Towards Predictive Maintenance For Radiation Generation Vacuum Components

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften (Dr.-Ing.)

von der KIT-Fakultät für
Elektrotechnik und Informationstechnik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

DISSERTATION

von

M.Sc. Con Tran Vu
geb. in Ho Chi Minh Stadt, Vietnam

Tag der mündlichen Prüfung:	01.08.2023
Hauptreferent:	Prof. Dr.rer.nat. Wilhelm Stork
Korreferent:	Prof. Dr.rer.nat. Cornelius Neumann

Abstract

Industry 4.0 in the manufacturing industry, and the cloud and subscription model in the software industry both represent the larger trend of utilizing data and connectivity to deliver unprecedented value to both customers and providers. In this larger context, a data-based Predictive Maintenance (PM) Solution, offered by a device manufacturer to their customer, help the customer to avoid unexpected downtime due to device breakdown. It also helps the service department of the manufacturer to organize efficient and cost-effective operations. The goal of this work is to lay out the foundation and take the first important steps toward the creation and integration of such a Predictive Maintenance Solution. The work will be realized in the context of X-Ray Fluorescence (XRF) measurement devices from a representative Small-Medium Enterprise (SME) manufacturer.

The development of a data-based Predictive Maintenance Solution requires data from device failures, for which a global Data Collection Campaign was organized. To enable the Campaign, we proposed and partially realized a System For Data Collection. With this, data can be generated from previously isolated, non-data-generating devices, and be securely brought back to the NoSQL Big Data server inside the SME manufacturer, ready for various data usage scenarios. From the collected failure data, it was shown that the Filament Current is the key signal in predicting the failure of the X-Ray Tube.

The onboard electronics of the devices only allow the data to be sampled at a low 8-bit resolution. This limits the earliest time point, since which a data-based failure prediction could be made. A simulation program was constructed to theoretically demonstrate the improvement in the prediction horizon resulting from the higher sampling resolution. Due to the exponential nature of the signal, 16-bit sampling would improve the prediction horizon by 20-fold compared to 8-bit sampling. A 12-bit sampling could be a good trade-off between cost and prediction performance. An experiment setup based on a real device, enhanced with a 16-bit data logger, was constructed. The data showed a clearly identifiable trend since very early in its lifetime, opening doors for early failure prediction.

For failure prediction, an algorithm architecture was proposed to remedy the limited availability of failure data in the short and middle term. The algorithm consists of the Anomaly Detection and Trend Prediction Block. It allows very efficient incorporation of data at all life stages of devices into the algorithm, leading to continual improvement in performance. The Trend Prediction was realized for the tube failure modes of Vacuum Leakage and Filament burnout. Analysis of data showed an unexpected linear dependence between the Filament current signal value and the logarithm of the remaining lifetime, i.e. I linearly dependent on (l.d.o.) $\log{(t_{fail} - t)}$. Even with low 8-bit resolution, failure prediction could be made up to 300 working hours before the failure. Within two weeks before failure, the exact time point could be predicted with up to ± 4 hours accuracy. This demonstrated the feasibility of Predictive Maintenance on the XRF device.

To incorporate Predictive Maintenance into the main measurement software, a software architecture was proposed based on the value propositions and features validated with internal stakeholders. The architecture allows maximum re-usability of software components, allows easy integration, and enables Over-the-air update of prediction algorithm for continual improvement of prediction performance.

In this dissertation, the architecture for the development and integration of a Predictive Maintenance Solution was presented and realized under real-life constraints. Algorithm realization showed the feasibility of the solution to deliver enough prediction performance to materialize the promised business values. Potential improvement in prediction performance based on additional high-quality data was investigated and demonstrated.

Zusammenfassung

Industrie 4.0 in der Fertigungsindustrie sowie Cloud- und Abonnementmodelle in der Softwareindustrie repräsentieren beide den größeren Trend, Daten und Konnektivität zu nutzen, um einen beispiellosen Mehrwert für Kunden und Anbieter zu liefern. In diesem größeren Zusammenhang hilft eine datenbasierte Lösung für Predictive Maintenance (vorausschauende Wartung), die von einem Gerätehersteller seinen Kunden angeboten wird, dem Kunden, unerwartete Ausfallzeiten aufgrund von Geräteausfällen zu vermeiden. Sie hilft auch der Serviceabteilung des Herstellers, den Betrieb effizient und kostengünstig zu organisieren. Das Ziel dieser Arbeit ist es, die Grundlage zu schaffen und wichtige erste Schritte zur Entwicklung und Integration einer solchen Predictive Maintenance Lösung zu unternehmen. Die Arbeit wird im Kontext von Röntgenfluoreszenz-Messgeräten eines repräsentativen Klein- und Mittelständischen Unternehmens (KMU) realisiert.

Die Entwicklung einer datenbasierten Predictive Maintenance Lösung erfordert Daten zu Geräteausfällen, für die eine globale Datensammlungskampagne organisiert wurde. Um die Kampagne zu ermöglichen, haben wir ein System zur Datensammlung vorgeschlagen und teilweise umgesetzt. Mit diesem System können Daten von zuvor isolierten, nicht datengenerierenden Geräten generiert werden, und in Sicherheit zum NoSQL Big Data Server innerhalb des KMU-Herstellers übertragen werden. Dort werden die Daten für verschiedene Datennutzungsszenarien bereitgestellt. Aus den gesammelten Ausfalldaten wurde beobachtet, dass der Filamentstrom das Schlüsselsignal für die Vorhersage des Ausfalls der Röntgenröhre ist.

Die On-Board-Elektronik der Geräte ermöglicht nur die Abtastung der Daten mit niedriger 8-Bit-Auflösung. Dies begrenzt den frühesten Zeitpunkt, zu dem eine datenbasierte Fehlerprognose möglich wäre. Es wurde ein Simulationsprogramm entwickelt, um theoretisch zu zeigen, dass sich der Vorhersagehorizont durch eine höhere Abtastenauflösung verbessert. Aufgrund der exponentiellen Natur des Signals, würde eine 16-Bit-Abtastung den Vorhersagehorizont im Vergleich zur 8-Bit-Abtastung um das 20-fache verbessern. Eine 12-Bit-Abtastung könnte ein guter Kompromiss zwischen Kosten und Vorhersageleistung sein. Es wurde ein Experimentaufbau auf einem realen Gerät entwickelt, das mit einem 16-Bit-Datenlogger erweitert wurde. Die Daten zeigten auffälligen Trends seit sehr frühen Lebensphasen und eröffneten Möglichkeiten für eine frühzeitige Fehlerprognose.

Für die Fehlerprognose wurde eine Algorithmusarchitektur vorgeschlagen, um die begrenzte Verfügbarkeit von Fehlerdaten in naher bis mittlerer Zukunft zu beheben. Der Algorithmus besteht aus dem Anomaly Detection (Anomalieerkennung) Block und dem Trend Prediction (Trendvorhersage) Block. Dadurch wird eine sehr effiziente Einbindung von Daten aus allen Lebensphasen der Geräte in den Algorithmus ermöglicht, was zu einer kontinuierlichen Verbesserung der Prognosenleistung führt. Die Trend Prediction Block wurde für die zwei Ausfallmodi, Vakuumleckage und Filamentausfall, realisiert. Die Analyse der Daten zeigte eine unerwartete Abhängigkeit zwischen dem Wert des Filamentstroms und dem Logarithmus der verbleibenden Lebensdauer, d.h. I ist linear abhängig von $\log{(t_{fail} - t)}$. Selbst mit einer geringen 8-Bit-Auflösung konnte eine Fehlerprognose bis zu 300 Arbeitsstunden vor dem Ausfall erfolgen. Innerhalb von zwei Wochen vor dem Ausfall konnte der genaue Zeitpunkt mit einer Genauigkeit von bis zu ± 4 Stunden vorhergesagt werden. Dies demonstrierte die Machbarkeit der vorausschauenden Wartung am Röntgenmessgerät.

Um die vorausschauende Wartung in die Hauptmesssoftware zu integrieren, wurde eine Softwarearchitektur vorgeschlagen, die auf den validierten Wertversprechen und Funktionen mit internen Interessengruppen basiert. Die Architektur ermöglicht eine maximale Wiederverwendbarkeit von Softwarekomponenten, erleichtert die Integration und ermöglicht Over-the-Air-Updates des Vorhersagealgorithmus zur kontinuierlichen Verbesserung der Vorhersageleistung.

In dieser Dissertation wurde die Architektur für die Entwicklung und Integration einer Lösung für vorausschauende Instandhaltung präsentiert und unter realen Bedingungen umgesetzt. Die Umsetzung des Algorithmus zeigte die Machbarkeit der Lösung, ausreichende Vorhersageleistung zu liefern, um die versprochenen Geschäftswerte zu realisieren. Potenzielle Verbesserungen der Vorhersageleistung basierend auf zusätzlichen hochwertigen Daten wurden untersucht und demonstriert.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisor Prof. Dr. rer. nat. Wilhelm Stork. Not only does he provide me with the greatest support and autonomy during my Ph.D. time, but he is also a great source of inspiration and guidance for me, and for all of his students.

I also want to express my sincerest thanks to Prof. Dr. rer. nat. Cornelius Neumann for kindly accepting to be the second reviewer of my dissertation.

My special thanks goes to Dr. Ashok Chandra-Sekaran. His most treasured support and guidance have made this dissertation possible.

I am also grateful to be able to work with the most amazing colleagues in ITIV. Cheers to Xiang, Simon, Gabriela, Nana, Toni, Dejenie, Anqi, Marius and Markus. Besides, this endeavour would not have been possible without the support of all the partners and colleagues, whom for various reasons I would like to mention by first name: Christian, Udo, Reinhard, Dmitrij, Michael, Jürgen, Volker, Rolf, Uwe, Kostas, Sven, Amel, Hagen, Jonathan, Pia, Steffen, Robert, Lars, Cay-Uwe, Bas, Timur, Thomas, Dokkun, Daniel, Mangesh, Amol, and all others.

Thank you to all my teachers, professors, friends, and others in Vietnam, in Germany, and elsewhere, who inspired me and gave me valuable lessons.

Words cannot express my gratitude to my parents. I still remember you carried me on motorbike to the evening tutor class, in the crazy traffic, under torrenting monsoon rain. Thank you for giving me all you have and could have had!

Omnis enim qui petit, accipit; et qui quaerit, invenit; et pulsanti aperietur.

Acronyms and Abbreviations

ACID	Atomicity, Consistency, Isolation, Durability
ADC	Analog-to-Digital Converter
AI	Artificial Intelligence
API	Application Programming Interface
ARIMA	AutoRegressive Integrated Moving Average
BDMS	Big Data Management System
B2B	Business to Business
B2C	Business to Consumer
CAP	Consistency Availability Partition-tolerance
CRF	Conditional Random Field
CRUD	Create, Read, Update, Delete
CPU	Central Processing Unit
DL	Deep Learning
DLL	Dynamic Link Library
DSL	Digital Subscriber Line
EDXRF	Energy-dispersive X-Ray Fluorescence Spectroscopy
EMP	Electromagnetic Pulse

ENC	Equivalent Noise Charge
ERP	Enterprise Resource Planning
FET	Field Effect Transistors
FIFO	First-In, First-Out
GB	Gigabyte
Gbps	Gigabits per second
GD	Gradient Descent
GPU	Graphics Processing Unit
HDD	Hard Disk Drive
HTTP	Hypertext Transfer Protocol
HV	High Voltage
IaaS	Infrastructure as a Service
IO	Input/Output
IoT	Internet of Things
IP	Internet Protocol
IPC	Inter-Process Communication
ISO	International Standard Organization
IT	Information Technology
ITIV	Institute for Information Processing Technologies
kHz	Kilohertz
KIT	Karlsruhe Institute for Technology
LAN	Local Area Network

l.d.o.	Linearly dependent on
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MB	Megabyte
Mbps	Megabits per second
MHz	Megahertz
ML	Machine Learning
MQTT	Message Queuing Telemetry Transport
MSE	Mean Squared Error
MTBF	Mean time between failures
MTTR	Mean time to repair
NoSQL	Not only SQL
NTC	Negative Temperature Coefficient Sensor
OS	Operating System
OTA	Over The Air
PaaS	Platform as a Service
PAN	Personal Area Network
PC	Personal Computer
pdf	Probability Density Function
PHM	Prognostic Health Management
PM	Predictive Maintenance
R&D	Research and Development

RAM	Random Access Memory
RDBMS	Relational Database Management System
REST	Representational State Transfer
RNN	Recurrent Neural Network
RTOS	Real-Time Operating System
RUL	Remaining Useful Lifetime
SaaS	Software as a Service
SCADA	Supervisory Control And Data Acquisition
SD	Secure Digital
SDD	Silicon Drift Detector
SGD	Stochastic Gradient Descent
SME	Small and Medium-sized Enterprises
SNR	Signal-to-Noise Ratio
SQL	Structured Query Language
SSD	Solid State Drive
TCP	Transmission Control Protocol
TTL	Transistor-Transistor Logic
UI	User Interface
USB	Universal Serial Bus
VPN	Virtual Private Network
WAN	Wide Area Network
Wi – Fi	Wireless Fidelity

WP	Work Package
XRF	X-Ray Fluorescence Spectroscopy
XRG	X-Ray Generator
XML	Extensible Markup Language

Contents

1 Introduction

In this chapter, an overview of the larger context and the specific domain of this research work is presented. In the first section, the visions of innovative business models and value propositions, enabled by data and connectivity, are discussed with a focus on Small-Medium Enterprises (SMEs). Predictive Maintenance (PM) is presented as one concrete application of this vision. The second section discusses the specific domain of this work, namely a Predictive Maintenance solution for X-Ray Fluorescence (XRF) measurement devices of an SME manufacturer. The last section lays out specific research questions for the realization of such a solution.

1.1 Motivation: Visions For Small and Medium-sized Enterprises

1.1.1 The Industry Is Entering The Information Age With The Power Of Data and Connectivity

1.1.1.1 “Industry 4.0” In The Manufacturing Industry

The term "Industry 4.0" originated in 2011 as a Future Project, part of the High-tech Strategy of the German Government [1]. It describes the integration of advanced technologies to completely revolutionize the way companies manufacture and distribute their products. In this way, it plans to bring about the Fourth Industrial Revolution, which aims at:

- increasing efficiency and competitiveness, and future-proofing the manufacturing industry of the nation,
- addressing the most pressing socio-economic challenges of the current age (demographic, environment), and,
- achieving a new level of economic welfare for the population.

The basis of Industry 4.0 is cyber-physical systems, in the same way as steam, electricity, electronics and IT have brought about the First, Second, and Third Industrial Revolutions, respectively [2]. Cyber-physical (or smart) systems are systems with deep integration of computation, communication, and physical processes. Examples include autonomous robots, smart wearable devices, and connected industrial machines. These systems are not only capable of efficiently performing their primary function in a complex, ever-changing environment but can also contribute in a meaningful way to other tasks which are traditionally performed by other agents. Those include but are not limited to self-diagnostics, recommendation formulation, or even decision-making.

Cyber-physical systems in the industry can work together to create "Smart Factories" [3]. In these factories, products, means of production (machines), and humans can generate information and communicate in real-time. This information flow enables new ways of production, value creation, and real-time optimization. The "walls" that previously existed between "information silos" are demolished.

Industry 4.0 has gained significant attention and adoption around the world, as companies and governments have recognized the potential of these technologies. In a survey conducted by McKinsey Global in 2019, 68 percent of participating companies regarded Industry 4.0 as a top strategic priority [4]. Among those, two-thirds were already testing or deploying new technologies to their production. The white paper "The Next Economic Growth Engine" in the World Economic Forum 2018 (Pre-Pandemic) estimated that Industry 4.0 potentially resulted in an estimated value-creation potential of $3.7 trillion in 2025 in the manufacturing and supply-chain industry [5]. Furthermore, the concepts and technologies of

Industry 4.0 can be applied across many other industrial segments such as oil and gas, mining, transportation, and others.

1.1.1.2 Advanced Digital Technologies Are The Enabler For The Trend In The Industry

The trend towards Industry 4.0 is characterized by the integration of advanced technologies such as artificial intelligence, the Internet of Things (IoT), and cloud computing into manufacturing processes.

At the foundation of those technologies are the technological mega-trends of increasing cost-effectiveness and capability of both connectivity and computational power. Large amount of data could be exchanged now at a speed never seen before. For example, WiFi devices used to support a transfer rate of 54Mbps with 802.11g generation [6]. Now at the 802.11ac generation, a transfer rate of 3.46Gbps is supported, which will then increase to 30Gbps in the coming 802.11be generation. At the same time, data could be generated, consumed, and analyzed by more powerful and cheaper computers as a consequence of Moore's Law.

These technologies have empowered the digital transformation of the manufacturing industry in many ways. Advanced sensors from multiple machines collect data. These are then analyzed with advanced data processing or Artificial Intelligence on local embedded computers, or on the Cloud. The analysis results deliver real-time visibility of manufacturing assets, thus allowing for better decision-making. Even higher value is delivered when data from customer orders are combined with operational data from the factory, supply chain, and other enterprise systems (such as ERP). With the optimized production process and supply chain, customers can then receive a completely individualized product at the same cost and time previously only possible by serial mass production. Furthermore, the machine could "interact" directly with the product, performing automatic quality inspection based on Artificial Intelligence, and delivering recommendations to the Human via Cloud or via Smart Glasses. In this way, both the manufacturer and customer benefit from higher productivity, improved quality, and reduced cost, and enjoy the value created by massive customization and enhanced customer interaction.

1.1.1.3 Subscription And Cloud Business Model In The Software Industry

Similar to the manufacturing sector, the software sector in recent years has also seen a paradigm shift towards offering subscription-based and cloud-based business model [7]. In contrast to the traditional one-time purchase option, customers in the subscription model pay a recurring fee to access a product or service, typically on a monthly or annual basis. The delivery of software in the subscription model is usually done with Cloud computing [8]. With this technology, many services in the software (e.g., data storage or user management), or even the whole software, do not reside locally user's computer, but are hosted on a server and accessible via the Internet.

There are three variants of this delivery model depending on how much the customer wants to develop and maintain the software they need [9]. At the minimum, Infrastructure-as-a-Service (IaaS) providers deliver customers with storage, server hardware, and networking. Based on this the customer have to develop the majority part of the software they want to host on the Cloud. Platform-as-a-Service (PaaS) Providers offer platforms, which are a collection of tools and services to assist the customer to develop and host the application with minimal effort. When the complete software is offered by the provider, the model is called "Software-as-a-Service" or SaaS. In this model, the customer can start using the software or service as soon as they purchase it [10]. When moving from IaaS to Paas to SaaS, the customers have less flexibility to develop software tailored to their needs. Nevertheless, the effort and cost required for development and maintenance were reduced as well. It is worth mentioning that a provider can develop their own software or service, host it on the IaaS of another provider, and sell it as SaaS to their customers.

The subscription model in conjunction with Cloud delivers several benefits to both customers and software companies. For customers, the benefits are delivered on three fronts: Cost, Quality of Service, and Personalization [11] [12].

- Customer can access the software in a more cost-effective and flexible way. Instead of making a large upfront purchase, customers can start using the software to generate value for themselves with a starting fee several times lower than the full price.
- Another dimension of the cost is the total cost of ownership. For many B2B software, there are costs associated with maintaining the internal infrastructure, personnel, and security. In subscribed and cloud-delivered software, this is performed by the software provider. Even though these costs will eventually be included in the subscription fee, the average cost will be lower due to economies of scale on the provider side.
- With the subscription and cloud delivery model, the Quality of Service is enhanced. Customers can receive the latest version of the software with minimal or no additional cost or effort. Customer support during training, usage, and troubleshooting of software is more efficient, as access to logging information and to the software itself is available.
- This delivery model enables deep Personalization of the software product depending on customer needs. Not only is the front-end (user interface) customizable, but customers can also choose the capability of software that is most suitable for the size and nature of the customer's organization. The software can also be module-based, and customers can choose which modules to use and to pay for. In the cloud delivery model, if the customer later chooses to upgrade the capability of the software (more storage, more features, etc.), this migration could be done smoothly with minimal transition cost.
- The delivery model allows for "Curation", a higher level of Personalization. In this, the provider company will compose the best features and/or contents before delivery, based on the specificity of the said customer, and on the

experiences with other customers. In this way, an exceptional and exclusive customer experience can be delivered.

Besides delivering immense value to the customer, software providers also adopt the subscription- and cloud-based model for financial and customer relation benefits [13] [14].

- The recurring subscription fee provides a predictable and stable cashflow for the company. On the one hand, this allows for easier budgeting and planning inside the company, which drives better business growth. On the other hand, the visibility and predictability of the revenue stream attract investors and drive up the valuation of the business. In fact, at the beginning of 2020 (Pre-pandemic), Salesforce, one of the pioneers of SaaS models, was valuated higher than SAP despite having only more than half of SAP's yearly revenue. This valuation still holds at the time of writing (Beginning of 2023).
- The model allows better expansion and retention of the customer base. The reason for this is the lower barrier of entry. Thus, even customers with lower purchasing power are enticed to try the product and software. The absence of a large upfront investment also means that the productivity and benefit delivered by the software or product are more visible to the customer. Besides increased customer satisfaction, there is also a psychological effect that helps increase customer retention. Researchers found that, as the customers are recurrently paying the subscription fee, they experience "sunk-cost fallacy" and attempt to use the product more, which in turn increases the "lock-in" effect of the product.
- The model enables additional revenue stream from up-selling and cross-selling to existing customers. In the traditional model, revenue is created once at the time of the customer's purchase. In contrast, after a certain period of subscription, the customers' usage of the product can increase or change due to business growth, or due to increased experience with the product. These usage statistics can be used to offer customer products

with better capabilities (up-selling), or relevant product which better fits the customer's need (cross-selling).

- Finally, the data and enhanced interaction allows companies to learn directly from the customers and improve their products and services. This can also result in products with completely new features, only possible with Cloud. An example is the live collaboration feature in Google Docs and Microsoft Office 365.

1.1.1.4 New Business Models in Other Industries

Outside the software industry, companies have applied the subscription business models for many other products and service [15] [16], which can be categorized under [17]:

- Information and Contents, these include magazines, music (Spotify), movies and TV shows (Netflix), or even workouts (Peloton)
- Consumables, which includes not only meal boxes or health and beauty products for B2C customers, but also chemicals, printer inks, and raw materials for the B2B segment
- Durable goods. For B2C customers, those can be electronics or household products (Grover). For B2B customers, machines for production, instead of being purchased, can be subscribed to. Example includes printing machines from Heidelberger Druckmaschinen [18].

Collectively, the value of subscription businesses grew 300% between 2012 and 2018 and could create an opportunity worth \$1.7 trillion to \$3 trillion for companies [19]. The business model opens new doors for growth and customer-centricity and is projected to become the norm and not the exception.

1.1.2 Benefits of Data and Connectivity For SME Machine Manufacturers

1.1.2.1 Overview

Small and Medium Enterprises (SMEs) are defined by the European Commission as those employing fewer than 250 persons with an annual turnover of less than €50 million and/or a balance sheet of less than €43 million. In the European Union, SMEs in the non-financial economy sector employ up to 64.4% of the workforce and contribute 52.4% to the total value added [20]. These enterprises are often considered the backbone of the European economy, providing jobs and growth opportunities.

SME machine manufacturers have an important distinction compared to other sectors: their products are a part of the value creation chain of other businesses. Thus, the concepts and technologies from Industry 4.0, Cloud, and Subscription business model benefit not only the value creation chain inside the SME (utilizing Smart Factories as mentioned above) but also the value creation chain of their customers.

The benefits, which SMEs can deliver to their customers in the manufacturing sector, are analogous to the benefits Industry 4.0 and Subscription/Cloud bring to all other economic sectors, namely:

- Improve the efficiency of production and Quality of products and services
- Lower entry barrier for customer and expand customer's base
- "Batch size one" - customized product at the low cost of serial production
- Improve customer retention, learn directly from customers, and enable new business models

1.1.2.2 Transformation Process To Realize The Benefits

SME machine manufacturers can contribute to the Industry 4.0 journey of their customers by offering innovative products, services, and business models. More concretely, the SMEs are embracing the transformation of their product range towards digitalization and servitization [21]. This means the transition from selling a pure product to selling a range of services and solutions that directly address the needs of customers. Tukker (2004) [21], extended by Lerch (2014) [22], described this transition in a three-step process:

- *Product-oriented service*: The product stands still at the center of business activities. However, manufacturers start digitalizing the service offerings including repair and maintenance, customer support, training, and consultation. The digitalized service serves as a differentiator to enhance customer experience.
- *Use-oriented service*: Instead of purchasing the ownership, customers pay only for the use of the product. Possible scenarios include product leasing, pay-per-use, product sharing, and consumables subscription, among others. At the same time, the product portfolio can be extended with pure digital services. Those include e.g. software platform, AI-based analytics.
- *Result-oriented service*: At this level, customers pay for the actual business value that they get, regardless of how, or whether the product gets used at all. With the product no longer at the center, manufacturers become solution providers. The customer can completely outsource an aspect of the business, e.g. quality control or asset maintenance, to the solution provider. The provider would apply a bundle of tangible products and intangible services and software to deliver the result.

