the study. Students who used VISIR generally had improved lab reports, improved lab examination results, higher grade distributions, statistically

significant correlations between VISIR accesses and the lab grade, and higher learning gains. According to the authors, for motivated students or students with a learning style more adequate to this kind of tool, VISIR is always useful. Still, particular conditions should be put in place to reinforce the effects for everybody, namely the following. (1) Students need a hands-on practice session before they start to use VISIR. In general, students encounter some difficulties when they start using the system; if teachers do not provide assistance and encouragement in overcoming these initial barriers, the probability of dropping out increases. The initial hands-on session helps to avoid this. 2) Both teachers and students consider VISIR more useful in introductory courses. [MVCL+14]

Axaopoulos et al. developed a series of simulated experiments for educational use and developed a system that can perform a real-time experiment remotely via the Internet in order to determine the characteristic curve of a photovoltaic panel. The panel, together with the instrumentation and automation modules, is placed on the roof of the laboratory and, using a web server, anyone interested can tilt the angle of the panel and observe the resulting energy conversion. The system consists of a PV panel, an electronically controlled ohmic load, automation and instrumentation devices, a web server with automation and instrumentation software and an outdoor web camera for live viewing of the PV panel. The automation and instrumentation devices are connected to the computer hosting the experiment through a twisted pair (UTP) cable in a daisy chain configuration, using the RS485 serial communication protocol. This set-up makes it possible to place the various devices at a total distance of 4,000 feet from the computer. In this way, critical signals from probes and other devices can be measured and controlled locally, avoiding signal loss due to attenuation on long cables. The servo motor operates on a closed loop so that the system has a feedback from the angle of the panel and, most importantly, can detect a malfunction on the panel's tilt mechanism. The software is implemented with Labview and has a web interface allowing for remote use, where multiple users can monitor the system concurrently, but only one can control it - whoever submits their choice for tilt angle first. Finally, the live view from a web camera gives the user a sense of personal presence in the place where the experiment takes place. [AMT12]

In an early work on remote real-time labs in electronics, Macias and Mendez criticize the inflexibility of common remote laboratories and that they only allow for the definition of initial conditions. Integrating data acquisition equipment controlled through a port in a computer and software specialized in measurement and control with the ability to reconfigure, in real time, a network/matrix of electronic components made it possible to deploy a remote Electronics laboratory, similar to a traditional laboratory. It makes measurements and control of the circuit fully available through the network without the user having to be physically present. For their work, the creation of an appropriate interconnection structure for remote laboratories was sought out, focused particularly on the Electronics area but with application to any other field with requirements for remote control and collaboration applications. The advance towards tele-engineering projects and educational laboratories, the authors state, leads to an integration of new data acquisition technologies, digital manipulation, and software-applied measurement

instrumentation - the so-called *Virtual Instrument (VI)s*. Their concept utilizes NI-ELVIS, a Data Acquisition (DAQ)-card and LabVIEW and project-specific electronics hardware. [MM07]

Stefanovic et al. [SCMS11] developed a remote experiment for controlling coupled water tanks using LabVIEW. The basic configuration of their system consists of a web server and one or more acquisition servers that control the experiments. Acquisition servers are personal computers with one or more Data Acquisition Systems (DASs) for measurements of physical quantities and control of the experimental equipment. Acquisition servers are connected with the web server using Win Sockets. Older DASs were programmed in C++, while the new DAS and the web server were programmed in C# .NET. The web server was implemented using Microsoft ASPX. Experimenters access the web lab via web browser as the only required tool on the user side. Each experiment can be controlled by only one remote user at a time, while others can observe the experiment from the VI and web cam live video. For the coupled water tanks experiment, the architecture had to be changed. No extra web server programming was required, as LabVIEW directly supports web access to experiments. Web server and acquisition server run on the same PC. The students who had access to the web lab (and could perform laboratory exercises on their own, repeat them, and analyze the results) achieved significantly better scores and fulfillment of educational goals compared to those who only practiced on the physical tanks. [SCMS11]

In their paper, Hossain et al. [HBRK+15] introduce the concept of interactive cloud experimentation for biology, which enables multiple users to execute live biology experiments over the Internet. Their system was not designed for real-time feedback control of one instrument by a single user, but aimed to enable high-throughput technologies to be shared across many users over the Internet concurrently while allowing iterative interactions. Many experiments were executed in parallel on individual Biotic Processing Units (BPUs), multiples of which operate autonomously and synchronously with two clock cycles. One clock polls the central server for a set of currently scheduled instructions that are multiplexed from several users; the BPU then compiles and executes these instructions. On the other clock cycle, the BPU takes experimental readings and demultiplexes these data for different users before sending them back to the central database (state data) and storage server (bulk data). Overall, this architecture is optimized to coordinate asynchronous user actions with synchronous equipment cycles to optimally utilize parallelized equipment. Further design goals were scalability and that users can access and run experiments anytime without having to book a time slot. [HBRK+15]

A distance learning laboratory for experiments on electronic instrumentation [CMZ06] has been developed at the University of Sannio in Benevento, Italy. The solution is integrated into the Learning Management System (LMS) of the university and can therefore be carried out using only a common web browser, with no need for specific software components on the side of the experimenter. Three user profiles have been created in its platform: student, teacher and administrator. The student has access to the following services:

- Experiment visualization: allows the student to display on his or her own computer a laboratory experiment typically held by the course teacher.
- Experiment control: allows the remote student to perform a pre-defined experiment, effectively controlling one or more actual measurement instruments.
- Experiment creation: allows the student to remotely create a new experiment.

The LMS provides user authentication and management as well as tracking the learning activities and progress. Figure 3.3 shows the building blocks of the Sannio solution. The *presentation tier* is based on the thin-client paradigm, which has been adopted

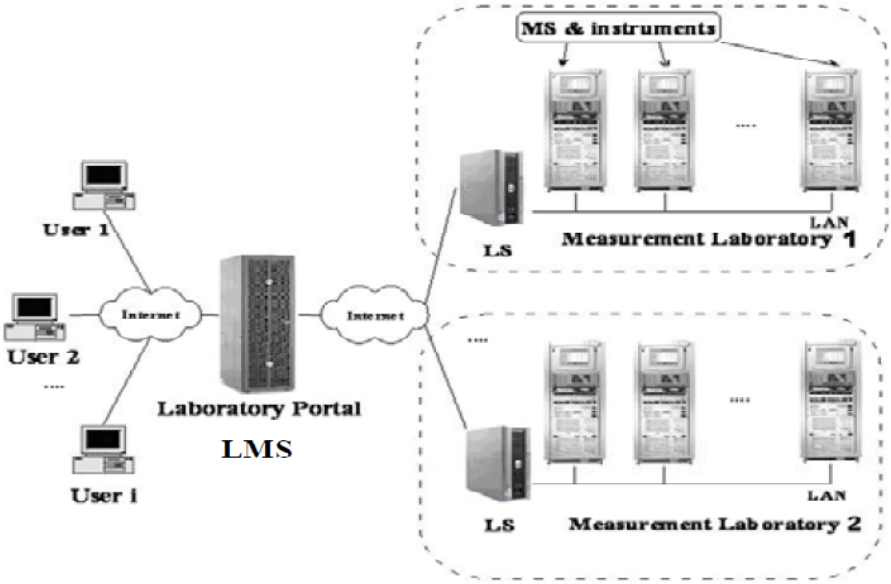


Figure 3.3: Block diagram of University of Sannio remote measurement laboratory. (from [CMZ06])

for realizing the remote access to VIs by using *ProperJavaRDP*, a Java-based remote desktop protocol implementation. The *middle-tier* consists of: (1) a web server hosting the LMS, serving as the so called *laboratory portal*; (2) one local *Laboratory Server (LS)* per laboratory to provide access to the lab equipment; (3) *measurement server*s (MS) which enable the interactions with one or more instruments. The server-side software component used to control the electronic instruments is LabVIEW.

After creating this remote experiment setup, a survey of seventy students was undertaken. The students had to perform magnetic measurement experiments in both the hands-on and remote laboratory. The remote laboratory was considered equally or more effective than the hands-on by 84.1 percent of the participants, 96.8 percent answered that applicable preparatory instructions were very important to successfully perform the experiments, while 87.3 percent demanded teacher presence for both types of experiments. [CMZ06]

3.2.2 Examples for simulated laboratories

Simulations can visualize additional information and the processes below the surface by abstraction and visualization. Finkelstein et al. [FAK+05] therefore state, "our results indicate that properly designed simulations used in the right contexts can be more effective educational tools than real laboratory equipment, both in developing student facility with real equipment and at fostering student conceptual understanding". Ever since, a lot of research has been done on this matter and various architectures have been created.

The virtual ChemLab project is a set of laboratory simulations for use in freshman- and sophomore-level chemistry classes and laboratories. The simulation software is a client-application that has to be installed on each computer before use, and is executed locally. Virtual laboratories in chemistry offer the advantage that time-consuming preparations such as setting-up the experiment are omitted, and the focus lies rather on the chemical principles and theories behind the experiment. They can, for the same reason of excluding the preparations, not replace chemistry wet labs entirely, but serve as additional practice lessons. A survey evaluating the success of the project found, that the average student generally did not like this kind of exercise, but showcased better understanding of the chemical processes. The survey results have been linked to another study analyzing the personality profile of those students and revealed some correlations. Utilization of the simulations was higher amongst creative learners compared to structured learners, whereas structured and precisely thinking students were more satisfied with the simulation than the intuitive, nonlinear and experientially oriented. [WAA+05]

Ding and Fang designed another local client application. Their simulation laboratory was created as a C++ program to explore the physical law of diffraction grating. The simulation was evaluated by 64 college students, including six interviews and 32 anonymous testimonies. Results attest that learning via simulation has potential in physics education as well and positively influences the motivation of students. [DF09]

The main contribution of Prieto-Blázquez et al. is the conception and definition of a general structure for virtual laboratories, applied to a specific virtual programming lab. Three key resource groups are identified: technological resources, pedagogic and strategic resources, academic staff resources. The authors name a number of simulators specific to information technology applications, but develop their own virtual lab as Java Applet that runs on web browsers. Another recommendation they give is creating virtual machines and allowing for remote connection to them for virtual or simulated lab experiments, such that operators do not have to go through a complicated installation procedure. Protocolling, logging, and - for educational purposes to reduce the workload - automatic assessment should be performed wherever possible. For evaluation, 284 distance learning students were asked about the presented solution. Availability and expertise of teachers, as well as learning methodology were mentioned as being of outstanding importance in the learning process. Critical for successful experiment realization was also the functioning of technological resources like simulator or virtual communication

environment. [PBHJGR09] Similar labs for networking and computation based on virtual machines (VMs) that demonstrate positive learning effects are presented in [HKG14] and [DWW+12].

In response to the Indian Ministry of Human Resource Development (MHRD) National Mission on Education through Information and Communication Technology (NME-ICT) Initiative, the Virtual and Accessible Laboratories Universalizing Education (VALUE @ Amrita) Virtual Labs Project was initiated to provide laboratory-learning experiences to college and university students across India who may not have access to adequate laboratory facilities or equipment. Adobe Flash is used for animating the virtual labs and Adobe ActionScript for the simulation engine that controls the animation. These virtual laboratories require only a broadband Internet connection and standard web browser with a Flash plug-in. The Sakshat Portal (www.sakshat.ac.in) is another component of the NME-ICT and is responsible for implementing a single web portal to provide access to all MHRD e-learning initiatives. At a workshop, 180 educators from more than one hundred educational institutions participated in a survey evaluating the Amrita Virtual Labs. More than 94% considered virtual laboratories to be an effective tool and 97% of the respondents felt such virtual labs sites assist in their job as a teacher. [ASS+11]

3.2.3 A virtual and remote robotic laboratory

The paper Chaos et al. [CCLOD13] describes the design and implementation of a virtual and remote laboratory for mobile robotics based on Easy Java Simulations (EJS) and LabVIEW. The application allows for interaction with a robot simulation (virtual laboratory) or with a real robot (remote laboratory), via the same Graphical User Interface (GUI) in EJS. This interface can be executed as a stand-alone Java application or integrated into *Moodle*, the LMS used by many schools and universities. In the EJS GUI, students can upload MATHLAB code for operating the robot, and observe its behavior via webcam.

The authors state many advantages of using the same interface for virtual laboratory and remote laboratory, since the operator can learn precisely how the laboratory works prior to using the real equipment.

- Efficient use of resources: Most of the assigned time on the real equipment can be used for actual operation because the users already know how to operate it. With other concepts, especially in the first learning phase, a lot of time would be spent on trial and error processes.
- Flexibility of timetable: The simulations are more readily available than remote laboratories, often around the clock. It can be practiced anytime and repeatedly. Real, hands-on or remote experiments must be supervised by a tutor in order to evaluate the practice and solve technical problems that may arise in the experimental session. Furthermore, simulations can be reset and require little, if any, changeover time.

- Freedom to experience: Learners may perform any experiment they want with the virtual lab without any danger to physical equipment. Thus, they can gain confidence in their work before testing on the real hardware. This increases the creativity during the design stage because they tend to be more conservative when they feel that the laboratory can be damaged.

Nonetheless, the virtual lab cannot substitute the real laboratory, as it is almost impossible to make a perfect model of a real system. Despite that, the virtual laboratory performs a model simulation of the robot that can be interacted with in the same manner as with the real robot. The virtual and remote laboratory architecture can be split into

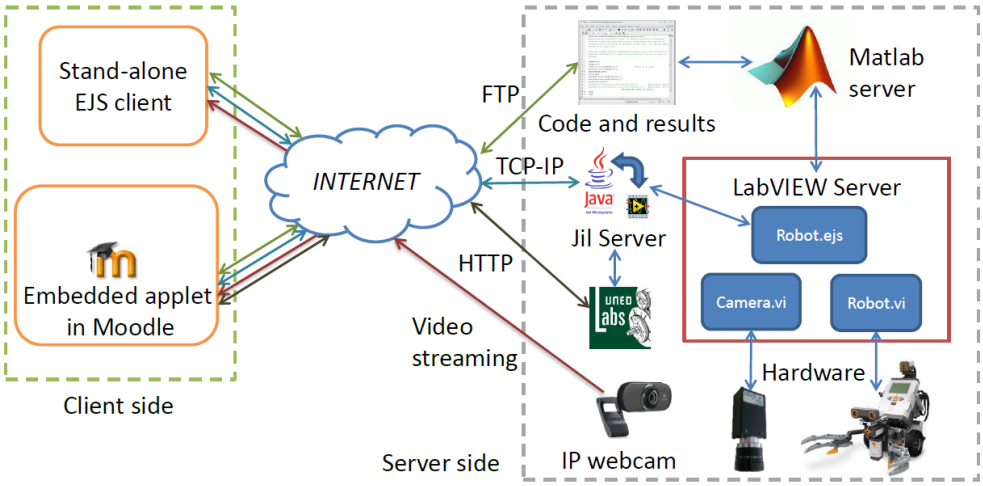


Figure 3.4: Laboratory architecture using EJS, Jil and Matlab. (from [CCLOD13])

client side and laboratory side, as depicted in figure 3.4. The client side is developed in EJS and either deployed as stand-alone application or embedded in Moodle as an applet. In the first case, the Java executable directly communicates: via FTP with a file server to load code for operating a robot to the server and save results; via TCP/IP to communicate with the Jil Server (Java Internet LabVIEW Server), a middleware which maps EJS variables to LabVIEW Virtual Instruments and can control the load and the execution of these VIs; and consumes the video stream from the webcam (available only on the remote laboratory). In the second case, the browser will additionally exchange information with the server that implements the Moodle courses. On the laboratory side, each robot is addressed via a controller.

In the presented experiment, the simulation of the environment only consists of generating an occupancy matrix and does not provide visual feedback in form of a camera or map of the simulated world. Other real-world issues that could not be reflected in the chosen simulation approach include: the model does not have the same complexity as the real robot. A real motor suffers from nonlinear effects like saturation on acceleration and dead

zones, in some situations wheels may slip; real sensors are affected by noise and outliers on the measurements; different sensors in reality have different delays; the Bluetooth communication with a real robot introduces a delay of 250 ms on the range measurements of the ultrasonic sensor, which makes the robot harder to control, while the simulation uses only 59 ms per picture; sometimes temporal failures in real communication lead to the loss of measurements, which can lead to mechanical problems of the vehicle, like collisions with obstacles, that are not simulated. [CCLOD13]

3.3 Initiatives for Sharing Remote and Virtual Laboratory Resources

The VALUE program [ASS+11] at Amrita Vishwa Vidyapeetham University, as previously described, provides virtual lab resources to people and regions that do not have access to real hands-on or remote laboratory equipment otherwise. Amrita University is part of a consortium of twelve institutions building over two hundred virtual labs covering nine key disciplines in science and engineering. The potential of virtual laboratories in developing countries is highlighted as the demand for high-skilled labor rises globally. However, remotely sharing laboratory resources is a growing trend for economic reasons and improving accessibility too.

With the iLab Shared Architecture (ISA), the Massachusetts Institute of Technology (MIT) developed a web service infrastructure to provide a unifying software framework that can support access to a wide variety of online laboratories. Experiments integrated into the ISA can be (ordered by their grade of interactivity): (1) batched experiments, in which the entire process is specified before the experiment begins; (2) sensor experiments, in which users monitor or analyze real-time data streams without influencing the phenomena being measured; and (3) interactive experiments in which the user monitors and can control one or more aspects of the experiment during its execution.

The main purpose of the ISA is to educate faculty and students, research is an added benefit, but not the initial goal. Educational institutions world-wide may integrate their remote laboratories instead of integrating them into own LMSs or having to provide other modes of access and distribution. The architecture is decentralized, in the sense that each organization manages its own students' accounts, lab time scheduling, data storage, etc., but provides a mechanism for authorization and access control. It also allows users of commercial software, particularly National Instruments LabView, to couple their applications with the iLabs web services.

Early experiments were operated via Java Applets in a standard web browser. Now, a variety of technologies can be used and web services serve as the communication framework for the ISA middleware. The architecture comprises both lab-side services (e.g. the online lab itself) and client or student-side services (authentication and authorization, class management, student data storage for experiment specifications and results as well as user preferences). [HDL+08]

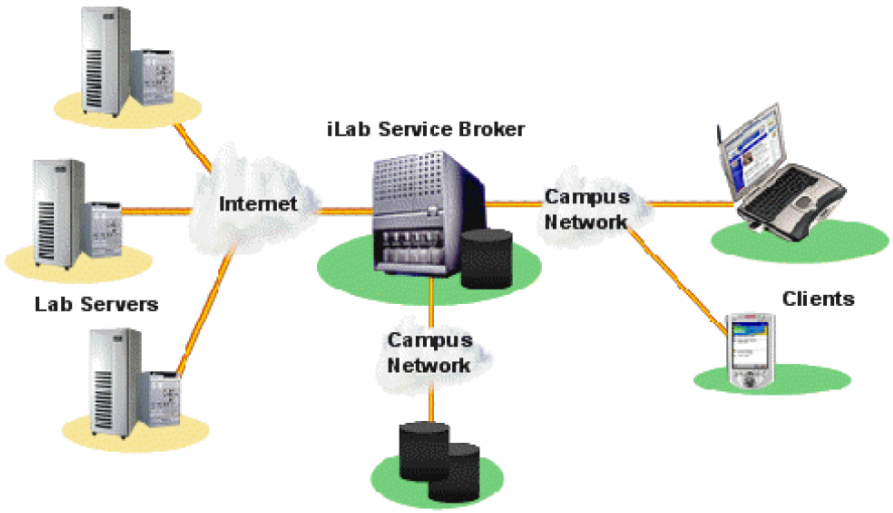


Figure 3.5: Three-tiered iLab shared architecture. (from [MIT18])

The iLab architecture separates online labs into three distinct modules connected by a Web service architecture.

- The Lab Server is located in the realm of the lab owner and deals with the actual operation of the lab hardware.
- The Lab Client runs on the user-end computer, and provides the interface to the operation of the lab.
- The Service Broker mediates exchanges between the Lab Client and the Lab Server and provides storage and administrative services that are generic. It can be shared by multiple labs within a single university. [MIT18]

The service broker can interoperate with any combination of client and lab server that implement the appropriate interfaces expressed in terms of web service Simple Object Access Protocol (SOAP)-calls defined in Web Service Description Language (WSDL). A student starts a session by logging on to the service broker using a standard web browser. Once the student chooses the experiment to execute, the client is launched and communicates with the service broker using the client to service broker web service. The service broker stores a copy of the experiment specification it received from the client, before forwarding it on to the lab server via a service broker to lab server web service.

For batched experiments, a lab client/server communication framework (LC/SCF) is defined in any text format, ideally Extensible Markup Language (XML). Using XML, the service broker needs only to be able to pass text strings in order to provide the communication of typed data records between any lab client/server pair. Java technologies are popular client development environments but PHP/Ajax/JavaScript frameworks or

Windows Forms (.NET) are also used. All communication between the client and lab server passes through the service broker, at interactive experiments this is not the case.

Allowing direct communication between the user client and lab server gives developers the freedom to choose their own communication protocol and to use third-party packages like LabVIEW and MATLAB in their development. The lab server should provide an abstraction layer that segregates generic modules from lab dependent code. In the interactive case, the middleware can have the following tasks:

- Experiment Storage Service (ESS): The service broker, the client, and the lab server may need to store information to record the complete experiment. An independent ESS is published as a Web service to handle the potentially high-bandwidth traffic from the client and lab server during experiment execution. Both XML and binary data from a particular experiment, but no corresponding administrative information is stored there.

- Scheduling Services: To sign up in advance for time on a particular piece of lab equipment. The scheduling application should also notify users if their reservation must be cancelled or changed and must allocate time for necessary actions before and after the experiment execution. Given the different requirements from the lab-side and the student-side perspectives, both a lab-side and a user-side scheduling service is useful.

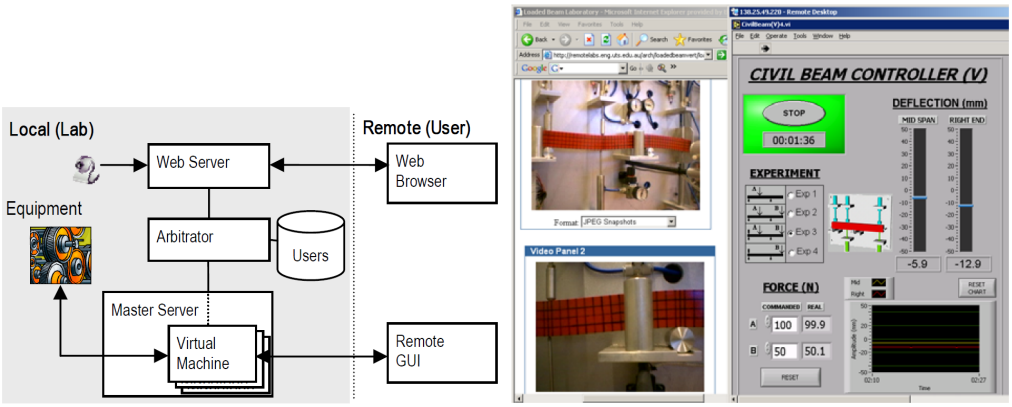
- Authentication and Authorization: Must grant permission to register for a time slot and to access the laboratory equipment. Performed by the Service broker. Different authentication mechanisms are possible.

Built upon the ILS generic classes, the LabVIEW Integrated Interactive Lab Server (LVILS) provides interfaces between the .NET web service application of the ILS and LabVIEW processes. Either a .NET page or the LabVIEW web server can be used as the front panel to an experiment. [HDL+08]

LabShare is a joint initiative of the Australian Technology Network, led by the University of Technology Sydney (UTS), aimed at providing shared remote labs across higher educational institutions in Australia. Remote laboratories have been developed for different engineering sectors, using a number of different technologies. Some involve the use of Linux hosted software development tools which are character-based and accessed through terminal sessions, yet provide a web-based output user interface. Others require Windows-based development tools to be available to the user in order to create control programs for industrial programmable logic controllers, and still others have been constructed to present a LabVIEW derived application to the user to manage the testing of control algorithms for coupled tank apparatus models. [MLL+08]

The resultant architecture is shown in figure 3.6a. A remote user logs-in through a web browser (with authentication managed by an arbitrator) and requests access to a set of

equipment that the arbitrator allocates if available. The student can then monitor the equipment auditorily/visually, through the web interface. For controlling the equipment and the differing user interfaces associated with the control applications, the arbitrator provides a Windows virtual machine on a master server (using VMware), that the authenticated student can now access. The student creates a remote desktop connection to this virtual machine, runs the control application and performs the experiments. An example of this kind of hybrid user interface of video stream in the browser and control via remote desktop client can be seen in figure 3.6b. Additional software on the client side is not required. When an experiment session is completed by a student, the arbitrator reclaims the apparatus, re-initializes it, and returns the device to the free pool. [LMLL09]



(a) UTS LabShare architecture. (b) LabShare hybrid user interface example.

Figure 3.6: UTS LabShare remote laboratory facility architecture (a) and hybrid user interface (b). (from [LMLL09])

In the course of the Labshare program, various studies have been carried out amongst the participants. One result was, that the most important factors influencing the perception of controlling real equipment were whether or not they had the right Programmable Logic Controller (PLC), the network had too much lag/latency, and especially whether a live video feed of the equipment was available. The long term outcomes depended more upon constructing a convincing scenario that envelopes an experiment. [LML08]

The *Library of Labs* (LiLa) project [RTB11] collects remote physical and virtual laboratory experiments on a European and world-wide basis, makes them available through an online portal that allows sharing and exchange of experiments, and offers lecturers the opportunity to download these experiments and integrate them into Learning Management Systems (LMS). It also provides a booking system to control the reservation and availability of access to remotely controlled lab equipment.

In comparison to MIT iLabs, which is based on SOAP - requiring the interaction of service brokers, lab server, student machine, and LabShare - LiLa does not specify how

a remote control of a laboratory should be established. Instead, its goal is to define a ”meta-infrastructure” that describes packaging and dissemination of access mechanisms to online equipment in such a way that any online lab using Hypertext Markup Language (HTML) to deliver its content to the user can be integrated into LiLa. LiLa offers experiments in the form of SCORM packages, which are basically HTML pages providing access to the experiment and its metadata, along with all necessary resources to render the page. Almost every LMS is able to accept and integrate those packages.