1.1.2.3 Challenges

Despite the promising prospects, SMEs face many roadblocks in adopting Industry 4.0 [23]. Bollhöfer (2014) [24] and Müller (2019) [25] [26] interviewed SMEs to investigate these difficulties. Some notable factors can be summarized as follows:

- *Lack of resource and competency*: New technologies require investment in hardware, software, and IT infrastructure. New competencies required the retraining of existing personnel. The development of technologies required new personnel with expertise in the field, who are often hard to come by and are attracted by large enterprises. Another key resource for development, namely data, is also lacking in quantity and quality.
- *Security concerns*: Data and connectivity are required for cyber-physical systems in Industry 4.0. Nevertheless, SMEs are very aware of the security concern of their customers, which include the fear of loss of data, public disclosure of data, and unwanted access to machines. All of them could result in consequences for the responsible employee (on the customer's side).
- *Unclear vision*: There is a lack of clarity in strategy and implementation plan as well as a target. Many SMEs show vagueness in deriving suitable value propositions. In many cases, the value propositions are assessed based on a short-term cost (or cost-saving) perspective. SMEs, who deliver those value propositions to customers, find it hard to estimate financial benefits for their own company. The cooperation between departments to achieve a synergy effect is also lacking.
- *Worries about technologies*: On the one hand, the employees are not clear yet about the benefits that Industry 4.0 bring to their day-to-day work. On the other hand, they worry that their jobs will either be replaced by technology or the technology will be used as surveillance on them. The fear of the loss of control was also mentioned.

Addressing those challenges requires a holistic approach consisting of but not limited to the following aspects: business model innovation, organization transformation, technological innovation, and clarification of legal framework.

1.1.3 Predictive Maintenance As An Application Enabled By Data and Connectivity

1.1.3.1 Overview Of Maintenance Strategies In Industrial Settings

On a more general level, maintenance of industrial machines is necessary to minimize the risk of failure and to reduce machine downtime [27]. In other words, it aims to maximize the Mean time between failures (MTBF) and minimize the Mean time to repair (MTTR), thus maximizing (percentage of) uptime, in which the machine is used productively. Failure leads to lost sales in consumer settings and loss of productivity in industrial settings. For measurement machines, even small failures can lead to reduced accuracy and inferior products. For safety-critical machines, failure can lead to catastrophic consequences.

Maintenance activities aim at preventing, detecting, and correcting defects before they become failure [28]. Activities such as cleaning or replenishing consumables (e.g. oil) were carried out to ensure the most favorable operation for the machine. Timely detection of small defects could prevent them to grow into large ones, which may lead to more costly repairs later. Corrective actions include remedying the defect or failure, such as swapping out defective parts. A proper maintenance record does not only increase uptime and equipment lifespan but also maintains the value of the machine (e.g. for resale) and ensures compliance with standards (ISO 9001) [29] [30].

There are multiple strategies to carry out the maintenance activities [31] [28]:

- Reactive Maintenance: The machine is fixed when it is broken down. No strategy is also a strategy.

- Preventive Maintenance: Maintenance activities are carried out at regular intervals. Interval is determined often by experience. Activities also include precautionary measures, such as swapping out parts ahead of time, even if there are not yet (detectable) signs of defect.
- Condition-based Maintenance: When signs of defects are detected, or certain parameters related to the machine's health are below/above a threshold value, maintenance activities are carried out.
- Predictive Maintenance: this is a subset of Condition-based Maintenance. The difference lies in the fact that the collection of data by sensors and the processing of data by intelligent algorithm are done automatically, and in real-time.
- Prescriptive Maintenance: The next level of the intelligent algorithm can anticipate many scenarios corresponding to many possible courses of action. Based on this it recommends the user the specific actions in order to achieve a certain outcome. For example, which additional actions are needed if the user wishes 99.9% of uptime instead of 99%.

1.1.3.2 Overview Of Predictive Maintenance (PM)

Predictive maintenance is a maintenance strategy that uses data and analytics to predict when equipment or machinery is likely to fail or require maintenance and to schedule maintenance activities accordingly.

The core principle of Predictive Maintenance is the three-step transformation from Data to Insight and finally to Action.

- *Data*: For *each individual* equipment, data should be captured with sufficient quality and quantity from a variety of sources. Operation and performance data come from sensors or logging devices. Data about the environment (temperature, shock acceleration, weather), as well as contextual data (user interaction, usage pattern, historical data, reports, schedules), is also collected for a reliable diagnostic of the assets' situation. Data from disparate sources is then collected via various connectivity methods and protocols to a central location for storage and processing. This location could be an embedded computer, a computer associated with the equipment, an on-premise server, or a public or private Cloud server.

- *Insight*: This data is then analyzed using various techniques to yield insights, or information, that may indicate a potential failure or maintenance need. More specifically, the analytics are done using advanced model-based and/or data-based algorithms, with the latest trends being Artificial Intelligence.
- *Action*: Actual value is delivered when the insights are used to optimize maintenance activities: adapting maintenance schedules, streamlining plans, and prioritizing resource allocation for various maintenance tasks. Digitalization and connectivity enable further optimization of the maintenance workflow throughout all steps of identifying, planning, performing, and evaluating maintenance work.

Predictive maintenance can be applied to a wide range of machinery beyond industrial equipment such as transportation systems, building systems, and energy systems. Predictive maintenance is most useful for industries, in which equipment failures can have significant impacts on operations and costs. General benefits of PM include a reduction in unexpected failure rate and optimization of service activities. According to an estimation, Asset availability could increase by 5-15%, and maintenance costs could reduce by 18-25% with predictive maintenance [32]. The concrete benefits of predictive maintenance in a specific SME context will be discussed in the next section.

1.2 Domain Analysis: A Case Study Of An SME Manufacturer Of X-Ray Fluorescence Measurement Machine

In this work, the concept of predictive maintenance should be applied in the context of an SME manufacturer of X-Ray Fluorescence (XRF) Measurement Machines. These commercial XRF machines are used to determine the elemental concentration and thickness of many materials such as metal or ceramics. The SME manufacturer encountered the problem, that many of the devices failed unexpectedly at customers' sites. This leads to significant consequences for both customer and the manufacturer. Among others, these include costly production interruption for customers, high service costs for both manufacturer and customer and reputation damage for the manufacturer. A predictive maintenance solution should be developed by the manufacturer, which should predict failures before they happen, and make it visible to both customers and manufacturers. This foresight enables effective planning and results in cost savings for both sides. Furthermore, predictive maintenance could be the first of many digital solutions, which would enable further value propositions.

The next subsections would take a closer look at the situation from the customers' side and from the side of the service team of the manufacturer. The ordinary workflow and the effect of unexpected failure will be discussed for both sides. These are followed by a discussion about how predictive maintenance would be an effective solution to this problem. Finally, a long-term vision for predictive maintenance as the enabler for further value propositions will be discussed.

1.2.1 Customer Usage Of XRF

XRF measurement machines play an important role in non-destructive analysis and quality control in many industries. For example, in the industrial galvanic industry, XRF measurement machines are used to control the thickness of the metal galvanized layer. In the gold and jewelry sector, XRF is used to determine the concentration of rare metals, thus determining the purity and value of the gold/jewelry piece. Customers could use XRF to carry out individual measurements, or they could place XRF in a conveyor belt for the continuous monitoring of the production process.

It is vital for many customers that the XRF devices function continuously without unexpected breakdowns. Firstly, in many cases, XRF is the only commercial method for (non-destructive) quality control of the output product. This means that, if the XRF machine failed and there is no reserve capacity, the production would likely be stopped or significantly slowed down. For retail customers, this results in a loss of sales. For industrial customers, this would mean all the fixed cost of production still incurs without any revenue being generated. All the loss would compound over each day of production stop. And it could last many days until the device could be repaired. Secondly, many customers do not possess reserve capacity for XRF measurement. Due to its price, the XRF measurement machine represents a significant investment. As a result, many customers have only one or a few devices. Certain large customers have multiple devices, but the significant device cost also means that the capacity of those devices is well-utilized, and not much reserve capacity remains. In fact, according to the service team of the manufacturer, 80% of all repair cases are considered "urgent" for the continuity of production of the customer. This fact will be important in the next subsection as it also affects the operation of the service team.

To keep the devices in functioning states, currently, the strategies of reactive and preventive maintenance are being utilized. For reactive "maintenance", the customer would use the device until failure occurs. Upon this, the service team of the manufacturer will be contacted for diagnosis and repair. Relevant details of this process would be presented in the next subsection. For preventive maintenance, the customer could sign a maintenance contract with the service team. Every year or half a year, the service technician would visit the customer to perform an inspection of the device. The device would be cleaned and checked for mechanical irregularities. Key components such as the X-Ray Source (often called "Tube") and the X-Ray Detector are checked. If the components have been operated exceeding a certain number of hours, it could be recommended to exchange them to prevent possible unexpected failure (thus the term preventive maintenance). This number of hours threshold was, however, determined based on experience. Besides hardware-based inspection, software-based inspection is also carried out. In this, certain standardized measurements are conducted, if the results are then inside the normal range, it would mean that the machine function properly.

The current scheme based on reactive and preventive maintenance was not very effective in helping prevent unexpected failure. Reactive maintenance provides absolutely no defense against either failure or unexpectedness. The inspection-based scheme in preventive maintenance only makes sure the device is working properly at the time of inspection (and might be a short time after that). The interval of half a year to one year between inspections would not be dense enough to prevent unexpected failure in between. Even more, as would be evident from the discussion in Chapter 4, the current method of using standardized measurement to diagnose device health would not be suitable for predicting health. The prophylactic exchange of components would do well in preventing failures. However, this possibly results in the exchange of still well-functioning components. The additional cost arises not only from the cost of components but also the extra time cost of the service technician. Opportunity cost also arises, as the device could be utilized for production, and the service technician could help other customers in the meantime.

1.2.2 Service Workflow

The service team of the XRF machine manufacturer is responsible not only for the inspection and repair of customers' devices. They are also the first point of contact whenever the customers need training, usage support, device calibration, and other after-sales services. In Germany, the manufacturer has thousands of customers but only a dozen of service technicians. This translates to a large amount of work for the service team. Planning and plannability are thus absolutely necessary to tackle this amount of work effectively.

Among the maintenance responsibilities of the service team, the repair activities are not easily plannable. It is after the failure has occurred that the customer became aware of the problem and then starts contacting the service team. From that point, it takes a few days up to a few weeks until the device is repaired. To understand why this is the case and how plan-ability is absolutely important for the improvement of MTTR, it is helpful to quickly examine the service process for repair. When the customer first contacts the manufacturer, the first level support would exchange a few key pieces of information to categorize what is the need of the customer (might be just training support or general question on the usage of the device). If the customer needs repair, a first-level support technician can guide them through some general instructions (e.g. check the power supply cable) and ask some further questions to initially identify the problem. If the problem persists, a support ticket is generated and assigned to a second-level support technician with available capacity. They will then analyze the information, call and discuss with the customer to diagnose or possibly remotely resolve the problem. When the diagnosis is finished and the problem persists, based on their experience, the technician would compose a service cost offer and send them to the customer. This includes the first estimation of component cost, technician cost, and travel cost for the repair. If the customer accepts the cost, the travel of a technician to the customer could be planned. The technician, who is stationed in the region closest to the customer, would then travel and repair the device at the customer's location. After the repair, the device would normally need calibration and other checks. If everything proceeds smoothly, only this repair and calibration process at the customer would last for one full working day (not including travel time). Finally, the invoice is generated. This amounts in many cases to be in the order of thousands of Euros, with a non-negligible portion attributed to travel and time cost.

Upon analyzing the repair process above, it would be clear that not only does each of the steps takes a lot of time, but the time between steps could also potentially take days. The key reason here is the dependence on the availability of technician capacity. An unplanned ticket can be assigned by first-level support to second-level only to the time slot when their capacity is not occupied by other pre-planned activities (such as inspection). There is also a delay between when the ticket is assigned and when it is processed, as the technician is occupied with other tasks in between. The analysis of the problem could also take a lot of time. The technician would guide the customer through many steps of troubleshooting to identify the problem. In many cases, the customer must guide the customer to carry out special operations to collect the data necessary for diagnosis. Both the communication and the operations may take hours. Even then, the diagnosis is not guaranteed to result in a precise service cost offer. In the case of a not precise service cost offer, the technician, being already at the customer location, may even have to drive back to obtain additional components.

After a concrete repair plan is formulated, the technicians are then assigned to travel to the customer and conduct the repair. This represents a significant pain point for both the service team and the customer. Clearly, the customer would like the service technician to travel right away to repair the devices (in fact, as mentioned above, 80% of the repair cases are considered urgent). Due to limited capacity, the weekly plan of the service team is usually strictly booked. The customer thus has to wait days or a few weeks until the travel could commence. If the repair case is considered "high priority" by the service team and immediate repair is required, there are two possibilities. One is to delay the inspection appointments of other customers to make a time slot. This results in not only a lot of communication and organization issues with the customer but also leads to dissatisfaction from the customers' side. The other option is to reassign a service technician to travel from one region to another (e.g. travel from North to South Germany) to conduct the repair at the customer. This leads to extremely high travel and time cost, and unnecessary extra environmental impact.

All in all, the current service workflow calls for more plan-ability in order to minimize repair delay and cost, and reduce stress for the service team as well as the customer.

1.2.3 How Predictive Maintenance Address The Domain's Problem

In Predictive Maintenance, based on the data collected from the device during the operation, an algorithm could give a prediction about the potential failure of the components in the device. This foresight delivers benefits to both customer and the service team of the SME XRF manufacturer and addresses the aforementioned pain points.

1.2.3.1 Predictive Maintenance For Customer

For the customer, the foresight offered by Predictive Maintenance could be used to initiate actions that minimize the MTTR. With the failure being **expected** to happen within a certain time period, the customer can plan for a "maintenance break" for the device. During this period, the customer could reallocate the capacity to other devices and reduce the load on the near-failure one. With this, even if the device failed, there is no sudden loss in capacity and production could continue (albeit possibly at a slower pace). The customer may also communicate with their own customer about the **planned** reduction in capacity so that the customer down the line could also prepare. Last and most importantly, the customer can contact the service team of the XRF manufacturer early. So that the service team could plan early and send a technician to repair the device before it even fails. In this way, it is entirely possible that a failure and production stop could completely be avoided.

Even if the failure could not be avoided, error diagnosis could be done more quickly with data available from predictive maintenance. Service technicians have more data to conduct the analysis, this data captures the component status over a longer period and not just a snapshot at one point in time. In this way, the service cost offer could thus be generated more precisely. These bring many benefits. First, this represents a time-saving in the back-and-forth communication for both customers and service technicians, which could span over many days, even weeks. Secondly, a more accurate diagnosis means the service technician could sufficiently bring all the components needed to the customer. The risk of having to drive back to obtain some more needed components is minimized. Thirdly, resulting from better diagnosis, a more precise service cost offer could be sent to the customer. This means that surprising additional costs would be less likely to occur and the customers could allocate the finance for the repair more reliably. All in all, the MTTR is reduced due to the fact that both customer and the service team have better preparation.

Finally, if the customer requires a high degree of availability of the devices, they no longer have to opt for prophylactic exchange of the components when it still functions completely well. The decision for exchange would now be based on the specific health status of the component, based on the data collected from it. Not only does this represent cost savings for the customer, but it also frees up the time of the service team to address other pressing matters, and it also reduces the environmental impact of maintaining the device.

1.2.3.2 Service Policy Optimization

For the service team, the availability of foresight offered by predictive maintenance would improve the planning of service activities. This leads to shorter reaction time, cost savings, and more customer satisfaction. The planning would be done with prior knowledge of device breakdowns that could happen in the planning period (e.g. within a working week). In this way, the planner could prepare a buffer time in the technician's capacity to deal with the breakdowns instead of planning all the capacity to the already plannable activities (such as inspection and training). In this way, the reaction time to a breakdown event would be shortened, due to the fact that unexpected events no longer have to wait until the next planning period (i.e. week) before it could be handled.

The amount of buffer time in each period could also be better planned to match the number of expected breakdowns. This is especially important in order to soften out the **peak demands** for repair. This could happen when a lot of **unexpected** breakdowns (and thus repair activities) occur at the same time. On the one hand, this creates a lot of stress for planners and technicians. On the other hand, customers also have to face an unusually long delay until being served in this case. With prior knowledge of the wave of incoming breakdowns, the planner could plan the activities over the planning periods or the next, even before the breakdowns have happened. This makes sure that the technicians have the capacity to serve the customer the best they could.

The data connectivity aspect of predictive maintenance could also streamline the process of error diagnosis. If a breakdown is predicted to happen, the software could automatically generate a service ticket with all relevant component data. For once, this accelerates the diagnosis process with all the benefits mentioned in the subsection 1.2.3.1. For another, the details of the required reaction time and impact of the failures could help the service team to better prioritize the repair action and improve customer satisfaction as a whole.

Another benefit could result from the combination of the prediction with the location of the customer. If it is known that two repair actions should happen at two customers, who locate near each other, the planner could combine the trip, and the service technician could only drive once to the locations and not back and forth. In another case, if a repair activity should happen shortly at a customer, who is nearby the current location of a service technician (maybe they are serving another customer), they can drive directly to the next customer without having to drive back to the headquarter. Time on the road is not productive time, and driving costs money and fuel. The foresight in planning would thus help the service activities to be more efficient.

All in all, Predictive Maintenance does not only initiate the whole process of repair (from first customer contact to on-site repair) before the breakdown actually happens. It also molds the process to become more efficient. An internal quick evaluation estimate a cost saving on the order of six figures for the service team per year. It also has the impact of improving customer satisfaction with the service and the manufacturer as a whole, which positively impacts sales. Finally, it also improves the well-being of service technicians and improves the environmental footprint of the whole service activity.

1.2.4 Additional Value Propositions Enabled By Predictive Maintenance

Besides the benefit to the customer and the service team, the infrastructure and data resulting from conducting predictive maintenance could be utilized as an enabler for other value propositions.

Inspired by the discussion about the subscription model in Section 1.1.1.3, instead of being offered as a one-time purchase, the XRF device might be offered under the scheme Device-as-a-service to the customer with all the aforementioned benefits. Predictive Maintenance would then provide information regarding the health and level of usage of the device. Based on this, the terms of the leasing contract (pricing, period) could be established and even customized depending on the usage profile of the customer.

Full deployment of a Predictive Maintenance solution would be accompanied by a system that provides connectivity from devices to private/public cloud. This infrastructure could be used to offer value propositions such as a cloud-based measurement service. The device no longer requires an external PC for device control and measurement calculations. Measurements could be done remotely with any connected device such as a tablet. This is a step towards a cyber-physical system (i.e. the key concept of Industry 4.0). Besides, all the benefits mentioned in section 1.2.3, the additional visibility to the measurement results could be offered to any stakeholders in real-time, regardless of their location. This potentially unlocks even further novel value propositions. For example, a company with many XRF devices (or even the manufacturer themselves) could offer measurement-as-a-service to a third party, who might require results from well-conducted measurements, but has not sufficient demand to justify purchasing a device.

Last but not least, the data collected as a part of the Predictive Maintenance solution helps the XRF manufacturer to gain more understanding of their customer and their own product. A deeper understanding of customers would help the company to properly target and improve the most important measurement tasks that the customer uses the device for. A better understanding of the product would identify problematic areas in device quality and device long-term performance, and contribute to the continuous improvement of products.

1.3 Research Goals

The research work represents the first steps in the digitalization journey that enable innovative products and services for the aforementioned SME company. Predictive maintenance is selected as the first innovative product in this journey. The research presented in this work lays the first steps towards creating a predictive maintenance solution for this SME company. Despite the specific focus on XRF devices, the work is applicable to other similar SMEs facing similar problems. Toward that vision, the research work has set the following goals:

- The first goal is to come up with an architecture that enables the data collection from machines to be used in the development of a predictive maintenance solution.
- The second goal is to realize important parts of the architecture into a data collection system and start collecting data from multiple devices at customers and in the laboratory.
- The third goal is the analysis of data and development of a proof-of-concept algorithm and demonstrate that predictive maintenance is possible on the XRF device from those collected data.
- The fourth goal is to investigate technical possibilities of improving the prediction quality and estimate how much it could be improved.

- The fifth goal is to derive an architecture for integrating predictive maintenance into the future product of the SME. For this, both the business and technical aspects should be considered.

2 Background

In this chapter, an overview of both fundamental and state-of-the-art background information related to a data-based Predictive Maintenance solution for XRF components is presented. In the first section, the principle of XRF, its components as well as degradation and failure mechanisms of the components are discussed. The second section discussed current practices and systems related to the handling of data and the development of data-related solutions. Data-based failure prediction methods are discussed in the third section.

2.1 Vacuum Enclosed Radiative Components for X Ray Fluorescent Spectroscopy

2.1.1 X Ray Fluorescent Spectroscopy

X-Ray Fluorescent Spectroscopy (XRF) is an analytical technique for qualitative and quantitative determination of the elemental composition of the sample. It functions based on radiative ionization and subsequent emission of characteristics X-Ray from the atoms of the samples. As a result, it is non-destructive, making it suitable for a range of applications from industrial quality control to archaeology.

In XRF, the sample is radiated with a primary X-Ray beam. The electron in the atom absorbs the photon and is ejected from the atom's shell, leaving behind a vacancy. Once an electron from an outer shell fills up the vacancy in an inner

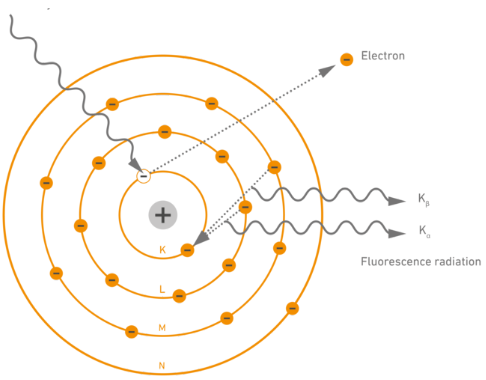


Figure 2.1: Principle of X-Ray Fluorescence Spectroscopy [33].

shell, X-Ray is emitted in form of lines, whose wavelengths are described by the Moseley's Law:

$$\frac{1}{\lambda} = k(Z - \sigma)^2 \tag{2.1}$$

where k is a constant depending on the type of transition, Z is the atomic number, and σ represents the screening effect of the electron shells. The dependence on Z means that the emission line is characteristic for each element.

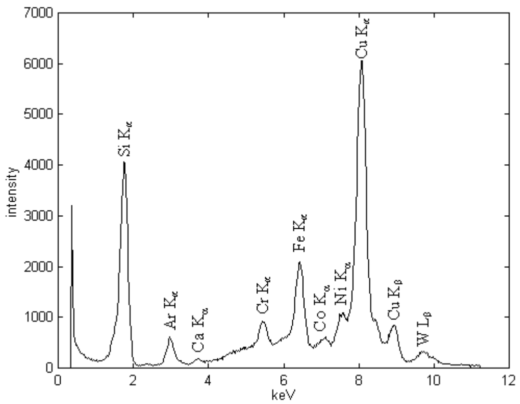


Figure 2.2: XRF spectrum collected from a quartz sample. The fluorescence lines were broadened into peaks due to various sources of noise [34].

In Energy Dispersive XRF, a detector would capture and analyze the spectrum from the sample. The spectrum diagram shows the intensity of X-Ray as a function of photon energy. The typical spectrum from samples consists of a series of peaks. The location of the peaks gives information about the elements in the sample. The area under the peak is related to the number of atoms of each element. This information could be used in conjunction with an algorithm to determine the quantitative information such as the concentration of elements, of the thickness of the layers of materials in the sample.

2.1.2 Commercial X-Ray Generator (XRG)

In commercial applications, continuous primary X-Ray beams can be generated by X-Ray Tubes. A schematic diagram of a type of tube usually used in dental applications is presented in Figure 2.3.

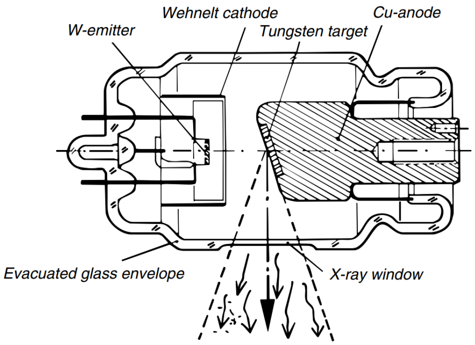


Figure 2.3: Schematic diagram of a commercial X-Ray Tube with stationary anode [35].

The filament (usually from Tungsten) on the cathode side is heated up by the filament current. At high temperatures, electrons will be emitted from the filament following a process called thermionic emission. The emitted electron current density is described by the Richardson equation:

$$J = A_G T^2 \exp - \frac{W}{kT} \tag{2.2}$$

with W is the work function of the metal, k is the Boltzmann constant, T is the temperature. The coefficient $A_G = \lambda_R \frac{4\pi m_e k^2 q_e}{h^3}$ depends on the material specific correlation factor $\lambda_R \approx 0.5$, mass of the electron m_e, charge of electron q_e, and Planck constant h.

In commercial X-Ray tubes, the filament is negatively biased with regard to its surrounding. The resulting electric field leads to enhanced electron emission following the Schottky effect. This is modeled by a decrease in the work function, lowering the barrier for the electron to escape the surface. The modified work function is:

$$\begin{aligned} W' &= W - \Delta W \\ \Delta W &= \sqrt{\frac{q_e^3 F}{4\pi\epsilon_0}} \end{aligned} \tag{2.3}$$

with F being the strength of electric field, and ϵ_0 is the electric permittivity of vacuum.

The beam of emitted electrons is shaped and focused by the Wehnelt cylinder. It is essentially a metal cup with an open aperture and is negatively biased relative to the filament.

The emitted electrons are accelerated by the Tube Voltage, which is on the order of tens of kilovolts. Electrons flow in vacuum and then impact the target surface of the Anode. This generates the primary X-Ray beam. A typical spectrum of the generated X-Ray can be seen in Figure 2.4.

The emitted spectrum consists of characteristic lines of the anode surface material imparted on the continuous background from Bremsstrahlung radiation. The Bremsstrahlung, as the name implied ("bremsen" means "to brake"), is the

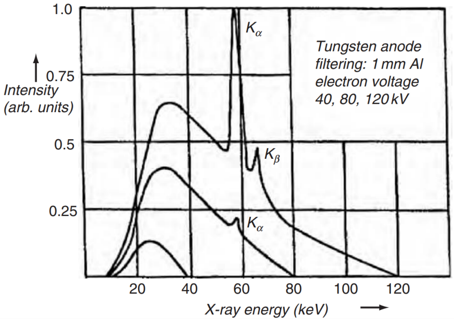


Figure 2.4: (Primary) Spectrum generated by a X-Ray Tube with at various Cathode-Anode voltages [35].

result of the rapid deceleration of high energy electrons following their inelastic collision with the Coulomb field of the atom. An important characteristic of the bremsstrahlung spectrum is the shortest emitted wavelength:

$$\lambda_{min} = \frac{hc}{q_e V_0} \tag{2.4}$$

with V_0 being the tube voltage. This wavelength corresponds to the highest energy X-Ray photon emitted by the tube. The photon energy should be high enough so that it can "knocks out" the electron in the inner shells of atoms in the sample. For example, the binding energy of the 1s (K shell) electron of Silver (Ag) is 25.5 keV [36]. Thus, in order to be able to obtain a Silver K_α line (i.e. photon from the transition of an electron from L to K shell), the tube voltage must be set higher than 25.5 kV.

The characteristic lines are generated in the same process as described in the previous subsection. However, these are undesirable in the primary X-ray beam, as they can be reflected and get mixed up with the secondary (fluorescent) X-Ray spectrum from the sample. To prevent that, filters (e.g. Aluminium or Nickel) can be set up on the path of the primary X-Ray beam to impart the desirable characteristics on the primary spectrum.

The majority of the energy of impact, however, is converted to thermal energy instead of X-Ray radiation. The efficiency of X-Ray Production can be calculated by Kramers' law:

$$Eff = 9.2 \times 10^{-10} Z V_0 \tag{2.5}$$

The tubes in practice have an efficiency of around 0.1 to 1% with the rest essentially becoming thermal energy [37] [38]. To dissipate a large amount of heat, the rest of the anode (apart from the e.g. Tungsten target) is made up of a massive body of copper. The copper anode extends beyond the glass envelope of the tube and is often attached to fins to increase heat dissipation.

The cathode assembly and the majority of the anode are enclosed in an evacuated glass envelope (hence the term "tube"). The vacuum helps ensure the electrons and resulting X-Ray photons were not impeded or absorbed by gas molecules. The glass tube is embedded into a bath tube of oil to help dissipate heat by convection. The outer wall of the bath tube is made from solid copper which provides good thermal conductivity and shielding against X-Ray radiation.

2.1.3 Silicon X-Ray Detector

In ED-XRF, the role of the X-Ray Detector is to capture X-Ray photons coming from the sample and convert each of them into an electrical signal proportional to the energy of the photon. There are multiple possibilities to realize a detector [39]. The proportional counter consists of a gas chamber. Upon collision of photons with gas molecules, an electric discharge is generated. Another possibility is the scintillation detector, which uses scintillation material to convert photons into electrons, which are then amplified by the subsequent electron multiplier.

A silicon-based detector offers much higher energy resolution than the proportional counter and the scintillation type. One reason is that one photon could generate many more electrons in silicon than in gas. This reduces the statistical

(Poisson) noise on the output. The amount of charge Q generated by a photon of energy E is:

$$Q = \frac{E}{\epsilon} \tag{2.6}$$

with $\epsilon \approx 3.6eV$ in Silicon, and $\epsilon \approx 25 - 26eV$ for Argon gas and scintillation material [36]. Another reason for the better resolution is that the signal from the Si detector can be read and amplified directly by on-chip, low-noise electronics instead of through the electron or photon multiplication process in gas-based detectors or scintillation detectors.

A better energy resolution means the characteristic lines coming from the sample are converted to narrower peaks in the detected spectrum. This helps reduce the overlapping of nearby peaks and improves the qualitative and quantitative evaluation of the spectrum.

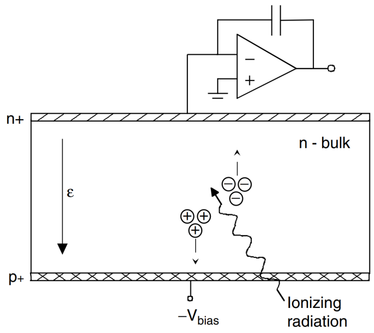


Figure 2.5: Schematic diagram of a Silicon X-Ray Detector with integrated electronics [35].

A basic schematic of a Silicon detector is shown in Figure 2.5. The pn-junction is reversely biased to create a depleted region. Upon interaction with the Si material, the X-Ray photons create electron-hole pairs, which are then separated and driven by the electric field toward the electrodes. This is where the charge is collected, resulting in a pulse-shaped current signal. The pulse is then converted to a step-shaped voltage signal by the subsequent integrating amplifier. The rise in step level is proportional to the charge, and thus to the photon energy.

Noise sources in a detector include statistical noise, noise from the charge multiplication and collection process, and electronic noise. The electronic noise is often expressed in terms of Equivalent Noise Charge (ENC), which is the amount of charge that creates a signal (pulse) whose amplitude equals the RMS value of the noise. In the detector, ENC could be written as [35]:

$$ENC^2 = (C_D + C_G)^2 a \frac{1}{\tau} A_1 + (C_D + C_G)^2 c A_2 + b \tau A_3 \qquad (2.7)$$

where C_D and C_G are detector capacitance and gate-source capacitance of the amplifier FET. τ is the width of the signal pulse. A_1, A_2, and A_3 are coefficients dependent on the shape of the pulse. The coefficients a, b and c represents the following noise components:

(a) Series white noise arising from thermal noise of the FET channel

(b) Parallel white noise arises from the leakage current of the detector and of the FET channel

(c) 1/f series noise arising from charge traps in the channel of the FET

To reduce the noise and improve energy resolution, it is important to keep the detector at a low temperature and reduce detector capacitance.

Figure 2.6 shows a detector assembly. The detector is mounted directly on the cold side of a multi-stage Peltier cooler. The temperature is kept very low (<-20^oC) to minimize the leakage current and reduce the thermal noise of the FET channel. The very low temperature requires both the multi-stage cooler and the complete prevention of heat convection by gas molecules. Thus the detector assembly is encapsulated in an air-tight shell and is evacuated.

A detector with a larger area is desirable so that more photons could be captured. However, in traditional detector shapes, such as in Figure 2.5, a larger area increases the detector capacitance and worsens the noise. Silicon Drift Detector (SDD) alleviates this problem by allocating electrodes with proper shape and bias on the sides of the detector. The resulting potential surface will "guide"

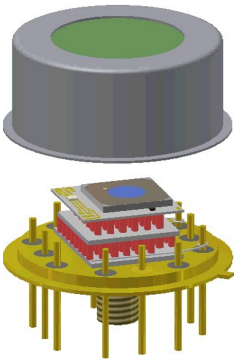


Figure 2.6: CAD Model of a silicon detector module. From top to bottom: Metal shell with a Be-window, Detector chip (blue), Two stages of Peltier cooler, Mounting base with connectors [40].

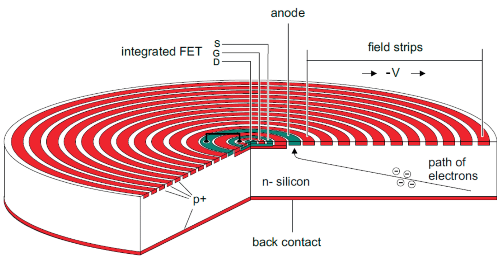


Figure 2.7: Diagram of a Silicon-Drift Detector. The electrode strips (red) are properly biased so that electrons, generated by radiation in the whole volume, will flow toward the small anode at the center [41].

the charges towards the electrodes which have a very small area (and thus small capacitance). The design of an SDD is illustrated in Figure 2.7.

2.1.4 Common Degradation and Failures Mechanisms In X-Ray Tube And Detector

2.1.4.1 In The X-Ray Tube

The most common degradation and failure modes will be presented for each important component of the X-Ray Tube, namely the Filament, Housing, Anode Target, and Beryllium window [42]. The cause of the degradation will also be concisely discussed.

The **Filament** is subjected to very high temperatures, which leads to:

- Thermal Evaporation: Over the long term, the Tungsten material will evaporate and lead to the *Filament Burn-out Failure Mode* (in which the filament would burn open). The degradation gets accelerated when a high filament current is applied, which could happen in High Anode Current, Low Tube Voltage conditions. In this condition, the field enhancement effect is reduced and the filament relies more on increasing temperature to achieve a high electron emission rate. Another important phenomenon is the formation of local hot spots due to imperfections in the structure of the material, or due to manufacturing problems. The material at the hot spot will evaporate faster, pinching the diameter of the filament at that point, increasing the local resistance, and then will lead to a higher local temperature at the hot spot, which finally leads to even faster evaporation.

The glass **Housing** should maintain the high vacuum inside the tube. The *Vacuum Leakage Failure Mode* happens when the vacuum is no longer maintained in the tube, leading first to the obstruction of the emission and the flow of electrons and X-Ray. Then if the contamination density is high enough, there is a conducting path between the Anode and Cathode and the tube stops working. The Housing may fail if there are:

- Damage to the glass: Imperfect manufacturing process may lead to pre-crack, or may leave behind residual stress in the glass material. This stress can cause cracks when the tube is subjected to thermal cycling (repeated operation in high and then low temperatures). Another possibility for damaging the glass tube could arise from arcing. The high voltage between the electrodes, and from the electrodes to copper housing leaves a potential for arcing, which may chip away, or burn a hole open in the glass tube.

- Damage to the joint between glass and metal parts: This can also arise from imperfect manufacturing or from thermal cycling.

The **Anode Target** is where the electron flow impacts and generates X-Ray. It can be damaged over time by:

- Evaporation: High flow of electron bombardment over time evaporates the material away, or leads to deformation of the surface. Another possibility for surface damage is the arcing between the cathode and anode. In both cases, the quality of the X-Ray Beam and the X-Ray spectrum is degraded, but it may not lead to catastrophic failure.

A **Beryllium window** (instead of glass) may be placed on the X-Ray beam path to minimize the loss of X-Ray intensity. The material is made into a very thin foil, which is then mounted to a frame and then to the glass tube. It can fail, which leads to loss of vacuum, in the case of:

- Mechanical damage: This may arise in case of vibration, shock, or mishandling of the device.
- Leakage: The leakage may develop due to thermal cycling, production error, or the high humidity in the environment could corrode the Beryllium material.

There are also failure and degradation mechanism, which includes but are not limited to:

- Contamination of the oil might lead to arcing.

- "Getter" material, which absorbs molecules to maintain the vacuum in the tube, can get exhausted.
- Residue gas molecule can stay in the vacuum after imperfect production process

2.1.4.2 In The Silicon-based X-Ray Detector

For silicon detectors, it is very important to maintain the vacuum inside the capsule and prevent heat convection. Otherwise, the Peltier elements (which have limited power) could not bring down the temperature to the necessary temperature, which leads to a very high noise level and bad measurement quality. There could be either a sudden loss, or a gradual degradation of the vacuum.

Sudden loss of vacuum happens when the capsule is no longer air-tight. The capsule, which is made of metal with a Beryllium window on the beam path, may have a leak following mechanical damage or mishandling. The Beryllium can also get oxidized and corrodes over time, leading eventually to a leak.

Gradual degradation of vacuum arises due to outgassing, in which the material in and around the detector assembly inside the capsule gets evaporated very slowly over time. There is normally "getter" material inside the capsule to absorb those molecules. However, when the getter runs out, the vacuum will gradually worsen. However, if there is no glaring manufacturing issue leaving unwanted materials inside the capsule, this phenomenon happens very slowly over periods of many years.

2.2 Data Engineering and Management

2.2.1 Data Engineering Lifecycle And Architecture

Development of a data-based Predictive Maintenance solution requires the availability of historical healthy-to-failure data from many devices. The provisioning of this data follows the Data Engineering Lifecycle, which comprises the stages of turning untapped raw data into useful end products.

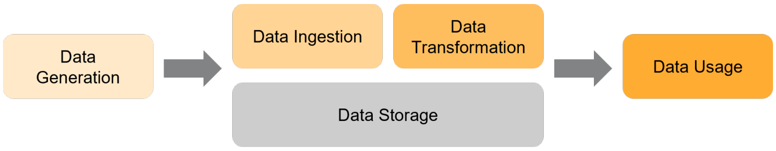


Figure 2.8: The stages of the Data Engineering Lifecycle (Adapted from [43]).

Figure 2.8 illustrates the five stages of the Lifecycle. Each stage performs the following tasks:

- Generation: Data is created by the source systems. These could include IoT devices, machines, or pre-existing data systems (database, SCADA). For the source system, it is important to consider the format, volume, and quality of generated data.
- Ingestion: In this stage, data is brought from the source systems into central storage. The subject of this stage consists of both the hardware aspects (network, connectivity) and the software aspects (protocols, interface) of data transfer. Data could be transferred by streaming (data is transferred as soon as it is generated) or in batch (data is stored locally between intermittent transfers).
- Storage: The storage system serves not only as a central place for all data to reside, but also as a platform to facilitate the usage of data. Thus it must

consider both the nature of data and its usage. This will be discussed in the next subsection.

- Transformation: The raw data must be transformed into a usable format. The tasks include e.g. extracting and formatting to the correct data types (date, numeric, text, etc.), eliminating duplicates, removing erroneous data, extracting useful data features, and so on. After this, the data is ready for long-term storage and usage.
- Usage: The data is finally turned into a useful product. Data could be presented in a dashboard for Business Intelligence. It could be analyzed to find trends and investigates product quality. It could be used to develop a Machine Learning algorithm.

The realization of the five stages of the data engineering life cycle requires an IT/-computing system. The system is often realized with an Edge-Cloud architecture, as demonstrated in Figure 2.9.

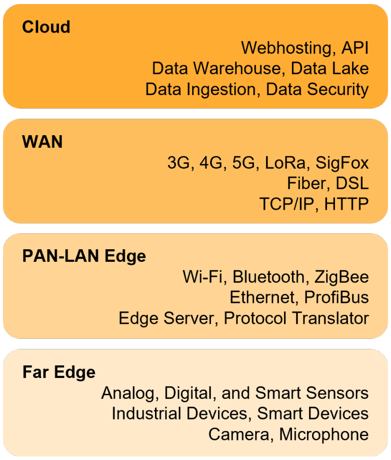

Cloud
Webhosting, API
Data Warehouse, Data Lake
Data Ingestion, Data Security
WAN
3G, 4G, 5G, LoRa, SigFox
Fiber, DSL
TCP/IP, HTTP
PAN-LAN Edge
Wi-Fi, Bluetooth, ZigBee
Ethernet, ProfiBus
Edge Server, Protocol Translator
Far Edge
Analog, Digital, and Smart Sensors
Industrial Devices, Smart Devices
Camera, Microphone

Figure 2.9: Reference architecture for an Edge-Cloud IoT solution (Adapted from [44]).

There are many variants, however, the architecture generally consists of 4 main layers:

- The Far Edge: This comprises non-smart and smart sensors (i.e. sensors with embedded data processing), industrial machinery, video cameras, and so on. The data is generated here.
- The PAN-LAN Edge: This comprises of the local connectivity infrastructure and near-edge computing resources. The connectivity provides a connection from the sensors to the router or gateway and could be realized with multiple technologies such as 802.11 (WiFi), Bluetooth, Zigbee, and so on. The near-edge computing resource facilitates further data processing, temporary data storage, protocol translation, and connection to cloud servers among others.
- The WAN Layer: Data is now transferred out of the local premise toward the cloud server. The transfer usually makes use of the Internet infrastructure, but can also be done via the cellular network, or in extreme cases the physical data medium (hardware) can be carried by a person to the destination.
- The Cloud Layer: The Cloud server provides endpoints for secured data reception, stores data, and facilitates usage of data. The server could be deployed as Public Cloud or Private Cloud. Public cloud services are offered by providers such as Amazon AWS or Microsoft Azure. It offers more flexibility and cost-effectiveness and requires less management and maintenance. In contrast, Private clouds are deployed on the IT infrastructure lying inside, or dedicated to the organization. It offers better data governance and regulatory compliance as well as maintaining data ownership. However, the setup and maintenance require more cost and effort.

2.2.2 Database Management Technologies And Big Data

The wide palette of currently available database management systems could be put into two categories. One of which is the Relational Database Management System (RDBMS) which often used Structured Query Language (SQL) to manage the data. The other is often called Not only SQL (NoSQL). RDBMS is more commonly used for transactional data (e.g. bank transactions), while NoSQL is more suitable for Big Data. The reason for this stems from their difference in the four aspects: data schema, scalability, consistency, and access methods [45].

With regards to those aspects, the **RDBMS** (or SQL database) exhibits the following characteristics:

- **Data schema**: Data is organized into tables. Each table contains one type of data (e.g. employee data, transaction detail). Each data record is a row in the table. Each record stores values in the columns. All records from the same tables have the same column. There is one column that uniquely identifies each record in the table, called Primary Key. A record in one table could refer to a record in another table by its primary key. This normalized and structured representation of data results in a very efficient use of storage space.
- **Scalability**: RDBMS deals with an ever-increasing amount of data by vertical scaling. It means that the server which hosts the database gets better CPUs, RAMs, and more storage capacity.
- **Consistency**: This means that the database is never in an invalid, or ambiguous state. Examples of invalid states could be, one record being inserted referring to another record, while that other record gets deleted at the same time, which might lead to a reference to a non-existing record. RDBMS ensures the ACID properties (Atomicity, Consistency, Isolation, and Durability) to maintain consistency at all times.

- **Access methods**: Basic data operations (CRUD - Create, Read, Update, and Delete) are carried out using SQL. RDBMS also supports "Join" operations, which combine records from two tables to create useful information.

In contrast to RDBMS, **NoSQL** database management system exhibits the following characteristics:

- **Data Schema**: There are no tables. The records are often stored as it is. Records can have a flexible schema with various numbers of fields. It means that each data records also have to store its own structure. Furthermore, as NoSQL traditionally does not support "Join", which means each record should (and can) store other records, so that each record contains enough information to stand on its own. Due to the reasons above, NoSQL stores data less efficiently.
- **Scalability**: NoSQL systems are often deployed on a cluster of machines and can scale horizontally. It means that to deal with more data, new commodity-grade computers are simply added to the cluster. This results in the potential of the NoSQL system to scale very large to handle Big Data.
- **Consistency**: NoSQL cluster is essentially a distributed data store, and thus subjected to the CAP (or Brewer) theorem. The CAP theorem states that the distributed system cannot concurrently ensure all three properties: **Consistency**, **Availability** and **Partition Tolerance** [46]. Availability means the system is always responsive to requests, even if some nodes are failed. Partition Tolerance means the system should still work if the communication between the nodes is severed. Many modern NoSQL systems ensure A and P, and support eventual consistency, which means that the system will return to consistent states after the nodes get synchronized after a breakdown.
- **Access methods**: NoSQL system, as the name implies, support access methods beyond just SQL. They often support scalable analysis of the massive amount of data using the parallel processing power of the cluster. One possibility is the use of the MapReduce framework, in which each

node executes the same code, and only at the end are the results aggregated to the main node.

The NoSQL data management systems are more suitable for handling Big Data. The rising prominence of Big Data comes from the proliferation of increasingly capable digital systems (e.g. the smartphone or social network), as well as the needs to "mine" Big Data for business values. The characteristics of Big Data are often referred to as the 3Vs [47] [48]:

- **Volume**: Big Data normally requires Terabytes, Petabytes, or larger storage.
- **Variety**: Big Data has a large diversity in form and format. The data points may not have a fixed structure that can fit into a row of a table (such as employee detail which normally contains name, ID, and department). Big Data also includes **unstructured data** such as multimedia, log files, or social media posts which can contain anything from text, links, images, audio, videos, and so on. In the context of IoT and Predictive Maintenance, **semi-structured data** often arises from the variety of smart sensors. A single data reading normally has a certain scheme. However, complications arise when there are different types of sensors generating data at different paces, the type of data from each sensor can also change accordingly to external conditions, and the data scheme can evolve with time as sensors get upgraded. The semi-structured data thus does not fit in RDBMS but is often stored in the NoSQL data system under formats such as JavaScript Object Notation (JSON) or eXtensible Markup Language (XML).
- **Velocity**: This refers not only to the huge rate at which Big Data gets created but also the high speed required to process this large amount of data and deliver the results as quickly as possible (and possibly in near real-time). The ever-increasing computational power required for this task is best provided by horizontal scaling in the NoSQL system. In this way, more commodity-level computer just needs to be purchased and add to the cluster and the algorithm execution speed (e.g. MapReduce) will scale.

This is in contrast to vertical scaling in RDBMS, which requires purchasing more expensive hardware for the single server, for which there is a limit.

- Additional Vs could include **Veracity**, which is the trustworthiness of data, or the **Value**, which expresses how relevant or informative the data was.

2.3 Failure Prediction

2.3.1 Overview

The central task of a predictive maintenance system is to produce failure prediction. Based on this information, the PM system can perform other tasks such as presenting the information to the user, or maintenance policy optimization (i.e. determination of the best time point for maintenance to optimize the cost-risk trade-off) [49]. There are different categories of technical methods for failure prediction, which include but are not limited to [50]:

- Survival analysis methods
- Anomaly-based methods
- Time-series methods

Survival analysis methods aim at producing the survival (or failure) probability of the system as a function of time. One possibility for the realization is, that the failure **probability** could be produced from the observed failure **statistics**. In this, all the failures of the device together with the failure time are recorded. From this histogram, a model could be applied to produce a continuous failure probability density function (pdf) as a function of time. Commonly used models in the field of Reliability Engineering include the Weibull model [51] and the Bathtub curve. The Bathtub curve is illustrated in Figure 2.10.

Another realization possibility is to adjust the survival probability based on the co-occurrence of other factors. This is commonly used in health care fields,

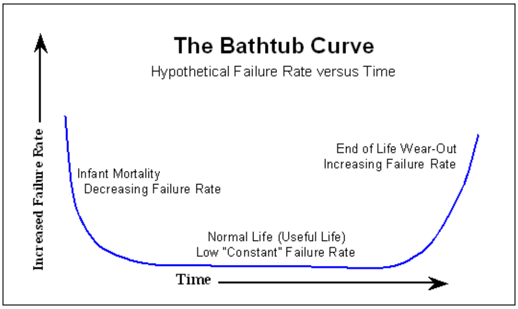


Figure 2.10: The eminent Bathtub curve from the field of Reliability Engineering shows the failure probability at various life stages of a product [52].

whereby the survival rate of a patient is determined based on covariates such as age, treatment methods, treatment plan, smoking and exercise habits, and so on. Common used methods include Kaplan-Meier method [53], Cox proportional hazard method [54] or Bayesian method [55].

Anomaly-based methods monitors the data and compares the current data with the "normal" data as expected from a healthy system. The failure probability is proportional to the deviation between the observation and the normal baseline. There are different possibilities for establishing this normal baseline:

- Expert-defined rules: the normal status is defined based on the technical specifications of the system or based on the experience of subject matter experts.
- System model: a mathematical-physical model of the system could be constructed to take in the current data and user inputs and produced the observable data and output.
- Deep learning model: Techniques such as autoencoder and variational autoencoder could be used to establish baseline behavior based on observation of many normal data points [56]. Each data point includes all the input, output, and data of the system. These models convert the data point to an internal (and often more concise) representation, from which the data

point could be reconstructed. All the internal representations of all normal data points define together a statistical distribution. The model could then be used to produce representations of new data points, which will then be compared against the known distribution to determine if it deviates from normal behavior.

Time-series methods observe the evolution of data points through time, extract the trends and characteristics from it, and predict the failure in the future. Possibilities for the realization of time-series methods are discussed next.

2.3.2 Time-Series Methods

The input to the time series methods is the data points z_t consisting of a time stamp t and the corresponding sensor readings, user inputs and outputs, and other data. From which the output y could be found by constructing the probability function $P(y|z_{t=t_0..t_{now}})$ or by finding the direct mapping $y = F(z_{t=t_0..t_{now}})$. The inclusion of timestamps as a part of the inputs z_t is important, as the output of the method is also a time, or time-relevant quantity y, such as the failure time point, the remaining useful lifetime (RUL), or the data points at the next time stamps.

Direct time series method aims at producing directly the final output such as failure time point or RUL. Currently, deep learning is used extensively for this purpose. A deep learning model consists of multiple layers, the output of one layer is the input to the next. Each layer is composed of weights $\boldsymbol{w}$, biases $\boldsymbol{b}$, and an activation function $\Phi(.)$ which provides non-linearities to the layer. One simple method for producing output in each layer is $y_{out} = \Phi(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x} + \boldsymbol{b})$. The deep learning model is trained by putting many training inputs x_{train} into the model to produce output y_{model}. The weighs and biases are then adjusted based on the Stochastic Gradient Descent algorithm to minimize the loss $L = (y_{train} - y_{model})^2$ between the true training output y_{train} and that produced by the model y_{model}. According to the Universal Function Approximation Theorem, the deep learning network could approximate any function $F(.)$, as long as the network has a large enough amount of parameters, and there is enough training data [57]. For the time series inputs, deep learning architectures such as Convolutional Neural Network [58], Recurrent Neural Network [59], and its variant Long Short-Term Memory [60] are commonly used.

Indirect time series method successively predicts (or forecasts) the future data points based on the observed points, from which the failure time point could be found when the future (predicted) data point crosses a certain failure condition. This type of time series forecasting has widespread applications such as in weather, climate, economic factor, or stock market forecasting, among others [61]. Popular realization possibilities include direct curve-fitting on the data (possibly with exponential smoothing of past data points), Autoregressive Integrated Moving Average (ARIMA) and its variants [62], or deep learning models [63].