At the core of the LiLa architecture is a web server, an experiment database, and a booking and reservation database for time slots management. How the back-end communicates to the plug-in or front-end in the SCORM package is entirely up to the content provider. For easy discovery of experiments, packages on the portal follow the meta-data specification of the Global Online Laboratory Consortium (GOLC). The LMS communicates to the LiLa packages through the standardized SCORM interface based on JavaScript and collects information on its user. Upon download from the portal by the teacher, the package is unpacked and re-wrapped by the access layer that contains the code to get user information, i.e. the user identity, from the LMS. This access layer checks with the booking system database if the corresponding user holds a valid reservation. For successful employment of virtual and remote experiments in lectures, LiLa content can also contain text documents, video and audio files, that provide didactic knowledge and background information. [RTB11]

The Lab2Go project [ZAMN10] identified the lack of information which describes resources as a general problem for many content types of special interested communities. As a solution to that problem, an online laboratory portal as a semantically linked repository for the global e-learning community was created. The focus in the process was on creating an ontology to describe the properties of online laboratories and experiments. Three experiment types are distinguished: Observation Experiments, Fixed/Controlled Experiments, and Adaptive Experiments. While both Lab2Go and LiLa aim at collecting online resources, Lab2Go provides references to online resources; whereas LiLa is an online repository, that is, it also offers such resources for download from its web portal, and offers the possibility to reserve and book them through a unified web interface. [RTB11]

The European Go-Lab federation opens up online laboratories (virtual and remote) and data sets from physical laboratory experiments for large-scale use in education. Renowned research organizations (e.g., CERN, ESA) participate in the consortium. [dJSG14] Inquiry learning spaces (ILSs), learning environments that can contain labs, learning resources and apps to enable inquiry learning, constitute the core of the architecture. To make labs interoperable with learning environments, in Go-Lab, online labs will be provided as smart devices. The smart device paradigm abstracts the details of each lab on the server-side by providing a specified set of web services, that labs have to implement. This interoperability layer also allows the ILS platform to reuse smart device compatible apps to operate numerous labs. To enable interoperability with labs where it is impossible to change the lab implementation, a smart gateway that transforms existing labs to conform

to the specification of the smart device is provided. The lab repository is implemented on top of the existing ROLE Widget Store that is built with Drupal, a widely used, open source content management system. The ILS management platform is implemented on top of GRAASP [BLL+12], which is a social media platform that supports personal and collaborative activities using resources and OpenSocial apps. [GCV+13]

3.4 Findings and Conclusion

The number of simulated laboratories that can be found in the literature and on the internet for trial is plentiful. Remote laboratories are less common but described for various engineering tasks. Amongst the systems, there are even less interactive real-time experiments conducted. An implementation of a complex control task that requires precise and true real-time control, and therefore consideration of delay at the transmission of control input, could not be found.

It was shown in multiple surveys, that simulations of processes that involve mechanical tasks are well-accepted, at least for training and education. The authors of [APA16] expect simulated labs to evolve with advances in technology and the expansion of virtual reality solutions. Both simulated and remote laboratories were highly-rated by students and can, due to their availability, lead to increased learning success compared to hands-on laboratories.

Even though, technologically experiments could perhaps be performed without the presence of experts and tutors, supervision still plays a vital role in the learning process. A frequent cause for inadequate use of remote laboratories is the lack of tutor assistance. It hinders motivation and makes it harder for a user to succeed in a task.

Some ideas, technologies and architectural designs of the presented solutions could be adapted for distance operation of a CMM, but none of them presented complex mechanical operations in 3D space, neither simulated nor remotely controlled. Therefore, for some parts, other technology has to be found.

3.5 Hardware and Organizational Changes in Automated Laboratories

Remote controlling of a physical laboratory ideally comes along with full automation. When it is no longer necessary that a lab is entered by a human aide, some sources of error can be eliminated. Automation, however, might require some changes in equipment and arrangement within the laboratory. The major differences are visualized in figures 3.7 and 3.8.

In the hands-on laboratory (figure 3.7), the target is to maximize the sample area in order to fit in more machines. The ratio between room length and width is close to the ideal ratio for area maximization. The archive is not necessarily located next to the probing room. Inside the probing room, the operator desks require a significant amount

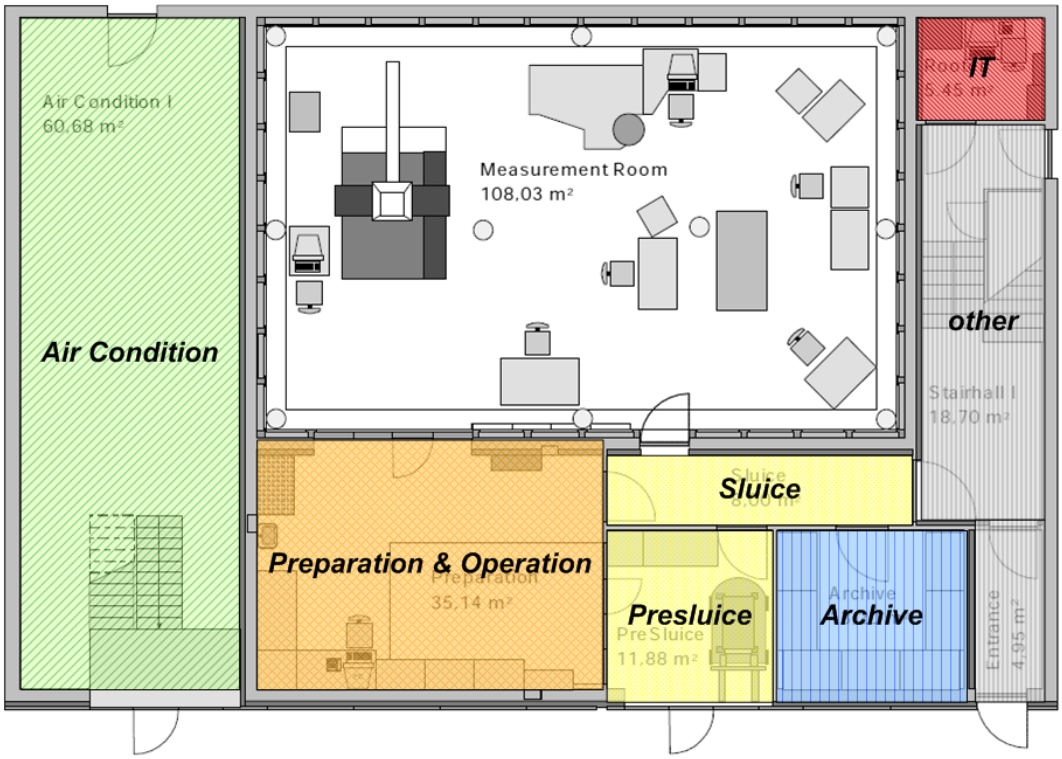


Figure 3.7: Floor plan of metrology laboratory Erlangen-Nuremberg. (amended from [WS00])

of space. They have to be located right next to the measuring devices, as those are operated partly manually via a console on the device and partly via directly connected computers. The operator might have to move back and forth between measuring device and computer workplace.

The probing room of the remote laboratory (depicted in figure 3.8) is shaped differently to the hands-on lab, rather like a corridor. This is a consequence of automation. Measurement objects are not mounted manually by the operator anymore, but via robotic arms that hand the objects over from the archive to the machines. The archive is therefore significantly bigger, and much longer than it is broad. It has to be located right next to the probing room. Windows are needed to allow for the transfer of the measurement objects from archive to probing room. The archive must have the same ambient condition as as the probing room. The room for preparation and operation can be used for supervision of the probing room, and for schooling. It is also sometimes beneficial to conduct inspections before starting a high-precision testing procedure. Manual gauging may be performed in the operation room when necessary.

In the remote laboratory, there is no space required for operator desks inside the probing room. This reduces the area occupied per metrology machine. One might expect, that

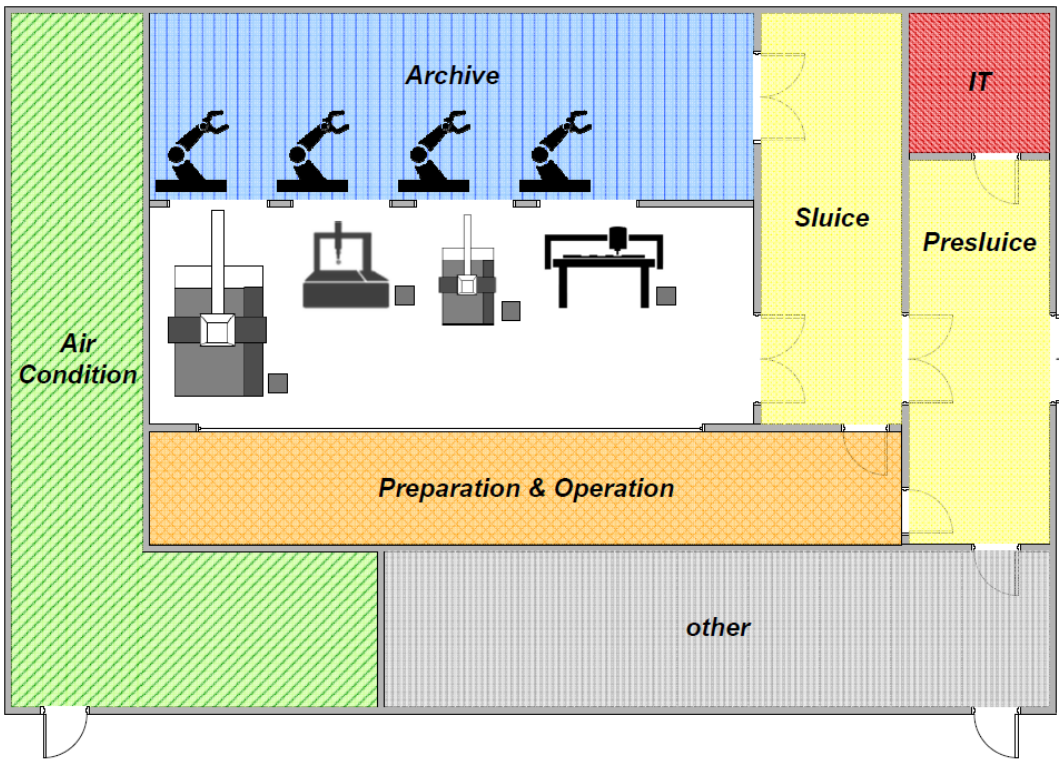


Figure 3.8: Floor plan of metrology laboratory redesigned for remote operation.

the probing room can then be shrunk. However, the ability to remotely control metrology devices from a distance, encourages cooperation between universities and/or research institutions. This reduces the financial burden that big investments pose, and allows for the acquisition of more specialized machinery. This diversification is indicated in figure 3.8 as well. As a result, laboratories that open up to a broader audience have been shown to increase in size, in terms of the number of machines and machine types. In the paper "Breaking the Lab's Walls" [BCQ+01], the authors aimed at enlarging the lab in all the dimensions of space, time, and available resources, through the use of internet technologies. A different approach is presented by Gustavsson et al. [GNZ+09]. Considering that the average university can only afford one or two workbenches, they established a collaboration between universities to form grid laboratories to allow for more students per instructor and for a greater variation of possible experiments.

In order to efficiently operate a laboratory remotely, a whole telepresence system has to be built up. This means that the experimental work is not only conducted in the framework of the measuring instrument software but also is supported by video and audio integration to the instrument room. USB webcams, mobile cameras and pan-tilt (ip) cameras carry out monitoring of the laboratory and experiments. The Vienna University of Technology (TU Wien) programmed a Zoneminder: an integrated set of applications which provide a complete surveillance solution allowing capture, analysis, recording and

monitoring of any CCTV or security cameras attached to a Linux based machine, for this task. It is designed to run on distributions, which support the video for Linux (v4l) interface and has been tested with video cameras attached to BTTV cards, various USB cameras and also supports most IP network cameras. [DBRB15]

All those pre-conditions and requirements show the heterogeneity within a typical metrology laboratory, and the need for either a standardization of interfaces or a highly flexible control solution.

CHAPTER 4

Design of a Networked Control System (NCS) for CMMs

4.1 Requirements of Control Software for Remote Operation of CMMs

One goal of this thesis is to investigate the requirements of modern operation software for coordinate measuring machines, particularly in connection to their intended remote operation. Those requirements should lead to a design suggestion that facilitates a broad range of operation modes and use cases.

4.1.1 Operation modes of coordinate measuring machines

A control system for CMMs has to allow for three different modes of operation:

- **Manual mode:** In manual mode, the operator is physically in the laboratory to shift the machine axes by hand. Operators have direct visual control that is not limited to the degrees of freedom of a camera setting. The immediate mechanical arrangement is intuitive to many operators and does not make a detour via human-computer interface. Nonetheless, the control software is used for taking measurements of the machine position, freeing an axis for manual shifting, and switching between manual and electromechanical control. In order to free the vertical machine axis, the operator must step on a mechanical foot pedal. This prevents the axis from falling and demolishing the measuring system. Data is, even in this mode, constantly collected and interpreted and calculations are performed after the probe by the software system. Manual mode is only possible if the motors of the axes do not block movement when they are deactivated.

- **Presence Mode:** In presence mode, the operator is still physically in the laboratory. Direct visual control of the CMM is possible, which allows for convenient observation of the machine transactions. However, the axes are addressed via the graphical user interface of the control software, which already allows for the performance of a complete measuring procedure.

- **Telepresence Mode:** The difference between presence mode and telepresence mode is, that in telepresence mode, the operator is not physically in the laboratory. Machines are operated entirely via the graphical user interface and visual control is only possible via cameras mounted in the laboratory. This mode has the convenience of not having to perform the provisions for entering a metrology laboratory before each measuring procedure and not influencing the measurement results by standing next to the machine. Disadvantages, however, arise with surging demands for internet security measures and a potential round-trip delay between sending a command, receiving it on the laboratory site, executing it, and providing visual feedback of the web cameras aligned at the CMM to the remote operator.

All those needs depending on the operation mode must be incorporated into the control software at design time.

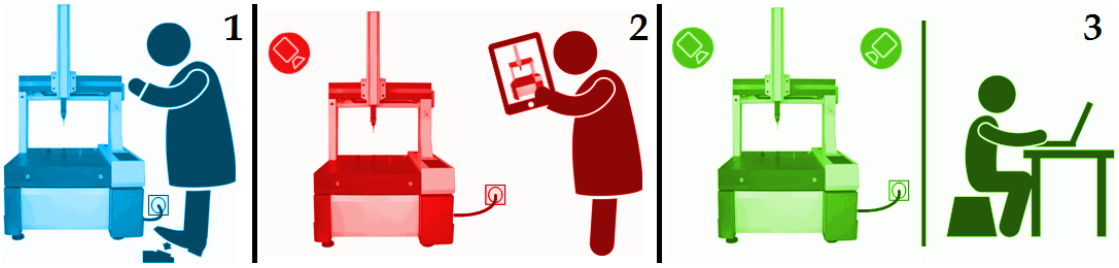


Figure 4.1: Operational modes of CMM software: (1) manual mode, (2) presence mode, (3) telepresence mode.

4.1.2 Actuators and sensors of coordinate measuring machines

Coordinate measuring machines have evolved over the past decades. From early category 1 machines to the modern category 4, operation modes and functionality were extended or deprecated (see subsection 2.2.1). In order to find an architecture that allows for operating machines of all four categories, commonalities and differences were investigated. Much progress, but the least disruptive changes, occurred on the level of mechanical components. For instance, all the machines in scope possess three axis that can be operated at at least two speeds, allowing for different grades of preciseness of control. It is therefore crucial to understand the mechanical constitution of CMMs. Investigations resulted in the following indispensable actuators:

Axis motors
Quantity: >= 1/axis
Type: Analog actuator
States: + and - movement within speed boundaries, no movement
Description: Each axis has at least one electric motor to move it. Motors can at each point in time either be inactive, cause a forward movement or a backwards movement. Two different control types are possible:

- *Discrete control* with a number of fixed speeds. This control type is mostly used when older category one or two machines are upgraded. Category one CMMs are mainly shifted manually. Manual movement is naturally not accurate enough for the usual precision metrology applications. Fine adjustments are therefore performed by twisting a set screw. Both the cart along the axis, and the set screw have to be connected to a motor but require different drives. They are either obtained by the use of multiple motors or by some form of gears. Discrete control is comparatively cheap but limits the usage to the provided speeds.
- *Continuously modulated control* with one motor that theoretically allows for an infinite number of different speeds. This control type is comparatively expensive and requires more complex hardware and software. It is used in modern CMMs, especially those with measuring probe functionality.

Connector pins
Quantity: >= 1/motor
Type: Binary actuator
States: [opened/closed]
Description: Each motor must be connected to its respective axis. The number of connector pins depends on the motor. In the case of a single motor per axis (e.g. at continuous control motors), when this motor has the ability to move both forward and backwards, the connector is usually embedded with the motor. It must either be non-blocking when the motor is turned off, or separately controllable to free its axis to allow for manual operation. In scenarios with multiple motors per axis, the connector pins can be closed to establish a connection, or opened to separate the motor from the axis. When moving an axis, only one of its connector pins can be closed.

Furthermore, the system state needs to be monitored constantly in order to prevent undefined machine states from causing damage to the hardware and to trigger automatic reactions. As an example, the control software must ensure that a motor only gets activated when all the connector pins of its axis are in the correct state. This is a quality control problem, for which the well-established PDCA cycle [Tag05] method is used. An interpretation for CMM control can be seen in figure 4.2

Commands for actions, which correspond to the *plan* activity, are received via the control software and forwarded to the machine. The CMM executes (*do*) the incoming command.

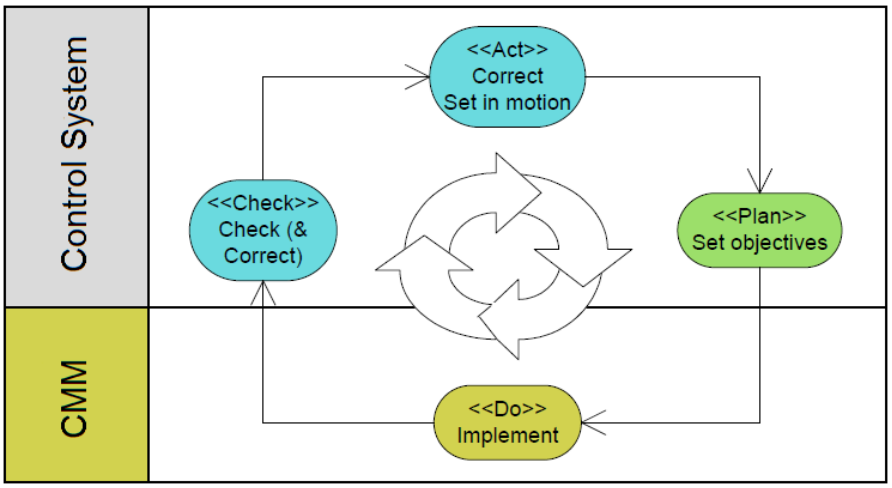


Figure 4.2: PDCA Cycle for CMM control (adapted from [Sch13]).

Sensors provide information to the control software, which *check*s whether the machine state after command execution matches the planned machine state. If a discrepancy between objective and actual state is detected, the control software decides on an adequate *act*ion. Such an action could be the re-execution of a command or setting the machine into a default state, both of which naturally have to be checked again after execution, and so on. Important hereby are also suitable measures to detect infinite loops and to inform the operator about ultimately failing command execution.

A variety of sensors are required to constantly observe the system state. They retrieve measurement data and allow for the aforementioned control cycles and feedback loops. Those sensors are namely:

Pneumatic switch sensor
Quantity: 0-1
Type: Binary sensor
States: [sufficient/insufficient pressure]
Description: Many CMMs operate pistons pneumatically to move their actuators or to counterbalance the weight of the probe shaft. [Hem00] In this case, a sensor has to permanently monitor whether sufficient air pressure is available in the system. Otherwise, no movement may be performed.

Foot pedal sensor
Quantity: 0-1
Type: Binary sensor
States: [pedal used/unused]
Description: This sensor is required for the manual operation mode. In manual mode, there is a mechanical foot pedal that cannot be controlled remotely. Only while the operator steps on it, can both connection pins of the vertical machine

axis be opened. Thus, the operator carries the weight of the vertical axis during the whole process. Another scenario would be that the connector pins of all axes can only be manually opened while the foot pedal is stepped on. This would be conceivable at CMMs that block shifting of the axes otherwise. The incorporated sensor signals to the control system whether or not the foot pedal is currently in use.

Connection sensors

Quantity: 1/pin
Type: Binary sensor
States: [connected/unconnected]
Description: There is one connection sensor per connector pin. It checks whether a connection between the motor and axis is actually closed. The state information collected by the connection sensors decides if a command from the operator may be performed, changes to the pin constellation have to be made, or if a command is rejected.

Displacement sensors

Quantity: 1/axis
Type: Analog/Digital sensor
States: numeric position within machine boundaries
Description: The displacement sensors together with the probing system, fulfill the actual purpose of a CMM - the measurement. The displacement sensors thereby register the exact position of the axes. More specifically, the measured position indicates the position of the centerpoint of the probing system at the moment of measurement. There are many different techniques to collect this information, based, amongst others, on magnetic, optic or piezo-electric principles. Information provided by those sensors can also be used to calculate whether or not each axis is moving, and at which speed. This is the most direct way of gathering this information. It is much more reliable than simply trusting that a command for the motor was correctly executed, and more accurate than calculations using the revolution counters of continuous control motors for that purpose.

Revolution counters

Quantity: 0-1/motor
Type: Digital sensor
States: rps between 0 and max. rps of motor
Description: Revolution counters observe the number of rotations of a motor per minute. They are used at machines with continuous control motors and *proportional–integral–derivative (PID) controller*s for double-loop feedback. PID control is by far the dominating control structure in industrial practice. [Årz99] An error value is continuously calculated as the difference between the planned value (*set point*) and the measured *controlled/process variable*. The controller tries to minimize this error and quickly reach a steady state around the set point. In

order to achieve that, it performs a correction based on a proportional, integral and derivative term. Figure 4.3 gives a graphical representation of an exemplary PID control curve. For more details about PID controllers see [FPW].

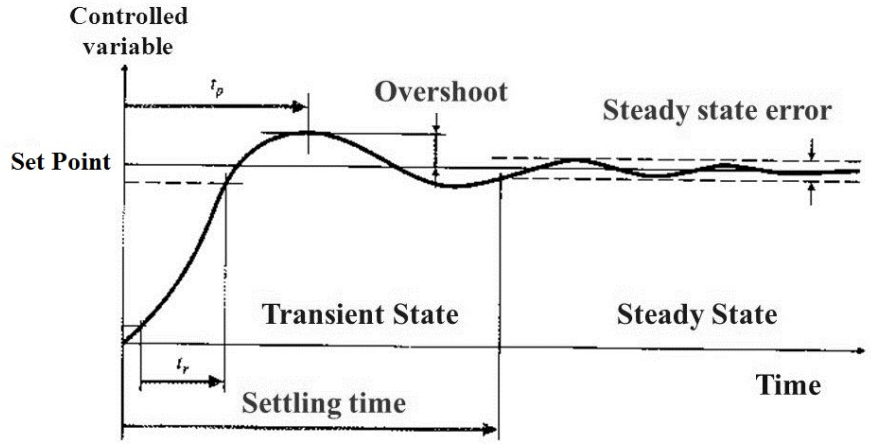


Figure 4.3: PID controller response curve (adapted from [Mar10]).

Single loop feedback: If an axis speed (process variable) differs from the intended axis speed (set point), the engine speed is adjusted. This occurs when the control receives a new command, and until a steady state around the set point is reached.

Double loop feedback: The controller checks whether the set engine speed results in the estimated axis speed. If not, parameters need to be tuned. There are various techniques for loop tuning. [ACL05] provides an overview of those techniques.

Probing system

Quantity: 1
Type: Analog/Binary sensor
States: [contact/no contact] (& contact properties for analog sensors)
Description: The probing system senses when there is contact with an object. Simple probing systems only provide the information that a contact occurred, but no information about direction and force of the contact. In those simple setups, the probing system serves as a binary sensor. More sophisticated sensors detect and encode information about the direction from which the sensing head is touched and the displacement of the tip from its basic position. In this case, the probing system serves as an analog sensor. Moreover, it is usually possible to mount multiple sensing heads onto the probe system suspension, or to combine multiple styli into a more complex sensing head configuration. Before the start of a probe, the sensing head configuration needs to be specified to the control software. A sensing head

within this configuration is defined by the three-dimensional relative position of its tip towards the center of the probing system. This means that length and mounting position and orientation of the stylus, as well as the radius of its tip have to be known. Those values can be entered completely manually, by selecting a configuration from a list of previous configurations, or automatically by importing the information from the measurement plan. Additionally, in the case of multi-head configurations, the control software must be informed of which sensing head tip will report the next contact with the object to measure. The control software uses all the information provided to obtain an accurate measurement result.

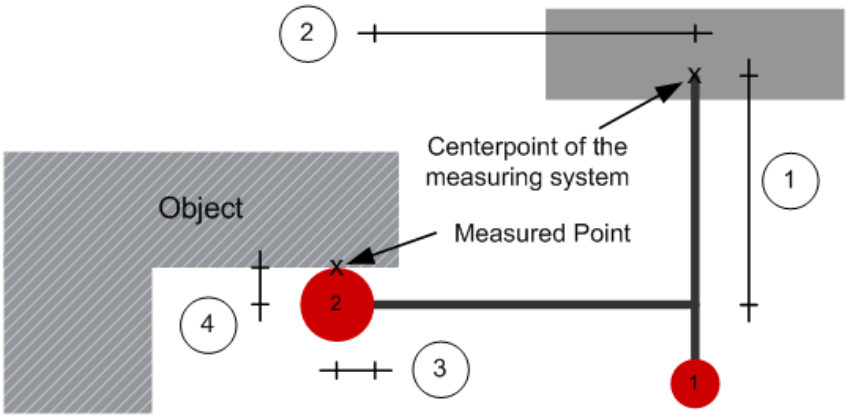


Figure 4.4: Software correction of machine position.

In the example of figure 4.4, the desired point is measured by the sensing head with the number two. This must be communicated to the control software before measuring, which then knows how to interpret the retrieved measurement position.

Step1 The mounting point of sensing head two on the stylus of sensing head one is known, its vertical distance from the centerpoint is subtracted from the measurement position.

Step2 The horizontal distance of the sensing head tip from the centerpoint, as well as the mounting angle are used to correct the measurement position in the two horizontal axes.

Step3 The radius of the contacting sensing head tip is added to the horizontal measurement correction.

Step4 Using the collected information about the direction from where the sensing head touches the measurement object, and the radius of the sensing head tip, an additional correction term must then be triangulated.

The calculations in Step3&4 are called *stylus radius correction*. Sophisticated modern measurement systems additionally recognize the displacement of the sensing

head tip, caused by the contact pressure, such that the control software may computationally eliminate the resulting measurement error. In any case, control software may ensure that the measurement object is touched only very slightly, by automatically moving a bit into the opposite direction at a contact.

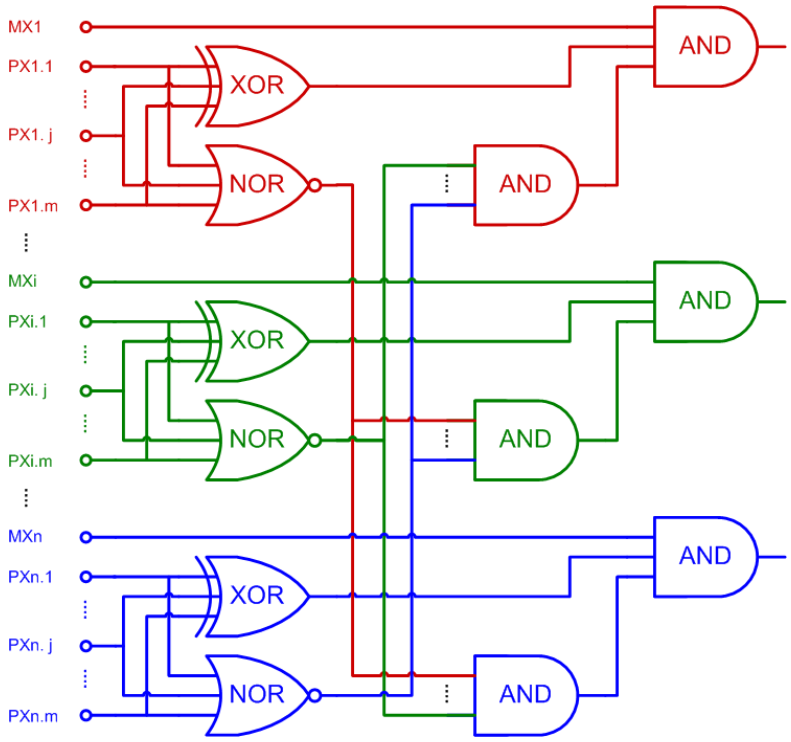


Figure 4.5: Actuator logic on the example of axis X.

Independent of its configuration, a basic logic must be implemented for the axes of a CMM. This logic is demonstrated in figure 4.5 in form of a logic diagram. In order to activate a motor, two conditions must be fulfilled. First, exactly one of its own pins must be closed. Oftentimes, there is only one pin per motor, but for realizing different gears or speeds there might be multiple pins. Second, none of the pins of the other motors may be closed. Following this logic, it is ensured that a maximum of one motor per axis can be active at each time and non-active motors do not block an axis. Motors of different axes can move independently of each other and may be active at the same time to allow for the implementation of the best-practice guideline of approaching the workpiece approximately orthogonally.

4.1.3 Use case scenarios

In short, the control system should offer the functionality to facilitate the measuring process introduced in figures 2.10 and 2.11, and create and store the corresponding

documentation. Extendible design should thereby allow for connecting additional data sources like CAD software exports. From this vision, several use case scenarios have been identified and will be discussed. Those use cases serve as the base for developing a *domain model* for the CMM control software system.

In the following, the use cases that motivate the design of the problem domain model, which will be specified in Section 4.1.4), are presented. Each use case consists of (1) a *Story* which introduces its context of use, (2) a *Discussion* which explains researched criteria and suggested specifics of realization, and finally (3) *related domain objects* which can be extracted from the respective *Discussion*. The subsequently presented domain model shall permit the discussed functionality.

UC1 - Initiating, accompanying and closing a measuring process

Story
The CMM control system facilitates appropriate (i.e. complete), reliable and efficient measurement and verification procedures. To this end, relevant information of both external data sources as well as information internally collected in the process must be storable. Especially in the planning stage of a measuring process, much specification is generated that is necessary for the further course of action. Everything is managed and archived within the control system to provide a centralized information base to check for inquiry.

Discussion: A measuring process is created in the beginning, and finalized in the end. It stores information about various stages of quality control processes. The stage in which a measuring process resides (e.g. Machine preparation, cf. figure 2.10), which artifacts are at hand that could help reducing the time and effort for planning and executing a probe (e.g. CNC probe program), and specification from the planning phase (i.e. probe plan) are documented. Further stored must be the protocols created in each phase and the date and time when each phase is scheduled.

Related domain objects:

- MeasuringProcess
- Artifact
- MeasuringProcessStage

UC2 - Creating probe plan

Story
The first stage of each measuring process is the planning stage. In this phase, the probe plan, a document describing the mandatory requirements for the measurement and the devised sensing head configuration, is generated. It is the instruction book for the remaining process. According to the probe plan, the machines that might be

considered for the probe are chosen, the setting must be established and preparations have to be made.

Discussion: The probe plan limits the choice of machines available for a probe due to precision requirements and degrees of freedom necessary for probing the specified object. Achievable precision also depends on the prevailing environmental conditions, and heavily on the choice of probing system and sensing head. The sensing head configuration is therefore an integral part of the probe plan. It is available to the CMM control software so that the software can computationally eliminate the position discrepancy between measured point and centerpoint of the measuring system (cf. Figure 4.4). This configuration consists of a collection of mounted sensing heads and their properties, as well as the mounting position and angles relative to the centerpoint. Many of the styli of the sensing heads can be docked with others, which can lead to complex tree-like structures.

Related domain objects:

- MeasuringProcessStage: MPSPlanning
- ProbePlan
- SensingHead

UC3 - Managing CMMs, their actuators and sensors

Story

The machines on which measurements are conducted with are modeled and then controlled, to effect a representation of the probing processes. All of their movements, sensing and observational activities find their counterparts within the software. Machines are configurable units. They usually do not change often, but are upgraded or changed at times. Changes must be reproduced accurately and promptly. Since each measurement requires an interaction with a CMM, they build the core objects within the system and which other activities make use of. The operating mode of CMMs has been discussed in Section 2.3, where the constitutive components are presented and explained. Flexibility and ability to store the various machine configurations and adequately react to changes, is crucial.

Machines are addressed manually, locally and remotely. They receive commands and periodically send their state to the control system. During a probe, each step taken and its timing are logged to ensure metrological reproducibility and traceability requirements.

Discussion: Measurements are performed on coordinate measuring machines (CMMs). Addressing a CMM means addressing all of its components to conduct the actual measuring. Machines and their components are managed by the control software. Components are either actuators or sensors and can be binary or digital within the control system, as the analog sensor signals are converted into digital values.

Cameras are visual sensors but might be processed differently, depending on their function. The video stream of a camera that is only used to observe the CMM, not to control it, can be forwarded to the user interface of the operator to improve processing times and not congest the transfer channel of the control traffic flow. Machines have one probing system, that is either touch trigger or a measuring probe system, the task of which is detecting occurrence and properties of sensing head contacts but not the absolute axis positions. It is a special sensor that nonetheless transfers its information together with the other sensors.

When a machine starts up, it establishes a session with the control system. During one session on a CMM, multiple probes may be conducted.

Related domain objects:

- MeasuringProcessStage: MPSMachinePreparation
- Machine (CMM)
- MachineState
- MachineComponent (Actuator, Sensor)
- Actuator: ActuatorDigital, ActuatorBinary
- Sensor: SensorDigital (ObservationCamera, ProbeSystem), SensorBinary
- CMMComponent (ProbeSystem)
- ProbeSystem: PSMeasuring, PSTouchTrigger
- MachineSession

UC4 - Provide user access management and machine time scheduling

Story

In the control system, it is relevant who performs an operation. Unauthorized users might pose a security risk, and untrained operators might damage machine or the sensing heads. In an industrial setting, job performance and output can be measured easily as employees actions are monitored. Performance monitoring is also relevant in an educational environment. Invited users (e.g. students and other researchers) are allowed to observe a machine session. On the other hand, non-authenticated or unpermitted requests have to be blocked. To enable all those task assignments, registration and distinct user roles are required.

Furthermore, even trained and registered users may not access machines all the time. Especially on scarce and expensive equipment, machine time is valuable. Machine access is therefore granted based on a scheduling system providing timeslots to authorized users.

Discussion: Permissions are granted for either a user or list of users. At each time, only one user can be in control of a machine. Additional users may observe the operation

but cannot actuate. Further user roles would allow experts (like a professor or tutor) to take over control for demonstration purposes from a regular user and hand it back afterwards. A user under administrator role definition may close machine sessions of non-administrative users. Registration for timeslots as well as user management should ideally be performed externally and information imported into the control system via a suitable interface, or machine access for regular users generally be permitted only through an access mode embedded into external software (e.g. LMS, ERP). For maintenance, calibration and testing, direct access must be possible.

Related domain objects:

- User
- MachineTimeslot
- MachinePermission
- Role

UC5 - Conducting probes on CMMs

Story

A probe is an actual interaction with a CMM with the goal to receive measurement data for a measurement object. The target of a CMM control system is a smooth, precise and convenient machine operation and to cover the measurement requirements even of highly complex objects. Therefore, the probe follows an exact probe plan and can either be executed by performing each step separately (i.e. interactively), or by running a previously recorded probe program. Gradually one geometry after the other is scanned and in this way the model completed. Furthermore, each step performed is recorded thoroughly and protocols are created that conform to metrology standards.

Discussion: A probe is conducted on a machine, by a user, at a specific time with a specific sensing head configuration following a probe plan. That means, to perform a probe, an authorized user connects to a CMM and thereby creates a user session on a machine. A probe is split into setting up the workpiece coordinate system and workpiece probing. The setup procedure resembles probing a corner point by probing the values for all three axes separately. After the coordinate system is set up, the object dimensions are determined. The overall process is divided into a series of machine operations that are equivalent to probing a property of a geometry or free form shape, like the orientation angle of a plane in space or a circle diameter, et cetera. Each operation may assume a set of states specific to the operation.

The probe can be conducted step by step, in interaction of machine and the operator, or via the probe program. In order to use a probe program, it must have been developed previously. There are two ways to define a probe program: (1)

the program is coded as addendum to the planning stage, and without machine interaction; (2) the program is recorded during an interactive probe by storing the machine position after each axis movement relevant to a future probe repetition. After its creation, a probe program can be called as often as desired. The usage of an existing probe program for a probe is called a probe program execution.

A probe session is logged from connection establishment until its termination, including the machine configuration (machine components and probing system configuration), all of the operations performed and the information required by traceability standards to guarantee that precision and measurement uncertainty lie within the prescribed boundaries.

Related domain objects:

- MeasuringProcessStage: MPSProgramming, MPSProbing
- UserMachineSession (ProbeWithoutRecording, ProbeAndRecording, ProbeProgramExecution)
- ProbeShape
- Shape
- MachineOperation
- MachineOperationType
- MachineOperationStep
- MachineOperationState
- ProbeProgram
- Protocol

UC6 - Calculating and evaluating measurement results

Story

Regardless of the probing type, whether interactive or automated, after the probing, geometrical and free form elements along with their properties are computed. The calculation engine uses the collected information in the form of points probed and shape information provided by the user or probe plan and converts it by conducting calculations, into measurement results, afterwards elements (plus geometrical distortions) and subsequently comparisons with the specifications (plus allowed deviations). In that manner, it creates a digital representation of the measurement object.

Discussion: The arithmetic component of the CMM software compares its measured results with the specification and calculates deviations and violations of tolerances. It is responsible for creating the metrological value. The complexity of this component is proportional to the functional range expected from the measurement setup.

Related domain objects:

- MeasuringProcessStage: MPSCalculation, MPSValidation
- CalculationEngine

4.1.4 Problem domain model

The listed use cases firstly motivate a domain model (cf. Figure 4.6) on a class diagram level of granularity. A class diagram describes the types of objects in the system and the various kinds of static relationships that exist amongst them. If they are more detailed, class diagrams also show the properties and operations (summed up under the term "features") of a class and the constraints that apply to the way objects are connected. [Fow00] The prevailing domain model diagram serves as a demonstration of classes and their interactions but does not contain all the properties and operations of those classes. The concepts of Unified Modeling Language (UML) version 2 packages and class diagrams are specified in [OMG10].

Within a MeasuringProcess, each stage may, but does not have to create its own Artifacts. An Artifact is an unstructured document for storing information in a human readable form. Information that should be used by the control system itself must be stored with the help of other domain objects such as the ProbePlan or, during a probe, within a MachineOperation. In the MPSPlanning, exactly one ProbePlan is created. It is used in other stages too and it can be fed also with data from external sources. The MPSPreparation exists within the system to furnish evidence that the configuration determined in the MPSPlanning has been established at the physical machine.

The MeasurementProcess knows all of its stages and each MeasuringProcessStage knows the MeasurementProcess. This allows MeasuringProcessStages to access a document owned by another stage. In the MPSProgramming, a ProbeProgram is created for later reuse. It is an optional stage that does not exist for each MeasuringProcess. The MPSCalculation and MPSValidation make use of a CalculationEngine, which performs all kinds of calculations and may compare measurement results and specifications in the ProbePlan. The CalculationEngine has a very complex internal structure, and will make use of a multitude of intermediate steps and helper objects that are not depicted in the domain model. Furthermore, external libraries will be used to perform the calculations and an external CalculationEngine might be integrated.

The SensingHead object represents the sensing head configuration that is decided on in the MPSPlanning. Multiple SensingHeads can be mounted on a base sensing head, thus a complex tree structure can be created. However, the hardware on a Machine may only be changed when the Machine is not connected to the control system. Otherwise, an operator could launch commands remotely, not knowing that the Machine is not ready. Therefore, before a Machine connects to the control system and thereby opens a new MachineSession, the demanded configuration must already be set up. As a result, there is just one SensingHead configuration per MachineSession.

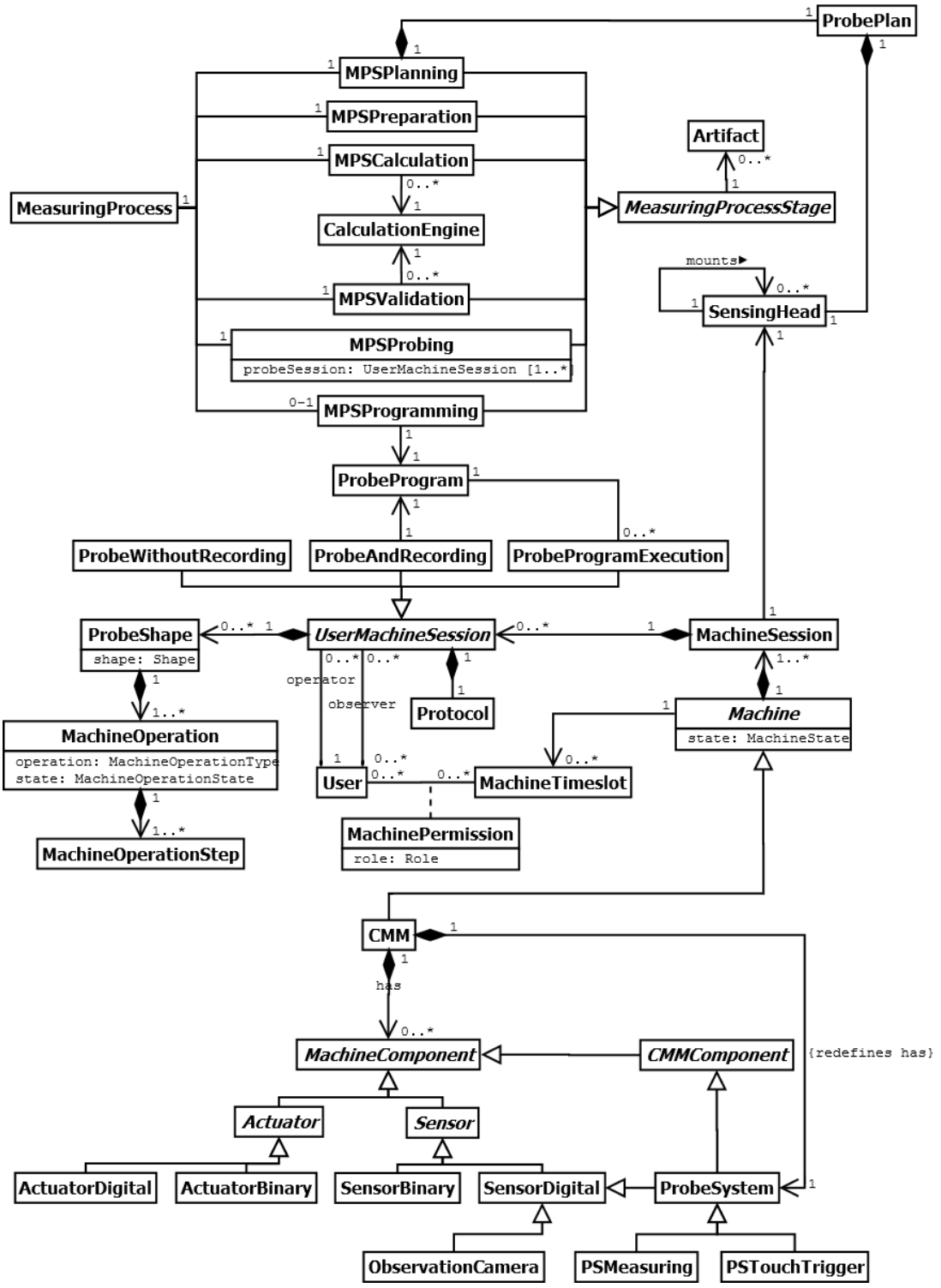


Figure 4.6: Class diagram of problem domain.

A Machine has one MachineSession per connection to the control system. When the Machine connects, a probe is started, and in some other situations, the MachineState it holds, changes.

The only type of Machine that is supported by the current control system design is the coordinate measuring machine (CMM). However, the control system functionality can easily be extended by adding classes for the new machine type and its components, as well as new MachineOperations. The components specific to CMMs are the CMMComponents. Important is thereby the ProbeSystem of which a CMM has exactly one. Of the other MachineComponents, a CMM can have an arbitrary number. Since the installed MachineComponents do not change frequently, they are stored directly in the Machine instead of the MachineSession like the SensingHead.

UserMachineSession is a non-instantiable probe class. There are three concrete implementations of UserMachineSession. A ProbeWithoutRecording is an interactive probe at which intermediate positions between multiple MachineOperationSteps are not stored. At ProbeAndRecording, a ProbeProgram is created for later, automated repetition of the measuring procedure. This probe type stores the intermediate positions and each measurement step. ProbePrograms created in this way or earlier in the MPSProgramming of the MeasuringProcess can later be used in automated probes, called ProbeProgramExecution. Via the respective MPSProbing, the UserMachineSession can be accessed. One Protocol is generated per probe. Out of the structured Protocol, a report Artifact is generated for the operator. Per MachineSession, multiple UserMachineSessions can be opened and probes performed.

Each probe is conducted as a series of ProbeShape calls. The ProbeShape stores the measured data points of a Shape. For each Shape determined in ProbeShape, a variety of MachineOperationTypes are at the disposal of the operator. Depending on the Shape probed, different MachineOperationTypes are possible. Each MachineOperation is of one MachineOperationType, which dictates the MachineOperationSteps required for its completion. Progress of the MachineOperation is tracked in its MachineOperationState field. Properties probed during the UserMachineSession are accessed by the CalculationEngine via the reference in MPSProbing.

Users may be imported from external sources such as a directory service. Considering the opportunity of outsourcing User access management and scheduling activities, external systems such as LMSs might create, update and delete MachineTimeslots and also set the respective MachinePermission. A MachinePermission is a property of a User for a specific MachineTimeslot. MachineTimeslots are stored to the Machine and reserved for/by a User, which then gets a specific Role assigned for the timeslot. Multiple Users and Roles may be assigned to a MachineTimeslot. The user interface for reserving MachineTimeslots might restrict the number of Users allowed, either per Role or in total. When Users want to create a UserMachineSession, it is checked whether there is a MachinePermission with sufficient privileges set for this User at the current timeslot. This is, of course, only possible if a MachineTimeslot for that time exists. The User that creates the UserMachineSession then becomes the operator. After the UserMachineSession is created,

other Users may join the session as observers, provided that they have the respective MachinePermission. If an administrative User takes over control from an operator, this administrator is the new operator.

4.1.5 Life cycle of machine sessions

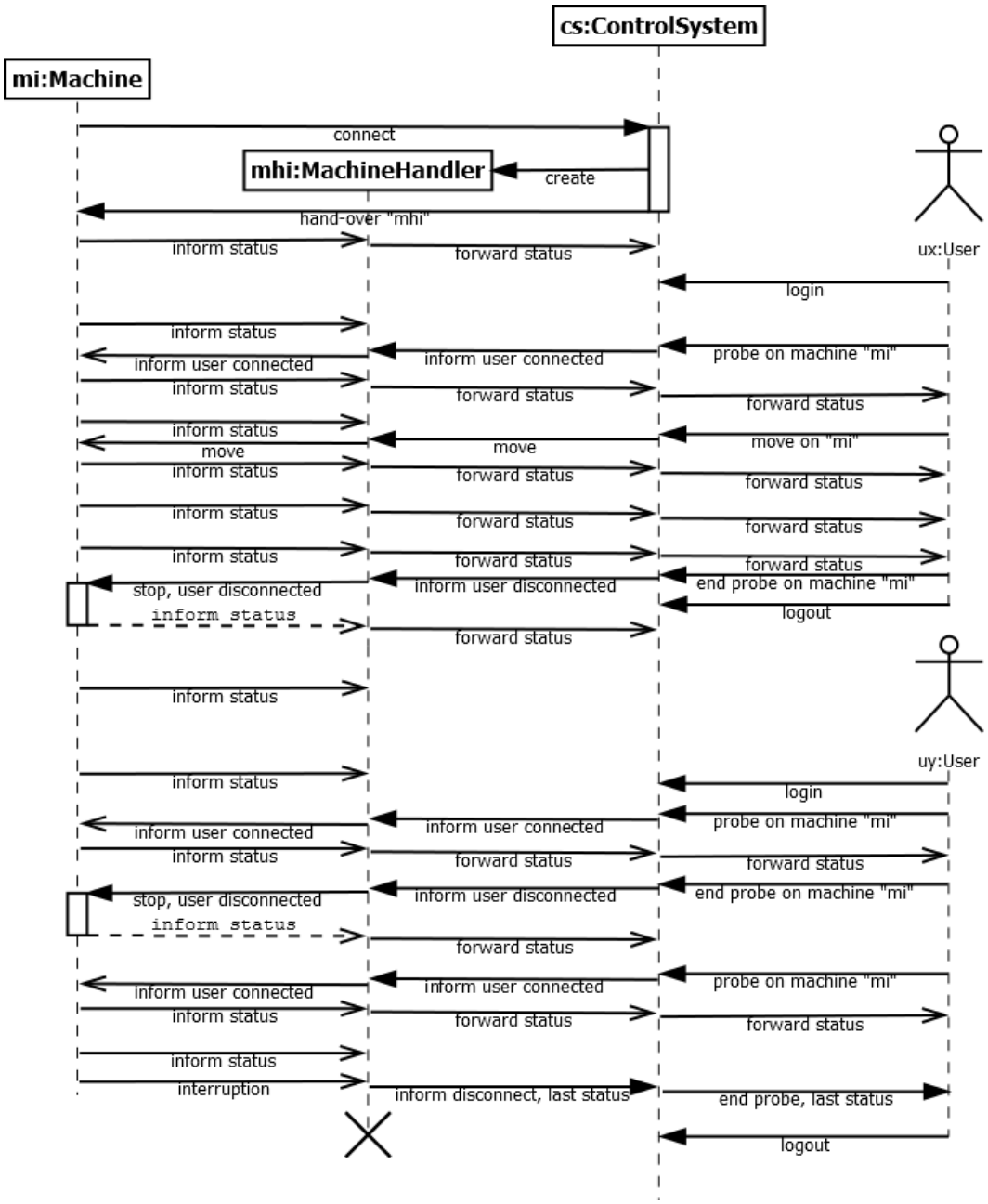


Figure 4.7: Sequence diagram of machine session life cycle.

The sequence diagram 4.7 shows a simplified, typical course of interaction between the control software, a machine that may conduct probes and two authorized users. It demonstrates the sequence of actions determined in the requirements engineering phase. Names of operations, as well as the number and names of operation parameters may change in the eventual design.

Machines cannot be set active when the control system is not started and accessible. A machine that does not get a confirmation that a control system connection was established successfully, will shut down again immediately. If the identification number of a machine seeking to connect is invalid, or a machine with an identical identification number is already connected, such a confirmation remains outstanding. Similarly, users must identify themselves at the control system as being registered and permitted to access the machine at the given time slot. Multiple machines and users may be connected to the control system simultaneously.

On startup the machine connects to the control system, which in turn creates a separate control handler for the machine and informs the machine about it. Further communication with the machine happens exclusively via its handler.

The machine continuously sends the state of its sensors to the machine handler created by the control system. This state information is forwarded to the main control system, but only in case there is a change in machine state compared to the previous state information message. Otherwise the forwarding is not necessary. If a user is connected to the machine via the control system, the state information is further relayed to this user. For performance reasons, and since it is impossible by design that an actuator moves when no user session is active, the notification interval of the status messages may be longer when no user is connected.

A user connects to a machine by starting a probe session. This session is managed by the control system. The machine, informed via its handler, only knows that an operator is connected now, but does not need information about the user. Access is governed by the control system. Establishing a connection between the user and machine constitutes a change in machine state, which is why the next state information message is forwarded to the control system and eventually the user. Once the connection stands, the user may send machine commands (Figure 4.7 insinuates a general ”move” command as a placeholder for a concrete command to move an actuator) to change the current machine state. As long as the state is changing, be it as a consequence of a command or due to the application of physical forces (e.g. a manual axis shift) to the machine, status information messages continue being forwarded. An arbitrary number of commands can be sent during a probe session.

When the user ends the probe session, the machine is informed via control system and machine handler that the operator disconnected. Notification intervals might now be enlarged again. Both the machine handler and the machine session stay upright and waiting for a new probe. Probes on one machine during one machine session might be performed by multiple users. One user may perform multiple probes on one or multiple

machines before logging out from the system again, as can be seen at user "uy" in the sequence diagram.