3 Conception

The domain analysis in Section 1.2 showed that a Predictive Maintenance solution offered by the SME manufacturer to the customer would address pain points and deliver benefits to both the end customer and the service team. This chapter would present an approach for the creation of such a Predictive Maintenance solution for XRF devices of this SME manufacturer. For the starting point, the current situation regarding the XRF device and the relevant infrastructure is presented. Requirements related to the creation of the solution, and to the solution itself, are then presented. After that, relevant works which address the topic of XRF Predictive Maintenance will be analyzed with regard to the aforementioned requirements. Finally, our approach, which addresses the requirements, will be presented in various aspects. Each aspect would address one stage of bringing about a Predictive Maintenance solution.

3.1 Situation Analysis

This section describes the starting point of the initiative toward predictive maintenance. A short description of the XRF devices and relevant resources will be covered.

The manufacturer delivers to the customer a setup consisting of the XRF device itself and an accompanying PC. Key components of the XRF device are the X-Ray Generator (also called "X-Ray Tube" due to the glass tube inside the copper housing) and the X-Ray Detector. Other components include the main board, the high-voltage board, housing, and other miscellaneous supporting components.

The XRF device receives command and measurement parameters (high voltage, anode current, and measurement time) from the PC. The boards will control the components and deliver a resulting spectrum back to the PC. On the PC, the main measurement software, which did send the command, receives the spectrum and conducts relevant calculations to deliver the measurement results.

The X-Ray Generator and Detectors are the most relevant part of Predictive Maintenance. They are the most expensive components of the device. Once they are broken down, the repair process is also very time-consuming as extensive re-calibration needs to be done to deliver accurate measurement results. Both components require a high vacuum for the operation, which means they are more sensitive to hostile operation environments than other components. Besides that, the X-Ray Tube is essentially subject to constant wear and tear. This is due to the high temperature which makes the emitting filament in the cathode evaporate over time.

Directly monitoring the X-Ray Generator by installing sensors in the vicinity of the X-ray glass tube was technically challenging. There is a high level of radiation that damages electronics over time. The space inside the copper housing is also very tight for any sophisticated sensor. Finally, there is a very high voltage (10-50kV) across the X-Ray tubes, which could potentially destroy the sensor. It is also not economical, until now, to develop or install a very specialized sensor to monitor the tubes.

Predictive Maintenance relies on observing trends from the data collected over a long term on the component to produce a prediction. Data collection is also a key part of the development process. The XRF setup mentioned above, however, does not collect the data. Fortunately, some key parameters related to the function of components are monitored by the main board and viewable from the main software. There is limited communication (a few bits) about errors between the device mainboard and the PC main software. The error communication is often "too little, too late", happening after the irreversible damage is done.

The absence of data collection also leads to some additional issues. There is actually no long-term monitoring of data that will assist the investigation in case some widespread quality issues occurred. In the case of the SME manufacturer, there was a series of premature device failures with serious financial and reputation damage. The problems were eventually figured out. Nevertheless, the investigation and damage control could have been done more efficiently, had a data collection system been in place.

Instead of long-term data collection, once a failure occurs, the service technician can access the "service menu" in the software and conduct a series of standardized special measurements. The results of this measurement help to determine the stability and functionality of the device at that moment in time. It is still to be determined how much the results of this special measurement could be an indicator of device status over time. Besides, there are also two problems with this approach (for Predictive Maintenance). First, the data is gathered **after** a failure may not be similar at all to the data **before** and leading to the failure. Secondly, in order to gather data using this special measurement, the customer has to actively take part in it (at least they have to place the physical measurement probe in the device). Thus regularly sampled data might not be possible to collect.

To support the customer, the manufacturer has a network of service offices in different countries. The service teams in the office have direct contact with the customers and could access many of the devices for inspection or repair. For that purpose, the support of the global service team is a valuable resource to be leveraged to support the collection of data.

The IT infrastructure is critical in supporting both the development and the operation of a Predictive Maintenance solution. On the side of the manufacturer, the (internal) IT department offers comprehensive support for global business activities. Internal, secured transfers of a large amount of data are possible. However, on the side of the customer, the IT landscape is diverse and disparate. Some customers connect the XRF PC only to the internal network. Some enable remote troubleshooting over the Internet. Others block the PC from any kind of network. For all customers, the protection of data and trade secrets is always a top priority.

3.2 Requirements and Analysis of the State of The Art

Starting from the situation described in the last section, the SME XRF manufacturer would like to develop a Predictive Maintenance solution with a data-based approach. Data from multiple machines should be collected. Based on this a model should be developed from the data to produce prediction. This model would then be integrated into XRF products and offered to the customer. The relevant requirements discussed in this chapter would be put into two categories. Requirements in the first category are relevant to the predictive maintenance solution itself. In other words, they specify what the solution should be able to perform, once it was developed. The second category of requirements is related to the development process. The majority of which comes from the data collection operation.

3.2.1 Requirements for the solution

The following requirements for the resulting Predictive Maintenance Solution could be categorized into functional and non-functional requirements.

Functional requirements specify what the product must do, or its intended function. In this case, those include:

- Data collection and processing in near real-time: Data related to the components, and from other auxiliary data sources must be collected and stored. The data analysis should be conducted as soon as enough data is available. Near real-time reaction (in order of hours) is sufficient as the degradation process happens over a longer period of time, and the XRF device is not safety critical.
- Failure Prediction of Key Component: From the data, predictions related to the failure time-point of the X-Ray Generator and Detector must be produced.

- Notification: The solution must convey the health status and possible failure of components to the end customers and service team.

Non-functional requirements specify how well the product should perform the intended functions. These include:

- Minimal disturbance to the main software: When the solution performs its function, the functionality, performance, and accuracy of the main measurement software must not be affected.
- Early Prediction Horizon: The failure prediction must be delivered early enough before the failure so that the customer could inform the service team and plan the production around possible disruption. The service team must be informed before the planning period (usually a week) so that a service visit could be planned in the buffer time in the work week.
- Accurate Prediction: The failure prediction should predict the failure point as accurately as possible, especially within the prediction horizon.
- Minimal additional cost: The final solution must be economical and the cost should be justifiable for all customer sizes. Even customers with only one device should be able to get the benefit of the solution, as those customers perceive the highest impact of failure.
- Adaptive to IT infrastructure: The solution should be able to function for many types of customers with heterogeneous IT landscapes. In short, wherever the main software could function, the solution also should be able to function.

3.2.2 Requirements for the Development process

The requirements for the development process were derived from the requirements for the solution. For the functional requirements, the development process should do the following:

- Large-scale data collection: During the development process, a large amount of data from various internal and external sources must be collected and evaluated.
- Validation of prediction algorithm on real data: The algorithm should be validated on data sets collected from real physical systems. Input data may be collected in real-time or pre-recorded and fed step-by-step into the system.
- Continuous improvement of prediction result: The development process and infrastructure should not only formulate recommendations on the further improvement of the prediction algorithm, but it should also provide the necessary instructions and resources so that the improvement can take place.
- Integration with the product: The development process should formulate a plan for integrating the resulting software part (data processing and prediction algorithms) and possible hardware parts with the XRF product.

Non-functional requirements:

- Minimal- or Non-disturbance to measurement activities: Data collection for the development should not disturb the day-to-day activities on the machine. The performance and accuracy of the performance must not be affected.
- Use of appropriate method for prediction algorithm: The algorithm should be developed with the most suitable method (including state-of-the-art methods) to deliver the best possible result with given data while maintaining the capability for generalization for unseen data.
- Data privacy protection: The data collected should be handled in a way that addresses the privacy worries of the customers. For example, the data should not be stored on a third-party server. There should be also guidelines for clear communication between the manufacturer and the customer about the use and handling of data.

3.2.3 Analysis on the State of The Art

In this chapter, the situation mentioned in Section 3.1 and the State-of-the-art in the relevant academic field are analyzed with regards to the requirements from Section 3.2.1. It would be clear that there is a massive deficit in data quantity and quality for the development of a prediction approach. In addition, the few previous attempts at failure prediction did not sufficiently address the requirements provided by the domain.

The long-term monitoring data of the X-ray components are **not** available, either internally from the manufacturer, or externally from the academics. From the situation summary in Section 3.1, it is clear that there had been neither the facility nor previous attempts at monitoring the long-term component data. The results based on special measurements after the failure often represent only a snapshot of the component status and a post-failure snapshot for that matter. Thus, it would be of limited usefulness for the deduction of possible data before the failure. With the absence of an internal data source, one possibility is to look at the academics for domain-relevant datasets. There are many useful data sources for the predictive maintenance of multiple systems available from many organizations, such as the jet engine dataset or the battery dataset from NASA [64]. For the domain of X-ray tube failure, there is so far not yet a relevant dataset.

Not only does the quantity but there is a lack of quality of the collectible data. As would be discussed in Chapter 4, the data, monitored by the XRF device control boards, is possible to be collected with some additional software implementations. However, the hardware which samples those data on the current generation of XRF devices, namely the onboard ADC, has only an 8-bit resolution. As the trajectory of the signal is expected to be exponential, the low bit resolution would limit the prediction horizon of the algorithm.

There have been a few works that addressed the topic of failure prediction for the X-Ray Tubes. They would be quickly summarized and discussed next.

The US Patent US6453009 presented a method for prediction of x-ray tube failure based on collected data [65]. The method covers the vacuum leakage failure mode, in which particulate matter enters the vacuum space between the anode and cathode. Those molecules create irregularities in the anode current, which is picked up by specialized circuits. The frequency of current irregularities would be compared with the baseline statistics to determine the health status of the tube. The approach requires additional hardware, which puts additional costs on deploying the solution. It also covers only one of the failure mechanisms of the tube. The other failure mechanism related to the evaporation of the filament, which is arguably more important, is not covered by this approach but is the subject matter of another work.

The work by Ma 2016 presents a method for predicting filament failure based on how much it was used [66]. Essentially, it is a filament heated under a certain current, the heated duration is divided by the expected lifetime of the filament under the condition of constant heating current to yield a parameter representing the wear and tear. This parameter is accumulated over time and the approach tries to predict the increasing trend and when it would cross a threshold to determine failure. However, due to the limited data available, it had not been possible for the author to determine this threshold. For this approach, the threshold is actually absolutely necessary. The way the wear and tear parameter is calculated means that the variances between different filaments would result in very different threshold values for this parameter. The method also made use of only input variables instead of a combination of input and output variables. Input variables are the current and voltage which are set and controlled by the operator and are kept at the input level regardless of the health status of the system. Output variables in the context of predictive maintenance are those signals resulting from, or required to keep the input variable at the input level. They can be a good representation of the actual status of the system. The wear and tear parameters in this method were calculated only from input parameters, which leads to some disadvantages. First, the input parameters could not be used to determine the failure threshold, as it is inherently system-dependent and require system-specific data (such as output variable values). Secondly, the wear-and-tear parameter actually represents the risks of system failure and not the actual status of the system. We can draw an analysis from the field of medical survival analysis. This is similar to determining the life duration of a patient only based on what they consume and how much they exercise, but not considering their genetic difference or blood pressure level.

The work by Nalbantov presents a predictive maintenance system for X-ray-based airport security scanners [67]. The approach collects machine logs and employs an external data logger to collect environment and system data. AI pattern recognition was employed on the machine logs to provide health status and prediction. The results are presented in a central dashboard. However, the article provided only very superficial information about the methodology and did not mention any performance metrics for reference.

In summary, there is a need for a predictive maintenance system for the X-Ray Tube and Detector, which directly use the input and output (currents and voltages) data of the components to produce the failure prediction. The approach should be economical and integrable to the hardware and software of the XRF devices.

3.3 Approach

The goal of the dissertation is to devise a plan and set into motion the first steps toward a data-based predictive maintenance solution for the XRF manufacturer and similar SMEs. In a data-based Predictive Maintenance solution, data from multiple devices from a healthy state until failure is collected. Based on this an algorithm is developed that essentially "learns" the trends and underlying models of data during the degradation process. After being deployed to a separate device, the predictive maintenance solution should collect and process the data. The algorithm would then take this data as input to produce the output, which is the failure prediction. The results would then be presented to the customers and service so that action could be taken.

Following the requirements for the development process in 3.2.2, the work in this dissertation was divided into 4 work packages (denoted WP1 to WP4). Each work package would address a research question:

- What infrastructure and process are needed to collect the required data for prediction?
- In case of lower quality of data due to limitations in the current XRF device, what technical improvements could be done to enhance the prediction quality?
- Given the data, is it possible to produce a failure prediction, and using which method?
- What should be done to integrate the predictive maintenance solution into the product and bring concrete features and benefits to customers?

In answering those questions, the work packages should aim at developing an end product that satisfies the requirements from Section 3.2.1. The development process itself should also adhere to a set of requirements from Section 3.2.2.

Given the limited time frame of the dissertation work, the practical constraints of deploying a data collection campaign, and the random nature of failure, we did not aim to present a complete Predictive Maintenance solution after a single dissertation. However, with all the preparation work done, the architectural design finished, and the demonstration of feasibility, further future work could be done to fully realize the solution.

3.3.1 Data Collection

The goal of the data collection work package is to collect as much full history-run data as possible. Full history runs are data starting when the XRF device is "healthy" up to the point of failure. The availability of many history runs would aim to sufficiently address the multiple failure modes and learn the underlying model for the device-to-device variances.

Clearly, to increase the chance of getting failure data, the data collection should be conducted on as many devices as possible. For that, the collection would be conducted from not only devices from our own laboratory but also other internal laboratories and showrooms. In addition, a data collection campaign would be conducted on the customers' devices.

Data collection from customers shall be done with the help of service teams from global branches. Discussion rounds should be conducted on the plenum to present the goal of the data collection campaign; the plenum would suggest and decide on a high-level strategy for deploying data collection capability and collecting data back. Based on that, individual discussions should be done with each country's representative so that a tailored plan is made for each country according to local data laws, customer privacy perception, methods of collecting data, timing issues, and other specifics.

To facilitate the data collection campaign, an architecture for the data collection system should be developed, which should address the generation, transfer, and central storage of data. In the architecture, the current XRF device should be enhanced with data collection capability. The possibility of using a data logger for additional data collection should be evaluated. The architecture should devise a method and infrastructure, with which the data from a multitude of customers and laboratories could flow back to the XRF manufacturer for storage and analysis. For the central gathering and analysis, the architecture should provide the capability for ingestion, pre-processing, and storage of a high volume of high-variety data. The needs of different internal stakeholders regarding the different uses of a centralized data platform should be considered. State-of-the-art Big Data storage technologies should be compared and the appropriate one should be selected.

The architecture should be conceptualized to be general enough so that it can be used by other manufacturers with similar endeavors. However, it should also be customized and adapted to the exact situation regarding the customer and service workflow of the XRF manufacturer. The exact type of data, which could be and should be collected, shall be investigated with the help of internal experts. A partial implementation of the architecture should be done to get the data collection campaign started. The enhancement of the XRF device for data collection should be done in cooperation with the software department. The deployment of the server and other data transfer architecture should be done in cooperation with the internal IT department.

3.3.2 Additional Data Generation For Improved Prediction

The goal of this work package is to evaluate technical methods to extend the prediction horizon of the failure of the X-Ray Tube beyond what is currently possible. The current prediction horizon is limited by the 8-bit ADC on the control board of the XRF devices, which leads to a low resolution of the current and voltage value. In addition, the key parameter which is the resistance of the filament is not measurable due to the absence of a signal for the voltage across the filament.

For improvement, the key signals should be measured with 16-bit resolution. Additional signals for the filament voltage should be measured. A high-speed data sampling rate should also be experimented with to explore transitory behaviors in current and voltages and their relevance to the tube's health. The result of the work package would be a proof-of-concept, from which a recommendation could be formulated about the integration of better signal sampling hardware into the XRF devices in the future.

First, the impact of replacing 8-bit measurement with 16-bit measurement is evaluated. Ideally, this would be done on an XRF device with both measurement setups and the result would be evaluated after the failure point. However, the failure would take a long time to happen (actually the device has not failed at the time of writing). As a result, the comparison of the measurement setups should be done on the simulated signal. For this, a simulation program should be developed, which simulates the filament evaporation failure as physically-correct as possible. The program should generate a signal of the types which is measurable by both the setup. For the sanity check, the signal should resemble the available fragments of collected data from real XRF devices. Once the simulated signal is generated, the improvement effect of the increase in bit-depth should be evaluated by how much the theoretical prediction horizon could be extended.

Once a possible improvement has been confirmed by the simulation analysis, a proof-of-concept setup should be built into a real XRF device. A high-precision, high-speed external measurement logger should be installed. The measurement points on the control board should be selected in consultation with internal experts. The realization of the hardware setup should be done with the help of the hardware team of the manufacturer because radiation protection should be maintained for the device. The software should be constructed to perform signal processing and extract important statistics of the signal before writing results to local storage. The goal is to capture sufficient health indicators without using a massive amount of storage. The output data is then put into the data server, resulting from the data collection work package, to be evaluated.

3.3.3 Prediction Algorithm

The main focus of this work package is the analysis of data and the development of the prediction algorithm. For which the availability of failure data is a prerequisite. Instead of waiting for enough data to become available to apply the desirable approach of Machine Learning, we opted for a strategy, which could deliver some prediction with the limited amount of starting data. At the same time, the strategy should also leave doors open for performance improvement as more data comes in.

Essentially, in the strategy, the work is divided into 3 stages depending on the increasing amount of data:

- Stage 1: only one or two failure data are available. The work concentrates on exploratory data analysis and producing a proof of concept, showing that failure prediction is possible. Prediction performance (i.e. accuracy) is not yet a top priority. Nevertheless, generality (the ability to produce prediction with unseen data) should be incorporated into the design.

- Stage 2: A dozen of data points are available, but the data is not yet comprehensive enough for all failure modes, conditions, and device variability. The work here concentrates on an approach that allows reliable prediction. At the same time, it must have a built-in feature, which enables improvement in performance over time as more data is available

- Stage 3: Comprehensive data is now available. At this stage, the machine learning approach could be used to learn hidden patterns in data and account for variability. The priority is now the high prediction accuracy in many conditions.

Because of the scarcity of data within the time frame of this dissertation, the work concentrates on stages 1 and 2. The work for stage 2 should actually be conducted first. A high-level design for the algorithm should be derived, which is applicable to prediction for both X-Ray Generator and Detector. The design should lay out methods to deliver continuous improvement of measurement results in the context of minimal, but increasing data amount. It should also be shown that the design addresses sufficiently the specificity of generators and detectors (such as the various operating condition of the X-Ray generator resulting in completely different signal levels). Realization options should also be discussed for the main building blocks of the design.

Based on the design, the work in stage 1 could be considered a partial realization of one of the main building blocks of the overall algorithm. In this stage, the vacuum leakage and filament burn-out failure modes, for which a limited amount of data was available, should be considered. Data should be analyzed to identify features or trends which enable failure prediction. Experimental algorithms should be developed to demonstrate the possibility of prediction. Following scientific practice, the algorithms should also be tested on real data from XRF devices or on equivalent data sets if no additional data is available.

3.3.4 Integration to Product

This work package focuses on the practical aspects of the predictive maintenance solution. First, the concrete features offered by the solution should be identified. Then, a proposal should be put forward on how to integrate the data processing and prediction algorithm into the main measurement software in order to realize those features.

The identification of concrete features should be done by presenting and discussing with internal stakeholders. Starting from the proposal of features, a shortlist should be extracted based on the feedback. The feature should include a description and concrete benefits. Related features should be grouped together into a value proposition, which should address the requirements presented in Section 3.2.1. From the features, visual wireframes and mock-ups should be built and demonstrated to the stakeholders to gather additional feedback for the refinement of features.

The predictive maintenance solution should be integrated as a module into the main measurement software. This requires first an understanding of the architecture of that software. The specific requirements for the module are then derived, based on which an architecture of the module is proposed. A detailed design of the important components in the architecture should be provided. Based on this, a partial realization of the software module should be developed and presented internally as proof of concept. A full realization and integration of the prediction algorithm are not in the scope of this dissertation.

4 Data Collection For Predictive Maintenance

The technical work in this chapter focuses on the system and process for the collection of historical data from devices to facilitate the development of predictive maintenance algorithms. Ideally, the obtained dataset should cover the entire history of each device, from its pristine state up to the point of failure. This is in contrast with post-defect data, which only covers the duration after the point of failure. Post-defect data, even though useful for failure analysis, is usually not suitable for failure prediction, since it may be entirely different from the data collected up to the point of failure. Due to the low probability of failures, data collection should be done on as many devices as possible.

The first section covers the architecture of a data system that enables the generation, transfer, and collection of large amounts of data into one internal server, which then acts as a single data warehouse for multiple internal uses. The second section deals with the actual execution of the campaign, including the process, coordination, real-life constraints, and the limiting factors that hinder the full realization of the architecture. It also discusses how certain parts of the data collection system are adapted so that the campaign can be initiated in the short term. The third section covers the partial realization of the architecture, that is, the development of key components of the systems. Finally, the results of the campaign are discussed in terms of the installation base, types of collectible data, as well as the analysis of data from a few captured failure cases.

4.1 Data Collection System

As established by the Situational analysis (Section 3.1), the current device lacked the ability to generate periodical data on component health, nor did it produce any data related to the environment. Besides this aspect of data generation, it is necessary for the proposed data collection system to cover all other aspects of the data collection process, which include data transferring, accumulating, and storage.

In addition to facilitating predictive maintenance, the data collection infrastructure also provides other benefits for internal stakeholders ranging from R&D to operative departments. This "feature wish list", which the data system should fulfill, is the topic of the first subsection. Derived from the features, the requirements should specify the target system. They should also take into account other constraints such as the condition and IT infrastructure of the SMEs which have similar hardware/software constellations. The design requirements for the system are outlined in the second subsection. Finally, the third subsection delves into the proposed architecture and its design choices.

4.1.1 Proposed Features and Benefits

In line with the discussion in Chapter 1 about the benefits of a data system, discussions with internal stakeholders from various departments were conducted. The goal was to define concrete features of the data collection system and their benefits.

As a result, two main features were defined:

- Edge Data Collection:
  - Collection, Processing, and Local Storage for long-term data (on field and laboratory test setup based) on every machine.

  - Transfer of locally-stored data into a global data platform by the service engineer or by the customer via the cloud.

- Central Data Platform:

  - Data is made globally available to internal stakeholders via a Central data platform
  - Each stakeholder can look up data for a component using serial numbers, conduct statistics or carry out deeper studies in specific cases.

The features would bring about the following concrete benefits:

- For component development team: the ability to do a long-term quality investigation of in-house produced components. Based on this, measures to improve quality can be decided
- For Service Team: the ability to perform quick diagnosis of error before visiting customer and thereby save time/cost
- For Quality Management: more convenience in carrying out jobs such as statistics making, root cause analysis
- For business development team: enable subscription pricing model. For which the systems enable the ability to do regular evaluations of the health of the leased machine, and adapt contract terms according to usage

4.1.2 Design Goals and Constraints

In order to realize the features and benefits above, the design of the technical solution for data collection should consider three aspects:

- **Data generation aspect** provides details about the types of data and methods in which data could be generated

- **Data transferring aspect** concerns the transfer of data from where it was generated to a centralized location
- **Data accumulation and storage aspect** should enable the data at the centralized location to be efficiently used for analysis or algorithm development.

Regarding the **data generation aspect**, the following three types of data should be generated:

- The component data of the two most important components, namely the XRG, and Detector: Component data should reflect the health situation of the device and should not be confused with the measurement results of the machine
- The diagnosis data: it is basically statistics over the results of a standardized measurement. Based on in-house experience, these statistics could be used to estimate the stability of the machine at the time the measurements are conducted
- The environment data: Not only could the thermal or acoustic signal around the component reveal the health condition of the component, but they could also directly affect the health status.

The design of the data generation portion of the data collection system is subjected to the following constraints:

- Physical constraints: The realization of the data generation edge should not require deep modification so that it can be widely rolled out. Hardware modification, either by modifying existing electronics, or installing an additional measurement module, is undesirable. The reasons for this include the high cost of purchase, the high labor cost for qualified installation, and the complexity of coordination and training of technicians. Exposed hardware collecting data could also raise a skeptical perception of the customer.
- Constraints imposed by measuring activities: The data collecting operation has to ensure that no ordinary measurement activities are slowed down or

affected in any way, Most importantly, the measurement accuracy has to stay perfectly intact

- Data protection constraint: No measurement data or business/operation-related data, or personal data should be collected. This comes from the general skepticism and fear of data loss as mentioned in Section 1.1.2.3. In the specific case of this SME, many customers use the device to serve customers who are operating in sensitive industries.

The **data transferring aspect** should consider not only the different scenarios in which data is transferred could happen, but also the data privacy and IT security concerns.

Two scenarios of data transfer are often available

- Direct transfer from edge to the cloud: Data collected from the edge is transferred directly to the SME server, often through LAN, WAN, and then the Internet. Streaming techniques are often proposed in the State of the art, but batch transfer should also be supported in case of network unavailability
- Data transfer with the assistance of third-party agent: Data is stored locally until the device comes into contact with a service technician during inspection or repair. Data could then be copied out in batch and transferred to the SME server by the service technician themselves

Ideally, both scenarios should be taken into account in the solution.

Regarding the data privacy concern, in both scenarios, direct point-to-point transfer without data residing on third-party servers is desirable. This means the data should not be uploaded to, or transited via third-party cloud providers e.g. Gdrive, Box, or similar. This is a result of skepticism, which arises from the mass surveillance program of certain governments. Furthermore, the notion of data ownership might be up to interpretation in certain countries, which worries the customer who lets their data out.