A disconnection of a machine may be induced by either (1) an administrative user via control system, (2) a general shutdown of the control system, (3) by an interruption on the side of the machine or of (4) the connection between machine and its handler at the control system. In the cases 1 and 2, the control system instructs the machine, via its machine handler, to shut down. The machine responds by executing the shutdown routine that consists of stopping all the moving actuators, attempting to restore a machine default state, confirming the shutdown (including the last machine state) to the control system, and eventually shutting down. In case 3, if the machine is shut down deliberately or detects a critical error that requires a shutdown, it may trigger the same shutdown routine as in cases 1 and 2. Otherwise, as well as in case 4, both sides detect the interruption. The machine stops all its movements and shuts down or attempts to restart and reconnect. The machine handler informs the main control system about the connection loss and the last known machine state, before it destroys itself. If there is currently a probe session running at the machine, the control system ends it and informs its operator. In all cases, the machine handler is destroyed, whereas the same physical machine, represented by the same domain object at the control system, can be reused later.

4.1.6 Non-functional requirements

Besides the aforementioned functional requirements, a list of Non-functional Requirements (NFRs) has been created in order to choose a suitable and effective system architecture. There is no exhaustive categorization for the identification of NFRs that is widely agreed upon, as the relevant criteria is always specific to the domain and application. Chung et al. [Chu00] discuss the dimensions: *development & operational costs*, *performance*, *reliability*, *maintainability*, *portability*. In [SL07], a framework for preventing conflicting NFRs is proposed. Especially the trade-off between the ISO/IEC 25010 [ISO11] quality attributes *reliability* and *efficiency* must be considered. Suggestions from these sources were followed in the process.

The following criteria were deemed most important for their respective dimensions. These are development targets during the design process and are kept in mind at each design decision.

- **Development & operational costs:** Free and open software should be favored when no significant convenience or performance loss is expected.
- **Performance:** The performance should be optimized by keeping the number of messages and amount of data transferred - in particular over the internet - low.
- **Reliability:** Fault and failure tolerance should be achieved by automatically detecting critical machine states and reacting in a way that protects equipment from damage.

- **Maintainability:** The system should be constructed in a way that allows for easy extension of functionality, especially the quick introduction of new machine types into the system. A further goal of the software design is adaptability to changes.
- **Portability:** The control system should provide a user interface that allows for CMM operation not only on stationary computers, but also mobile devices like tablet computers and smartphones. Operating remote laboratories should not require the user to install specific software. This improves convenience for the user and simplifies the distribution of updates.

Alongside these dimensions, two more goals were verbalized in the interviews with domain experts. The user interaction should be intuitive and close to the original control process with the proprietary local control software, so that only little initial training and explanation is required and improper usage is minimized. The system should provide a higher degree of abstraction and automation and react dynamically to the situation by highlighting operations that are currently useful. Those two ideas are partly contradictory and a compromise between keeping the original control experience and automation must be found.

4.2 Proposed System Architecture

4.2.1 Constituent parts of the control system

As a consequence of the functional and non-functional requirements collected, a number of constituting components of the control system have been identified. Each of those components realizes a share-functionality. Multiple functional components can later be arranged, combined into one physical component, located and connected by interfaces and communication protocols, to form a system architecture for performing probes. The constituents are:

Graphical User Interface (GUI)
: The means for the user to set up a connection and give commands to a machine. The graphical interface provides the necessary control elements to initiate all currently available and allowed machine operations.

User Request Handling (URH)/User Access Management (UAM)
: A request-based interface for receiving user commands, managing user sessions, performing the user authentication and checking if a request is valid. If the session is established and everything about the request is OK, the input of the user is forwarded to those components that handle and process them.

Command Preprocessing (CPP)
: A component for querying steady machine information and changing the machine state on control system-side when manipulating the machine itself is not required.

This includes all set commands other than those to position machine actuators, e.g. starting the procedure for measuring a new shape or its properties. Those commands that do require access to the actuators or sensors of a machine, are forwarded by the preprocessing component to the correct machine handler.

Persistence (PRS)
A data base where information about performed activities (e.g. protocols of sessions and operations), as well as named objects that are used by the control system (e.g. equipment, users, process artifacts) is stored permanently. Run-time properties, like machine and operation states, are persistently stored in case of interruptions or to handle concurrency, even though those fields will be overwritten frequently during a probe.

Machine Access Management (MAM)
A gateway that manages machine connections. It holds a list of all active machines and provides an interface that waits for user commands to operate the machines. In contrast to its request-based user communication, the MAM component is in constant interchange with the machines and receives periodic updates on their states.

Machine I/O Control (MI/O)
The Data Acquisition System (DAS) is in permanent contact with the real CMM or virtual instrument (simulation). Its task is to collect the measurements of the machine sensors and forward them to more distant processing components of the control system. Role and functionality are identical for both VIs and physical machines, but the technologies used and the implementation may vary depending on the controller type and available programming languages. If real CMMs are controlled, commands are converted into electronic signals. Physical quantities measured by sensors of the machine are converted into digital data.

Live View (LVW)
The live view component allows for remote visual process control. It is a necessary component as a consequence of introducing remote operation. Different solutions for remote and virtual laboratories are required.

Virtual Instrumentation (= Simulated Machine) (VIN)
A software simulation, which serves instead of the physical machine for studying the essential concepts of CMMs during design and implementation stage, for testing the control software, or when the physical machine simply is not available. It is further used to educate students and future metrology professionals and provide practice opportunities in addition to the experience they gain on the physical machine. The simulation exhibits all the essential addressable parts that the physical machine also possesses and might furthermore display extra information, not collected by the physical machine. In the literature, e.g. [FAK+05], [MN06], [dZ13], the term for such simulated laboratory equipment is *Virtual Instrument*.

4.2.2 Architectural overview

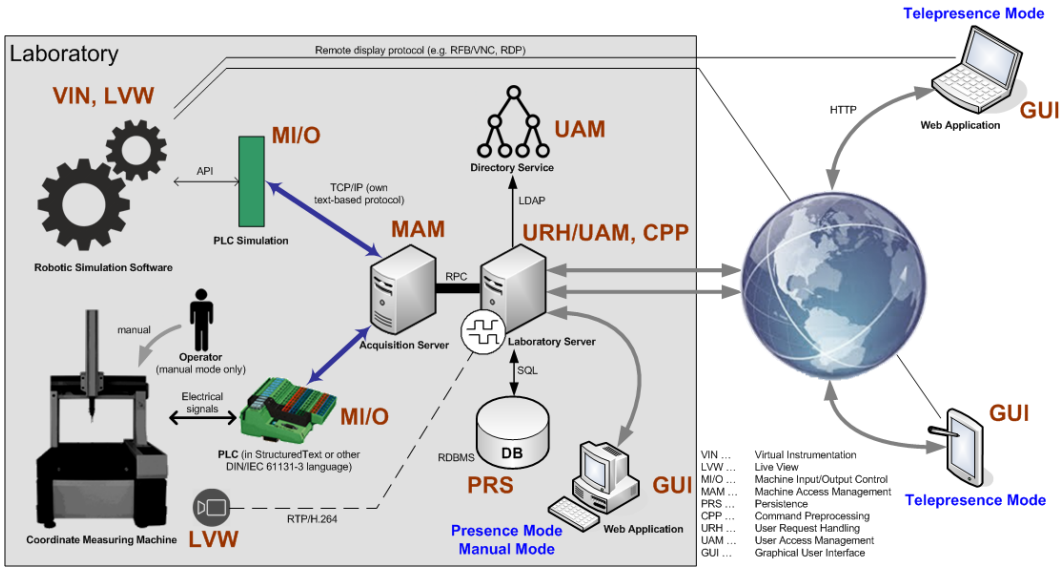


Figure 4.8: Architectural overview of the control system.

Decisive for the performance of a distance control system is the physical arrangement of its components and the logical distribution of its tasks. Figure 4.8 depicts how the share-functionalities defined in Subsection 4.2.1 are arranged to best fit the criteria specified in Subsection 4.1.6.

The first design decision that had to be made was, whether an already established architecture was suitable for the control system or a new architecture had to be invented. One way to preserve the proprietary control software that CMMs are equipped with at delivery was pointed out by Lamberti and Sanna ([LS11]). They introduce a way of virtualizing a local desktop application without re-coding by migrating it into a Web OS framework and recording its visual appearance and dynamic behavior. However, such approaches lack in reliability and responsiveness. Moreover, machines of category 1 (cf. 2.2.1) cannot be upgraded following this approach. Access control might also be an issue at cloud-based approaches. Instead of engaging with the user interface, a re-engineering and imitating of the communication between the machine controller and server software was thinkable too. A major disadvantage of this approach would have been that the calculation engine and complex logic already implemented into the proprietary control software, could not have been preserved, which significantly cuts its benefit. Therefore, it was decided that the architecture should replace vendor-specific software entirely and actuate the machines on an electronic level.

Physical machines are addressed using a PLC. These controllers have a number of output pins, each of which connects to an actuator and a number of input pins that connect to a sensor. The controller varies from machine to machine and can be of different types

and vendors. In some machines, the control unit is physically integrated and not easily exchangeable. If that is not the case, even simple electrical controllers with the necessary number of analog and digital input/output pins and a port for TCP/IP communication come into consideration. However, in order to achieve the best performance, industrial PLCs from specialized vendors should be used. They possess better hardware, which is optimized for short cycle-times and therefore can react faster to the changing input. That way, machines can be positioned more precisely, and the risk of damaging equipment is reduced. There are a number of different languages specified in industry standard IEC 61131-3 [DIN13] that may be used for programming industrial PLCs. When a new machine is to be operated via the control system, the controller must be programmed according to the specification. When something about the sensors or actuators of a machine changes, re-programming is required as well. The choice of language used for implementing the controller belongs to the responsible administrator for the machine, as long as it allows for using TCP sockets. The exact functionality and control cycle of the PLC is specified in Subsection 4.2.3. Data collected by the PLC is sent to an Acquisition Server (AS), and the AS hands over commands to the PLC for the actuators of the machine.

Instead of a physical machine, a simulated CMM can be used for experiments too. In that case, the machine must be modeled precisely in an appropriate simulation software. The intended application of the experiment determines the choice of simulation software. For some educational tasks software that offers more opportunities for interaction with the environment of the simulation might lead to a better learning success, whereas higher precision and more realistic mechanical behavior can be achieved by using specialized *robotic simulation software* (RSS). The use of RSS is essential for conducting research on the behaviour of a remotely controlled CMM.

Vendors provide their own Application Programming Interface (API) for interacting with the robotic simulation, as a function library written in at least one common programming language. It is often either a procedural language (e.g. C), or an object-oriented language (e.g. C++ or C#). This allows for easy access to actuators and sensors of the simulated machine and manipulating environmental conditions of the experiment. In order to operate a simulated CMM, the course of action of a PLC that controls a real CMM is exactly imitated. Thus, the *PLC Simulation* also follows the control cycle from Subsection 4.2.3. Just like the real PLC, it exchanges sensor data and machine commands with the AS.

The AS uses the same protocol for communication with the physical and the simulated PLC. It has been developed for that particular purpose and is presented in Subsection 4.2.4.

The communication of the control system with the machine PLC is periodic whereas communication of the control system with the operator is request-based. This results in highly inefficient resource scheduling that slows down any system that combines these two interaction schemata. Therefore, a clear logical separation into two components was chosen. *Acquisition Server* and *Laboratory Server* (LS) build the core of the control

system. They are both implemented using object-oriented programming (OOP) concepts like encapsulation and subtyping to facilitate adaptability and extensibility. In order to perform any kind of operation via the control system, AS and LS need to be on.

The *Acquisition Server* consists of a central service that waits for connection requests and one autonomous handler for each active machine connection. The IP address and the port of the central service are stored in the configuration of the PLCs of the CMMs. When a machine is started, it tries to establish a TCP socket connection with the AS. After successfully connecting, the AS receives the state information as a text from the machine and forwards it to the *Laboratory Server*. The machine handlers contain little program logic other than a check whether the input data is in the correct format. Yet they do compare state information messages to previous messages and filter them out if no digital value changed and changes of analog values lie within a characteristic threshold, subject to the machine. In the other direction, the machine handlers at the AS receive commands from the LS, convert them into the format to satisfy the text-based communication protocol with the PLCs, and transmit them.

Laboratory Server and *Acquisition Server* communicate via Remote Procedure Call (RPC). The LS holds a list of currently active machines and their respective handlers at the AS. Although the two servers may be programmed in different languages, the communication is faster when both are using the same language as language-specific protocols can be used. The AS does not have direct access to the database and has no knowledge about the domain objects. When a CMM attempts to connect, the AS asks the LS if the connection request is to be permitted. Commands from the LS for a specific machine are sent directly to the correct handler and consist of the identification number of the targeted pin at the PLC and the analog or digital value to set.

The LS possesses many interfaces to combine the subsystems. Also, much of the logics operations and computational tasks are performed by the LS. Due to the number of tasks the LS has, it is described separately in Subsection 4.2.5.

Like in the system presented in [SCMS11], the AS and LS may run on the same physical computer. However, due to the clean separation of concerns, the load can also be split onto multiple physical entities. It is also possible to control some laboratory-based CMMs via one LS, and others via another LS.

The use of multiple physical servers requires coordination between the different LS and concurrency control at accessing laboratory resources. Therefore, all the servers in one laboratory access a common database. Relational database systems (RDBS) are perfectly suitable for this task. The structure of the data to handle also makes them superior to NoSQL databases for this application [Nay13]. The database is capable of storing all the domain objects presented in Subsection 4.1.4. Nonetheless, if an LMS or an appropriate Content Management System (CMS) is available at the institution and already used for the *user access management* and tracking of learning activities, it might also be used for tasks related to the measuring process, e.g. managing its artifacts.

Regular users (like a student or regular researcher) need a reservation for a timeslot,

which grants a temporary right to use a specific machine. Furthermore, they can proceed with a measuring process and add artifacts. Usage of a machine can be operation or observation. In an operator role, the user conducts probes either step by step or using a probe program and may record the operations as a new probe program. As an observer, a user can log into a running probe session to see the actions of the operator at the graphical user interface and observe the machine state. Expert users (like a professor or tutor) have additional privileges and options to manage and grant other users timeslots. They can change machine configurations and take over control of a session for demonstration purposes or as an opportunity to intervene in case of improper machine handling or conduct. The ability of expert users to take control of a session might be limited to particular machines or cases when the session was created in the course of a timeslot that the respective user issued. Administrator users can take control of each running probe session and have full privileges to the control server. This allows for functions like creation, update and deletion of machines and management of probe logs and probe programs regardless of their authors.

Criteria for the GUI were portability, maintainability, functionality and design. It must be usable on different types of displaying devices like desktop, laptop and tablet computers. Access to local resources on client-side is not necessary and there is no local database. As the ideal candidate, and as already successfully proved and tested at the solutions presented in [SCMS11], [CMZ06] and [ASS+11], a *web application* was chosen as a means for the operator to access the machines. The main advantage is the simple roll-out of changes to the software, compared to software clients that are installed at the client. For conducting experiments at a remote laboratory, a web browser-based graphical user interface is the only tool required. Since the CMM is out of sight at such experiments, one or more web cameras record the proceedings in the laboratory and live-stream a video directly to the *Laboratory Server*. The web cam live stream is embedded into the GUI. For simulated laboratories, the realization of the *Live View* functionality, depending on the choice of simulation software, requires some remote connection for displaying the simulation window. The Live View functionality is explained in more detail in Subsection 4.2.6.

Material may be provided via LMS, timeslots reserved and remote laboratories completely integrated within a website. At simulated laboratories a full integration is not possible when the simulation environment does not provide means for a Live View. In that case, web links to the machine for remote connection can be provided by the LMS and user access on the remote machine handled via the directory service of the organization, while the simulation window is displayed externally. In industrial applications or when there is no LMS yet, another CMS or the database of the CMM control system are used for the LMS tasks.

In the following, some of the components are explained in more detail.

4.2.3 Functional specification of the controller

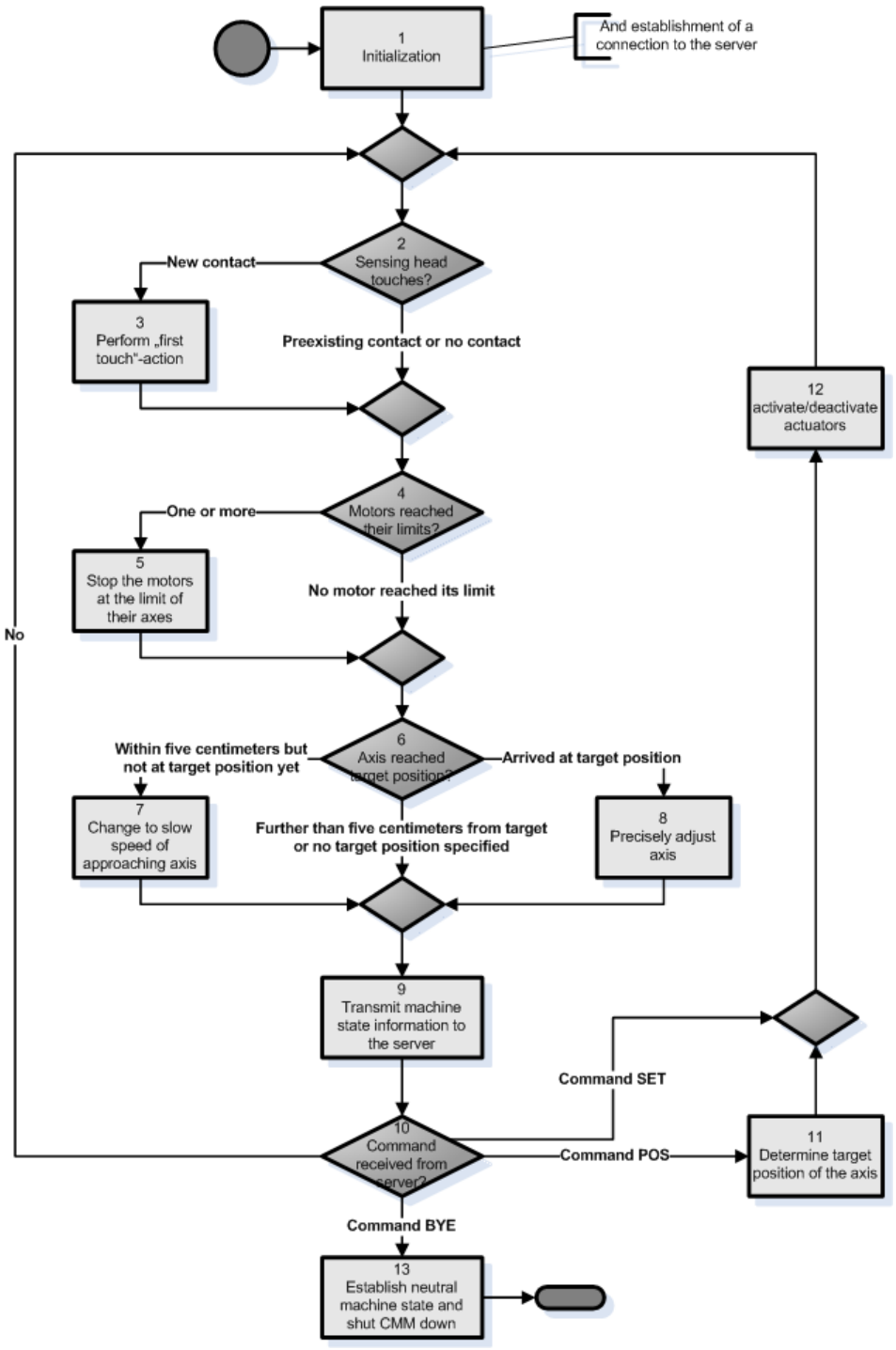


Figure 4.9: Control cycle of the PLC.

The PLC can be of various types (industrial PLCs or other microcontrollers), individual to the machine and is implemented by the laboratory who owns the project itself. Even though the controller implementation depends on the hardware used, the underlying concepts are identical. Each controller realizes the guidelines specified in this subsection.

Basically, the complex logic operations are performed in the server. The PLC essentially converts signals and relays them between machine and AS. It unifies the DAS for reading out the sensor data and the unit for feeding the machine with the control input. However, in order to reduce reaction times some logic should also be integrated into the PLC. Another advantage of immediate control by the component closest to the machine is that it limits the quantity of things that can go wrong.

Discrete control can be realized within a single control cycle and can therefore be applied on controllers that do not support multiple parallel threads. If PID control is desired, hardware choices are limited to controllers that allow for parallel calculation of the P, I and D terms.

Controllers follow a standardized control cycle that is depicted in Figure 4.9.

Step 1:
The first thing the controller does when the machine is switched on is to establish a connection with the AS and authenticate the machine via its identification number. If this process fails at some point, the machine is shut down immediately. After successfully connecting, all the variables are initialized and actuators set into their default state. In the default state one motor connector pin per axis is closed and no motors are moving. If there are more motors than one per axis, the connector pin of the motor that induces the slowest axis movement is closed. After initialization the control loop starts.

Steps 2 and 3:
At the beginning of each control cycle, the PLC checks if the sensing head registered a contact. In case of a contact, it checks whether there was a contact in the previous control cycle or not. If it is a new contact, a routine called *first touch action* is performed, which stops all of the axes in order to prevent the sensing head from damage. It is one of the logic operations explained in more detail later in this subsection. If the contact preexisted or there is no contact detected, no action is performed.

Steps 4 and 5:
Some motors and the axes have extreme positions specified. When those limits are reached, the motor concerned must be stopped. The PLC checks those conditions and acts accordingly as part of a routine called *check boundary reached*.

Steps 6 to 8:
An axis can be moved in two ways. Either one motor gets activated and stays active until a stop command is received from the server or an axis or motor limit has

been reached, or a *position* command determines a target position for an axis. If a target position is specified, the PLC must observe it. Four cases are distinguished that require different actions or no action. The necessary checks and actions are performed as a part of the *check boundary reached* routine, which will be explained in more detail later as one of the logic operations of the PLC.

Step 9:
The values of all the sensors mounted to measure the machine state are transmitted to the AS. If an actuator does not have a sensor to verify its state, the value that the PLC has stored for the actuator must be reported. However, this violates the feedback control rules and such values cannot be considered as reliable as values measured by sensors.

Step 10:
The PLC checks whether it received a command from the AS. If a known command was received, it must initiate an adequate action. If no command was received or the PLC received an undefined command, the control cycle starts over again at Step 2.

Step 11:
If the PLC received a *position* command, it checks if the specified position lies within the boundaries for the respective axis and sets the variable storing the target position for the axis. Afterwards, Step 12 is performed.

Step 12:
Three cases are distinguished:

Set position: The PLC received a *position* command. It (1) stops any movement of the axis, (2) connects the fastest motor of the axis and disconnects the others and (3) activates the fastest motor of the axis to move towards the target position.

Set pin: The PLC received a *set* command for a connector pin. If the pin is not already in the specified position, the PLC (1) stops any movement of the axis and (2) opens or closes the connector pin according to the command received.

Set motor: The PLC received a *set* command for an axis motor. If the command tells a motor to stop, the respective motor is stopped. If a move is requested, the PLC (1) checks the connector pin positions of the axis, (2) activates the correct motor. At incorrect pin positions the move is rejected.

Afterwards, the control cycle starts over again at Step 2.

Step 13:
If the PLC received a *bye* command, it (1) re-establishes the default position (c.f. Step 1), (2) confirms the shutdown to the AS and (3) shuts the CMM down.

An interruption of the network connection to the AS at any point of the control cycle leads to a stopping of all motor movement and continuous reconnection attempts. Any other kind of interruption initiates a shutdown of the machine according to Step 13.

Even though the concept of the control system defines that operations that require some kind of control logic should be performed by the server components, it is advised to include those routines into the controller, which are required to work quickly and reliably to prevent the machine from damage. Those routines are:

First touch action: This routine should prevent sensing head collisions by accelerating the reaction to a contact with an object. Due to performance optimizations of industrial PLCs, they can perform simple control tasks some orders of magnitude faster than a regular computer. The detection by the PLC instead of the server, hence, has the potential to reduce the reaction time from dozens or several hundreds of control cycles to less than one control cycle.

The *first touch action* is activated when the PLC detects a sensing head contact in one control cycle and there was no contact in the previous. At first it halts all machine movements. It then initiates the smallest possible move into the direction the contact came from by activating and immediately deactivating the slowest available motors of the axes that were moving when the contact occurred. This is repeated until no contact is detected, and followed by a single move into the direction that originally caused the contact. The result is the slightest possible sensing head contact.

This is done to relieve the force on the sensing head that the contact causes and as a consequence, should prevent it from breaking. Another advantage is that the distortion of the measured position is minimal, even when the measurement system does not have the ability to recognize and computationally eliminate the displacement of the sensing head tip caused by the contact pressure.

Check boundary reached: Since the standard command to move a motor activates it without a guarantee of a stop command ever arriving, it becomes necessary to regularly check the positions of motors and axes to prevent them from damage. One functionality of the *check boundary reached* routine is therefore to stop movements when they reach the absolute boundaries defined for those elements.

The second task of this routine is important for the *position* command. If an axis is moving and there is a target position defined for it, the current axis position is compared to its target position. If the axis gets within a specified distance to the target position, its fast movement is stopped and a slow movement initiated instead. This method is useful, since the chosen target position is usually close to or at an edge of the measurement object. Inaccurate production or fixation of the object can lead to a collision which at fast speed might do damage that is not done at a slow speed.

In order to execute the initialization phase, the PLC must read some parameters defined in the configuration file. The configuration contains the following information:

- Information about the machine such as its identifying machine number (ID)
- The information required to connect to the *AS* - its IP address and port
- The period of the *periodic state information* message: Due to the very fast control cycles of industrial PLCs, sending the machine state information at each control cycle might congest the server in a way that slows down its other processes. Therefore, the number of control cycles that the PLC should wait until sending the state information message again can be configured.
- The distance of the probing system from the target position of the *position* command at which a change to slow movement of an axis should be initiated (in millimeters)

4.2.4 Communication between Acquisition Server and CMM

Industrial PLCs know some or all of the IEC 61131-3 languages, whereas controllers like Raspberry Pi or Linux OS-based microcontrollers are object-oriented programmable. A common base are only their Ethernet ports that allow for TCP/IP transmission. Hence, there is no programming language defined to use for the PLC implementation, but a protocol for the communication between PLC and AS. In order to use the AS code for all the CMMs without reprogramming, the protocol must be followed by all the machine controllers. The protocol is based on plaintext and transferred via TCP sockets. All the available commands are listed in Figure 4.10. The messages sent by the AS always start with a three-letter identifier, whereas the messages sent by the PLC start with a two-letter identifier. Synchronous messages are transmitted only on creation and destruction of a connection. When the PLC establishes a TCP socket connection the AS responds with a request for identification in the form of a *WHO* message. The PLC answers this request by sending an *ID* message with its machine identification number as a parameter.

The regular *SET*, *POS* and *IN* operations are asynchronous. The TCP handles transmission errors, which ensures that the messages sent arrive at their recipient or the sender is informed if the transmission was not successful. No additional redundancy is necessary.

The commands the AS sends to actuate the machine are *SET* and *POS*. The *SET* command sets the value of a specified pin. Therefore, it has the pin number and the value to set as its two parameters. Mostly, the value is a binary indicating the opening/closing of a connector pin or activation/deactivation of a motor. An analog value is used to set the speed only at a continuous control motor. In contrast, the *POS* command expects one of the letters *X*, *Y* or *Z* as parameters for indicating an axis, and a numeric digital position value. The PLC is then responsible for moving the axis to the specified position (c.f. Subsection 4.2.3 Step 12).

The periodical state information that the PLC sends to the AS is identified by the keyword *IN* and further contains a list of pin numbers of the sensors and their alternating

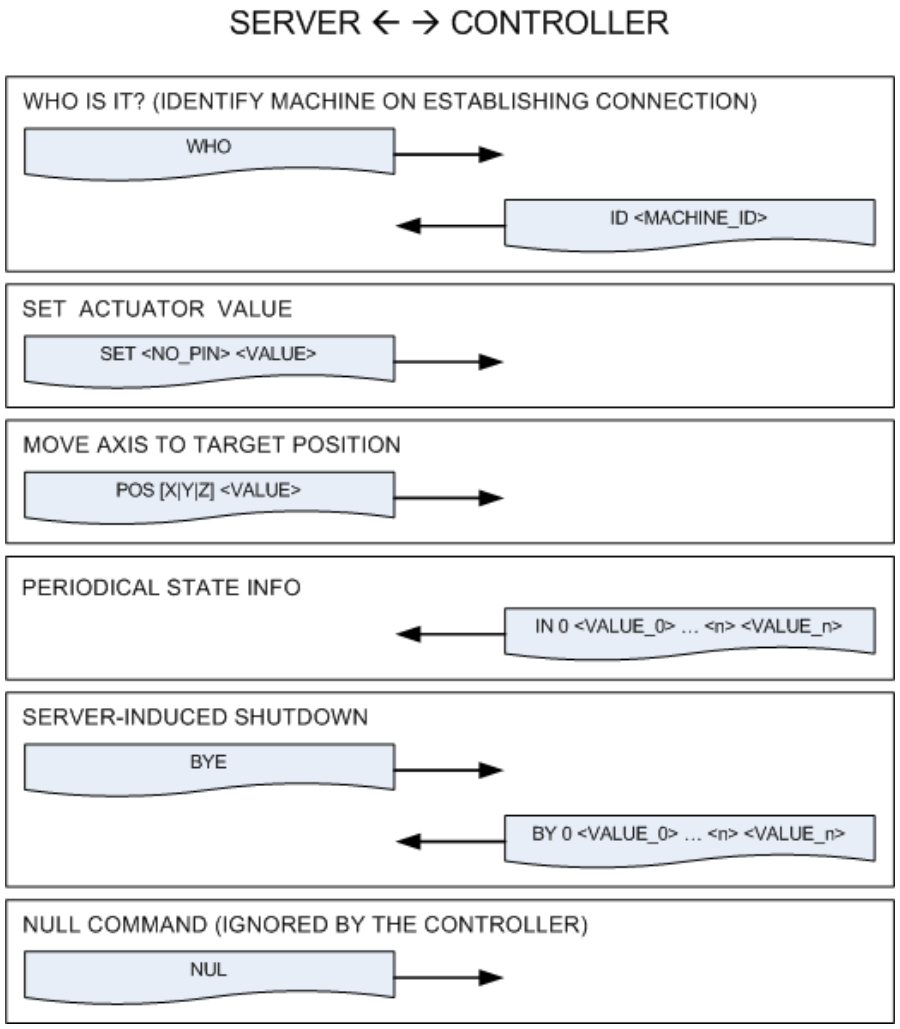


Figure 4.10: Communication protocol between AS and PLC.

values separated by whitespaces. The server holds the list of components of the CMM and maps them onto the pin numbers.

If the server needs to disconnect a machine it sends a *BYE* command without parameters. The PLC confirms with a *BY* message in the same format as a regular *IN* message just with the different identifier. The server can store this last machine state in its database.

Other commands sent from the server are ignored by the controller and could thus be used for checking the underlying connection. However, it is encouraged to use the designated *NUL* message without parameters for such purposes.

4.2.5 The Laboratory Server

The *Laboratory Server* connects many subsystems and performs the majority of the processing. It is a web application that is controlled by the users and based on the Model-View-Controller (MVC) pattern to separate the data model from the processing tasks and the graphical representation. The main task of the LS is to listen for user requests and respond to them appropriately. The necessary actions include:

Executing login requests: When users initially call up the website they land at the authentication page. If a directory service is configured for user authentication, the LS queries the necessary information from there. If not, it checks the database for the user information. As a result, access is rejected or granted and the website for probe execution displayed.

Further requests after successful login can be:

Machine management: Users with the administrator role have access to machine management functionality. They are provided with a list of all machines and their components. The current machine states of the CMMs are shown as well. Machines can be created, updated or a deleted flag can be set for a machine. If a machine that is currently active is manipulated, a user-machine-session is created by the LS with the administrative user as the operator and a BYE command is sent to shut down the machine.

Probe session: Users can create a probe session if a timeslot is available or reserved for them or if they are administrators. During the probe, the LS dynamically changes the GUI to only offer elements for the functions that are allowed to perform at the current machine state. User commands are executed by either interacting with the database or forwarding them to the AS. The LS issues all the necessary actions at the machine to implement a requested move or measurement operation. The command execution is logged and a preview of the probe protocol updated in the GUI. When requested, the LS ends a probe session.

Calculation engine: After ending a probe session, the LS executes all the due calculations in the calculation engine. The results are stored to the database, the probe protocol updated and the user informed in the GUI.

The LS additionally provides:

State information processing: As long as a machine is connected to the server, the LS receives machine state information messages from the AS. The LS processes the state information, forwards it to the user if a session is running and updates the database if necessary.

Experiment observation: At remote experiments on the real machine, the LS processes the webcam video. A live stream is embedded into the GUI and presented to the remote user. The Live View is presented in detail in Subsection 4.2.6.

4.2.6 The Live-View

When operators are standing right next to a machine, they can observe with their own eyes the position of the probing system relative to the measurement object. At remote or simulated experiments the operator is not present in a laboratory where the experiment takes place. This raises the need for technology to provide real-time visual control to the operator.

The obvious solution for experiments on physical machines is to mount cameras for viewing the experiment setup. The cameras must provide perspectives that allow for comfortable probing of different object shapes and sizes. Cameras might also be mounted directly onto the machine to keep the sensing head centered. Some cameras even offer remote control and can rotate or move in space.

Streaming a video file is easy, whereas real-time streaming is harder to realize. A streaming server is required that can convert the data stream from the webcam into a format to embed it into the website. One way would be to split an MP4-stream into small segments that are transferred as separate video files. However, this adds a delay of about two seconds. Since audio is not required for CMM control, another way would be to convert the video into *jpeg* images and transfer the single images. Furthermore, Javascript HTML5-video player implementations[1] [2] are available, which should work on all common browsers. The video stream is thereby sent to the client via web socket, decoded frame per frame and displayed in a canvas. However, the preferred approach is adaptive bit-rate streaming. It offers significant advantages in terms of both user-perceived quality and resource utilization for content and network service providers [ABD11] and is constantly tried to improve [BKTA17] for real-time interactive applications.

Adaptive streaming via Hypertext Transfer Protocol (HTTP) allows for browser integration without the need to include a video player. It splits the live video stream into a sequence of small HTTP-based file downloads. For Apple iOS with the Safari browser, the HLS (HTTP Live Streaming) codec is available. Other platforms implemented MPEG-DASH (Dynamic Adaptive Streaming via HTTP) [Sto11], which is the international standard ISO/IEC 23009-1:2014, specified in [ISO14]. It makes use of MSE (Media Source Extension) - an API allowing media data to be accessed from HTML video and audio elements - which is natively supported by the most common browsers. There are encoders that encode MP4 or x.264 streams into MPEG-DASH[3] and Apple HLS[4]. Integration into the GUI only requires linking Javascript libraries within the HTML definition. They are then executed on the client-side.

[1] https://github.com/131/h264-live-player
[2] https://github.com/Streamedian/html5_rtsp_player/wiki/HTML5-RTSP-Player
[3] https://github.com/slederer/DASHEncoder
[4] https://supportdownloads.adobe.com/product.jsp?product=160&platform=Macintosh

For simulated CMMs, the Live View is significantly different. Theoretically, a camera could also be pointed onto the graphical representation of the machine within the simulation software, which would save development effort. Unfortunately, this solution is far from elegant and would not allow for a switch of camera perspectives. Ideally, simulation software would allow for embedding a video stream of the simulation execution into a website. Whereas some simulation environments (e.g. LabVIEW) provide a graphical representation that can be embedded, specialized robotic simulation environments currently only provide a graphical representation directly at the machine the simulation runs at. However, robotic simulation software is required to accurately model the CMMs, which means that for the Live View functionality the operator must use external software. Therefore, a remote or virtual machine connection to the simulation server is established. As a *Remote Administration Tool (RAT)*, the Remote Framebuffer Protocol (RFB) which is used by Virtual Network Computing (VNC), Remote Desktop Protocol (RDP) and others is suitable.

The best convenience for the operator could be achieved by integrating the remote connection window into the GUI like at physical remote experiments. No separate user access management would be neccessary for the remote connection. In the past, such functionality was often realized via plug-ins based on NPAPI or ActiveX. Now, they are outdated since they are insecure, not supported by all common web browsers and require installation. A more modern approach are clients based on Javascript[5]. Despite the standardization advances in web technologies, there are still multiple implementations needed to cover the most common browsers. Web browsers are frequently updated and the embedded clients partly access browser internal interfaces that might change, which causes temporary service outage until the library is updated. There might soon be universal libraries for embedded clients but at the moment the implementation and maintenance effort seems to high to justify. Instead of an embedded solution, an external client is preferred. This client can be either stand-alone or browser-based.

Regardless of real or simulated laboratory, it should be possible for other authenticated users to observe an experiment. In order to achieve the best learning effects for the observers, they should not only see the Live View but also which control elements the operator presses. Hence, rather than including observation functionality into the program code, established screen sharing technologies can be recommended. The main benefits are: no individual development is required; screen sharing software often comes with IP telephony that allows group calls; and no security challenges arise. There are screen sharing tools available that are browser based[6] and do not require the installation of client software. Some can even be integrated into LMSs[7].

[5] e.g. Mstsc.js - https://github.com/citronneur/mstsc.js

[6] e.g. Google Hangouts - https://hangouts.google.com/

[7] e.g. Apache OpenMeetings - https://openmeetings.apache.org/
Moodle Video Conference - https://moodle.org/plugins/mod_videoconference

CHAPTER 5

Manejo - An Implementation of a NCS for CMMs

5.1 Setup and Functionality

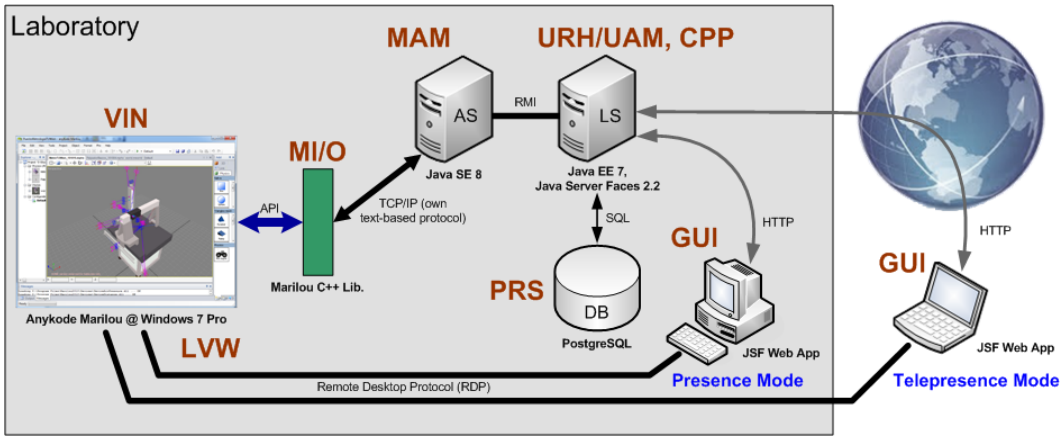


Figure 5.1: Overview of the control system for a simulated CMM.

For verification of the architectural design and to perform experiments (c.f. Chapter 6), a sample solution has been implemented. The implemented system is named *Manejo*, which is a Spanish acronym that stands for *MANejar, EJecutar, Organizar* and refers to three of the main tasks of modern CMM control systems - handling (*manejar*) the actuators, executing (*ejecutar*) CNC programs and organizing (*organizar*) machines. Manejo enables touch trigger probing of a measurement object both locally and via the internet. Properties of geometric shapes, such as the inner and outer diameter of a cylinder, the orientation of a plane in the three-dimensional space or the distance

between two points can be calculated. Measurements are performed on a simulation of a CMM (c.f. Section 5.2) which is controlled via a web browser-based GUI. User access management is done by the LS using the database and not a directory service.

As Manejo works with simulated CMMs, a suitable RAT is required for the Live View functionality. Due to the extensive implementation and maintenance effort that embedding a remote connection window into the GUI website causes (c.f. Subsection 4.2.6), Manejo uses third-party software instead. The simulation is deployed on a *Windows 7 Professional* operating system (OS). A *Remote Desktop Protocol* (RDP) session can be set up to the simulation server via pre-installed software that was optimized for this OS.

5.2 Simulation of a Numerex CMM in Anykode Marilou

A retrofitted machine of the vendor *Numerex* was chosen as the CMM to be simulated. It was refurbished in the AUM metrology laboratory such that each axis can be controlled in two different speeds. The machine is a bridge design with frictionless air bearings on each axis and was manufactured in the early nineties. It belongs to type *1818-10*, which means it has a measuring range of 18 inches at the X and Y axes and 10 inches at the Z axis. Further technical specifications are listed in Table 5.1. In order to collect meaningful results from experiments, it is crucial to accurately model the machine parameters.

Measuring Range:	475.2 x 475.2 x 254 mm
Approximate Weight:	400 kg
Maximum Distributed Work Piece Weight:	500 kg
Linear Accuracy:	+/- 0.00381 mm per axis
Volumetric Accuracy:	+/- 0.01143 mm per axis
Overall Height x Width x Length:	1803.4 x 863.6 x 863.6 mm
Clearance Under the Bridge:	355.6 mm

Table 5.1: Technical specifications of the Numerex 1818-10 CMM.

The CMM possesses twelve all-binary actuators and eleven sensors that were to be simulated. Each of the three axes has one fast and one slow *motor* that can be connected to their axes via a *connector pin*. The fast motors are connected with the axis bodies and are able to shift them directly, whereas the slow motors are connected to screws for fine tuning. A slow motor, when it is activated, rotates the screw of its axis and thereby adjusts the axis position precisely. There is one sensor per connector pin that checks whether the motor is connected to the machine or not. However, motors do not have sensors that report their states to the controller. Therefore, the machine controller can only inform the server about the signal it sends to the motor, and not if this signal causes an actual effect. Each axis also has one analog displacement sensor which measures the axis position.

Axis X

MOTOR ACTUATOR			SEN.	CONNECTOR PIN ACTUATOR			CONNECTOR PIN SENSOR		
PIN NAME	TYPE	VALUES		PIN NAME	TYPE	VALUES	PIN NAME	TYPE	VALUES
01 motorX--	binary	0: inactive/1: active	-	12 connectorXfast	binary	0: opened/1: closed	20 sXfastConnected	binary	0: opened/1: closed
03 motorX-	binary	0: inactive/1: active	-	13 connectorXslow	binary	0: opened/1: closed	21 sXslowConnected	binary	0: opened/1: closed
02 motorX+	binary	0: inactive/1: active	-	13 connectorXslow	binary	0: opened/1: closed	21 sXslowConnected	binary	0: opened/1: closed
00 motorX++	binary	0: inactive/1: active	-	12 connectorXfast	binary	0: opened/1: closed	20 sXfastConnected	binary	0: opened/1: closed

Axis Y

MOTOR ACTUATOR			SEN.	CONNECTOR PIN ACTUATOR			CONNECTOR PIN SENSOR		
PIN NAME	TYPE	VALUES		PIN NAME	TYPE	VALUES	PIN NAME	TYPE	VALUES
05 motorY--	binary	0: inactive/1: active	-	14 connectorYfast	binary	0: opened/1: closed	22 sYfastConnected	binary	0: opened/1: closed
07 motorY-	binary	0: inactive/1: active	-	15 connectorYslow	binary	0: opened/1: closed	23 sYslowConnected	binary	0: opened/1: closed
06 motorY+	binary	0: inactive/1: active	-	15 connectorYslow	binary	0: opened/1: closed	23 sYslowConnected	binary	0: opened/1: closed
04 motorY++	binary	0: inactive/1: active	-	14 connectorYfast	binary	0: opened/1: closed	22 sYfastConnected	binary	0: opened/1: closed

Axis Z

MOTOR ACTUATOR			SEN.	CONNECTOR PIN ACTUATOR			CONNECTOR PIN SENSOR		
PIN NAME	TYPE	VALUES		PIN NAME	TYPE	VALUES	PIN NAME	TYPE	VALUES
09 motorZ--	binary	0: inactive/1: active	-	16 connectorZfast	binary	0: opened/1: closed	24 sZfastConnected	binary	0: opened/1: closed
11 motorZ-	binary	0: inactive/1: active	-	17 connectorZslow	binary	0: opened/1: closed	25 sZslowConnected	binary	0: opened/1: closed
10 motorZ+	binary	0: inactive/1: active	-	17 connectorZslow	binary	0: opened/1: closed	25 sZslowConnected	binary	0: opened/1: closed
08 motorZ++	binary	0: inactive/1: active	-	16 connectorZfast	binary	0: opened/1: closed	24 sZfastConnected	binary	0: opened/1: closed

Other Sensors

PIN NAME	TYPE	VALUES
18 sAirPressure	binary	0: no pressure/1: pressure
19 sContact	binary	0: no contact/1: contact
26 sPositionX	analog position	in meters
27 sPositionY	analog position	in meters
28 sPositionZ	analog position	in meters

Other Pins

PIN NAME	TYPE	VALUES
ETH PLCtoServer	TCP/IP	text

Figure 5.2: Actuator & sensor specification of the simulated Numerex CMM.

In addition to the sensors that observe a specific axis, there are two sensors that report about the global machine state. One sensor checks the air pressure for the air bearings of the axes. If there is insufficient pressure, no movement may be triggered and the machine has to shut down immediately. At the simulation, the pneumatic system does not have to be modeled. It can be assumed that there is always sufficient air pressure in the system. Finally, there is the measuring system. It is binary, which means that it only recognizes whether a contact occurs but neither does it detect the direction nor the force of the contact.

These components are to be modeled together with the fixed parts of the CMM, later addressed by the controller and eventually configured at the server.

The CMM simulation was developed using *Anykode Marilou*, which is a robotic simulation platform focused on "an engine that reproduces, with an extremely high level of reality, the behavior of sensors and actuators with respect to real properties in a physical environment"[1]. *Marilou* was chosen because it allows setting mechanical constraints and surface properties like friction, calculates inertia and contact forces, offers various kinds of joints and sensors, motors and cylinders and has been used successfully at the UTN in Buenos Aires multiple times for complex robotic simulation tasks. *Marilou* simulations are based on one *world* file that models the surroundings of the scene, one or more *physics* files that specify the simulation models, one *configuration* file that defines environmental conditions as well as which files should be used, one executable *simulation script*, and one *simulation project* file that puts it all together.

The machine model was created in a sequence of steps:

Step1 - Building the machine body: At first, every part of the physical CMM was measured. With the measurement results, the rigid machine base could be created from the rubber feet on the bottom to the granite table on the top. For each machine part the position, dimensions, mass and contact properties were defined. In the end, all the parts were grouped to constitute one common *rigid body* called the "base". In the same way, the moveable bridge was created as three rigid bodies - one per axis. The screws for fine adjustment are also part of the original machine and were added as separate bodies.

Step2 - Adding motor parts and pins: As a next step, the mechanical parts of the motors and the connector pins were added. The pins that connect the fast motors were modeled in purple color for easy recognition, those of the slow motors in light blue. Each body of a motor or pin was linked to the body it should interact with using a *slider* joint, which allows for translational movement. At the sliders, the position limits were set according to the measuring range denoted in the machine specification. Additional force parameters like the force restitution on the limits and the torque and force that lead to a breaking of the joint were configured.

[1] Anykode Marilou Website: http://www.anykode.com

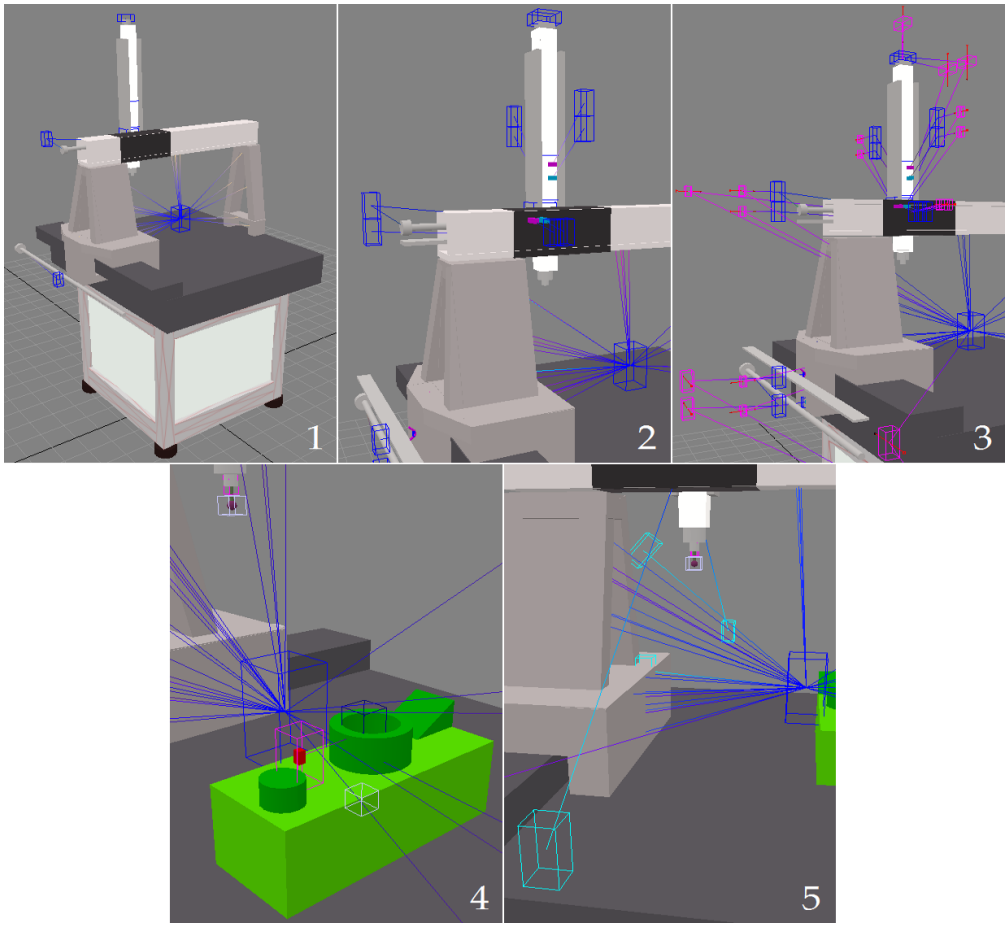


Figure 5.3: Modeling steps of the simulated CMM.

Step3 - Adding logical devices: Each slider was equipped an actuating cylinder that is addressable by the controller using the *Marilou Open Devices Access* (MODA) API. For a cylinder the maximum force and speed was set and either air or mechanical - via screw - constraint chosen.

Step4 - Adding probe head and measurement object: The probe head and measurement object were both created in a separate *physics* file and later linked to the machine. That way, they are easily exchangeable for other experiments and might also be added to other machines. A sphere with a diameter of five millimeters on a twenty millimeters long stylus of two millimeters radius was modeled for the probe head. It was linked to the machine model and breaking force and torque were configured. The measurement object was linked via a *fixed* joint because it is mounted firmly to the CMM during the probe. It exhibits a number of different standard geometries that can be used to test all the measurement functionality implemented by the Manejo control system (c.f. Subsection 6.2.3).

Step5 - Adding viewpoints: *Marilou* offers the *viewpoint* render utility for simulating cameras. During a simulation, the operator can switch viewpoint at any time, using a drop-down menu in the simulation window. These viewpoints can either be placed at a fixed location or added to a body such that they move along when the body is moving. Different sets of viewpoints holding *camera* devices were experimented with until a configuration was found that can facilitate visual control of the whole probing process. As a further aid to orientation during the probing process, arrows were placed at strategic points of the CMM to indicate the directions.

The simulation script can be developed in C# or C++, it communicates with the model using the MODA API and can be compiled into an executable file. It follows the control cycle specified in Subsection 4.2.3. For the Numerex simulation, C++ was used as the programming language. Core class and class containing the main method of the simulation is *NumerexControl*. The C++ simulation has the advantage of allowing multithreading, hence the server communication is carried out in an own thread named *InputThread*. Some selected parts of the simulation script are presented in the following.

The program is run via a batch file that loads the *Marilou* project and connects it to a new simulation instance. The simulation script establishes a blocking TCP socket connection to the AS in the new thread and initializes auxiliary variables and machine devices. Each device can be used to control one actuator or receive information from one sensor. When everything is started up, the actuators are set to their initial position. Ultimately, a new InputThread is started that uses the TCP connection to receive commands from the AS and executes them by calling functions of the *NumerexControl* class.

Listing 5.1: Simulated PLC NumerexControl.cpp - initialization

```
int contact = 0; //0 no contact, 1 approach autom., 2 contact maintained
...
NumerexControl::NumerexControl() { inputThread = new InputThread(this); }

void NumerexControl::runInputThread(void * arg) {
    while(repeat) { inputThread->receiveInstruction(); }
}
void NumerexControl::stopInputThread() { repeat = false; }

int _tmain(int argc, _TCHAR* argv[])
{
    NumerexControl * control = new NumerexControl();
    ...
            /* 1. INITIALIZATION - Find devices */
            pDevice_AXCarro= robot->QueryDeviceActuatingCylinder("
                sliderFrontal/a1/xSlider");
            ...
                /* 2. INITIALIZATION - Est. initial state */
                pDevice_AXPinzaFino->GoPosition(PINZA_X_POS_CLOSED,
                    SPEED_FAST);
                ...
```

```
        _beginthread( NumerexControl::runInputThread, 0, 0); //listen
            for incoming commands from server and execute them
```

After the initialization is finished, the control loop starts. The same steps are repeated until an interruption occurs.