The danger of hacking and cyber-attack is always present in any data-transferring method. The threat exists whenever any point of the system is exposed to the internet. Potential damage includes not only the loss of data but the possibility of privilege escalation for the attacker with massive consequences. As a result, the number of exposed points should be minimized and should be hardened (with password, encryption, or similarly) and partially isolated to prevent escalation.

The portion of the system responsible for **data accumulation and storage aspect** should possess the ability to ingress and store the high volume and variety of data that would be generated by multiple machines over many years. For this purpose, the use of NoSQL technologies should be considered. Furthermore, the system should enable:

- the various modes of access to the stored data (individual data readings or in batch, raw data, or processed data)
- the various usage of stakeholders. The internal stakeholders require the ability to speedily conduct analysis and create visualizations with an easy-to-use user interface. However, it is not desirable when these activities require the user to write code or script extensively or have to research too much into technical aspects.)

4.1.3 Proposed Architecture

The proposed architecture for the data collection system is presented in Figure 4.1. The architecture followed the classic edge-cloud architecture with a twist that allowed a third agent: the service technician. The next subsections discuss will discuss the detail of the three main parts of the architecture:

- The edge includes the XRF device and the accompanying PC
- The cloud includes the internal servers inside the SME
- The service technician agent as a person together with their PC or laptop

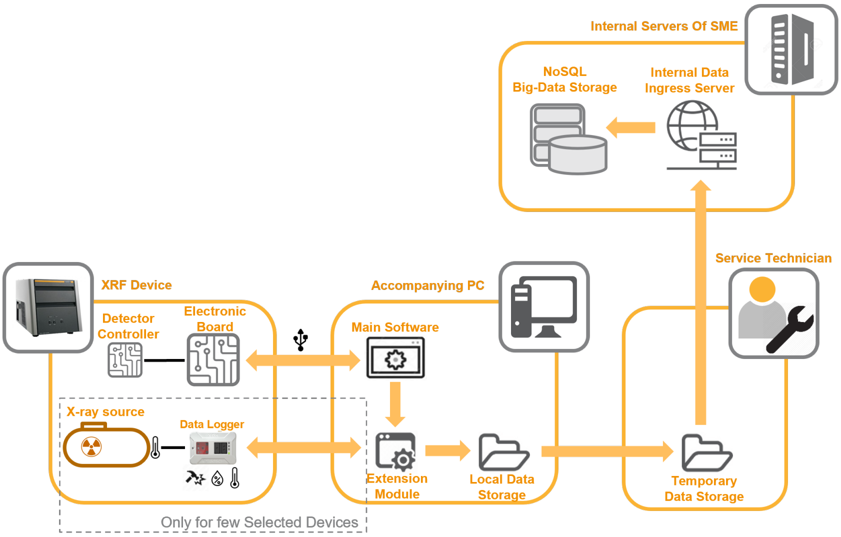


Figure 4.1: The proposed architecture for the Data Collection System.

4.1.3.1 At the edge

The edge is where data is generated. It comprises the device and accompanying PC as the main platform. The device is built up from, among other, electronic boards with firmware which controls and monitors the key components. As mentioned in Section 3.1, the software on the PC controls the device through communication with the electronic board. In designing the edge, the key principle is to reuse as much as possible the capability offered by the platform to satisfy the constraint of minimal invasive modification as mentioned in the last section.

The next paragraphs would discuss how the required three data types will be generated.

A complex device such as a measurement machine generates by themselves already a lot of component-related data. In many cases, the data stays on the firmware level to be used in the e.g. control loop for adjusting the component operation. The accompanying measurement software on the PC also yields data during the operation, such as command logs, operation logs, and other internal diagnosis data. In many cases, the data is transient in nature, i.e. they are not stored. Examples of that are data used as input to the control loop. As a result, both the software and the firmware must be modified to write this data out and not discard them. To realize this, the required modification is kept at a minimal level by developing a software extension module. The module would be discussed in detail later.

On the measurement software on the PC, it is possible to conduct a set of specific measurements with a standardized probe, similar to the calibration. These measurements are conducted over a longer period of time (over many hours). The resulting measurement results and the statistical description of the results, based on in-house experience, can be used to evaluate the health of the device. The measurements are done by placing the probe and selecting a specific mode in the software. The results could be exported by setting up the software accordingly. All of the steps are organized into a shortcut in the software so that the user must only place the probe in and start the process with one click. Nevertheless, this level of comfort still needs some effort to be set up. Besides, these measurements should be conducted a regular intervals for a sensible result.

Depending on the device type, some sensors may be available for monitoring the environmental condition, such as the component temperature, air temperature, vibration, acceleration, etc. In the specific case of the XRF device from the SME, such a sensor does not exist. Thus an additional data logger, with optional extending electronics should be installed to obtain those monitoring data.

The key point is to bring all those existing but disparate data generating mechanism together and make the data persists. The architecture proposes an extension module attached to the main measurement software on the PC. This module:

- offers a unified API that acts as a data sink. Multiple components from different parts of the measurement software can "pour" data into this sink in the "write and forget" fashion. In this way, the modification and additional programming in the measurement software are kept minimal

- is thread-safe as different data source will write data at their own pace and does not (need to) care about synchronization
- takes care of persisting the data. Basically, this means the module takes care of writing and organizing data in the local storage
- handles the communication as well as obtains, and stores data from the external data logger. This operation should be isolated from the implementation of the API above, as API needs to be responsive to the component of the main measurement software which interacts with it
- is lightweight and performant so as to not hinder the normal measurement operation as per requirement
- is well isolated from the measurement software so that does not affect the measurement result, nor the measurement program will not crash if the module does not exist or has errors
- handles the communication with the data server of the SME and streams data securely to the remote server

Behind the scene, this constellation of software and hardware modules at the edge function as follows. Component data, generated in the electronics boards, is read out by a routine in the measurement software. This routine then calls the API of the software extension module and hands over the data to it. Environment data generated by the external data logger is read out by a separate routing inside the software extension module. Both of the data flow is then written to a local storage. If possible, the data flow is also streamed to an external server by another routine inside the software extension module. In this way, the software module acts as a data broker, linking various data sources and data sinks on the edge. The decoupling of sinks and sources in separate processes inside the extension module ensures that the main software would not be hindered during normal measurement operations.

4.1.3.2 On the cloud

The cloud portion of the architecture consists of servers on the premise of the SMEs. All data from all devices will be transferred to this central point, processed, stored, and made ready for use, whether for analysis or for algorithm development. The cloud portion is made up of two logical servers, which could lie on the same or on different physical servers. They are the data ingress server and the big data server.

The data ingress server is the central endpoint for all data being uploaded. It also takes care of pre-processing the data, converting it to the required format, and transferring it to the big data server. This server is the single only part in the architecture whereby an endpoint (e.g. an IP port) must be exposed to the internet. This minimizes the exposed surface to a potential cyberattack. In the case of this server being compromised, the design choice for the separation of the data ingress server from the big data server ensures a level of isolation and impedes further escalation. Nevertheless, it is necessary to harden this exposed endpoint. Possible measures include the use of Secured Socket Layer (SSL) for encryption, requiring authentication and authorization (by password or by certificates), or requiring the use of a secured tunnel (e.g. with a Virtual Private Network - VPN) for access.

The NoSQL Big data server, as the name implies, serves basically as a data warehouse for all the data collected. It also serves as a "data workshop", providing necessary access to all tools for the diverse data usage of the organization. For example, beyond providing basic querying capability, a data server could provide a user interface for data analysis. Users can query for required data, apply filters, and create visualizations, all in one web interface. The capability should be provided to authorized internal stakeholders. Advanced users could of course only use the querying capability from the provided server, and conduct advanced analysis using another tool of their choice.

In summary, the two logical parts of the cloud portion ensure security and provide the capability for diverse data usage. The data ingress server enables secured, direct point-to-point data transfer with lowered attack surface. Data lies only in the hand of the customer and of SME, without residing on a third-party server. The big data server provides a one-stop solution for storage and analytics.

4.1.3.3 The Service Technician Agent

In many cases, no direct connection between the edge to the cloud can be established. In this scenario, a third agent, the service technician is incorporated into the whole system. This enables an elegant solution for the transferring of data between measurement PC and SME server. The agent consists of the service technician, the PC, and possibly additional software and hardware for reading out the data from the customer's PC.

Essentially, when the service technician has the opportunity to come into contact with the device, data is transferred from the measurement PC to the service PC. Data is then uploaded by the technician to the data ingress server. Data is thus transferred in batch and not streaming and thus not suitable for real-time use. However, for the data uses laid out in the Features and Benefits section, real-time-ness is not a core requirement.

For the practical scenarios of data transfer, the proposed inclusion of the service technician offers an elegant solution to the requirement of cyber-security and data privacy. Physically, the service technician agent is close to the devices, but from the IT point of view, they are closer to the SME server (as they are often on the same VPN and has access to many internal IT infrastructure despite being possibly in other countries). This allows possibly never exposing data to the danger of the internet, as the agent can transfer data from the XRF device to their device in the customer network (or with a physical medium) and transfer data to the SME server inside VPN or with secured channels.

4.2 Data Collection Campaign

As mentioned in the last section, the data collection system architecture serves many data needs of internal stakeholders. In the data collection campaign, the system is used specifically to gather historical device data to aid the development of the predictive maintenance system. To become useful for predictive maintenance, the whole data history of a device up to failure should be captured. The failure rate of the device of the SME is relatively low. As a result, in order to capture failure data, the data collection has to be deployed to a large number of devices.

To make this wide deployment possible, the architecture, specifically the edge side, has to be adapted to accommodate the "denomination" of a variety of deployment scenarios. The constraints and corresponding adaptation are the topics of the first subsection. The second subsection presents the concrete plan for the preparation and deployment of the campaign, in which the data collection was rolled out to customers' machines in the field.

4.2.1 Adaptation Of System Architecture To Real-Life Constraints

In this subsection, the constraints, technical and non-technical, during the practical deployment of the architecture will be discussed. These came most likely from inputs from the technicians on the field and stakeholders who have close contact with customers. The corresponding adaptation and concrete implementation choice are also discussed.

4.2.1.1 On the edge

Based on the discussions, there are four constraints for the realization of the data collection edge: transparency, performance, radioactive protection, and preference for passive data generation.

The generated data should stay on the local PC and be really transparent to the customer. It means the customer can read and understand the collected data. There are two reasons for this transparency:

- Customer could know exactly which type of data was collected and confirms that it is not against their consent
- Service technician could diagnose the problem (during repair or inspection) and "prove" their diagnosis to the customer by showing the data easily in plain form

With limited performance and storage place, the PC accompanying the measurement device imposes constraints on the data generation. The data collection computation has to be simple enough. The data sampling frequency has to be not too high, or else too many computational and storage resources would be taken

The device is radioactively protected, which means its housing is made from metal plates with practically minimal openings. which means:

- Any electronic modification or installation of an additional data logger requires the opening of that housing by a certified technician, which leads to a lot of effort
- An internally installed data logger would have a problem wirelessly transferring the data to outside/the PC
- It would require a lot of explaining to the customer when the housing is opened, or when a data logger is dangling around internally or externally

To explain the preference for passive data generation, first, the difference between passive and active data collection should be discussed.

- Passive: Data is generated in the background, invisible and independent from customer measuring activities, without the customers having to actively do anything. Generation of component data and environmental data belongs to this category.

- Active: Data is generated only when the customer manually initiates it. For example, the standardized measurements are started when the customer places the correct probe and presses the start button. In order for the data to be relevant to predictive maintenance, this standardized routine has to be done regularly (e.g. 15 minutes every day). The statistics of the special measurements belong to this category.

It is very hard to instruct and motivate customers on taking up a routine that blocks up their normal daily activity.

The four constraints above lead up to the following decision for the adaptation and the realization of the data collection architecture:

- On the edge, the data collection campaign will concentrate on passive data collection, with the software extension module being at the core. The main measurement software also needs to be updated to send data to this extension module. The extension module will store data as text files on local storage inside the installation path of the main program. Sampling rates for data will be in the order of seconds.
- On very few selected devices, an additional data logger will be installed. The communication and data transfer will be taken care of by the extension software module. The medium for communication will be wired, via a USB connection, and not wireless (despite the availability of the feature).

4.2.1.2 Data transfer and Data server

Data transfer by streaming (i.e. the first scenario mentioned in the requirement section above) is not well received. IT department of customers rarely approves when a program automatically sends data to a server outside the firewall, through the danger of the internet. In fact, a discussion with peers in the industry shows that 95% of their customer base does not want to send any data at all over the internet. As a result, many XRF devices are completely isolated from the internet. Another reason for this isolation is simply the lack of suitable internet infrastructure in some parts of the world where the XRF is used.

On the SME server side, there are considerations for the data ingress server. Namely, exposing a particular IP port to the internet brings high risk, particularly when the exposed surface were not hardened, properly maintained, and regularly patched or updated.

With these difficulties, the architecture was adapted as follows for the realization:

- The service technician agent portion in the architecture will be an indispensable part of the majority of data collection cases.
- Data transfer (in batch) will happen only when the service technician comes into contact with the device. This may happen during yearly inspection or repair.
- Data were then uploaded to a data exchange server using a password credential inside a VPN. The data exchange server uses an existing solution, which was already used in the SME for sending and receiving data from external organizations. The infrastructure for it was already maintained by the IT department so no additional effort is necessary. Nevertheless, the part of the data ingress server, where data extraction and conversion were done before being uploaded to the Big Data server, has to be realized separately.
- In certain countries and for certain retail customers, the emphasis on stringent data security is lower. In this case, batch data transfer via team viewer is possible. The service technician can access the PC remotely, and copy the data. The uploading to the data exchange server was done as in the previous case
- In very rare cases, the customer was informed about the data collection campaign and wanted to upload data themselves. In this case, access to the data exchange server will be granted to them.

4.2.2 Planning and Execution of The Global Campaign

The data collection is conducted on a global scale to maximize the installation base. In the next paragraphs, the method and the plan of execution for the global data collection campaign are presented.

Considering the method, three scenarios for deploying data collection to devices were derived for a variety of customer situations:

- On new device sales: data collection capability was enabled after production, or by a service technician at the office in the country before the device is delivered to the customer
- On existing devices with only software update: the target devices are already located at the customer. To deploy data collection, either the technician has to visit the customer during the yearly inspection, or the devices were sent to countries office for inspection and repair. In this case, only the extension module will be deployed with a software update, no additional hardware will be installed
- On existing devices with additional data logger installed: in comparison to the case above, the hardware module will be deployed on very few devices, this has to be done by a well-trained technician and require high effort

The concrete plan for the preparation and execution of the campaign consists of 5 steps.

- Step 1 - Preparation of the technical infrastructure by a partial realization of the system: More specifically, the minimum viable product, i.e. the critical parts from the data collection system, has to be prepared. This includes the software extension module, the firmware for the data logger module, the modification of the main measurement software, the data extraction program, and the big data server.

- Step 2 - Communication with representatives of service technicians from subsidiaries: This includes the initial communication to determine which countries will participate. A presentation and discussion in a big plenum with country leads followed. Further details were discussed during follow-up individual meetings will all service technicians from specific countries.
- Step 3 - Acquisition and communication with customers: The specificity depends on the specific scenario and specific country. In some countries, all existing devices will be updated with data collection unless the customer opt-out. Self-evidently, customers were informed and the data were shown to them to assure data privacy. In other countries, initial communication has to be conducted with customers and they have to actively opt-in before any installation can happen.
- Step 4 - Preparation of necessary documentation: These include instruction manuals for service technicians for conducting the software update (scenario 2), and for installing the data logger (scenario 3). These manuals also include guidelines for communication with customers. There is also instruction for the preparation and upload of data to the data exchange server. In case the customer wants to do the upload themselves, another instruction manual was also prepared with step-by-step instructions and credentials required for the upload.
- Step 5 - Maintenance of documentation about the installation base: More specifically, the information about the devices with data collection, which typically includes serial number, version, dates of installation, and location, among others.

4.2.3 Data Collection on Laboratory Devices

To maximize the installation base, many devices from the laboratory and exhibition room were also installed with data collection capability. These include available devices from the showrooms, which were not used in the development and validation. Some additional devices from R&D were borrowed on a timely basis to run stress tests, in which specific usage patterns were applied to accelerate the degradation of the device in the hope of catching a failure.

In comparison to data collection on the field, the lab-based collection has the following advantage:

- No communication with the customer is required. No coordination is necessary to collect the data back due to (mostly) the close physical distance between R&D office and the device.
- Specific stress test can be conducted to shorten the amount of time until device failure

However, it has some shortcomings:

- The device in the showrooms are most likely not heavily used
- Specific stress conducted on lab devices test does not fully represent real usage pattern of the device

4.3 Partial Realization Of Data Collection Infrastructure

Under consideration of the requirements and adaptation, important parts of the infrastructure of the data collection system were realized as the first step in the Data Collection Campaign. This realization is only the first version. Besides, this realization was customized to fit the situation of this particular SME. The scale of the realization was small enough to accommodate the trial first phase (for proof of concept). Full realization is recommended as the next step if the SME wants to develop a full commercial Predictive Maintenance solution, which is outside the scope of this work.

The first subsection describes the realization of components on the edge, which include the software extension module and the firmware for the data logger. The second subsection describes how the collected data are extracted and uploaded to the big data server. And the third subsection describes the selection of big data technology and the setup of the big data server.

4.3.1 Data Collection Setup

4.3.1.1 Software Extension Module Realization

The software extension module is the centerpiece of the data collection system on the edge side. As the name suggested, it extends the main measurement software and receives all the data generated by the firmware, software, and data logger and stores them locally, and/or transmits them to the server. As an extension module and not integrated into the main software, it maintains certain independence from the main software, which:

- does not impact the main software performance and stability
- allows easy enable and disable data collection feature
- allows more freedom to integrate more features

Regarding the implementation of the extension modules, it helps to look at the software platform of the main software. It is an x86 application developed for Windows PC. In this Windows environment, a dynamic-linked library (DLL) is most suitable for the realization. In comparison to a static library, the code in the DLL is not integrated into the main executable in compile time. In run time, a DLL can be loaded dynamically, i.e. whenever the code in the DLL is needed and not necessarily when the program starts. The DLL was implemented in C++ for performance and granular control of object and thread lifetime.

As mentioned earlier, the program consists essentially of data sinks and sources. Each of the sink and source was implemented as an Object, which is specially implemented to execute their code in different Threads. This allows a high level of decoupling from each other and from the main program. So that in the event of a crash or unavailability, the other parts of the module would function normally. The data sinks include:

- The receiving endpoint for data from the main measurement program. For this purpose, this data sink would expose APIs. These could be called by the main program, by which the data could be handed over. The required modification to the main program was done by the software team of the manufacturer.
- The receiving endpoint for data from an external data logger. This endpoint takes care of creating and maintaining the connection, as well as the sending of commands to and receiving the data from the data logger.

The data sinks then put the data into a central repository, from which the data could be buffered, and then consumed and written to a local or remote location by a data source [1]. The data sources include:

- A component for managing and writing data to the local storage.
- A possible component for streaming data to a remote location was also planned but not yet realized.

To enable data generation on a device, the main measurement software should be updated to a certain version. After that, the DLL extension is simply copied to the same directory as the main program folder. It will be dynamically loaded (after the next time the program starts) and data is generated. With this method of deploying, data generation could be deployed to customers' devices with the least amount of effort.

[1] The terms "sink" and "source" here are defined relative to the external objects and agents, not relative to the internal repository.

4.3.1.2 Data Logger For Environmental Data Collection

Environmental data such as ambient temperature and temperature of key components can be an indicator for, or key factor affecting the health condition. In the exploration phase for the proof-of-concept, it is desirable to collect as many data types as possible. However, many industrial machineries, including the XRF device in this work, do not include sensors that measure (or can be repurposed to measure) environmental data. As a result, the onboard electronics of the XRF device were enhanced with an additional data logger for this purpose.

Considering the technical situation of the device, and the environment (at the customer or in the laboratory) in which the data will be operated, the data logger should satisfy the following criteria so that it can perform its mission. It should:

- have industrial-grade reliability: the device may undergo various adverse operating conditions, such as high, low, and rapidly changing temperature, shock, diluted radiation, and so on.
- have the capability for extension with other sensors: Extensibility is useful when data should be collected in positions that are not in the direct vicinity of the logger, or when types of data other than those supported by integrated sensors should be collected.
- have permanent internal storage capability: Because the logger is an external part, it may get dislodged erroneously when used. The data should still be collected and stored in this case.
- be programmable: The data logger should be able to interface with many types of sensors. Some onboard data processing is also desirable to save space on local storage.

With these requirements, the data logger model XDK 110 from Bosch was selected [68]. The data logger has a small but rugged form factor and can measure temperature, acceleration, humidity, and light with onboard sensors. The logger is further extended with additional temperature sensors. These were mounted on the component. This allows the temperature of some components in small niches to be measured while the logger is positioned on the housing wall. As for the type of temperature sensor, the negative-temperature-coefficient (NTC) type was selected after being tested against the diode-based types (e.g. LM35). The reason was the better SNR of the NTC (after similar signal processing steps). The NTC was soldered to a simple low-pass filter and made a voltage divider with a reference resistor. The signal was then sampled by an ADC with high internal resistance, which was built into the data logger.

The data logger was programmed to conduct data collection, processing, and storage on the SD card. The firmware is implemented as a program running on RTOS. Upon startup, the program executes a routine to initialize hardware subsystems. The sensors, storage drivers, ADCs, and communication drivers are checked for operation. After that, a number of recurrent threads are started. Each of the two "threads" executes endless loops. One of the loops keeps reading and processing the sensor values after every fixed interval. The details of processing will not be covered in detail here for concise reasons. The data is then written onto the SD storage so that even with mishandling on the users' behalf, the environmental data is still available. The other loop implemented a state machine for maintaining communication with the software extension module to receive commands and send data.

On the field, or in the lab, the data loggers are deployed as add-ons to the data collection edge. On the field, the deployment process was started by first installing the XDK with an additional SD card. Then it is flashed with the binary of the firmware. The device housing should be opened and the logger (together with additional extended sensors) was mounted. It is connected back to the measurement PC with a USB cable. The DLL module would then coordinate all the data transfer and store it in the same place as component data. However, due to the high effort and high cost of deployment of the loggers on the field, the data logger was installed only on a very limited number of devices in the laboratory to investigate certain hypotheses.

4.3.2 Data Extraction

As a result of the transparency constraint discussed in the last section, the generated data is stored in plain text in the local storage. When this data is uploaded, it is not yet ready to be taken in by the big data server. Data must be extracted from data files and converted to the appropriate format. This is the role of the data ingress server.

In this partial realization, due to the smaller scale of the proof of concept, the implementation does not need dedicated computing resources as a server. However, the same functionality was implemented as a script running on a PC in the R&D department. At the moment, the script is still manually triggered. However, it can be easily automated later to run on the server.

Data extraction was done in the script using regular expression. Regular Expressions (regex) is a formal language used for matching patterns in text and capturing data from the matches. The syntax of RegEx is built around a series of characters that define a pattern to be matched in a given string. These characters may include literal text, meta-characters, and quantifiers. An example for a regex pattern is `^Serial\s*=\s*([0-9]+)$`, which will match pattern such as `Serial = 1234` and will extract the value `1234`. Here, the caret (`^`) and dollar sign (`$`) denote the start and end of the string. the token `Serial` will match literally those characters. the meta-character `\s` with quantifier `*` will match any number of free spaces. The match group `[0-9]+` will match one or more consecutive digits. This group was wrapped in parentheses, which indicates that the match value should be "captured" and returned as output. Regex is a powerful tool for text manipulation. But it also requires a solid understanding of the syntax and experience in order to be used in practical applications.

Data extraction is only one of the steps that was performed by the script to ingress the data to the server. For this purpose, the script carries out the following tasks:

1. The directories are scanned for all relevant data files

2. All the files are searched for metadata. These include the serial number, device model, hours under operation, and so on.
3. For each of the data sources, namely data from the main software, data logger, and special measurement, one subroutine will be run. In each routine, the files are scanned line-by-line and matched against a predefined Regex pattern.
4. The output of each sub-routine is stored in data frames. These are then serialized into JSON format.
5. The connection to the Data Aggregation Server is initiated. Serialized data is converted into queries and the data is thus handed over to the server.

4.3.3 Data Aggregation Server

After the data coming from all the devices has been processed, it is then uploaded to the data aggregation server for storage and for further analysis. As mentioned in the requirements section, the data server not only serves the development of predictive maintenance but also acts as a source of device history data for all other relevant departments. These are the department, in which the long-term statistics of a large number of devices are of concern, These include departments such as component development or quality assurance.

The data stored in the server could be considered as Big Data. It has great volume and variety and required high velocity for analysis. The volume comes from a high number of machines in the world over an extended period of time, each has a decent number of data features and sampling rate. The variety depends on the scenario after which the data collection has been set up in the device. Some have additional data types to be collected such as environmental data or measurement data. Some other setups generate completely unique types of data. All this variety and volume should be analyzed at high velocity. The end-user of the data system may have queries to investigate certain aspects of the whole device fleet and wish to get the results in a reasonable amount of time. Most importantly, this analysis feature must be easy and intuitive to use, as end-users do not wish to learn another query language.

The technology for big data management has been growing and becoming more prevalent in the last few years. There have been many well-implemented, matured platforms for big data management. Thus it was not necessary to implement the part from scratch. The first following subsection details the selection of the suitable database according to the design goals laid out in Section 4.1.2. The second subsection presents the necessary steps for resource calculation and deployment of the server.

4.3.3.1 Selection of Big Data Management System (BDMS)

For the data aggregation server, we took a look at state-of-the-art BDMS at the time. Similar to the rest of the Data Collection System, the candidate for BDMS should be able to be deployed completely internally inside the SME IT infrastructure. In this way, the data never lands on or transits through third-party servers. For practical reasons, the system must be also either open source, or there is no associated license fee to use. For these two reasons, the BDMSs which rely upon 3rd party cloud (examples include Azure CosmosDB, Amazon DynamoDB of Google BigTable) or which require a license fee for commercial use (e.g. MongoDB) were not considered.

At the time of the partial realization, the following BDMS were the leading technologies:

- **Cassandra** is an open-source wide-column store NoSQL database. It has a master-less architecture, which means all nodes in the cluster play an equal role. This results in high availability of the cluster as there is no single point of failure. Consistent hashing is used to determine the read or write location of a key-data pair [69]. Thus, Cassandra has an advantage in write performance. However, in read operations, or operations in which data has to be scanned not by primary key, Cassandra has a disadvantage compared to others [70].

- **Elastic** is a search engine based on a document-store NoSQL database. In elastic, the documents (essentially key-value pairs with schema-free value) are indexed. This means not only the key to value mappings were stored, but also the mappings from value content (e.g. word or numerical value) to the key were also computed and stored. This means that the write operations are much slower. However, the read operations and especially the analysis of a large amount of data based on value (and not based on primary key) are very fast.
- **HBase** is an open-source wide-column store that is a part of the Hadoop framework for BigData. HBase uses a master-slave architecture with a focus on consistency. However, availability can be impaired if the master node gets disconnected. Analysis of data can be carried out very efficiently on HBase due to its high read performance and also by using the MapReduce programming model [71].

With regard to the requirements and the intended usage of the BDMS, availability, and consistency are not the most important criteria in this case. The intended central workload of the server is the analysis workload. This will not happen 24/7 and if the server failed, the analysis could wait. Thus availability is an important criterion, but not the deciding one. Similarly, it is unlikely that during analysis work, a lot of reading and writing of the same data points are happening simultaneously over different servers on the cluster. Thus eventual consistency (which is supported by all aforementioned BDMS) is good enough.

The key deciding factor is the efficiency of the analysis workflow. This includes exploring, filtering data, doing simple processing on values, and presenting analysis results. In this regard, Elastic is the most suitable BDMS. The indexing mechanism allows very good reading and filtering performance (which is expected from a search engine). Aggregation operations (such as moving average of values) and transform operations (basically computation scripts) are supported. They are very easy to use with the help of JSON REST queries without the overhead of having to write and configure MapReduce programs. Most importantly, all of the analysis features are available through an intuitive Kibana web interface. The analysis could be done with a drag-and-drop-based interface and the results could very easily be put into graphs for visualization. The result is a very easy to use and yet powerful toolset for the analysis of time-series data without the need to learn additional programming language.

4.3.3.2 Planning and Deployment of Server

In the planning stage, the amount of computation resources necessary for the server was determined. This is followed by the deployment stage, in which the actual work for setting up the server was carried out.

The key concept in the resource planning stage is the trade-off between the speed of access to data and the amount of computational resources required for that speed. An analogy could be drawn from the computer architecture with storage on the on-die CPU cache, RAM, SSD, and HDD. Following that order, each of the storage tiers has faster access speed but is more expensive (and thus more limited in capacity). Similarly, data stored in an Elastic server can reside in the hot, warm, and cold tiers. Data in the hot tier provides the fastest access but requires more RAM to save, among others, indexing data to facilitate that speed. Data in the hot tier is also not compressed. In the lower tier, data is compressed and less RAM per data amount is needed, which results in lower access speed.

To address the trade-off, the concept of data life-cycle management was used. The newly imported data and the most used data are stored in the hot zone with the fastest access. After a certain time, data is moved to the warm zone with compression and lower access speed. Finally, after a number of years, the data could be "retired" and normalized and stored in e.g. an SQL database for very compact long-term storage but the slowest speed for analysis.

Apply the concept of data life-cycle management, planning, and computation of required resources could be made. An example is presented next, which includes the determination of the amount of data, the data life-cycle policy, and the resulting calculations.

Amount of data:

- Number of devices generates data: 400
- Amount of data: 0.5GB per device per year
- Resulting in 200GB per year

Data life-cycle policy:

- Data retention for 1 year in the hot zone for fast analysis
- After which, data is moved to the warm zone and compressed with a 1:4 ratio
- After 10 years, data is moved to the cold zone

A node in the hot zone requires a 30:1 data-to-memory ratio. In the warm zone, the required memory reduces to a 160:1 data-to-memory ratio [72]. Assuming each node in the cluster has 16GB of RAM. The resulting calculation is:

- 200 GB/year $\times$ 2 Replicas $\times$ 1.25 overhead $\div$ 30 Data-to-memory ratio $\div$ 16 GB of RAM/node $=$ 1.04 node(s) in the hot zone
- 200 GB/year $\times$ 10 years $\times$ 2 Replicas $\times$ 1.25 overhead $\div$ 4 Data compression ratio $\div$ 160 Data-to-memory ratio $\div$ 16 GB of RAM/node $=$ 0.48 node(s) in the warm zone

As a result, one can choose to plan 3 nodes in the cluster. 2 nodes in the hot zone (with one node dedicated to running Kibana and 1 node in the warm zone.

The deployment of the BDMS was carried out with the help of the IT department. They provide both computing and networking infrastructure. The computing infrastructure includes a number of servers with the Linux-based operating system. The networking infrastructure includes the connectivity, IP and `iptable` configurations, and domain name for accessing the server. The installation and configuration of the cluster were conducted following the official guide. In the next section, some examples of collected data from failure will be presented.

4.4 Results

4.4.1 Installation Base

During the data collection campaign, the representatives of service branches in many countries have been contacted. Eight foreign branches agreed to join the campaign, 3 from Europe and 5 from Asian countries. In the one-year duration up to the point of writing, a total of 80 devices at the customer site have been installed with data collection capability. In the laboratory and showrooms, there is an additional 20 devices that were installed. The collection back of data happens only if the device has failed, which during the course of the campaign, has not happened. Another data collection possibility was during the yearly inspection, which happens only once per year. Further data sets are projected to be delivered in the following years. Clearly, with the high lifespan expectancy of devices, a data collection campaign for the statistics on the history of devices should be conducted over many years.

4.4.2 Data Collection Table

Regarding the real-life technical and non-technical constraints, a partial realization of the data collection system was carried out. The resulting edge setup could generate a number of types of data. The data types were organized into three main categories:

- Component data was generated by on-board electronics and collected by the software with an extension module

Table 4.1: Types of data that could be generated from a modified XRF device

Component/System Data			
Device Operating Time	Xray Tube Operation Time	Watt x Hours	Filament Current
High Voltage Set Value	High Voltage Measured Value	Anode Current Set Value	Anode Current Measured Value
Detector Set Temperature	Detector Measured Temperature	Detector Cooling Voltage	Detector Cooling Current
			🟨 Implemented in Device
			▨ External Datalogger Required
			🟦 Special Measurement Routine
Device Environment Data			
Xray Tube Temperature	Air Pressure And Humidity	Temperature inside Housing	Vibration And Shock
Measurement Statistics			
Count Rate	Position Of Dominant Peak	Width Of Dominant Peak (FWHM)	

- Environmental data was generated by the additional data logger
- Special Measurement data was generated by carrying out the measurement with a special probe over a long period of time. Results from a few consecutive measurements are combined into blocks and the mean and deviation were calculated for each block. The change in these statistics over time was observed.

Table 4.1 presents the types of collectible data types. Note that many potentially useful signals are not measurable. For example, Vacuum quality is not directly measurable due to the constraint of small space, high radiation, and very high voltage, all of which results in hostile condition for measurement electronics. As another example, the voltage of the filament is not measurable. This means the resistance of the filament, an important indicator for XRG health, is not available. However, a decent failure prediction could still be made based on the filament current alone. This will be discussed in detail in Chapter 6.

Not all XRF setups will generate all three data types. In fact, the majority of data collection devices will only be generating the first category (component data). The reason is that this type of data can be generated passively in the background without the user having to do anything. The required effort for setting up the collection of the first category of data is also minimal by just copying and updating the software. The second category (environmental data), even though also passive data generation, was limited by the high effort for hardware modification, and the high cost of the additional hardware itself. Finally, the third category (special measurement data) requires the user to actively carry out the measurement regularly.

4.4.3 Captured Device Failures

During the course of the data collection campaign, which has been approximately 1 year up to the point of writing, there has been no failure captured at the customers' site. Fortunately, in laboratory devices, it was possible to capture partial history up to failure from 3 devices. All the failures have happened to the X-Ray Generator. The defect components are examined by internal experts and concluded to have come from 2 types of failure:

- Vacuum leakage failure: there was 1 device failure of this type.
- Filament burn-out failure: partial history data up to the point of failure was captured in 2 devices

According to the bathtub failure model, the failure probability of a system is at the highest in the earlier or the late stages of the lifetime. Early failures are most likely due to manufacturing errors. And due to the random nature of the errors which may happen, it is very hard to capture those failures or to induce them in a systematic way. The single available case of vacuum leakage failure fell under this category.

Late failure in the devices may take many years to develop. The device, on which data collection was started, may fail in a few years' time, if not later. The devices, which failed in the time frame of the project, have unfortunately not yet been set up with the data collection software.

To obtain the necessary data for the investigation of the possibility of predicting filament burn-out failure, the failure was induced to happen earlier in 2 laboratory devices. The hardware of the device was modified to "unlock" the anode current which is a lot higher than the allowed maximum. The onboard control electronics had then to increase the filament current to achieve this set value for anode current. This lead to a higher temperature and faster evaporation of material from the filament. Within the limited time frame for which the laboratory devices could be used for this purpose, two failure cases were captured.

Selected data from the Vacuum leakage failure cases are presented in Figure 4.2 and 4.3. At first glance, Figure 4.2 clearly demonstrates the behavior of signals at the point of failure of the XRG. Both signals (and also other not shown signals) show a drastic and catastrophic drop from the level they were before. This means that the post-failure data collection will obtain signals at a completely different level than that up to that point (i.e. pre-failure signal), rendering it not valuable for Predictive Maintenance.

Another observation from Figure 4.2 is that the signal for Anode Current stays at a very stable level right up to the point of failure. Other not shown signals demonstrated the same behavior. There are two reasons for this. First, many of the signals are controlled signals, meaning they are outputs of control loops. This arises from a need to keep the output X-Ray intensity to be constant throughout the operation. Secondly, maybe the gradual changes during the degradation in the signal are too small for the low-resolution ADC to detect.

This leaves out the Filament current as the key signal for predicting the failure. In this failure mode, the filament current signal started to rise less than 20 days (24-hour day) before the failure point and reached a threshold. After that, it drastically decreased similar to other signals. Due to the availability of only one failure run, the exact value, or value distribution of the threshold value could not be concluded. However, the rising behavior of the signal seems to agree with the theoretical expectation of the subject matter expert inside the SME. Thus this trajectory here could be used as a pattern, upon comparison with which the failure in other devices could be predicted.

Finally, Figure 4.3 demonstrated an important characteristic of the filament current. It has different values depending on the operation condition. The operation condition is the Tube Voltage and Anode Current value required for the specific measurement, which the customer wishes to carry out. A higher Filament Current is required to generate a higher Anode Current in a lower Tube Voltage condition. The operation condition is driven by application and not by degradation, meaning it can change very quickly and very frequently during the lifetime of a tube. Every time the condition is changed, it results in discrete jumps in signals. This creates difficulty for failure prediction and should be addressed appropriately.

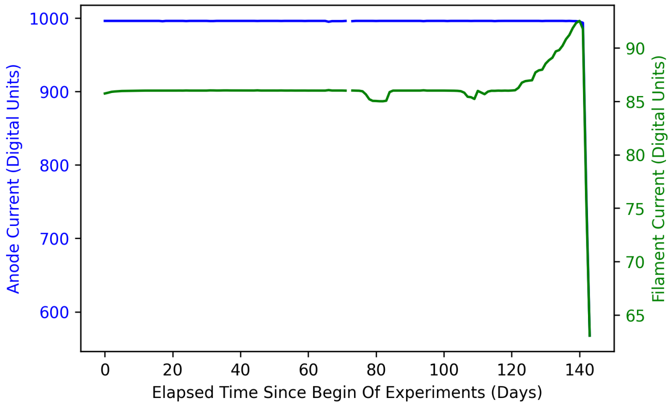


Figure 4.2: Digital values of Anode Current (blue) and Filament Current (green) leading to the failure due to Vacuum Leakage mode. The anode current and many other signals stay the same up to the very point of failure.

The operation condition can be utilized to accelerate the degradation of the XRG and induce the Filament Burnout Failure mode. Two devices in the lab were set up and modified to realize this test. In this, the Tube Voltage was set to the lowest possible value. The device hardware was modified to enable the highest possible anode current without instantaneously evaporating the filament. Clearly, such a setup could only be realized in a lab environment. To "accelerate" the failure even more, the XRGs used in the test were not new ones. They were both taken from customer devices, which were swapped out during inspection, even though both still functioned properly at that time. One tube has already been used for 61.000 working hours, the other one for 53.000 working hours. With this starting point, the devices were put into 24/7 operations for the test.

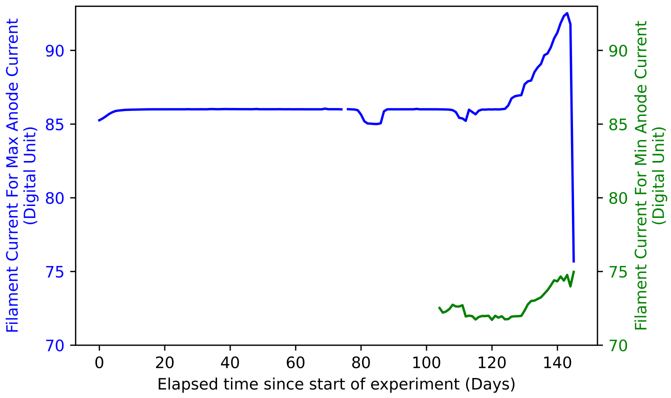


Figure 4.3: Digital value of the Filament Current signals for the different operation conditions of the same failure. The higher curve (blue) is the filament current required to generate the maximum allowed Anode current. The lower curve is for the minimum allowed Anode current.

The test resulted in two failure runs for this failure mode, whose selected data are shown in Figure 4.4 and 4.5. Similar to the last failure mode, other signals except for the filament current stay very stable up to the point of failure, after that there is a drastic breakdown in the levels of all signals.

In contrast to the Vacuum leakage mode, the Filament Current in this failure mode shows a decreasing trend. Furthermore, the trend is not (near) linear but more exponential. This means the majority of the movement in the signal would bundle up at the very end of the lifetime. This delays the time point, after which the failure prediction could be made reliably.

Both the Figures also clearly show the signal with discrete staircase-like jumps between digital levels. The reason for this is the low digital resolution of the signal. The component data is measured by the 8-bit on-board ADC. This complicates the determination of the exact signal value when it is resting at one level of the staircase. Averaging the signal over time would not be very useful for recovering the signal level for two reasons. First, the exponential nature makes it difficult to choose the length of the averaging window. Secondly, averaging filter would not recover the signal if the true signal level does **not** resulting in a proportional transition probability between bits. These points would be further discussed in Section 6.2.2.1.

A final observation is that the two XRGs have a different range of signals. The difference in starting value could be contributed to the different starting working hours. However, the difference in the final values of the two tubes means that there is no fixed threshold for the filament current, at which the filament will break.

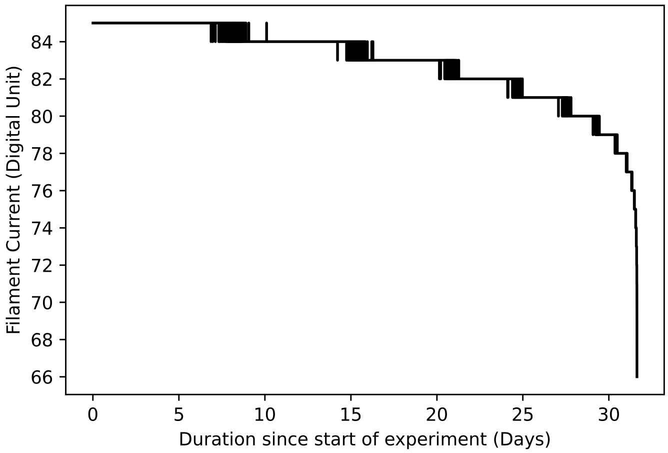


Figure 4.4: Digital value of the Filament current up to the failure due to Filament Burn-out. The filament current is the only signal that showed changes during the lifetime. The data comes from an X-ray tube with 61000 working hours before the experiment started.

From the discussions above, it is evident that the changes in data, which could be used as an indicator for failure, happened only a short time before the failure. This severely limits the time horizon in which the prediction could be made. The late appearance of observable change is suspected to be a result of the low ADC resolution. Furthermore, the range of change in signal due to degradation is much smaller than the range of signal during its normal operation. More clearly, during boot-up, the filament current increases from 0 (digital value) to e.g. 90 after it

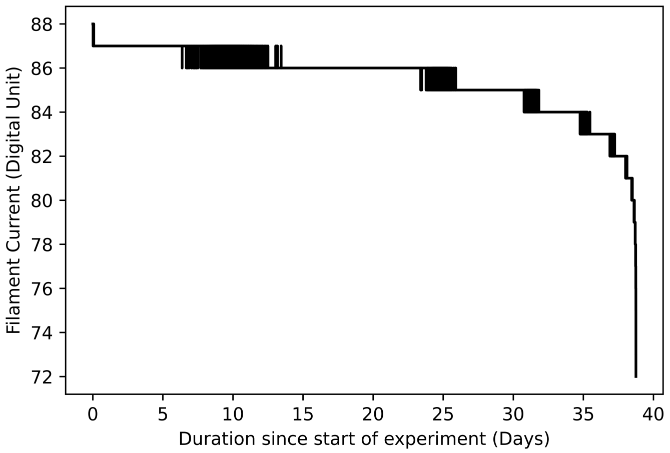


Figure 4.5: Digital value of the Filament current up to the failure due to Filament Burn-out from an X-ray tube with 53000 working hours before the experiment started. Notice the difference in value range between the two cases.

stabilizes. Once degradation happens, this stabilization level of the signal may decrease or increase to e.g. 95. The signal conditioning electronics must be designed to capture the whole signal range for controlling purposes. This means the range of change, due to degradation alone, gets captured with effectively much less than 8 bits.

It was expected (and demonstrated from the Filament burnout cases as well as in chapters 5 and 6) that the signals have an exponential trajectory. This means the signal change rate is very small for the majority of the lifetime. This minuscule change was not able to be recorded with the very low bit resolution. Furthermore, the key indicator of the status of the filament, namely its resistance, was not available due to the lack of onboard capability to capture filament voltage. All of those deficiencies would be addressed in the next chapter. In that, the measurement of component data was enhanced above what had currently been available, namely with a high-resolution data logger. It would be shown theoretically and experimentally that a prediction (at least for the filament burnout failure) could be made with a much longer prediction horizon.

A detailed discussion with physical explanations of the signal behavior would be carried out in Section 6.2.2.1 when the signal is used for developing the prediction algorithm.

4.5 Conclusion and Next Steps

This chapter presents the technical and non-technical aspects of starting a Data Collection Campaign. They represent the first steps toward building a data-based Predictive Maintenance solution.

The Data Collection System was discussed first. It is the technical system for the generation, collection, management, and facilitation of data usage. Such a system is not only useful for Predictive Maintenance development but also for the diverse use of Data and Big Data in the organization. Based on the needs of internal stakeholders, an architecture for the system was prepared. The architecture aims at upgrading the fleet of existing and new devices with the ability to generate data useful for the aforementioned purpose. The proposed architecture also oversees the method for transferring this data back to the internal server of the manufacturer. Direct data streaming is possible. However, batch transfer with the help of the Service Technician offers an elegant solution that addresses many security concerns. In the architecture, all collected data ends up in a central server inside the premise of the manufacturer. This server would ingress, store, and enable analysis of the data.

To collect the history run (i.e. data from healthy point up to failure) of as many devices as possible, a Global Data Collection Campaign was conceptualized and executed. This campaign aims to collect data from devices at customers. The goal was to maximize the installation based to increase the probability of "catching" a failure case. The Data Collection System was partially realized to enable the campaign. Extension systems to both software and hardware were implemented to enable data generation. The programs and the server to facilitate the data flow between the edge to the central storage server were implemented. The central Big

Data server was set up. This partial implementation follows an adapted version of the architecture, which enable the campaign to be deployed on many customers despite the multitude of their situations and concerns.

Parallel to the campaign, some devices in the laboratory were set up with tests. In these tests, the machines were put into special operation conditions to induce the failure to happen earlier. Even though the data collection at the customer did not deliver a failure history run yet, the lab-based data collection has resulted in 3 failures of two different failure modes.

Inspection of the failure data revealed that the majority of the signals stay at a fixed level up to the very point of failure. They would not be very useful for predictive maintenance. On the other hand, the Filament current was found to be the key signal which has shown changes leading to the failure. This means that the prediction of failure was possible from the collectible data types. However, the signal was sampled with an 8-bit ADC, which leads to many difficulties such as a decreased prediction horizon and problems in determining the true signal level for failure prediction. These topics will be addressed in the upcoming chapters.

5 Additional Data Collection and Generation

The collected data from device failures in the last chapter has demonstrated the difficulty in producing predictions based on 8-bit data.[1].The absence of changes in signal level due to low bit resolution made predictions become available only starting at a very late stage of the life of the devices. Considering the required reaction time for the service department to carry out the procedures, the device time-out may not be completely avoidable. We thus explore the idea of the possibility of having the signal sampled at a much higher resolution and investigate the effect, as well as the implementation of the idea.

Among the components of the XRF device, the theoretical and experimental work in this chapter concentrates first on the X-Ray Tube for the following reasons:

- Beside the detectors, the tube is the most expensive component in the XRF device
- From the experience of the SME, the number of service cases related to the tube is much higher, thus a priority on the tube would create the highest financial impact.
- Detector failure mechanisms are very much random or related to production issues. Examples are vacuum leaks due to puncture or production errors.

[1] Essentially, if no change could be detected in data, the prediction could still be done based on the statistics over the fleet of devices, taking into account the specificity of the way the device was used. Nevertheless, statistics over the fleet of devices were also missing.

> While the tube has failure mechanisms that essentially result from wearing. This makes it predictable

The research task in this chapter can thus be summarized into two key research questions:

1. If the 8bit was replaced by a 16-bit ADC, how much improvement will it bring to the prediction of tube failure? More precisely, how much earlier the failure can be detected?

2. If a 16-bit setup is desirable. How can we implement a possible technical realization for this 16-bit setup on a current device?

To answer the first question, a program was created to simulate the data coming from the X-ray tube over its lifetime. The data is then sampled as if there were an 8-bit or 16-bit ADC to see how earlier the change in the signal can be detected. The details of this will be presented in the first subsection.

To create a proof-of-concept for the enhanced measurement, a special measurement setup was developed. The second subsection details the architecture, hardware, and software realization of this setup

5.1 Simulated Model for Degradation in X-Ray Tubes

As mentioned in Section 2.1.4, tube failures can be due to random events, due to production-related issues, or due to wear, tear, and degradation. In the scope of this work, we assume that there are no problems in the tube due to production and random events, and concentrate on simulating the wearing phenomenon happening to the filament of the tube, namely the gradual evaporation of the filament.

To investigate the effect of higher bit resolution on the prediction, a high-fidelity signal reflecting the health status of the tube should be available throughout the lifetime. Such a signal can be captured experimentally by a high-resolution test setup. Indeed, such a setup was built and a test has been running. Nevertheless, at the point of writing a failure has not yet happened so an experiment-based conclusion could not be drawn yet. As a result, a simulation program is needed to construct the high-fidelity signal.

Even without the simulation program, certain hypotheses could be made about the form of the signal, namely exponential. Thus, a simulation is required so as to theoretically investigate this hypothesis. To do that, the simulation has to be based as much as possible on physics but also take into account the measured data for corrections.

In the future, as the simulation program is fully developed, it could be used in helping directly the failure prediction. One possibility is to use it in model-based prediction methods. As another possibility, it could be used to create training data for an AI model.

In this simulation program for the X-ray tube, we would like to get the filament current as the output signal. On the one hand, the filament current is a controlling, not a controlled signal. Controlled signals are outputs of control loops, which are essentially kept at a constant level right up to the very last moment. Thus they are not very useful for telling the component health. For that, only the controlling signals, i.e. signals which were used to adjust controlled signals, are relevant. On the other hand, the filament current does surprisingly give a lot of information on the health status of the X-ray tube, in particular of the vacuum and of the filament.

Normally, to determine the lifetime of the filament, the filament was heated with a constant filament current. But in real XRF devices, it is actually the anode current that is kept constant. The reason is that the intensity of X-ray beam is proportional to the anode current. Thus the filament current is adjusted in real-time to keep the anode current constant. Under this condition, the following phenomena should be taken into account:

- When the filament gets thinner, the resistance is higher. Thus, less filament current is required to reach the same overall temperature to emit the same number of elections.

- Hot spot (pinching) effect: irregularities in the filament construction may lead to one small spot with a smaller diameter. This spot will heat up faster, gets thinner more quickly, and then heat up even more than its neighbor. As the electron emission is proportional to the exponential of inversed negative of temperature, this small spot may emit a very large number of electrons, which reduces even more the need for filament current.

In effect, instead of having a constant filament current throughout, a real tube in a real device will have **reducing** filament current during its lifetime.

The first Section covers the simulation program. The second Section would provide an analysis based on the output of the simulation.