First, the routine checks whether a new contact occurred since the last cycle. This is the case when the probe head did not sense a contact before (variable *contact* has value ”0’)’ but does so now. The *firstTouchAction*-function takes as parameters the information whether and at which speed the axes have been moving at the time of the contact. This information is necessary for the controller to perform the counter-movement. While the *firstTouchAction* is performing, the variable *contact* is set to ”1”. When the action is finished, it is set to ”2” and is reset to ”0” when the contact has ended.

Next, the *checkBoundaryReached*-function might stop or slow down movement of the axes.

The cycle is concluded when the machine state information message to the AS is sent, using the respective function of the inputThread.

Listing 5.2: Simulated PLC NumerexControl.cpp - control cycle

```
/*****************************************************/
/* 3. END OF INITIALIZATION, START OF PROGRAM LOOP! */
/*****************************************************/
while(repeat) {

    /* check for new sensing head contact */
    if(pDevice_SensingHead->IsTutching() == 1 && contact == 0) {
        control->firstTouchAction(xMoving, yMoving, zMoving);
    }
    else if(contact == 2 && !pDevice_SensingHead->IsTutching()) {
        contact = 0;
    }

    /* check for motor positions reaching their extrem values */
    control->checkBoundaryReached();

    /* send periodic status information */
    std::string s = control->getMachineState();
    char *data = new char[s.length() + 4];
    strcpy(data, "IN ");
    strcat(data, s.c_str());
    NumerexControl::inputThread->connection->sendMessage(data, (int)
        strlen(data));
    delete [] data;
}
```

The *receiveInstruction*-function of the inputThread is called continuously until the simulation is finished. It waits for incoming commands from the AS and extracts the command identifier (keyword).

If the command is a *SET* command to set the state of a specific actuator, the pin number for identifying the actuator and the value are extracted. The actuator and value information is then used to call the correct function with the right parameters. This function establishes the prerequisites for the move and finally the move of the actuator itself.

If a *POS* command to move an axis to a specific position is received, the axis letter and position value are extracted. The respective *move<Axis>ToPosition*-function is then called with the desired position and the instruction to use the fastest motor as parameters. The function then sets the preconditions and target position for the axis and initiates a fast move of the correct actuator. When the *checkBoundaryReached*-function detects that the axis is within five millimeters from the target position, it automatically slows down the movement.

If the inputThread receives a *BYE* command, both the receiveInstruction-function and the main control cycle are set to not repeat anymore and all movements are stopped. A *BY* message is sent to the AS, containing the last machine state before shutdown and the simulation is ended.

In every other case, the input is ignored and the *receiveInstruction*-function is called again.

Listing 5.3: Simulated PLC InputThread.cpp - receiveInstruction()

```
...
if(keyword.compare("SET") == 0) { //command: SET
    ...
    /* CONTROL ACTUATORS! */
    if(strcmp(actuator, "0") == 0) control->moveX(1, 0, value == '1'? true :
        false);
    ...
    else if(strcmp(actuator, "11") == 0) control->moveZ(0, 1, value == '1'?
        true : false);

    else if(strcmp(actuator, "12") == 0) control->setXFastConnected(value ==
        '1'? true : false);
    ...
    else if(strcmp(actuator, "17") == 0) control->setZSlowConnected(value ==
        '1'? true : false);
}
else if(keyword.compare("POS") == 0) { //command: POS
    if(control->getContact() == 1) { control->setContact(0); }
    ...
    if(strcmp(axis, "X") == 0) {
        control->moveXToPosition(position, true);
    }
    else if...
}
else if (keyword.compare("BYE") == 0) { //command: BYE
    control->stopInputThread();
```

```
    control->haltMachine(true);

    std::string s = control->getMachineState();
    char *data = new char[s.length() + 4];
    strcpy(data, "BY ");
    strcat(data, s.c_str());
    connection->sendMessage(data, (int) strlen(data));
    delete [] data;
}
else {} //command: WHO or undefined command
```

The simulation was later used to test and adjust the motor power for smooth movement of the axes without causing vibrations, followed by experiments to find out which motor speeds allow for efficient operation without breaking off the sensing head. The fast motors were set to five centimeters per second, the slow motors to two millimeters per second.

5.3 Implementation Details of the Acquisition Server

The AS was realized in plain *Java SE 8*. It is a stand-alone Java application that does not require access to external data sources or libraries. It consists of a *connection service* and a *communication service*. The connection service performs the connection management with the CMMs, while the communication service is responsible for sending commands to the machines and receiving machine state information. There is only one instance of the connection service running, whereas each machine has its own communication service thread. The communication services correspond to the MachineHandler in Figure 4.7.

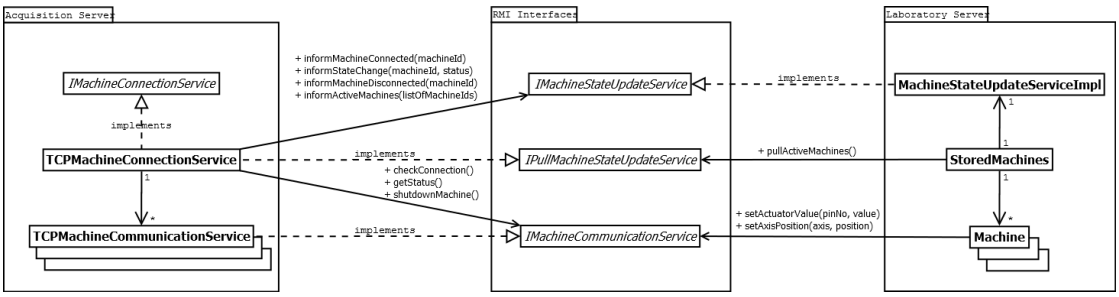


Figure 5.4: RMI communication between AS and LS.

Towards the LS on the other side, the AS has a Remote Method Invocation (RMI) - the Java version of the RPC - connection (see Figure 5.4).

The *TCPMachineConnectionService* class is the TCP implementation of the general *IMachineConnectionService* interface and holds a map that links the IDs of all the connected machines to their communication service handlers. It additionally implements an *IPullMachineStateUpdateService* that allows the LS to trigger it into sending the list of the IDs of the active machines to the LS by using the *informActiveMachines*-method of an implementation of the *IMachineStateUpdateService* interface. The

IPullMachineStateUpdateService interface is implemented on the LS side by the class *StoredMachines*, the *IMachineStateUpdateService* interface is implemented by the class *MachineStateUpdateServiceImpl*.

Additionally to the LS-side pull, the *TCPMachineConnectionService* informs the *MachineStateUpdateServiceImpl* whenever a machine connects, disconnects or when a machine state changes. The AS notifies a change in machine status when the value of a binary sensor changes or a change of the value of a numeric sensor exceeds a certain threshold. Furthermore, the AS sends a list of all active machines on startup or after the LS restarted.

The *TCPMachineCommunicationService* class is the TCP implementation of the general *IMachineCommunicationService* interface. Each instance handles the communication with one machine and offers checking the connection of the AS to the machine, retrieving the machine status, setting the value of an actuator, setting the position of a machine axis to a specified value and shutting down the machine. The LS only uses the methods for setting the actuators and positions of the axes in its *Machine* class. This means that a User currently cannot shut down a single machine remotely and that the only class retrieving the status of the machines is the *MachineStateUpdateServiceImpl*. The methods *checkConnection*, *getStatus* and *shutdownMachine* are used by the *TCPMachineConnectionService* for machine management purposes. The *IMachineCommunicationService* interface could also be split into separate interfaces for the methods that the AS uses and those that the LS uses, but that limits the possibilities for future extension of functionality of the LS.

The communication service waits for incoming state information messages from the CMM via TCP socket. When a message arrives, an internal method called *updateSensorValues* is called. If the TCP connection is interrupted, the CMM is removed from the list of active machines.

The machine state is stored in an object of a class called *StatusMessage* that has to fields: *current* and *previous* and a method called *changed* with return type *boolean* that compares the current value with the previous one. When a new machine state is received, the method *updateSensorValues* changes the field *previous* of the *StatusMessage* object to the old machine state and the field *current* to the new machine state. If the method *changed* returns the value *true*, the communication service informs the LS about the change of a machine state by calling the method *informStateChange* of the class *MachineStateUpdateServiceImpl*.

Listing 5.4: Acquisition Server - TCPMachineCommunicationService.java

```
public void run() {
    ...
        ... //code for binding the service to the RMI registry
        String response;
        while ((response = this.receive()) != null) {
            String[] parts = response.split(" ");
```

```
            if (parts[0].equals("IN")) {
                ...
                this.updateSensorValues(response);
                ...
            }
        }
        ... //close socket and unbind the service from the registry
    this.cs.removeFromActiveMachinesList(this.machineId);
    ...
}

private void updateSensorValues(String response) throws
    ServerNotFoundException, IncompatiblePinException {
    ...
    if(this.status.getCurrent() != null)
        this.status.setPrevious(this.status.getCurrent());
    this.status.setCurrent(response);
    ...
        if (this.status.changed()) {
            ...
                this.stateUpdateService.informStateChange(this.machineId,
                    this.status);
            ...
        }
    ...
}
```

5.4 Implementation Details of the Laboratory Server

The LS handles requests of registered users primarily for machine control, but also for user and machine management. These tasks can be fulfilled from all input devices that support modern web browsers. Hence, the GUI is implemented as a request-based web application that supports HTTP sessions.

The application should follow the Model View Controller (MVC) design pattern for higher flexibility and better maintainability. There are two types of frameworks that facilitate the development of MVC web applications: component-based and request/action-based MVC frameworks. Request-based frameworks offer more fine-grained control over the rendered view, whereas component-based web MVC frameworks take care of input validation, conversion and updating the model. The flexibility and scalability advantages of request-based technologies do not add much of a benefit to Manejo and server-side session management is desired.

Therefore, Manejo uses the component-based framework *Java Server Faces (JSF) 2.2* for the front-end development at which simple *JavaBean* classes build the Model and an *XHTML* file builds the View. Instead of using Javascript code, UI components are annotated with tags. In the scientific literature, simulated laboratory setups have often used Java for browser-based applications - *Applets* back in the earlier days and *Servlets* more recently. [ÖKA10], [PBHJGR09], [DWW+12], [HDL+08], [CCLOD13], [APA16]

The open-source Rich Internet Application (RIA) development framework *ICEFaces 4*[2] extends the application by additional UI components and *Ajax Push* functionality. As a database, *PostgreSQL* was used but all common relational database management systems (RDBMS) that use *SQL* as their query language are equivalent for the purpose.

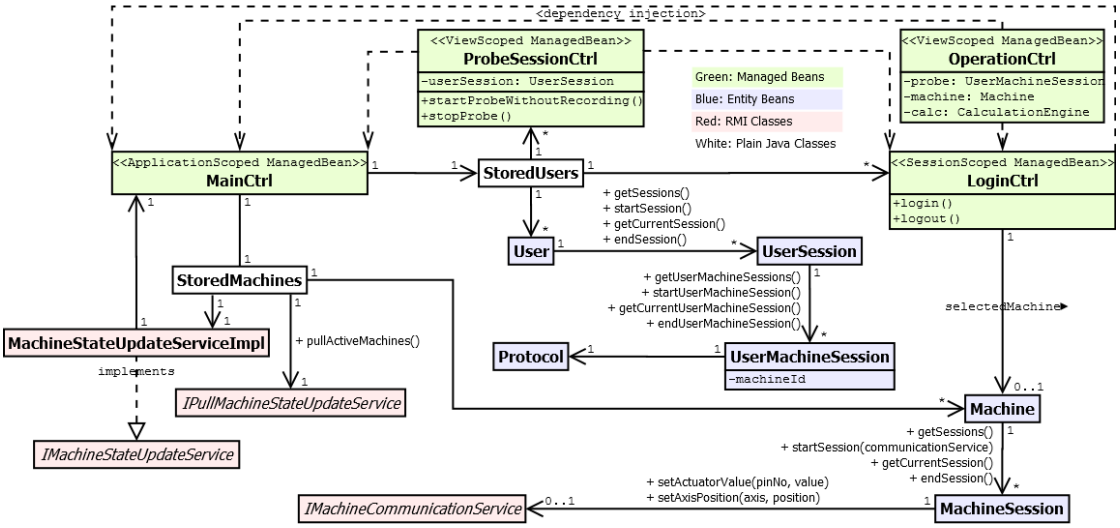


Figure 5.5: Core architecture of the LS.

The Figure 5.5 is a diagram of the core architecture of the LS, that provides an overview of its structure and interactions. For the sake of simplicity, it only depicts the classes that are most important for conducting a probe and omits most of the domain entity classes like machine components or probe types as well as classes for machine configuration and probing process management tasks.

On startup of the LS, the class *MainCtrl* starts the user management *StoredUsers* and machine management *StoredMachines*.

The *StoredUsers* class provides access to the *User* entities in the database and links HTTP-session state information to those users that are online. This information contains the *LoginCtrl* managed bean, which authenticates a user and grants access to the XHTML site for machine control ("machinesession.xhtml") on login and invalidates the session and withdraws access on logout. The *LoginCtrl* creates, via the *StoredUsers* class, a new *UserSession* entity on login and ends it on logout. The second HTTP object that the *StoredUsers* holds available for the user is the *ProbeSessionCtrl* managed bean. It is created when the *machinesession.xhtml* site is loaded and can start and end *UserMachineSessions* for the *UserSession* that was created by the *LoginCtrl*.

When a *UserMachineSession* is started, the *OperationCtrl* managed bean gets loaded. It injects the *MainCtrl*, accesses the *Machine* object that was selected to perform the

[2]ICEFaces JSF Framework - http://www.icesoft.org/java/projects/ICEfaces/overview.jsf

probe on via *StoredMachines*, and uses the *IMachineCommunicationService* to operate the machine according to the user input via the *machinesession.xhtml* site. Every action is added to the *Protocol*.

The *StoredMachines* class provides access to the machines in the database and for the initialization of the RMI connection to the AS on startup by creating a *MachineStateUpdateServiceImpl* object and registering it at the RMI registry. Whenever it is required, it further triggers the AS into sending a list of connected machines to the *MachineStateUpdateServiceImpl* by using the *IPullMachineStateUpdateService*.

The *MachineStateUpdateServiceImpl* is contacted by the AS on startup of the AS, when a CMM connected or disconnected. It then uses the methods of the *StoredMachines* class to create and end machine sessions and to update the values of the machine components.

Listing 5.5: Laboratory Server - StoredMachines.java

```
public void createMachineSession(long machineId) throws ... {
    if(this.machines.containsKey(machineId)) { //if machine exists ...
        if(this.machines.get(machineId).getState() == EMachineState.INACTIVE)
             { //... and no session is running
            ...
                this.machines.get(machineId).startSession((
                    IMachineCommunicationService) registry.lookup(this.
                    bindingNameCS + machineId));
            ...
}

public void endMachineSession(long machineId) {
    if(this.machines.containsKey(machineId)) {
        ...
            this.main.getUserMgmt().getProbeSessionBean(machineId).stopProbe
                ();
        ...
        this.machines.get(machineId).endSession();
    }
    ...
}
```

A new machine session is only created, if the requested machine exists in the database and has no session running yet. That is the case when its state is *INACTIVE*. The method *startSession* of the class *Machine* adds a new *MachineSession* object to the list of sessions of the machine and changes the machine state to *CONNECTED*.

When a machine session should be ended, *StoredMachines* queries the user management to get the *ProbeSessionCtrl* object of the operating *User* of a possible *UserMachineSession*. If a *User* is currently connected to the *Machine*, the method *stopProbe* of the *ProbeSessionCtrl* object is called. Afterwards, the method *endSession* of the *Machine* deletes the *IMachineCommunicationService* from the *MachineSession* and resets the machine state to *INACTIVE*.

A probe begins when one of the methods *startProbeWithoutRecording*, *startProbeWithRecording* or *startProbeProgramExecution* of the class *ProbeSessionCtrl* is called. They all add a new entity of the respective subtype of *UserMachineSession* to the list of probes of the *UserSession* and change the machine state to *PROBING_WITHOUT_RECORDING*, *PROBING_AND_RECORDING* or *PROBING_PROGRAM_RUNNING* and keep a reference to this newly created *UserMachineSession*.

The probe is stopped by setting an end-time for the *UserMachineSession*, changing the machine state back to *CONNECTED* and resetting the *Machine*, the *UserMachineSession* reference and the *OperationCtrl* object of the *ProbeSessionCtrl*.

Listing 5.6: Laboratory Server - ProbeSessionCtrl.java

```
...
private UserMachineSession activeSession;
...
public void startProbeWithoutRecording() {
    long machineId = this.loginCtrl.getSelectedMachine().getId();
    this.loginCtrl.getUser().getCurrentSession().startUserMachineSession(new
        ProbeWithoutRecording(machineId));
    this.loginCtrl.getSelectedMachine().setState(EMachineState.
        PROBING_WITHOUT_RECORDING);
    this.activeSession = this.loginCtrl.getUser().getCurrentSession().
        getCurrentSession(); //UMS
    ...
}

public void startProbeWithRecording() { ... }

public void startProbeProgramExecution(ProbeProgram program) { ... }

@PreDestroy
public void stopProbe() {
    this.loginCtrl.getUser().getCurrentSession().endUserMachineSession();
    this.loginCtrl.getSelectedMachine().setState(EMachineState.CONNECTED);
    this.loginCtrl.setSelectedMachine(null);
    this.activeSession = null;
    FacesContext.getCurrentInstance().getViewRoot().getViewMap().remove("
        operationCtrl"); //Remove operationCtrl from context
    ...
    }
```

The class *OperationCtrl* implements an *IMachineControlService* interface that defines the functions that a user can perform during a probe. There are two types of functions - such that send a command to operate the actuators of the CMM and such that change values in the database. The former connect, disconnect, move or stop a motor. The latter define what is measured (by starting an operation), change the state of the measurement (by measuring the current machine position), cancel or finish an operation. When an operation is finished, the calculation engine calculates the properties of the workpiece measured.

Listing 5.7: Laboratory Server - IMachineControlService.java

```
/* FUNCTION GROUP 1: DIRECT ACTUATOR CONTROL (simple movement, no state) */
public boolean moveX(int axisMotorNo, int move) throws ...;
public boolean moveY(int axisMotorNo, int move) throws ...;
public boolean moveZ(int axisMotorNo, int move) throws ...;

public boolean setXConnected(int axisMotorNo, boolean connected) throws ...;
public boolean setYConnected(int axisMotorNo, boolean connected) throws ...;
public boolean setZConnected(int axisMotorNo, boolean connected) throws ...;

public boolean moveXToPosition(double relativeTargetPos) throws ...;
public boolean moveYToPosition(double relativeTargetPos) throws ...;
public boolean moveZToPosition(double relativeTargetPos) throws ...;

/* FUNCTION GROUP 2: PROBE OPERATIONS (operations that have a state) */
public boolean measurePosition() throws ...;

public boolean calibrateMachine() throws ...;
public boolean probePoint() throws ...;
public boolean probeInnerDiameter() throws ...;
public boolean probeOuterDiameter() throws ...;
public boolean probePlane() throws ...;
public boolean probeDistancePlanePoint() throws ...;
public boolean probeDistanceParallelPlanes() throws ...;

public boolean cancelOperation();
public boolean finishOperation();
```

CHAPTER 6

Research on Remote CMM Operation

6.1 Properties of Internet Delay

The Internet has evolved from a simple store-and-forward network to a very complex communication infrastructure. Present-day demands for flexibility, performance and security necessitate that network traffic is processed repeatedly at the origin, destination and even inside the network. Protocols and applications that require such additional processing include Network Address Translation (NAT), firewalling, and Virtual Private Network (VPN) tunneling. Increasingly complex services that demand more intensive processing contribute this development. [RWW04] More and more security features and services will have to be implemented in the future. [Kiz17] Long distances must be bridged and transmission channels expanded as services are targeted at global audiences. [Mit16]

Technological development enables this progress, but each layer of complexity adds to the time it takes to transfer the data between server and client. The sum of this time is called *latency* and is particularly relevant for real-time applications. Four types of delay contribute to the total latency:

Transmission delay: The time it takes to send the data packet onto the wire.

Propagation delay: The time it takes to transmit the packet via the wire.

Processing delay: The time it takes to handle the packet on the network system.

Queuing delay: The time in which the packet is buffered before it can be sent.

Historically, the biggest contributors to the total latency were *propagation delay* and *queuing delay*. However, the complex services and security measures mentioned above increase the relevance of the *processing delay*. The processing delay is mostly generated at the endpoints of a communication and can only be reduced by improving router hardware or software for the different computationally intensive procedures like NAT or VPN. [RWW04]

The transmission delay depends on the available data rate of the transmitting link and is not correlated to the distance between the client and the server. It is usually small for fast links and average packet sizes. As the transmission delay is produced mostly within the realm of the organization that hosts the laboratory, it can be minimized by providing a suitable network connection with sufficient bandwidth.

Queuing delay arises if the packets arrive at a faster rate than a router is capable of processing and they must temporarily be stored in a buffer. It is very difficult to predict - there might be no queuing delay at all or the packet might be dropped and have to be resent after a timeout. The queuing delay can be reduced by developing better routing mechanisms [ZM04][LPJ16] or queue management algorithms [NJ12] but not by a regular service provider such as a remote laboratory. The relative importance of queuing delay compared to the other types of delay is decreasing.[JT01]

Much research is carried out by analyzing and modeling the network delay produced by those factors. It can be modeled as a constant delay, an independent random delay, and a delay with known probability distribution governed by the Markov chain model [LY05][RRPI03]. Models can also be generated using system identification tools [KM06]. Similarly, attempts are made for the analysis of the available bandwidth of a network.[JD02]

The information gathered is used for designing and implementing different kinds of delay reduction methods. These methods comprise mathematical-, heuristic-, and statistical-based approaches.[NBW98][ZLL+17] They follow the principle that if the delay can be predicted, then enough resources can be allocated or a better routing path can be found.

Another approach is to improve the user experience by stabilizing instead of reducing the delay. Multiple methods exist for this approach, of which the buffering/queuing method is the most common.[RC05][TdS05] The buffering method can be used even if the transfer network is not under control of the laboratory provider. However, it is more difficult to guarantee a stable round-trip time from the moment a command is issued until the moment when visual feedback arrives and is displayed on the screen of the operator.

Propagation delay is dictated by the distance and the medium through which the signal travels. The speed of light is the maximum speed at which all energy, matter, and information can travel. This observation places a hard lower boundary on the propagation time of any network packet. The speed of light in an optical fiber channel, through which most of our packets travel for long-distance hops, is around 200,000,000 meters per second.

Route	Distance	Time, light in vacuum	Time, light in fiber	Round-Trip Time (RTT) in fiber
New York to San Francisco	4,148 km	14 ms	21 ms	42 ms
New York to London	5,585 km	19 ms	28 ms	56 ms
New York to Sydney	15,993 km	53 ms	80 ms	160 ms
Equatorial circumference	40,075 km	133.7 ms	200 ms	200 ms
Vienna to Buenos Aires	11,819 km	39 ms	59 ms	118 ms

Table 6.1: Signal latencies in vacuum and fiber. (adapted from [Gri13])

Delay	User perception
0–100 ms	Instant
100–300 ms	Small perceptible delay
300–1000 ms	Machine is working
1,000+ ms	Likely mental context switch
10,000+ ms	Task is abandoned

Table 6.2: Latency and user perception. (adapted from [Gri13])

The numbers in Table 6.1 show the time a packet would travel on the shortest path between the cities. In practice, the packet will take a much longer route and each hop along this route will introduce additional routing, processing, queuing, and transmission delays. As a result, the actual RTT between New York and Sydney works out to be in the 200–300 millisecond range.

Studies have shown that most users report a perceptible ”lag” at a delay of over 100–200 milliseconds. Once the 300 millisecond delay threshold is exceeded, the interaction is often reported as being "sluggish". At the 1 second barrier, many users have already performed a mental context switch while waiting for the response. After 10 seconds have passed, unless feedback on the progress is provided, the task is frequently abandoned. Those thresholds (c.f. Table 6.2) remain constant, regardless of the type of application (online or offline), or medium (laptop, desktop, or mobile device). [Gri13]

Significant latency is introduced not only at long distances. The local Internet Service Provider (ISP) needs to route the cables throughout the neighborhood, aggregate the signal, and forward it to a local routing node - all of which cause delays. The U.S. Federal Communications Commission (FCC) annually measures the latency from the user to the closest node within the core network of the ISP for thirteen providers that serve about 80% of the population. Their annual *Measuring Broadband America* report [FCC16] shows that latencies have remained relatively stable over time, a condition that is known as *the last-mile problem*. As it can be seen in Figure 6.1, they lie between 10 and 75 milliseconds, depending on the provider and technology used. Although the study was

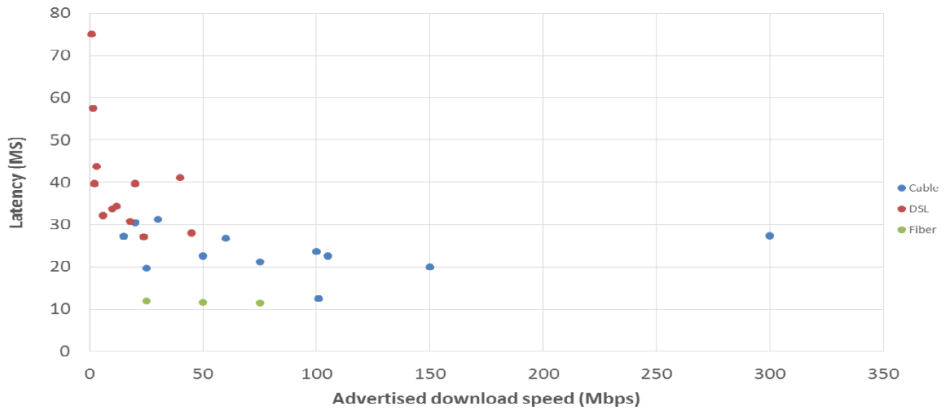


Figure 6.1: Latency for terrestrial ISPs by technology and advertised bandwidth. (from [FCC16])

conducted in the United States, last-mile latency is a challenge for all ISPs, regardless of location.