5.1.1 Implementation Of The Simulation Program

5.1.1.1 Overview Of The Program

The overall inputs to the simulation program include:

- Filament parameters including its length l and radius r
- Operational parameter that is the required anode current I_{anode}
- Simulation time step size Δt that is the duration between each update of all mutable parameters
- End condition, which would be checked after each simulation step to stop the simulation.

There are some possibilities for the end condition. It could either when the mass of the filament is reduced by 10% [42]. In this simulation program, we used the end condition observed from the real failures, that is when the anode current is reduced down to 70% of the original value. Note that the exact value does not matter that much, because at the end of the lifetime, the anode current decrease very rapidly over a very short time. As a result, the variation in exact end value leads to a negligible difference in the total lifetime.

The overall output of the program is an array consisting of filament current values for each and all of the time steps.

The class `Experiment` is the highest-level structure in the program. An object of the class `Experiment` is instantiated to represent one full simulation run over one filament. It contains:

- An internal storage of the parameters. This allows multiple `Experiment` objects to be created, each having (slightly) different parameters. Together, the objects correspond to a range of parameter values. This is particularly useful for:
    1. Scanning and evaluating the effects of changing one or more input parameters.
    2. Stochastic simulation, whereby each parameter can be sampled from a distribution
- All variables for state parameters. State parameters represent the state of the filament and are mutable (i.e. will be changed) during the course of the simulation run.
- A function to calculate new values for the state variables after each single time step
- A function which implements the loop of the simulation program. The loop repeatedly calls the single-step function until the end condition is checked. All intermediate values of the state variables are saved to be used for plotting or as return values.

In the main program, first, all the input parameters are defined in a dictionary structure. The simulation inputs, namely filament and operation parameters, are fully defined by this one structure[2]. Then Experiment objects are then instantiated. For a single run, only one experiment object is needed. For a parameter-scanning experiment or a stochastic experiment, a corresponding iteration logic is implemented and one Experiment object is created in each scanning run, or each Monte Carlo run. On each object, the loop function is called and its return values are saved into the output storage.

5.1.1.2 One Updating Step

In the Experiment class, the function refreshParam(delta_t) will calculate and update the values of all state variables after one timestep Δt. Not all variables can be updated in parallel due to their logical dependence. In the implementation, the order of variables being updated is presented in Figure 5.1. Each step starts with the current mass of the emission segment of the filament and ends after the new mass is calculated. All the calculations necessary for going from the old to the new value of filament mass belong to the main loop. This does not include the filament current. As a result, the current will be calculated as a leaf branching from the main loop.

Before the main loop can even be started, the starting values of the state variables are calculated. First, the filament mass is calculated from the radius and filament length. The radius can be read out from the technical drawing of the filament, as well as the length of the helical section of the filament. The emission length approximates the length of the helical section and is adjusted slightly according to real data. Calculation of the filament mass is done as follows:

$$m = \pi * l * r^2 * \rho_W \tag{5.1}$$

[2] For stochastic simulation, an additional dictionary structure can be defined which contains a statistical description (e.g. standard deviation) of each parameter. The filament and operations parameters are then randomly sampled from those two structures.

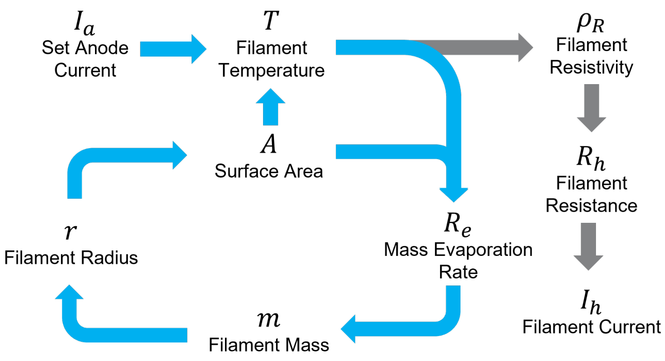


Figure 5.1: A single loop which runs every time increment Δt to update the value of the parameters. The main loop (blue) starts and ends from the Filament mass$r$. The leaf (gray) is needed to calculate the value of the Filament current I_h. The parameters are updated in the order following the arrow direction.

where m is the filament mass, l is the length of the emission portion of filament, and $\rho_W = 19.25 * 10^3 kg * m^{-3}$ is the mass density of Tungsten. Besides that, the work function of Tungsten is adjusted to take into account the Schottky effect according to Equation 2.3.

As the simulation loop started, the `refreshParam(delta_t)` will be called in each iteration of the loop. Starting from the current filament mass, the radius of the filament is then calculated as follows:

$$r = \sqrt{\frac{m}{\pi * l * \rho_W}} \tag{5.2}$$

After which, the evaporating area of the filament can be found:

$$A = 2 * \pi * l * r \tag{5.3}$$

The temperature of the filament is then calculated. Depending on whether the failure due to the pinching effect should be taken into account, the calculations follow different methods. Details of this step are discussed in Section 5.1.1.3.

With T having been found, the evaporation rate is then calculated as [73]:

$$R_{eva} = A_{eva} * e^{-\frac{q_0}{k*T}} * A \tag{5.4}$$

where $A_{eva} = 6.67 \times 10^8 kg \cdot m^{-2} s^{-1}$ is the the evaporation constant and $q_0 = 1.333 \times 10^{-18} J$ is evaporation energy of Tungsten. A_{eva} should not be confused with the area A of the filament.

Assuming a constant evaporation rate in each time step, the new value of the filament mass is then found:

$$m' = m - R_{eva} * \Delta_t \tag{5.5}$$

This concludes the main loop. To find out the filament current, the resistivity of the filament is calculated after the new values for temperature and radius have been calculated in the main loop. The resistivity is approximated as follows:

$$\rho_R = \alpha_\rho * T + \beta_\rho \tag{5.6}$$

where ρ_R is the resistivity and should not be confused with Tungsten mass density ρ_W. The coefficients α_ρ and β_ρ represent the linear approximation of resistivity against temperature in the relevant range. They were calculated by doing linear regression on the values from Table 5.1.

Finally, the filament current can be calculated. For which, the electrical power could be set to be equal to the radiating power, which is given by the Stefan-Boltzmann law:

$$\frac{P}{A} = \sigma T^4 \tag{5.7}$$

with P is the radiation energy of the black body, A is the area, T is the temperature and $\sigma = 5.68 \times 10^{-8} W \cdot m^{-2} K^{-4}$ is the Stefan-Boltzmann constant. The

Table 5.1: Numerical values for tungsten filament resistivity as a function of filament temperature [74].

R/R_{300K}	Temperature (K)	ρ ($\mu\Omega$cm)	R/R_{300K}	Temperature (K)	ρ ($\mu\Omega$cm)
1.0	300	5.65	10.63	2100	60.06
1.43	400	8.06	11.24	2200	63.48
1.87	500	10.56	11.84	2300	66.91
2.34	600	13.23	12.46	2400	70.39
2.85	700	16.09	13.08	2500	73.91
3.36	800	19.00	13.72	2600	77.49
3.88	900	21.94	14.34	2700	81.04
4.41	1000	24.93	14.99	2800	84.70
4.95	1100	27.94	15.63	2900	88.33
5.48	1200	30.98	16.29	3000	92.04
6.03	1300	34.08	16.95	3100	95.76
6.58	1400	37.19	17.62	3200	99.54
7.14	1500	40.36	18.28	3300	103.3
7.71	1600	43.55	18.97	3400	107.2
8.28	1700	46.78	19.66	3500	111.1
8.86	1800	50.05	20.35	3600	115.0
9.44	1900	53.35			
10.03	2000	56.67			

radiation energy is provided by the electrical energy from the filament current. As a result:

$$\begin{aligned} I_h^2 R_{fil} &= \sigma T^4 A \\ I_h^2 \frac{\rho_R l}{\pi r^2} &= \sigma T^4 2\pi r l \end{aligned} \tag{5.8}$$

As a result, the filament current is thus finally calculated as follows:

$$I_h = \sqrt{\frac{T^4 * 2 * \pi^2 * r^3 * \sigma}{\rho_R}} \tag{5.9}$$

The steps above are for the case without consideration of the pinching effect. To take this effect into account, some adjustments were necessary:

- The length of the filament is divided into 2 sections: the so-called "pinch section" or "hot section" with smaller radius, higher temperature, and thus higher evaporation rate, and the normal "cold" section with lower temperature.

- During the simulation, two set of parameters $(r, T, R_{eva}, m, \rho_R)$ will be maintained and updated for each section. The anode and the filament current are the common parameters due to Kirchhoff's law.
- The updating order of the parameters follows the same steps as discussed. Most of the steps are just carried out twice, one for each section. However, the calculation of the temperatures of the two sections from the anode current has to be done simultaneously, as both sections contribute to electron emission. Besides that, the calculation of filament current just needs to be done in one section due to the continuity of current. [3]

Here we aimed to build a simplified model. Thus the heat diffusion between the pinched and normal sections, which requires finite element methods, was beyond the scope of this work.

5.1.1.3 Solving for the Filament Temperature

As detailed in the overall introduction to this section, in devices, the filament current was controlled in real-time to achieve the desired value for anode current. Thus, during the simulation, the anode current is actually an input parameter, the task in each updating step is to find the filament current from this anode current, given the current other parameters of the filament.

The filament current is directly related to the filament temperature by the Stefan-Boltzmann law (Equation 5.7). Thus the tasks above can be rephrased as finding the filament temperature from the anode current. Starting from the Richardson

[3] Actually, the current flowing into the filament is the sum of that flowing out of the filament and the anode current. However, we can ignore the anode current for the calculation of the filament current for 2 reasons. First, the anode current is around three orders of magnitude smaller and is thus negligible. Secondly, the anode and filament currents are controlled by two separate circuits, thus by the principle of superposition, the filament current could be considered in isolation.

equation for thermionic emission with adjustment by the Schottky effect, the anode current can be found as:

$$I_{anode} = A_G * T^2 * e^{-\frac{W'}{k*T}} * A * c_1 \tag{5.10}$$

where $A_G * e^{-\frac{W'}{k*T}}$ is the emission current density as discussed in Section 2.1.2. A is the filament surface area and c_1 is a correction coefficient to account for the fact that some emitted electrons may not reach the anode.

In the case when the pinching effect is not considered, the filament temperature T can be found by solving the equation 5.10. The analytical solution for this equation exists as follows:

$$\begin{aligned} a &= \frac{I_{anode}}{A_G * A} \\ b &= \frac{W'}{k} \\ c &= \frac{b}{2 * \sqrt{a}} \\ T &= \Re\left\{\frac{b}{2 * W_0(c)}\right\} \end{aligned} \tag{5.11}$$

where $W_0()$ denotes the principal branch of the Lambert W function. The Lambert function $W_k()$ is a special function that offers the complex-valued solution to the equation $z = xe^x$ for the complex value z and unknown variable x. For a single value z there may be multiple solutions for x, denoted by the branch k of the function. As the temperature is a physical quantity, we take the Real part of the result with the operator $\Re$.

When the pinching effect is taken into consideration, the anode current is now the sum of the contributions from the electrons emitted by the pinched section and the "normal" section of the filament length. The pinched section has a smaller cross-section and thus higher resistance. As the same filament current must flow through both sections, the thermal power emitted by the pinched section must be higher. As a result, the temperature of the pinched section is higher than the rest of the filament, and it will be denoted the "hot" section, in contrast to the remaining "cold" section. It is also worth noting that, the higher temperature also leads to a further increase in the resistivity of the hot section and makes it even hotter. Finally, the anode current is modeled as:

$$I_{anode} = (T_h^2 * e^{-\frac{W'}{k*T_h}} * A_h + T_c^2 * e^{-\frac{W'}{k*T_c}} * A_c) * A_G * c_1 \tag{5.12}$$

where A_h and A_c are effective emission areas of the hot and cold sections of the filament. The temperature of the two sections, T_h and T_c, are the unknown variables that should be solved.

Because there are two unknowns, yet another equation is needed for the solution. From the Kirchhoff law, the filament current flowing through the hot and the cold sections of the filament should be the same.

$$I_h = I_c \tag{5.13}$$

Technically, there is also the anode current flowing through part of the filament to the anode. However, the anode current is very small (hundreds of micro-amperes) in comparison to the filament current (amperes) and is thus negligible. Following the Stefan-Boltzmann law, the equation above could be rewritten as:

$$\begin{aligned} I_h^2 &= I_c^2 \\ \Rightarrow \frac{\sigma T_h^4 * A_h}{R_h} &= \frac{\sigma T_c^4 * A_c}{R_c} \\ \Rightarrow \sigma T_h^4 * 2\pi r_h l_h * \frac{1}{R_h} &= \sigma T_c^4 * 2\pi r_c l_c * \frac{1}{R_c} \\ \Rightarrow \sigma T_h^4 * 2\pi r_h l_h * \frac{\pi r_h^2}{\rho_h l_h} &= \sigma T_c^4 * 2\pi r_c l_c * \frac{\pi r_c^2}{\rho_c l_c} \\ \Rightarrow T_h^4 * \frac{2\sigma\pi^2 r_h^3}{\rho_h} &= T_c^4 * \frac{2\sigma\pi^2 r_c^3}{\rho_c} \end{aligned} \tag{5.14}$$

denoting $M_h = \frac{r_h^3}{\rho_h}$ and $M_c = \frac{r_c^3}{\rho_c}$, the second equation could be written as:

$$T_h^4 * M_h = T_c^4 * M_c \tag{5.15}$$

Solving equations 5.12 and 5.15 simultaneously would yield T_h and T_c.

However, this system of non-linear, transcendental equations does not have an analytical solution and has to be solved numerically. For this purpose, the system of equations can be rewritten as follows:

$$\begin{aligned} (T_h^2 * e^{-\frac{W'}{k*T_h}} * A_h + T_c^2 * e^{-\frac{W'}{k*T_c}} * A_c) * A_G * c_1 - I_{anode} = 0 \\ T_h^4 * M_h - T_c^4 * M_c = 0 \end{aligned} \tag{5.16}$$

or

$$f(T_h, T_c) = 0 \tag{5.17}$$

One possible method to find the results is Newton's method [75]. This is an iterative method in which the guess to the correct solution is improved successively given the gradient of the function at the current guess. Starting with an initial guess x_0, the next guess is $x_1 = x_0 - J_f^{-1}(x_0)f(x_0)$. The quantity $J_f^{-1}(x_0)$ is the inverse of the Jacobian of function f at position x_0. The process is repeated until the error $f(x_n)$ is smaller than a certain threshold ϵ.

Another possibility is the use of the Gradient Descent (GD) method, or its modified version Stochastic Gradient Descent (SGD). In essence, the GD method moves the guess point in the direction where the function has the steepest descent. From the previous guess, the next guess is calculated as $x_1 = x_0 - \gamma \nabla f(x_0)$ [76].

The calculation of gradient $\nabla f(x_0)$ is usually calculated automatically by packages such as PyTorch or TensorFlow [77]. In comparison to Newton's method, there is no need in GD to analytically calculate the Jacobian matrix, which is a quantity very specific to each equation system. As a result, the same code base implementing GD can be reused to solve other equation systems or optimization problems.

The learning rate γ determines how fast the guess point should move in the negative gradient direction. Learning rate scheduling methods such as AdaGrad, RMSProp, or Adam dynamically adjust the learning rate [78]. This brings about benefits such as faster convergence, and, most importantly, avoidance of local minimum.

For these reasons, in this work, the GD method was applied to solving the equation system 5.16.

5.1.2 Findings And Analysis

In the experiment, we set up the simulation program with two sets of operational parameters corresponding to two conditions:

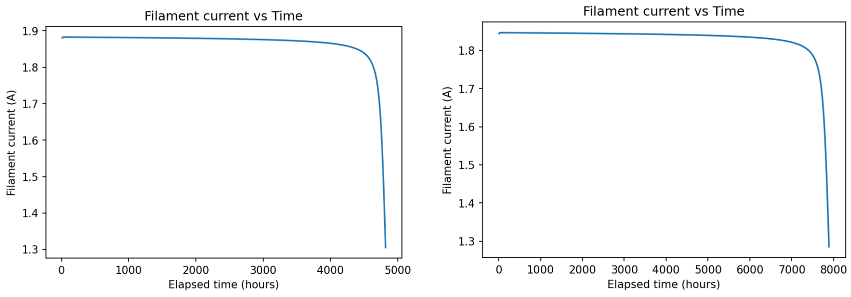
- The overdrive condition (denoted C1) with $I_{anode} = 1.2mA, V = 10kV$: This is the condition that the two real devices in the laboratory were tested. To achieve this condition, the devices were actually modified and thus can be operated at a much higher current than normal.
- The maximum allowed condition (denoted C2) with $I_{anode} = 0.9mA, V = 10kV$: This is the highest current value that is allowed to be set in unmodified hardware.

For both conditions, the device should produce the variable that is observable by the device, namely filament current I_h.

To start the experiment, the geometric parameters of the filament parameters are set as followed:

- Length of emitting portion of the filament $l = 20mm$
- Length of the pinching higher-temperature portion $l_h = 0.16mm$
- Radius of filament $r = 0.055mm$

The `Experiment` object will be instantiated. From which the `loop()` function is called. This function will repeatedly do the updating steps. After each step, the failure condition is checked. Here we defined the failure condition based on the observed data from the two failure cases. The simulation is stopped when the value of the filament current is 30% less than the original value.



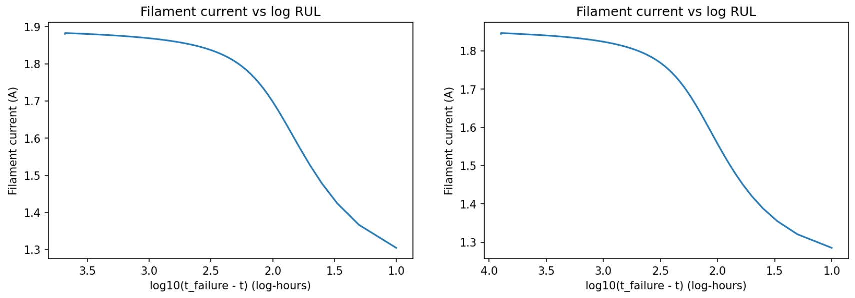
(a) For 1.2mA anode current

(b) For 0.9mA anode current

Figure 5.2: The filament current trajectories during the whole lifetime of the tube resulting from the simulation.

Figures 5.2 show the trajectories of the filament current for the conditions C1 and C2 through time. The pinching (or hot spot) effect was taken into account. It shows that during the lifetime, the trajectory of the filament current could be divided into 2 sections. In the starting section, which lasts for the majority of the lifetime, the signal decreases at a very low rate. Towards the end, there is a sudden drop in the signal and it decreases at an exponential rate.

In Section 6.2.2.1, based on the analysis of the collected data from real filament burn-out failures, the relation $y = a\log(t - t_{failure}) + b$ (where (y, t)s are pair of signal value and timestamp, and a,b are scalar parameters) could be established from the signal. As a sanity check, the signal value from the simulation output was plotted against the logarithm of the remaining lifetime for both conditions C1 and C2. The plots are shown in Figure 5.3. The lines show a reasonable agreement with the real data. Two comments can be made about the difference in the shape of the simulated and real data. First, the ranges of time were



(a) For 1.2mA anode current

(b) For 0.9mA anode current

Figure 5.3: Plot of Filament current against $\log(t_{failure} - t)$.

different with the simulated time ranges between around 10^4 to 10^1 hours (due to simulation granularity) and the real data time ranges between around 10^3 and 10^{-3} (data sampling granularity). This difference in ranges may lead to differences in the shape of the plots. Secondly, further investigations could be made on the simulation to also take into account further possible phenomena and to improve the agreement with real data.

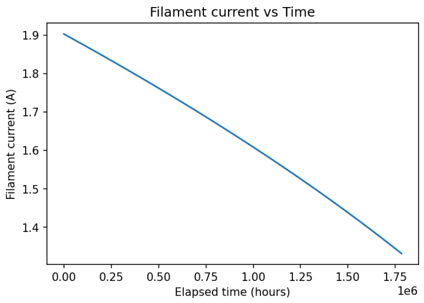


Figure 5.4: Filament current over time when the hot spot phenomenon is not taken into account. Note that the signal trajectory is not in the exponential form as it should be.

The simulations above took into account the pinching effect and produce reasonable results. If the effect was **not** taken into account, the output signal turns out to be very different than the real signal. Figure 5.4 show the filament current throughout the lifetime in this case. Comparing to Figure 5.2 with hot spot (pinching) effect, it was clear that the expected behavior of slow decrease followed by exponential decrease was not observed. Instead, the filament current decreases at a steady, slightly quadratic, rate.

Finally, with the simulation results being agreeable with the real data, the output filament current could be used to evaluate the improvement in the prediction horizon between the 8-bit and 16-bit data collection setups. As will be established in Chapter 6, the filament current is the most important measurable signal reflecting the health status of the XRG. To determine the prediction horizon, the earliest time point at which a prediction could be made should be identified from the trajectory of the output filament current. First, the digital values of the current should be determined. For this discussion, a simplified calculation approximates that which happens in the device. The current would flow through a $R = 1\Omega$ resistance. The voltage across is digitized by an n-bit ADC, whose range is set to $5V$. The digital signal is thus $I_{digital} = \frac{I_{filament} \cdot 1\Omega}{5V} 2^n$. Following the discussion in Section 6.2.2.2, a prediction could first be made when three transitions from a digital value to the next lower digital value have happened. This means that the first instance when $I_{digital} = I_{digital,0} - 3$ should be found.

Table 5.2: Expected lifetime and prediction horizon at different bit resolutions. The unit is Hour.

	Condition C1	Condition C2
	Ianode = 1.2mA	Ianode = 0.9mA
Expected Lifetime	4820	7890
Prediction Horizon at		
8-bit	190	390
12-bit	2310	4030
16-bit	4590	7470

The table shows the expected lifetime and the prediction horizons for the 8-bit, 12-bit, and 16-bit measurements for the operation conditions C1 and C2. As can be observed from the table, the transition from 8 to 16-bit measurement massively improves the prediction horizon. In the 8-bit cases, the prediction could only be made relatively late in the lifetime. This horizon, while still enabling planning and communication between the customer and the service team to ensure no production interruption, leaves little room for error. In the 16-bit cases, the prediction could be made from a very early stage in the component's life. This would enable a very high margin of safety for the system due to the plenty of reaction time for both the customer and service team. The single issue may be the cost of the 16-bit ADC, in which case, a compromise could be made and a 12-bit could be used. As demonstrated in Table 5.2, even though the margin of security is not as high as 16-bit, the 12-bit version still improves the prediction horizon ten-fold in comparison to the currently available 8-bit version.

5.2 High-Precision Measurement of Operational Parameters Of X-Ray Generator For Improved Prediction

Based on the analysis in the last section, we have established that the 16-bit measurement of the component data will improve the prediction horizon. The topic of this section is the construction of a set-up around an existing device, but the component data will be measured in an enhanced resolution of 16-bit. Not only does this setup helps to confirm an improved prediction horizon, but it will also help the investigation of further quantities:

- On this setup, it will become possible to measure filament resistance. The measurement of resistance was not possible with the original in-device electronics.
- As mentioned in Section 2.1.4, over time, Tungsten will evaporate from the filament due to high temperature and from the anode surface due to electron impact. Combined with existing impurities, it will make vacuum quality deteriorate. Arcing between anode and cathode could happen. If

the measurement setup could also operate at a high sampling rate, it might be useful to investigate how the statistics of the arcing change over time.

The first subsection will discuss an overview of the architecture of the setup. Hardware realization will be concisely discussed in the second subsection. Software realization with a discussion of the signal pre-processing algorithm will be the topic of the second subsection. The resulting data and some analyses are discussed in the third subsection.

5.2.1 System Architecture and Hardware Realization

An overview of the system is presented in Figure 5.5. The system was realized based on the basic device setup with the device and PC. Inside the device, the X-ray tube (with the glass housing) and the high voltage cascade reside inside a copper housing, with provided radiation protection. The housing has plugging slots for cable, which feeds current to drive the filament and the high voltage cascade. The other side of the cable leads to the high-voltage controller board, which has a high voltage side and the low voltage side. The low voltage side operates at TTL level (+- 5V) and has electronics to convert high current/voltage to the level so that it can be measured by the microcontroller.

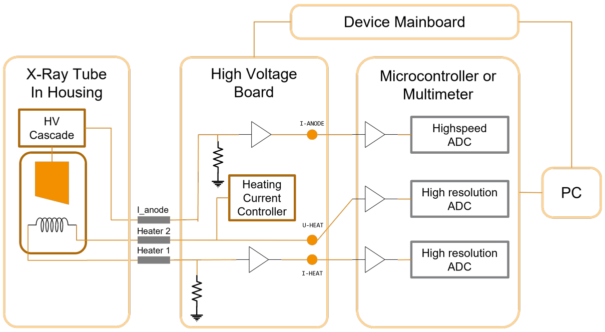


Figure 5.5: Architecture of the setup, in which the component data of the X-Ray tube from the XRF device was measured with enhanced accuracy

Table 5.3: The signals from the high voltage board, captured by the six 16-bit channels of the data logger.

ADC Channel	Measured Data Type
0	Filament Voltage
1	Filament Current
2	Anode Current (Measured)
3	Anode Current (Set)
4	Tube Voltage (Set)
5	Tube Voltage (Measured)

For the enhanced measurement setup, an industrial-grade measurement module was used. The selected measurement module offers 6 buffered ADC channels. Each channel has a 16-bit resolution at a maximum of 750kHz sampling rate. Its input ports will be connected (by wires) to the low-voltage side of the electronic boards. The wires were soldered to the board on the points, from which the corresponding signals (e.g. anode current) were also sampled by the board's own micro-controller. The 6 signals monitored by the module are presented in Table 5.3.