The FCC method is defined as follows: *"Latency and packet loss: Measures the round-trip times for approximately 2000 packets per hour sent at randomly distributed times. Response times less than 3 seconds are used to determine* ***median latency****. Acknowledgements not received or received with a round-trip time greater than 3 seconds determine* ***packet loss****."* [FCC16]

The effect that the long distance between Austria and Argentina has on the signal latency can be determined in almost the same manner as the FCC researched the last-mile latency. To this end, a number of tests were performed locally in Vienna and between Vienna and Buenos Aires. More than 3000 *ICMP echo* ("ping") messages were sent on different days of the week and at different times of the day, in a variety of packet size combinations, to each of the destinations. Sending very small and the largest possible packets in parallel to both destinations turned out to be one of the more revealing experiments.

Figure 6.2 plots a test series of four parallel runs of the ping command with 300 messages each. The web server of the UTN in Buenos Aires limits the buffer size of ping messages to 996 bytes, which is why this size was chosen to determine the latency for the larger packets. The small packets had a buffer size of 32 bytes. The resulting diagram reveals that clearly the delay patterns are very similar for the short and long distance. This suggests that most of the delay variation ("jitter") is produced in the source network and on the first and last mile respectively. Another observation is the effect that the packet size has on the overall latency. Whereas the mean latency for small packets inside Vienna was 22.66 milliseconds (median: 11 ms), the mean for large packets was 33.09 milliseconds (median: 22 ms) - a difference of about 11 milliseconds. At the long distance from Vienna to Buenos Aires, the mean and median for small packets were 259.92 and 247.5 milliseconds and for large packets the mean and median were 273.18

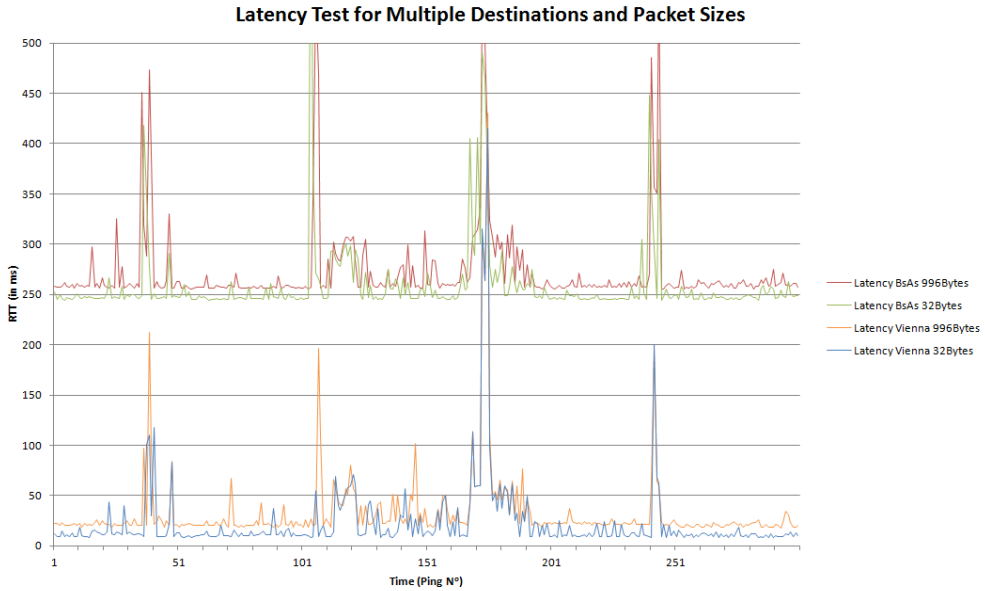


Figure 6.2: Results of Latency Tests.

and 259 milliseconds.

The difference between the mean and median values indicates a skewed distribution at which outliers towards the higher end tend to be much bigger than outliers towards very low latencies. In fact, there is a strong lower boundary defined by physical limits and network characteristics. Kalman and Girod [KG04] have shown that shifted gamma-distributions can be good models for the packet delay. Sato et al. [SAO05] pointed out that the heavy-tail of the RTT distribution can be modeled by the mixture of non-heavy-tailed distributions (i.e., normal distribution). However, hybrid simulations are preferable to modeling the whole internet delay for experiments (c.f. Subsection 6.2.1).

6.2 Experiment Preparation

6.2.1 Simulation of the distance effect of the latency

The delay below 300 milliseconds measured in the previous tests can only be achieved under the condition that the internet connection bandwidth of the laboratory where the control system operates the machine, as well as the internet connection bandwidth of the user are not limiting factors for the control and the video stream transmission.

The similarity between the long-distance and short-distance latency patterns allows one to perform meaningful delay experiments in two ways: from the actual distance in Buenos Aires or via the internet from within Vienna by simulating only the additional delay that long-distance operation would produce. In order to create a realistic delay simulation the difference in results between the tests to Vienna and to Buenos Aires that were collected

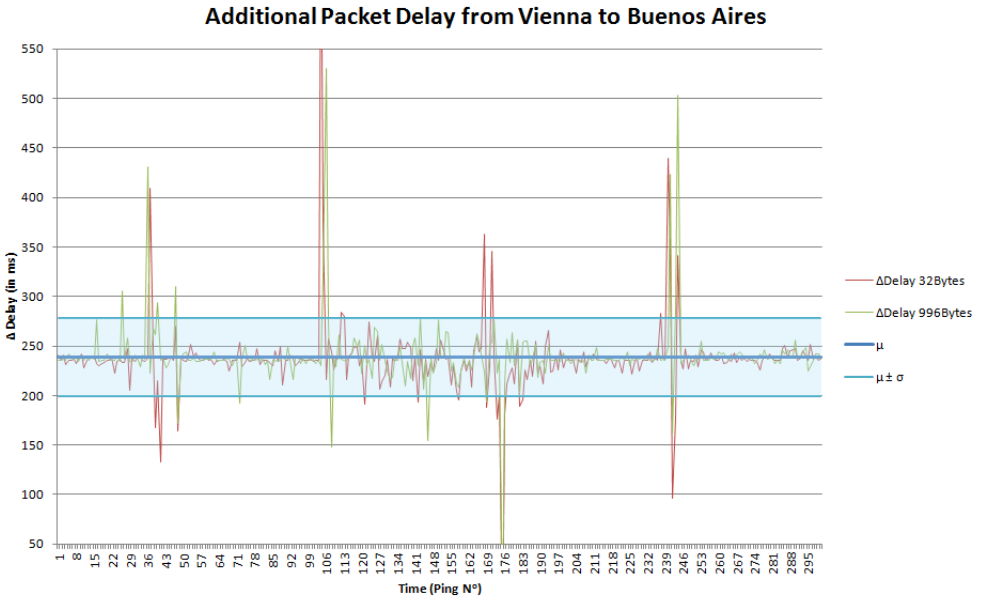


Figure 6.3: Mean and standard deviation of latency from long-distance transmission.

in the end of Section 6.1 was extracted and analyzed. The Figure 6.3 plots this difference over time in a graph which resembles a normal distribution. The mean is at 238.68, the median at 236 and the standard deviation at 39.675 milliseconds. During the tests, one out of every 1,200 packets got lost, which is a packet loss rate of 0.00083.

However, the time between the issue of a command by the user until displaying the results in the form of numeric or binary state information as well as visual feedback via remote desktop connection can be much longer than the RTT of a single packet. This so-called *"Page Load Time" (PLT)* includes also the necessary time for processing and preparing the answer to a user request on server side, the processing time for displaying the response in the web browser of the client, as well as the time it takes the server to put all the required packets of an answer on line. Figure 6.4 depicts the message flow between the user and the control system and shows how the PLT comes off. The HTTPS uses a TCP connection with *Transport Layer Security (TLS)*. Thanks to the *HTTP persistent connection* technique, a connection is kept alive for a number of minutes without user interaction so that a new TCP handshake and TLS authentication is not required every time a command is sent to the server. This saves two RTTs delay per command and keeps the distance effect of the latency between Vienna and Buenos Aires at one RTT.

A method to generate a normally distributed random delay with a mean of 239 milliseconds and a standard deviation of 40 milliseconds was written in the class *TCPMachineCommunicationService* of the *AS* to simulate this distance effect.

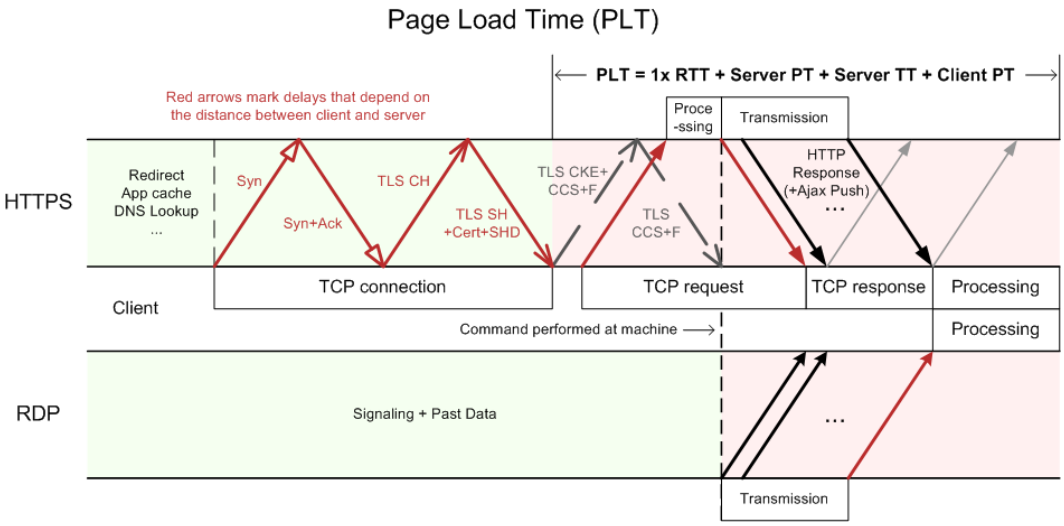


Figure 6.4: HTTPS and RDP message flow and resulting Page Load Time.

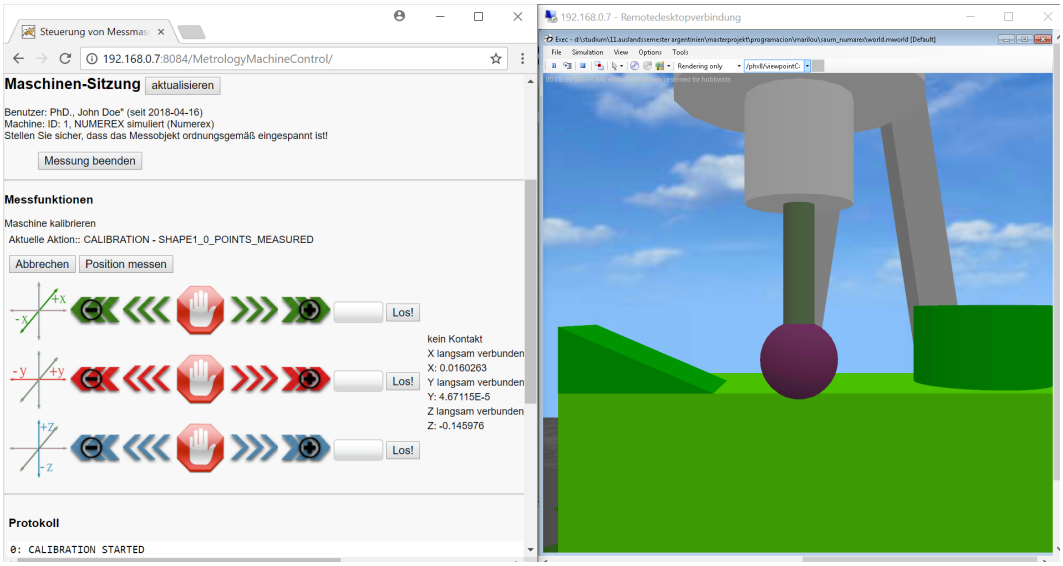


Figure 6.5: Control Interface (left) and Live View (right) for the experiments.

6.2.2 The experiment setup

The experiment server used the *Windows 7 Professional* operating system with an *Apache Tomcat 8* web server. As the *Marilou* CMM simulation does not support live streaming of the simulation window, the *Windows Remote Desktop Connection* tool was used for the Live View. The tool uses the AVC/H.264 codec for the graphical representation. The bitrates achieved and delay added to the Live View of the simulation should therefore be similar to those that would have been created by a webcam video stream of a physical

CMM. Due to its optimization for a Windows platform, *Remote Desktop Connection* is more efficient in this setting than implementations of the flexible, pixel-based *VNC* protocol. Both the GUI and the Live View were accessed via a VPN connection for security at the distance tests via the internet. Figure 6.5 shows the GUI used for the experiments.

As a reaction to the unreliable connection, the control mode was designed such that rather than sending a constant signal just the start/stop signals for the motors were sent. Because of that, the control input need not be buffered to prevent jitter and can be forwarded immediately. Only the Live View should be smoothened. Buffering and other activities to make the user experience as pleasant as possible are already implemented into the Javascript library of the webcam stream or the remote access tool. The logic for detecting and reacting to a probe head contact was put into the PLC and the PLC can intervene and stop motors when necessary.

6.2.3 The experiment workpiece

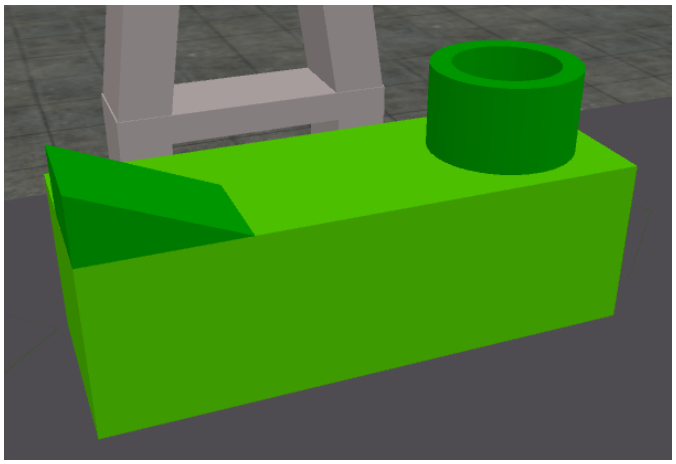


Figure 6.6: Sample workpiece for the probe experiments.

The experiments are performed on a workpiece model in *Marilou* which was mounted to the machine model via fixed joint. Its main body is a cuboid of size 250x90x80 millimeters. The cuboid can be used to measure its basic dimensions and parallelism of its opposing sides. On top of the cuboid there are two objects mounted:

- A ramp with a rectangular base area of 70x50 millimeters. Along the x-axis its height declines in a 23.199° slope from 30 millimeters to zero. Its main purpose is to test the *orientation in space* function.
- A pipe or hollow cylinder of 25 millimeters radius, 10 millimeters thickness and a height of 40 millimeters. This results in an *inner diameter* of 50 millimeters and an *outer diameter* of 70 millimeters, which to determine is subject to the experiments.

6.3 Experiment Execution and Results

In order to evaluate the system architecture and control mechanisms, the prototype should allow for performing an interactive point probe like the one described in Subsection 2.3.2 and depicted in Figure 2.11 on the workpiece specified in Subsection 6.2.3. The procedure is divided into three phases. In the first phase, the workpiece coordinate system is set up by using as the origin the position of the top corner of the body of the workpiece that faces the minimum positions of the x-axis and y-axis. As a second phase, the pipe is probed by measuring three points along its circumference and the position of its center-point and its outer diameter is calculated. Finally, in the third phase the length of the main body of the workpiece is determined.

This procedure should be performed five times from the local network and five times remotely via the internet. The data collected while conducting those operations should be analyzed such that the research questions defined in Section 1.2 can be answered.

6.3.1 Results on measurement accuracy

The research question

Q1: *How should a software-system for operating CMMs be designed to allow for the same accuracy at both remote local and internet control?*

has been answered in Chapter 4 by presenting a software architecture and control mechanism to achieve accurate remote control. However, it must be evaluated whether or not the measurement performance achieved by the control system is actually equivalent for the local and the remote control.

In the course of the experiment, for each experiment type 25 independent axis values of which the true values are known, have been probed directly. After the experiments, the deviation of the values from the true values was calculated. The resulting data set of two groups of 25 measures each was used for analysis. The means were 7.898e-06 (local) and 4.934e-06 (remote), standard deviations were 6.77574e-05 (local) and 4.55648e-05 (remote).

The equivalence of the local and remote measures was tested with a *TOST (two one-sided tests) equivalence test*, based on *Welch's test* [Wel47]. This test requires the sample sets to be normally distributed. For testing the normality, the *Shapiro-Wilk test*, published in [SW65] was used. It is more powerful than other normality tests, especially for small data set sizes below fifty values. [RW$^+$11]

The null hypothesis of the *Shapiro-Wilk test* is that the sample was drawn from a normally-distributed population. The alternative hypothesis is that the distribution of the underlying population is not normally-distributed. A significance ($p < 0.05$) would mean that the null hypothesis is rejected. The test returns a "W" statistic, which is a fraction of two estimates of the population variance. If the data are actually normal, W

equals one. This statistic is used to calculate the p-value. The p-value is the probability of finding a W statistic at least as small as the one observed if the null hypothesis is true.

The tests were performed in the language *R* in version 3.3, using the function *shapiro.test*. The results were as follows:

- local values: n=25, W = 0.9784, p-value = 0.8506
- remote values: n=25, W = 0.9663, p-value = 0.5546
- all values: n=50, W = 0.979, p-value = 0.5119

In all three tests, the null hypothesis could not be rejected. The data set can therefore be used in a *TOST equivalence test* based on *Welch's test*. A graphical representation of the distribution of the deviations from the true value is provided in 6.7.

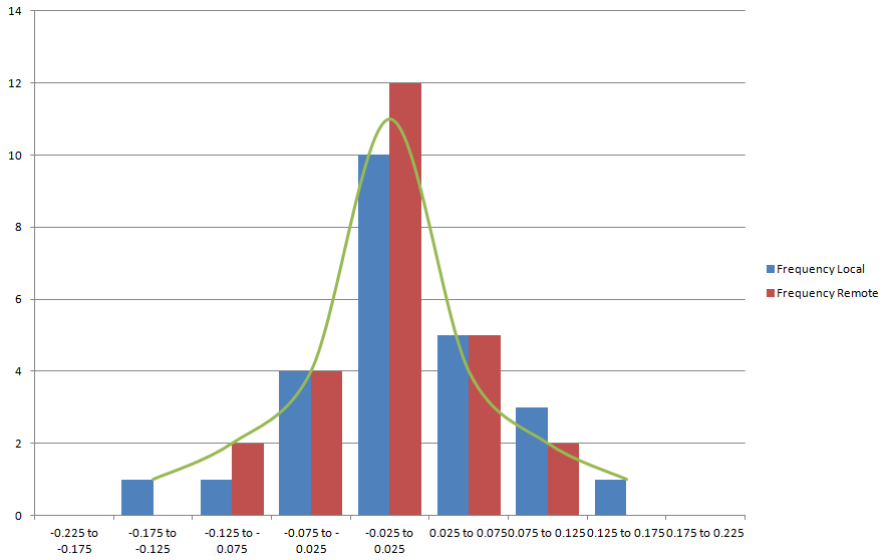


Figure 6.7: Histogram of measurement deviations from true value.

The *Welch's test* is used to test the null hypothesis that two sample populations have equal means. It is a generalization of the *Student's t-test* that is more reliable when it cannot be assumed that the two samples have equal variances. However, the null hypothesis of equal means can only be rejected, not confirmed.

In contrast to classic hypothesis tests, equivalence tests test for the null hypothesis of an effect large enough to be deemed relevant. Therefore, *"equivalence bounds"* must be defined prior to an equivalence test. The test then calculates a confidence interval for the difference statistics which must lie entirely within the *equivalence bounds* in order for the two distributions to be considered *equivalent*.

The local and remote measurements are considered equivalent if the difference in the means of their underlying distributions is less than 0.1 millimeters. The null hypothesis for the *TOST equivalence test* therefore is that the difference in population means is greater than 0.1 millimeters. The *TOST equivalence test based on Welch's test* performs two one-sided Welch's tests with the null hypotheses that the difference in means is $\leq$ -0.1 and $\geq$ +0.1 millimeters. A null hypothesis is rejected if the test is significant ($p < 0.05$).

The tests were performed in *R* in version 3.3, using the function *TOSTtwo* of the package *TOSTER* written by Daniel Lakens[1]. The results were as follows:

- Equivalence region: -0.1mm to +0.1mm
- H0: $\Delta \leq$ -0.1mm : t = 6.304972, p-value = 7.214372e-08
- H0: $\Delta \geq$ +0.1mm : t = -5.941972, p-value = 2.402691e-07
- Using $\alpha = 0.05$ the equivalence test based on Welch's t-test was significant, t(42.02106) = -5.941972, p = 2.402691e-07
- TOST confidence interval (90%): -0.02450299mm to +0.03043099mm

The test statistics of both one-sided tests and the equivalence test were significant, the null hypothesis that there is a relevant difference in means was rejected. The 90% confidence interval lies entirely within the equivalence bounds. The accuracy of the local measurement and the remote measurement can be considered equivalent.

6.3.2 Results on experiment duration

In addition to knowing the accuracy that can be achieved by measuring remotely via the internet, it is also relevant for operators and institutions to know whether distance influences the time it takes to conduct a probe. The experiments performed on the sample work piece were designed to answer the research question:

Q2: *How does CMM control via the internet perform in terms of duration of operation compared to remote control via local network?*

In the course of the experiments, five runs were performed per experiment type. Out of the ten measured experiment durations, two had to be excluded from analysis. The first local run was excluded as it was the first overall run and used to explore the most efficient way of probing the workpiece. The second remote run was excluded because a collision happened that broke off the probe head and the simulation had to be reset before continuing. Both of the events led to an untypically high process duration. The

[1] http://daniellakens.blogspot.co.at/2016/12/tost-equivalence-testing-r-package.html

remaining four durations of each experiment type were analyzed using *Fisher's exact test* [Fis37]. A graphical representation of the duration of the experiment runs is provided in 6.8.

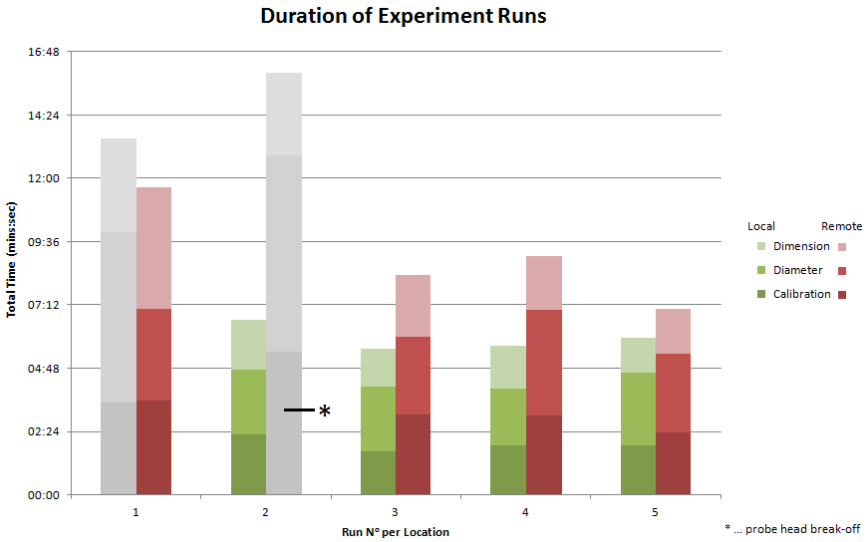


Figure 6.8: Duration of the experiment runs.

The *Fisher exact test* is a widespread *permutation test* for statistical significance testing and is used especially when sample sizes are small. It makes no assumptions of distribution and hence does not require knowledge about the underlying distribution types. The test removes the data group labels from the sample values and produces all possible permutations of assignments of values to the groups. It is therefore also known as *"randomization test"*. The null hypothesis for a two-sample permutation test is that the values of the two samples are equally distributed and hence it does not matter which data group a value is assigned to. The difference in means between the newly assigned groups is calculated for each permutation and compared to the difference in means of the original group assignment. The p-value is exact for all sample sizes and represents the share of permutations that result in a higher absolute mean difference than the original data group assignment. The null hypothesis is rejected if the test is significant ($p < 0.05$).

The tests were conducted in *R* version 3.3, using the function *oneway_test* for non-paired values of the package *coin*. Tests for all three possible alternative hypotheses were performed and resulted in $Z = 2.0655$ and the following p-values:

- H1a: μremote - μlocal > 0 : p-value $= 0.01429$
- H1b: μremote - μlocal $\neq 0$: p-value $= 0.02857$

- H1c: μremote - μlocal < 0 : p-value $= 1$

The equality of sample means had to be rejected under alternative hypotheses H1a and H1b but could not be rejected under alternative hypothesis H1c. It can therefore be assumed that the mean duration of the remote operation is (H1a) greater than and (H1b) not equal to the mean duration of the local operation. It cannot be assumed that it is (H1c) less than the mean duration of the local operation.

CHAPTER 7

Conclusion

7.1 Main Findings

This thesis described the concept of remote CMM probing for metrology research, stated its importance for engineering education and presented a reference software architecture. The operative mechanisms of tactile workpiece probing via CMMs were analyzed and a generic measuring procedure was illustrated. This procedure can be used for tactile workpiece probing on various generations of CMMs as well as physical and software-simulated machines. Control systems that serve so many different machines must be able to communicate with machine controllers of both the major manufacturers of specialized industrial controllers as well as manufacturers of regular PLCs.

The proposed software architecture is based on the concept of a centralized control system that is able to serve multiple machines that are active at the same time and perform complex operations and calculations. The server thereby distinguishes between the commands that require machine interaction and those that do not. Machine controllers only execute actuator movement commands, respond to the server by sending periodic state information and perform event-triggered safety measures. Their control cycle and communication with the server has been defined in detail in this thesis. In a well-designed system, the machine controller is the only part that needs to be coded when a new machine is introduced. At the control server, the machine can be created by simply configuring master data information and the actuators and sensors mounted.

The control system server is split into two components. The *Acquisition Server (AS)* is responsible for continuous communication with the machine controller, whereas the *Laboratory Server (LS)* provides a gateway for request-based user input. The amount and size of communication messages is reduced towards the user end of a communication. Duplicate information is filtered, the update frequency lowered and only significant changes forwarded to the distant operator. This traffic reduction should improve the latency for the user.

Regarding the choice of user interface technology for remote systems, recent developments strongly favor web services and *browser-based solutions*. Advances in the standardization of web technologies simplify their development, as well as the embedding of video streams into user interfaces or of whole control components into CMSs and specialized LMSs. The necessity of providing real-time video arises from the remote control as the CMM that is worked on is out of sight. At simulated laboratory experiments, the graphical simulation window is made accessible instead. As the experiments are only observed via cameras, the orientation in space is more difficult and there is a higher risk of handling errors.

If the internet is used as the communication channel, the control over the *page load time (PLT)* - from issuing a machine command to seeing its effects on the screen - is small. The *HTTP persistent connection* functionality of modern browsers is helpful in reducing the *PLT* to one RTT plus local processing and transmission delay.

As a reaction to the unreliable connection, the control mode was designed such that rather than sending a constant signal just the start/stop signals for the motors were sent. Some of the responsibility is taken away from the user and handed over to the PLC. The controller detects contact of the probe head or axes approaching specified limits and is able to automatically stop or slow down movements. As precise positioning is difficult considering the internet delay, the PLC does not only protect from damage. At contact, it automatically adjusts the machine to only very slightly touch the measuring object. This *first touch action* avoids lengthy fine tuning and improves measuring accuracy.

Controllers are designed for short cycle times to react before damage occurs. However, there exists a trade-off between duration of operation and equipment safety. Available axis speeds as well as the distance of an axis from a target position that triggers the *check boundary reached* routine to slow down the movement are configurable to match the experience and skill level of the operator. The probability that the PLC can prevent that the probe head breaks off varies, depending on the movement speed at contact. High speeds should therefore not be used when closely approaching the measuring object.

Embracing all those design provisions, it could be shown that the presented software architecture produces equivalent measuring accuracy for long-distance control via the internet and control via local network. The duration of operation via the internet was found to be longer than via local network.

7.2 Future Research

The reference software architecture introduced by this thesis is meant to encourage the creation of further implementations of location-independent CMM control systems. However, *Manejo*, the implementation presented, is already suitable for conducting both local and remote control studies. Some opportunities for research are mentioned below.

A *Live View* component for visual observation of the procedures on the CMM is necessary for remote experiments. Thereby, the field of vision and freedom to move around in space

is limited in comparison to being physically present in the laboratory. In the course of this thesis, it was shown that the internet operation takes longer than the operation from within the local network. The durations of the experiments in both settings appeared to be long for the number of points measured. Therefore, it would be interesting to compare the duration of operation on a physical machine while watching the machine directly to when a video stream is used for visual control. The assumption that the indirect visual control prolongs the probe duration could be affirmed or rejected with additional research.

The current control mechanism allows for the usage of axis speeds at which the PLC cannot reliably prevent the breaking of a probe head. This rare event occurred once during the internet control experiments. Performing a much higher number of experiment runs, it could be ascertained with statistical methods whether the internet control is more prone to damaging the probe head than the remote local control.

Surveys have shown that the introduction of simulated and remote laboratories can improve the learning success for students. If the *Manejo* implementation is used in engineering education over a longer period, the scores of students who use the software can be compared to a control group and conclusions be drawn about its learning effects.

List of Figures

List of Tables

Listings

Acronyms

API Application Programming Interface. 63, 71, 82

AS Acquisition Server. 63–70, 82–86, 88, 100, 109

CMM Coordinate Measuring Machine. 2

CMS Content Management System. 64, 65, 110

CNC Computer Numerical Control. 1, 18, 20, 51, 79

DAQ Data Acquisition. 28

DAS Data Acquisition System. 61, 65

EJS Easy Java Simulations. 31, 32

FCC U.S. Federal Communications Commission. 97, 98

GD&T Geometrical Dimensioning and Tolerancing. 9, 13

GOLC Global Online Laboratory Consortium. 37

GPS Geometrical Product Specification. 8, 9

GUI Graphical User Interface. 31, 65, 70–72, 80, 87, 102

HTML Hypertext Markup Language. 37, 71, 88

HTTP Hypertext Transfer Protocol. 71, 87, 88, 100, 110

ILAC International Laboratory Accreditation Cooperation. 10

ISA iLab Shared Architecture. 33

ISO International Organization for Standardization. 9, 11, 59, 71

ISP Internet Service Provider. 97, 98

JSF Java Server Faces. 88

LMS Learning Management System. 28, 29, 31, 33, 36, 37, 54, 57, 64, 65, 72, 110

LS Laboratory Server. 29, 63, 64, 70, 80, 85–88, 109

NAT Network Address Translation. 95, 96

NFRs Non-functional Requirements. 59

PLC Programmable Logic Controller. 36, 62–69, 102, 109–111

RPC Remote Procedure Call. 64, 85

RTT Round-Trip Time. 97, 99, 100, 110

SOAP Simple Object Access Protocol. 34, 36

UML Unified Modeling Language. 56

VI Virtual Instrument. 28, 29, 32, 61

VISIR Virtual Instrument Systems In Reality. 26, 27

VPN Virtual Private Network. 95, 96, 102

WSDL Web Service Description Language. 34

XML Extensible Markup Language. 34, 35

Bibliography

[ABD11] Saamer Akhshabi, Ali C Begen, and Constantine Dovrolis. An experimental evaluation of rate-adaptation algorithms in adaptive streaming over http. In *Proceedings of the second annual ACM conference on Multimedia systems*, pages 157–168. ACM, 2011.

[ACL05] Kiam Heong Ang, Gregory Chong, and Yun Li. PID Control System Analysis, Design, and Technology. *IEEE transactions on control systems technology*, 13(4):559–576, 2005.

[AHS14] Firoz Alam, Roger G Hadgraft, and Aleksandar Subic. Technology-enhanced laboratory experiments in learning and teaching. *F. Alam (Ed.), Using technology tools to innovate assessment, reporting, and teaching practices in engineering education*, pages 289–302, 2014.

[AMT12] Petros J. Axaopoulos, Konstantinos N. Moutsopoulos, and Michael P. Theodoridis. Engineering education using a remote laboratory through the Internet. *European Journal of Engineering Education*, 37(1):39–48, 2012.

[APA16] Tareq Alkhaldi, Ilung Pranata, and Rukshan I. Athauda. A review of contemporary virtual and remote laboratory implementations: observations and findings. *Journal of Computers in Education*, 3(3):329–351, 2016.

[Årz99] Karl-Erik Årzén. A Simple Event-Based PID Controller. In *Proc. 14th IFAC World Congress*, volume 18, pages 423–428, 1999.

[ASS+11] Krishnashree Achuthan, K. S. Sreelatha, Shone Surendran, Shyam Diwakar, Prema Nedungadi, Steven Humphreys, C. O. S Sreekala, Zeena Pillai, Raghu Raman, Ani Deepthi, Rathish Gangadharan, Saritha Appukuttan, Jyothi Ranganatha, Sreedha Sambhudevan, and Suma Mahesh. The VALUE @ Amrita Virtual Labs Project: Using web technology to provide virtual laboratory access to students. *Proceedings - 2011 IEEE Global Humanitarian Technology Conference, GHTC 2011*, pages 117–121, 2011.

[BCQ+01] A Bicchi, A Coppelli, F Quarto, L Rizzo, F Turchi, and A Balestrino. Breaking the Lab's Walls - Tele-Laboratories at the University of Pisa. *Proceedings of IEEE International Conference on Robotics & Automation. Seoul, Korea.*, pages 1903–1908, 2001.

[BIP18] BIPM. International Vocabulary of Metrology. https://jcgm.bipm.org/vim/en/2.2.html [Online; accessed 2018-04-20], 2018.

[BKTA17] Ahmed Badr, Ashish Khisti, Wai-Tian Tan, and John Apolstolopoulos. Perfecting protection for interactive multimedia: A survey of forward errror correction for low-delay interactive applications. *IEEE Signal Processing Magazine*, 34(2):95–113, 2017.

[BLL+12] Evgeny Bogdanov, Freddy Limpens, Na Li, Sandy El Helou, Christophe Salzmann, and Denis Gillet. A social media platform in higher education. *IEEE Global Engineering Education Conference, EDUCON*, 2012.

[CCLOD13] Dictino Chaos, Jesús Chacón, Jose Antonio Lopez-Orozco, and Sebastián Dormido. Virtual and remote robotic laboratory using EJS, MATLAB and LabVIEW. *Sensors (Switzerland)*, 13(2):2595–2612, 2013.

[Chu00] Brian A Chung, Lawrence and Mylopoulos, John and Yu, Eric and Nixon. *Non-Functional Requirements in Software Engineering.* 2000.

[CMZ06] Drago Cmuk, Tarik Mutapcic, and Francesco Zoino. Remote versus classical laboratory in electronic measurements teaching - effectiveness testing. *XVIII IMEKO world congress proceedings*, 2006.

[CSC02] Chetz Colwell, Eileen Scanlon, and Martyn Cooper. Using remote laboratories to extend access to science and engineering. *Computers & Education*, 38(1-3):65–76, 2002.

[CSS11] Horst Czichos, Tetsuya Saito, and Leslie E. Smith. *Springer Handbook of Metrology and Testing.* 2011.

[DBRB15] Numan M. Durakbasa, Gokcen Bas, David Riepl, and Jorge M. Bauer. An innovative educational concept of teleworking in the high precision metrology laboratory to develop a model of implementation in the advanced manufacturing industry. *Proceedings of 2015 12th International Conference on Remote Engineering and Virtual Instrumentation, REV 2015*, pages 242–248, 2015.

[DF09] Yimin Ding and Hao Fang. Using a simulation laboratory to improve physics learning: A case exploratory learning of diffraction grating. *Proceedings of the 1st International Workshop on Education Technology and Computer Science, ETCS 2009*, 3:3–6, 2009.

[DIN94] Wörterbuch der Metrologie, International Vocabulary of Basic and General Terms in Metrology. Standard, 1994.

[DIN95] Leitfaden zur angabe der unsicherheit beim messen. german translation of "guide to the expression in uncertainty in measurement". Standard, 1995.

[DIN13] DIN. DIN/IEC 61131-3:2013. "Programmable controllers - Part 3: Programming languages". Technical report, 2013.

[dJSG14] Ton de Jong, Sofoklis Sotiriou, and Denis Gillet. Innovations in STEM education: the Go-Lab federation of online labs. *Smart Learning Environments*, 1(1):3, dec 2014.

[DOAS01] Numan M. Durakbasa, Herbert P. Osanna, and A. Afjehi-Sadat. A general approach to workpiece characterization in the frame of gps (geometrical product specification and verification). *International Journal of Machine Tools & Manufacture*, 41:2147–2151, 2001.

[DOBB12] Numan M. Durakbasa, Herbert P. Osanna, Jorge M. Bauer, and Gökcen Bas. Innovation in production metrology for precision engineering and to support sustainablity and improvement of process and product quality in modern manufacturing industry. *MSD journal*, 2:5–11, 2012.

[Dur03] Numan M. Durakbasa. *Geometrical Product Specifications and Verification for the Analytical Description of Technical and Non-Technical Structures, printed in Austria (TU Wien: Abteilung Austauschbau und Messtechnik)*. 2003.

[DWW+12] Razvan I. Dinita, George Wilson, Adrian Winckles, Marcian Cirstea, and Aled Jones. A cloud-based virtual computing laboratory for teaching computer networks. *Proceedings of the International Conference on Optimisation of Electrical and Electronic Equipment, OPTIM*, pages 1314–1318, 2012.

[dZ13] de Jong, Ton and Linn, Marcia C. and Zacharias C. Zacharia. Physical and Virtual Laboratories in Science and Engineering Education. *Science AAAS*, 340(April):305–308, 2013.

[ET09] Yasser . H. Elawady and a. S. Tolba. Educational Objectives Of Different Laboratory Types: A Comparative Study. *International Journal of Computer Science and Information Security*, 6(2):89–96, 2009.

[FAK+05] N. D. Finkelstein, W. K. Adams, C. J. Keller, P. B. Kohl, K. K. Perkins, N. S. Podolefsky, S. Reid, and R. Lemaster. When learning about the real world is better done virtually: A study of substituting computer simulations for laboratory equipment. *Physical Review Special Topics - Physics Education Research*, 1(1):1–8, 2005.

[FCC16] FCC's Office of Engineering and Technology and Consumer and Governmental Affairs Bureau. Measuring Broadband America Fixed Broadband Report. pages 1–78, 2016.

[Fis37] Ronald Aylmer Fisher. *The design of experiments*. Oliver And Boyd; Edinburgh; London, 1937.

[FK13] V. Fečová and M. Kočiško. The principle of measuring by coordinate measuring machines and new possibilities of work with these machines. *SAMI 2013, IEEE 11th International Symposium on Applied Machine Intelligence and Informatics*, 2013.

[Fow00] Kendall Fowler, Martin and Scott. *Distilled UML*. Addison Wesley Longman. Inc., Reading, Mass, 2000.

[FPW] Gene F Franklin, J David Powell, and Michael L Workman. Digital Control of Dynamic Systems. pages 1–12.

[GCV+13] Sten Govaerts, Yiwei Cao, Andrii Vozniuk, Adrian Holzer, Danilo Garbi Zutin, Elio San Cristóbal Ruiz, Lars Bollen, Sven Manske, Nils Faltin, Christophe Salzmann, Eleftheria Tsourlidaki, and Denis Gillet. Towards an online lab portal for inquiry-based STEM learning at school. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8167 LNCS:244–253, 2013.

[GNZ+09] Ingvar Gustavsson, Kristian Nilsson, Johan Zackrisson, Javier Garcia-Zubia, Unai Hernandez-Jayo, Andrew Nafalski, Zorica Nedic, Özdemir Göl, Jan MacHotka, Mats I. Pettersson, Thomas Lagö, and Lars Håkansson. On objectives of instructional laboratories, individual assessment, and use of collaborative remote laboratories. *IEEE Transactions on Learning Technologies*, 2(4):263–274, 2009.

[Gri13] Ilya Grigorik. *High Performance Browser Networking: What every web developer should know about networking and web performance*. O'Reilly Media, Inc., 2013.

[HBRK+15] Zahid Hossain, Paulo Blikstein, Ingmar H. Riedel-Kruse, Xiaofan Jin, Engin W. Bumbacher, Alice M. Chung, Stephen Koo, Jordan D. Shapiro, Cynthia Y. Truong, Sean Choi, and Nathan D. Orloff. Interactive Cloud Experimentation for Biology. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*, pages 3681–3690, 2015.

[HDL+08] V. Judson Harward, Jesus A. Del Alamo, Steven R. Lerman, Philip H. Bailey, Joel Carpenter, Kimberley DeLong, Chris Felknor, James Hardison, Bryant Harrison, Imad Jabbour, Phillip D. Long, Tingting Mao, Loai Naamani, Jedidiah Northridge, Mark Schulz, Daniel Talavera, C. Daniel Varadharajan, Shaomin Wang, Karim Yehia, Rabih Zbib, and David Zych. The iLab shared architecture: A web services infrastructure to build communities of internet accessible laboratories. *Proceedings of the IEEE*, 96(6):931–950, 2008.

[Hem00] Donald K Hemmelgarn, Thomas L and Bell, Frederick K and Raleigh, Freddie L and Greier. Patent "US 6058618 A". United States Patent and Trademark Office (USPTO). Publishing Date: May 9th 2000., 2000.

[HKG14] Wu Yuin Hwang, Chaknarin Kongcharoen, and Gheorghita Ghinea. To enhance collaborative learning and practice network knowledge with a virtualization laboratory and online synchronous discussion. *International Review of Research in Open and Distance Learning*, 15(4):113–137, 2014.

[ISO11] ISO. ISO/IEC 25010:2011. "Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models". Technical report, 2011.

[ISO13a] Geometrical product specification (gps) – geometrical tolerancing - tolerancing of form, orientation, location and run-out. Standard, International Organization of Standardization, 2013.

[ISO13b] Geometrical Product Specifications (GPS) – Inspection by measurement of workpieces and measuring equipment - Part 1: Decision rules for proving conformity or nonconformity with specifications. Standard, 2013.

[ISO14] ISO. ISO/IEC 23009-1:2014. "Dynamic adaptive streaming over HTTP (DASH) - Part 1: Media presentation description and segment formats". Technical report, 2014.

[JCG12] International vocabulary of metrology- Basic and general concepts and associated terms (VIM). Standard, 2012.

[JD02] Manish Jain and Constantinos Dovrolis. *End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput*, volume 32. ACM, 2002.

[JT01] Ramesh Johari and David Kim Hong Tan. End-to-end congestion control for the internet: Delays and stability. *IEEE/ACM Transactions on Networking (TON)*, 9(6):818–832, 2001.

[KG04] M. Kalman and B. Girod. Modeling the delays of successively-transmitted Internet packets. *2004 IEEE International Conference on Multimedia and Expo (ICME) (IEEE Cat. No.04TH8763)*, pages 2015–2018, 2004.

[Kiz17] Joseph Migga Kizza. *Guide to computer network security*. Springer, 2017.

[KL16] Alexandra Kraemer and Gisela Lanza. Assessment of the measurement procedure for dimensional metrology with x-ray computed tomography, 2016.

[KM06] Ehsan Kamrani and M Hosein Mehraban. Modeling internet delay dynamics using system identification. In *Industrial Technology, 2006. ICIT 2006. IEEE International Conference on*, pages 716–721. IEEE, 2006.

[KPS+05] H. Kunzmann, T. Pfeifer, R. Schmitt, H. Schwenke, and A. Weckenmann. Productive metrology - adding value to manufacture. 2005.

[LML08] David Lowe, Steve Murray, and Euan Lindsay. Reflecting professional reality in remote laboratory experiences. *REV 2008: Remote Engineering and Virtual Instrumentation*, pages 1–5, 2008.

[LMLL09] David Lowe, Steve Murray, Euan Lindsay, and Dikai Liu. Evolving remote laboratory architectures to leverage emerging internet technologies. *IEEE Transactions on Learning Technologies*, 2(4):289–294, 2009.

[LPJ16] Joan Meseguer Llopis, Janusz Pieczerak, and Tomasz Janaszka. Minimizing latency of critical traffic through sdn. In *Networking, Architecture and Storage (NAS), 2016 IEEE International Conference on*, pages 1–6. IEEE, 2016.

[LS11] Fabrizio Lamberti and Andrea Sanna. Migrating desktop applications to the internet: a novel virtualization paradigm based on web operating systems. *J. Web Engineering*, 10:234–272, September 2011.

[LY05] Fu-Chun Liu and Yu Yao. Modeling and analysis of networked control systems using hidden markov models. In *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*, volume 2, pages 928–931, 2005.

[Mar10] Robert Marmelstein. "control systems for robots". cpsc 527 – robotics. East Stroudsburg University, 2010.

[Mit16] Sunil Bharti Mittal. The trends of globalization and digitalization are changing the market contexts. *China's Foreign Trade*, 5:030, 2016.

[MIT18] MIT. Mit ilab shared architecture documentation. https://wikis.mit.edu/confluence/display/ILAB2/Developers [Online; accessed 2018-04-22], 2018.

[MLL+08] Steve Murray, David Lowe, Euan Lindsay, Vladimir Lasky, and Dikai Liu. Experiences with a hybrid architecture for remote laboratories. *Proceedings - Frontiers in Education Conference, FIE*, pages 1–5, 2008.

[MM07] Manuel E. Macías and Israel Méndez. eLab - Remote electronics lab in real time. *Proceedings - Frontiers in Education Conference, FIE*, pages 12–17, 2007.

[MN06] Jing Ma and Jeffrey V. Nickerson. Hands-on, simulated, and remote laboratories. *ACM Computing Surveys*, 38(3):7–es, 2006.

[MVCL+14] Maria A. Marques, Maria Clara Viegas, Maria Cristina Costa-Lobo, André V. Fidalgo, Gustavo R. Alves, João S. Rocha, and Ingvar Gustavsson. How remote labs impact on course outcomes: Various practices using VISIR. *IEEE Transactions on Education*, 57(3):151–159, 2014.

[Nay13] Ameya Nayak. Type of NOSQL Databases and its Comparison with Relational Databases. 5(4):16–19, 2013.

[NBW98] Johan Nilsson, Bo Bernhardsson, and Björn Wittenmark. Stochastic analysis and control of real-time systems with random time delays. *Automatica*, 34(1):57–64, 1998.

[NJ12] Kathleen Nichols and Van Jacobson. Controlling queue delay. *Communications of the ACM*, 55(7):42–50, 2012.

[ÖKA10] Mehmet Efe Özbek, Ali Kara, and Musa Ataş. Software technologies, architectures and interoperability in remote laboratories. *2010 9th International Conference on Information Technology Based Higher Education and Training, ITHET 2010*, pages 402–406, 2010.

[OMG10] OMG. Unified Modeling Language (OMG UML), Superstructure, Version 2.3. Technical report, Needham, MA, May 2010.

[ORDH95] Herbert P. Osanna, K. Rezaie, Numan M. Durakbasa, and C. R. Heiss. Form measurements - a bridge between production metrology and biomechanics. *International Journal of Machine Tools & Manufacture*, 35:165–168, 1995.

[PBHJGR09] Josep Prieto-Blazquez, Jordi Herrera-Joancomarti, and Ana-Elena Guerrero-Roldán. A Virtual Laboratory Structure for Developing Programming Labs. *International Journal of Emerging Technologies in Learning (iJET)*, 4(0):47–52, 2009.

[RC05] Tyler Richards and Mo Yuen Chow. Performance characterization of IP network-based control methodologies for DC motor applications - Part I. *IECON Proceedings (Industrial Electronics Conference)*, 2005:2405–2410, 2005.

[RRPI03] P Salvo Rossi, Gianmarco Romano, Francesco Palmieri, and Giulio Iannello. A hidden markov model for internet channels. In *Signal Processing and Information Technology, 2003. ISSPIT 2003. Proceedings of the 3rd IEEE International Symposium on*, pages 50–53. IEEE, 2003.

[RTB11] Thomas Richter, Yvonne Tetour, and David Boehringer. Library of Labs - A European project on the dissemination of remote experiments and virtual laboratories. *Proceedings - 2011 IEEE InternationalSymposium on Multimedia, ISM 2011*, pages 543–548, 2011.

[RW+11] Nornadiah Mohd Razali, Yap Bee Wah, et al. Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. *Journal of statistical modeling and analytics*, 2(1):21–33, 2011.

[RWW04] Ramaswamy Ramaswamy, Ning Weng, and Tilman Wolf. Characterizing network processing delay. In *Global Telecommunications Conference, 2004. GLOBECOM'04. IEEE*, volume 3, pages 1629–1634. IEEE, 2004.

[SAO05] Yasuhiro Sato, Shingo Ata, and Ikuo Oka. Using Mixed Distribution for Modeling End-to-End Delay Characteristics. pages 422–433, 2005.

[Sch13] Walter S A Schwaiger. It-based management "designing cybernetic mgt-systems". lecture script ws2013/14. Vienna University of Technology, 2013.

[SCMS11] Miladin Stefanovic, Vladimir Cvijetkovic, Milan Matijevic, and Visnja Simic. A LabVIEW-based remote laboratory experiments for control engineering education. *Computer Applications in Engineering Education*, 19(3):539–549, 2011.

[SL07] Vishal Sadana and Xiaoqing Frank Liu. Analysis of Conflicts among Non-Functional Requirements Using Integrated Analysis of Functional and Non-Functional Requirements. (Compsac):0–3, 2007.

[SNRPK02] Heinrich Schwenke, Ulrich Neuschaefer-Rube, Tilo Pfeifer, and Horst Kunzmann. Optical methods for dimensional metrology in production engineering. *CIRP Annals - Manufacturing Technology*, 51:685–699, 2002.

[Sto11] Thomas Stockhammer. Dynamic adaptive streaming over http–: standards and design principles. In *Proceedings of the second annual ACM conference on Multimedia systems*, pages 133–144. ACM, 2011.

[SW65] S. S. Shapiro and M. B. Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611, 1965.

[Tü15] Sarp Türkmen. Design and implementation of a remotely controlled high precision measurement laboratory using telepresence and teleoperation in coordinate metrology applications. Vienna University of Technology, 2015.

[Tag05] Nancy R Tague. The quality toolbox. *Milwaukee: ASQ Quality Press. Published: 2005 (1995)*, page 390–392, 2005.

[TdS05] P L Tang and Clarence W de Silva. Stability and Optimality of Constrained Model Predictive Control with Future Input Buffering in Networked Control Systems. pages 1245–1250, June 2005.

[WAA+05] Brian F Woodfield, Merritt B Andrus, Tricia Andersen, Jordan Miller, Bryon Simmons, Richard Stanger, Gregory L Waddoups, Melissa S Moore, Richard Swan, Rob Allen, and Greg Bodily. Teaching with Technology The Virtual ChemLab Project : A Realistic and Sophisticated Simulation of Organic Synthesis W. *Journal of Chemical Education*, 82(11):1728–1735, 2005.

[Wel47] Bernard L Welch. The generalization of 'student's' problem when several different population variances are involved. *Biometrika*, 34(1/2):28–35, 1947.

[WS00] A Weckenmann and W. Scharf. Conception of a high precision measurement room. volume 8, pages 315–319, XVI IMEKO World Congress, Vienna, 25.-28.09.2000, Proceedings, 2000.

[ZAMN10] Danilo Garbi Zutin, Michael E. Auer, Christian Maier, and Michael Niederstätter. Lab2go - A repository to locate educational online laboratories. *2010 IEEE Education Engineering Conference, EDUCON 2010*, pages 1741–1746, 2010.

[ZLL+17] Junzhi Zhang, Yutong Li, Chen Lv, Jinfang Gou, and Ye Yuan. Time-varying delays compensation algorithm for powertrain active damping of an electrified vehicle equipped with an axle motor during regenerative braking. *Mechanical Systems and Signal Processing*, 87:45–63, 2017.

[ZM04] Taieb F. Znati and Rami Melhem. Node delay assignment strategies to support end-to-end delay requirements in heterogeneous networks. *IEEE/ACM Transactions on Networking*, 12(5):879–892, 2004.

[ZOP08] Zacharias C. Zacharia, Georgios Olympiou, and Marios Papaevripidou. Effects of experimenting with physical and virtual manipulatives on students' conceptual understanding in heat and temperature. *Journal of Research in Science Teaching*, 45(9):1021–1035, 2008.

[ZXK+11] Yaoyao Zhao, Xun Xu, Tom Kramer, Fred Proctor, and John Horst. Dimensional metrology interoperability and standardization in manufacturing systems. *Computer Standards & Interfaces*, 33:541–555, 2011.