The measurement module does not perform signal processing. As a result, all the readout signal is transferred to the PC with a USB connection. On the PC, a program will read out the signal and do signal processing before saving the results to the local storage. The details of the software will be discussed in the next subsection.

For the hardware realization, the high voltage board was taken out of its own EMP protection housing inside the device. As the EMP housing does not offer any free space for additional electronics. The soldered wires have to be routed out of the metal cage. Care was taken to minimize interference on the readout signals. On the one hand, the measurement module was placed as close as possible to the EMP cage. On the other hand, the connecting wires are twisted pairs, with one of the pairs soldered on the signal point, and the other of the pairs soldered to a

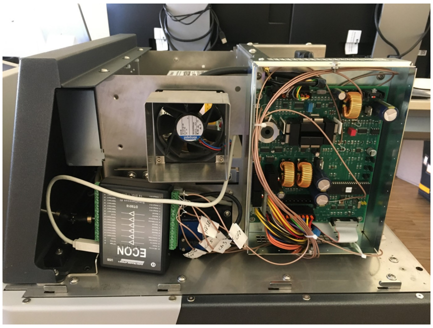


Figure 5.6: Hardware realization of the setup with all housing and EMP shield removed. Right: high voltage board. Bottom left: 16-bit data logger.

nearby ground point. Figure 5.6 shows the finished setup with the top of the EMP cage and housing still removed.

5.2.2 Software Realization

The software was developed to control the measurement module, collect the measurement and pre-process the measurement data before saving it to local storage. The software would run on the computer accompanying the XRF device that was a part of the test setup. It will run independently from the main measurement software, as the setup is only experimental.

In contrast to other usual measurement loggers, the selected measurement logger does not feature a programmable board, i.e. it does not execute custom-made programs on its microprocessor. This is actually a very sensible design. The logger generates a very large quantity of data, that the on-board computational power may not be able to process. Instead, the logger is controlled with the help of a driver program on the PC. A custom program is then developed to use the APIs from the driver. Data would flow directly from the internal buffer of the data logger to the custom program and the more powerful PC processor would then process the data.

Instead of writing raw data to the local storage with limited capacity, the program pre-processes the data to extract and save only important features. For 6 channels each 16-bit and 10kHz sampling rate, the raw data amounts to around 120kB a second or 412MB per hour, which will fill up the hard disk very quickly. Instead of that, for each second of data, only important statistical characteristics will be extracted, namely mean, median, standard deviation, skewness, and kurtosis. During the degradation, changes in signal level are expected to happen over a time scale of hours. Thus the duration of one second between stored data points may be sufficient.

On the short end of the time scales, the arcing between the cathode and anode should also be investigated. These arcing events create signal spikes in the order of milliseconds according to internal experts. Thus the sampling rate of 10kHz should be sufficient. However, what is important to investigate is not the individual spikes, but how the statistics of these spikes change over a longer time scale. For that, the statistics of those spikes within each 1-second portion of data are extracted and stored. The statistics include the number of spikes in one second, and the average and maximum height and width of the spikes.

To identify a spike, the z-score of each signal point is compared to a threshold. The z-score represents the statistical anomaly of the value y and is calculated as $z = \frac{y-\overline{y}}{\sigma_y}$. The σ_y is the standard deviation of the overall signal. The $\overline{y}$ is the expected value of the signal at that point, whose value is determined by the value of the best line regressed to the signal. If the z-score goes beyond the threshold, the point would be identified to be a part of a peak. Consecutive points which exceed the threshold are grouped together and identified as one peak, from which its height and width are calculated. The overall statistics of all peaks in one second are then calculated and stored.

With all the building blocks already discussed, the flow of the program could now be described. At the start, the program will start up the communication of the data logger and set the operational parameters. These include sampling rate, bit depth, number of channels, buffer capacity per channel, and other parameters. Basically, buffers are data buckets that are first handed over empty to the data logger. The logger will read out the data, fill up one bucket at a time, and hand back the full buckets to the program. The program would then read the bucket and offload the data to a FIFO queue for processing. It will then hand the empty bucket over back to the logger.

Due to a very large amount of data and processing, the processing is packed into routines, which are then executed (possibly in parallel) by threads in the Thread Pool. Each thread reads the data portion of the queue and executes exactly the same algorithm for the extraction of data statistics as described above. The results of the parallel processing are then put into another queue. At the end of this second queue is a single thread, writing data to the local storage. This single thread manages the file system and file names. In contrast to processing, disk-based IO operations are normally not conducted in parallel due to either not being supported, or due to the possible data corruption. The whole software program is realized in C# language based on .NET 4.0 framework as required by the driver API.

5.2.3 Results of Data Collection and Recommendations

The modified hardware and software were put into continuous operation mode until failure. The main measurement software was the one used in the data collection campaign with the ability to collect component data in 8-bit. The operation condition was set to 10KV, 0.9mA anode current (condition C1) so that the evaporation rate of the filament is maximal within the allowable limit. The data logger was set to conduct measurements with 6 channels, each 16-bit at 10kHz with a 5V voltage range. In addition to component data, an additional XDK data logger was used (Section 4.3.1.2) to monitor the temperature of the environment.

The graphs of the results will be presented next. Not all results, but only the more interesting ones will be presented. It is worth noting that the values in the graph which were measured by the 16-bit ADC are not necessarily the actual values. Instead, they are just digitalized values, mapped to the 5V range, i.e.

$y_{shown} = \frac{y_{digital}}{2^{16}} 5V$. Analogously, the 8-bit values measured by the onboard electronics are presented by their digitalized value (i.e. 0-255). Conversion to real values could be done simply but will not be done here for confidentiality reasons.

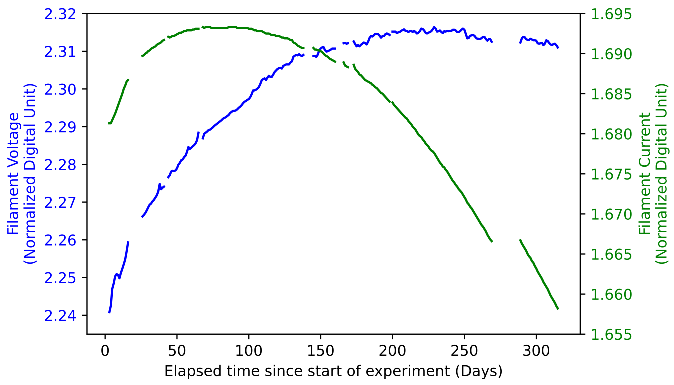


Figure 5.7: Filament Voltage and Filament current collected from the setup. The values are digital values, normalized to the 5V range.

Figure 5.7 presents the value of filament voltage and filament current over time, measured in 16-bit. The filament voltage can be seen to rise over time but starts to plateau out towards the end. The reason is the rising filament resistance (which will be shown next) and the initially rising filament current. The plateau (and possibly a drop at the end) is due to the value of filament current required for the same anode current dropping more quickly. The dropping value of the filament current is the expected behavior, which is also predicted by the simulation. Such behavior is indeed seen from the later portion of the graph for the current. In the early portion, the filament current value actually rises. This behavior is neither expected nor known yet by both the academy and internal experts. One possible reason for this behavior could be, that the filament needs some initial conditioning so that either the surface or the lattice structure got changed and improve the electron emission efficiency.

The filament resistance, obtained by dividing filament voltage by current, is obtainable for the first time with the help of the extra logger. As can be seen in

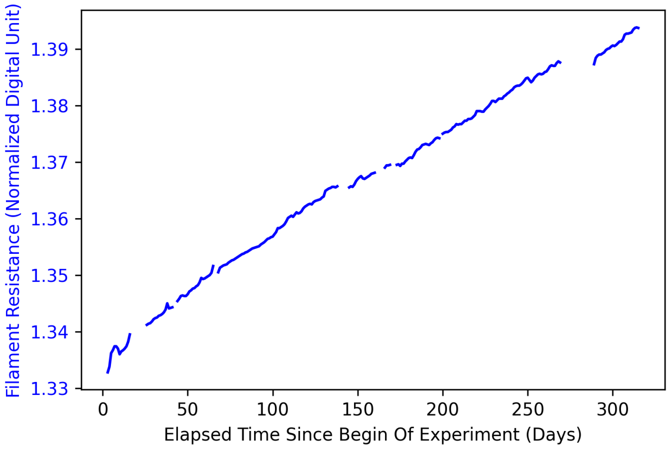


Figure 5.8: Electrical resistance of the Filament.

Figure 5.8, the resistance value increase very stable (and even very linear) over time. This means that, if it can become measurable in future devices, it can be very useful for predicting the lifetime or failure of the filament.

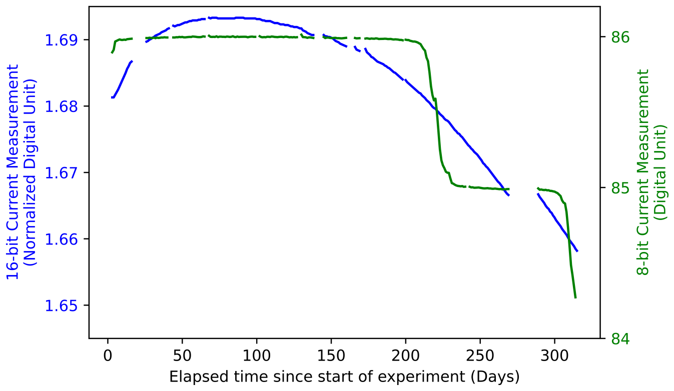


Figure 5.9: The filament current measured by the 16-bit setup (blue) and on-board 8-bit electronics (green). Notice the staircase shape of the 8-bit curve.

Among the current measurable values, the filament current is the key to predicting failure, as will be discussed in chapter 6. Referring back to the central research question of this chapter, Figure 5.9 clearly presents the advantage of the 16-bit ADC over the on-board 8-bit measurement. The 16-bit filament current shows a line smoothly transitioning between values, following possibly a second-order trajectory. This is very useful for the feature extraction and pattern recognition required in the prediction algorithms. On the other hand, the 8-bit signals show the staircase pattern between the digital values. There are very long stretches of time where the value essentially stays at the same level, having no information value. The transition between levels is also abrupt. Essentially, it means that for the 8-bit ADC, the probability of a bit flip between digital levels is not proportional to the actual value. Thus even an averaging of values over a longer period of time does not help to recover the true value of the signal. The negative implication of this behavior will be discussed closer in Section 6.2.2.1.

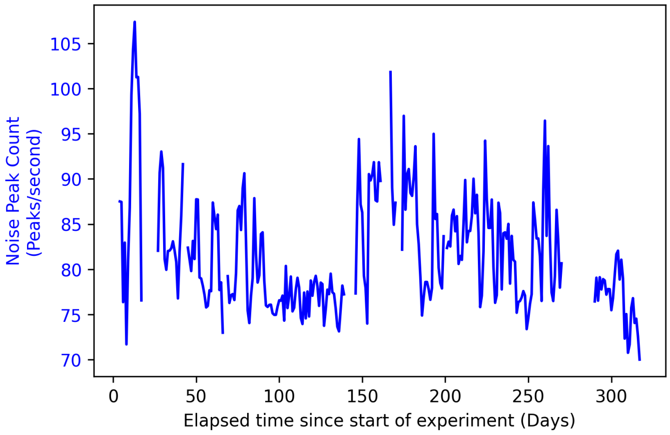


Figure 5.10: The count of noise peaks per second over the course of the experiment. No clear trend is observed.

Finally, the statistics of the signal spikes over time (which relates to the arcing between anode and cathode) are shown in Figure 5.10. As can be seen, the signal behaves sporadically showing no clear trends. This is to be expected, as the arcing is more relevant to the quality of the vacuum. For a tube that was not degrading according to the vacuum leakage failure mode [4], it is likely that the arcing statistics would not be changing very much.

5.3 Conclusion and Next Steps

In this chapter, a method for improving the prediction performance based on high-precision digital measurement was theoretically and experimentally investigated. More specifically, a higher bit resolution could extend the prediction horizon (i.e. the earliest time point starting from which a failure prediction could be made) to more than what is currently possible with the in-built 8-bit measurement electronics in the XRF device.

First, a theoretical analysis was carried out by developing a physics-based simulation model. This means the model simulates the physical phenomena that happen during the process of filament degradation and produces as output the measurable signal over time until failure. It was found that it is important to take into account not only the evaporation of tungsten from the filament surface but also the phenomenon of local hot spots.

Based on the simulated signal, it was evident that a 16-bit resolution would extend the prediction horizon massively (by at least 20-fold). Basically, the failure prediction horizon could cover the majority of the lifetime and prediction could be made even very early into the life of the filament. The analysis also showed that a 12-bit resolution could also deliver a very good prediction performance. This could be a sensible choice if the cost of 16-bit electronics was considered too high.

In light of these positive results, an experimental test setup was built based on an XRF device. The electronics of the device were extended with a high-precision

[4] The tube was being operated to induce the filament burn-out failure mode

16-bit data logger. Software was developed to control the data logger and to preprocess the huge amount of data to compress them to a manageable volume.

The experimental data confirm the hypothesis by showing the change very early (even since the start) in the lifetime. Even though the XRG has not failed yet at the time of writing, the fact that the filament current shows a slow, steady, and measurable trend means that the prediction is possible from an early stage. Besides that, the test setup enables the filament resistance to be measured, which is an improvement over the current device. The filament resistance also shows a very static trend of increasing over time.

From the results, a recommendation could be made to utilize high-bit-resolution electronics to monitor the signal of the component. In that way, the prediction could be made earlier. This allows the customer more time to plan around the failure. It also means that the Service Team would have more time to react. Even more, the massive increase in prediction horizon would enable additional value propositions. As one example, the customer could decide to prophylactically exchange the components during a yearly inspection and not have to wait until a few weeks before the actual failure. This saves both time and money for the customer and service team. Another example is to reduce the required inventory in each of the service offices. Instead of maintaining a large inventory of spare parts, the service office would order a part from headquarters when the failure is predicted for the part. This helps optimize the operation cost of the service activities.

As a next step, the simulation model could be improved so that it can generate synthetic data for the training of a deep learning model. For that purpose, the physical accuracy of the simulation model should be improved by incorporating the heat diffusion mechanism between the hot spot and the rest of the filament. The simulation program should also be extended with the ability to conduct stochastic (Monte Carlo-type) simulation runs. For the experimental setup, it has to wait until the point of failure. After that, the data could be analyzed and a definitive data-based conclusion about prediction performance could be formulated.

6 Prediction Algorithm

As mentioned in the background subsection, and following the concept laid out in subsection 3.3.3, a data-based approach to predictive maintenance was adopted. During the data collection, the full history consisting of data points, each with a timestamp, becomes available from many devices. After development, a prediction algorithm internalizes the (stochastic) mapping between data points and timestamps. When it receives new data points during deployment, time-related prediction, such as remaining useful lifetime is returned as output.

A data-based approach is more suitable than a model-based approach for the object of this work. Even though the model-based approach could deliver more accurate and explainable results, such a model is not available before and during the time scope of this work. The reason is the complexity of constructing such a model. It is also worth noting that the physical-based model constructed in the chapter 5 is not yet suitable for generating prediction. First, the model only covers a simplified version of one among many failure mechanisms. The full and extensive coverage of many mechanisms requires much more effort beyond the scope of this work. Secondly and more importantly, the constructed physical model is a generative model. It takes initial conditions as input and generates as output a series of data-timestamp pairs. As the final timestamp (at the breakdown) is known, the remaining lifetime could be inferred. A predictive model, on the other hand, takes data-timestamp pairs as input and output remaining lifetime as output. The mathematical-physical calculations are often very different between the two types of models. This leads to the unavailability of the exact model for the model-based approach. A generative model could only be "converted" into a predictive model if the following conditions are satisfied:

- The underlying physics model is a Markov model. This means that the next data point can be sufficiently inferred from one (or only a few) data points before it and not require the whole data history since the beginning.
- The initial and operation parameters are known. Those are required for generating the next data points. In many cases, the parameters are unknown and parameters scanning must be carried out to identify them, which is computationally expensive. In many other cases, multiple sets of initial parameters lead to the same data points, which complicates the calculation of the next data point. In those cases, stochastic techniques must be applied.

Under these conditions, the generative model can keep generating the next data points from existing ones until the breakdown or end condition is met. At which, the time until failure of the given data point can be back-calculated. This approach should also be investigated in the future, once a very comprehensive generative model would become available.

Among data-based approaches, machine learning-based methods have demonstrated their strength in approximating very complex models and producing excellent predictions. However, this approach requires a diverse, large, and balanced data set, which contains many full history runs. "Diverse" means the majority of the many operation conditions are covered. "Large" means for each condition, many full runs must be available so that aleatoric uncertainty could be sufficiently accounted for. "Balanced" means the amount of data for each condition should be similar, so that overfitting into a few conditions will not skew the output prediction. In order to collect such a large amount of data, a large-scale data collection campaign must be conducted over a long time. This has been discussed in Chapter 4.

Until the required data has been collected, a transitory solution is required. At this stage, some amount of full run history has been collected. But the data has not yet been able to cover the complex interaction between component data variables and operation condition variables. Thus, the direct mapping between multi-dimensional, time-varying input to single time-varying output could not yet be modeled. The idea is to separate the multi-dimensionality and the time-variance aspects of the problems and create two parts that deal separately with them. The two parts are called the Anomaly Detection Block and the Trend Prediction Block. This design of the algorithm will be discussed in detail in the first Section.

At the time of writing, with the availability of a total of only three data runs for two failure modes, only an experimental implementation was conducted. The goal is to show that it is possible to predict failure and remaining lifetime with the collected data. This implementation corresponds to the second block of the two-part algorithm design mentioned above. An elaborate discussion of the implementation will be the topic of the second Section.

6.1 Design of the Prediction Algorithm

6.1.1 Overview

The prediction algorithm takes the data-timestamp pair as input. Data here consists of component data and operation parameters. The produced output is the remaining lifetime, or a range of this quantity, possibly also with an associated certainty level.

Similar to other prediction problems, the input is a high-dimensional vector, whose value changes as the time (and thus the degradation) proceeds. The input vectors in this specific prediction problem exhibit some unique characteristics. First, different degradation processes result in very different trajectories of movement of this vector on the hyperplane. Sometimes the direction could be the polar opposite of each other, even for the very same component. Thus, to determine the degradation, it is also important to consider the divergence of the data vector from the "normal" (or pristine) data. Secondly, each change in operating condition results in a completely different value of the input vector. This means that during daily operation, the data vector could jump between very different trajectories.

It also means that the "normal" data is not just a point on the hyperplane. The "normal" state would be a line if there is one operation parameter, a plane if there are two operational parameters, and so on. Thirdly, there is a certain device-to-device variant. Upon analysis of the data, the normal state value of one device could fall into the region corresponding to the degraded state of another device. The final point regards the determination of the point of failure. In this case, it would require a seamless inclusion of expert knowledge in the form of failure threshold values into the algorithm. A common solution would be to include the threshold as one of the inputs.

As a result, the output variable $y \in R^1$ is not only dependent on time t and the sensor data $\boldsymbol{x} \in R^n$. But it also depends on operation parameter $\boldsymbol{p} \in R^m$, normal condition $\boldsymbol{x_0} \in R^n$, and a multitude k of thresholds $\boldsymbol{\theta} \in R^{n*k}$. The algorithm is a model which performs the mapping:

$$\begin{aligned} f_M : R^{1+n+m+n+n*k} &\mapsto R^1 \\ (t, \boldsymbol{x}, \boldsymbol{p}, \boldsymbol{x_0}, \boldsymbol{\theta}) &\xmapsto{f_M} y \end{aligned} \tag{6.1}$$

Some of the inputs contain further dependencies to other inputs, $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{p}, t)$, $\boldsymbol{x_0} = \boldsymbol{x_0}(\boldsymbol{p})$, and $\boldsymbol{\theta} = \boldsymbol{\theta}(\boldsymbol{p})$. A monolithic model approach would train a single model (such as a neural network) to approximate this mapping. This has a few disadvantages:

- The data requirement is very large to cover the whole input hyperspace $(t, \boldsymbol{x}, \boldsymbol{p}, \boldsymbol{x_0}, \boldsymbol{\theta})$
- A monolithic black box is less explainable to the customer why a certain output was produced
- Changes in user-defined values such as threshold θ, or in naturally variant values such as normal state $\boldsymbol{x_0}$ could put the model to an unknown input region, which may not guarantee a meaningful output.

With these insights, an approach was developed to reduce the data needs and improve the explainability of the output. In this approach, the algorithm consists of two sub-blocks. The first time-invariant sub-block compares the current data with the normal data. It produces an output that represents the divergence, or how abnormal the current data is. The second sub-block tracks and predicts this abnormality with time and product prediction. Details will be explained in the next subsection.

6.1.2 The Two-Block Design

With the divide-and-conquer strategy, instead of consisting of a monolithic block, the algorithm is designed from two connecting blocks aptly named "Anomaly Detection" and "Trend Prediction" Block. The output of the Anomaly Detection block is low dimensional (ideally one-dimensional) and is the input to the Trend Prediction block, which produces the prediction such as remaining useful lifetime. The task of developing the algorithm could then be mathematically specified as a two-step process.

6.1.2.1 The "Anomaly Detection" Block

In the first step, the mapping function g for the Anomaly Detection should be found such as:

$$\begin{aligned} g : R^{n+n+n*k} &\mapsto R^1 \\ (\boldsymbol{x}(\boldsymbol{p},t), \boldsymbol{x_0}(\boldsymbol{p}), \boldsymbol{\theta}(\boldsymbol{p})) &\overset{g}{\mapsto} z \end{aligned} \tag{6.2}$$

whereby $z \in R^1$ is the anomaly score. A trivial implementation of g could be:

$$\begin{aligned} z &= g(\boldsymbol{x}(\boldsymbol{p},t), \boldsymbol{x_0}(\boldsymbol{p}), \boldsymbol{\theta}) \\ &= \left(\frac{\boldsymbol{x}(\boldsymbol{p},t) - \boldsymbol{x_0}(\boldsymbol{p})}{\boldsymbol{\theta}(\boldsymbol{p}) - \boldsymbol{x_0}(\boldsymbol{p})} \right)^2 \end{aligned} \tag{6.3}$$

This demonstrates the reason why z is called the anomaly score. In a well-behaving scenario, the divergence between the actual data and "normal" data grows with time. In other words, the data become more "abnormal". This potentially results in a monotonous trend for the anomaly score z, for example, z is always increasing from 0 to 1 with the trivial implementation above. Such monotonousness is very desirable for the next block which predicts the trend in the z value. With such trivial implementation, or in another more elaborate implementation, the monotonousness condition is not always satisfied and should be further enforced by the additional constraints.

The first constraints enforce either "hard" monotonousness with time:

$$z(t') \geq z(t) \quad for \quad t' > t \tag{6.4}$$

, or "soft" monotonousness with time, i.e. finding g so that

$$\max_g P(z(t') \geq z(t)|t' > t) \tag{6.5}$$

in this case, time-monotonousness is tried to be achieved as well as possible. It allows g to compromise between this constraint and other constraints. In other words, if training g would require maximizing a certain target function, the "soft" version of constraints allows the target function to be constructed as a weighted sum of all the constraints.

The second constraints enforce a certain independence of z on $\boldsymbol{p}$. This seems counter-intuitive, as $\boldsymbol{p}$ appears extensively in the specification 6.2 above. Nevertheless, during real usage, the operation parameter could be changed, and thus $\boldsymbol{p}$ takes a different value. It does not make sense for z to also take an abrupt change. After all, z can be considered as an indicator of health, and the momentary health status does not simply change when a user decides to switch the device operation mode. Similar to the other case, this condition can be enforced "hard":

$$z(p', t') = z(p, t) \quad for \quad t' = t \tag{6.6}$$

or in a "soft" way:

$$\max_g P(z(p', t') = z(p, t)|t' = t) \tag{6.7}$$

Before implementing g, certain data should be available. First, the "normal" data for all different operation conditions $\boldsymbol{x_0}(\boldsymbol{p}), \forall \boldsymbol{p}$ should be captured. This could be done by carrying out a so-called "sweep test". An automatic script could set the machine to "sweep" through all the possible combinations of operation conditions and register the data. If one operation parameter is a continuous variable (such as a set level for anode current), some quantization could happen and this parameter could be sampled at discrete intervals.

Another required data are the threshold values $\boldsymbol{\theta}(\boldsymbol{p})$. The threshold could be set by domain expert knowledge, technical specification (e.g. maximum allowed power), or by observing other breakdown cases.

Those data are the bare minimum for the trivial implementation in Equation 6.3. To enforce monotonous conditions 6.4 and 6.5, however, further data should be available. For the first condition 6.4 (time-monotonousness), one or more full run history $\boldsymbol{x}(\boldsymbol{p}, t), t \in [0..T]$ could be used. Nevertheless, regarding the fact that the condition 6.4 enforces the **order** of the data with regards to time, and not the exact value of the timestamp, one could also construct multiple simulated data trajectories. For example, if during degradation, a data parameter always decreases or increases, a series of (ordered) data points $\boldsymbol{x_i}, i = [0..N]$ could be heuristically constructed for enforcing the monotonousness condition. Thus the need for real data trajectories could be reduced. For the second condition 6.6 or 6.7, however, the data at all the operation conditions must be collected along the degradation process, i.e. $\boldsymbol{x}(\boldsymbol{p}, t), \forall \boldsymbol{p}, t \in [0..T]$. It means that during the full run history of the device, sweep tests must be conducted. In this case, the exact time stamp is again not necessary, but far more important is the sampling of data (at different operation conditions) at the same time. This set of data is not only used during implementation of the g function, it is also the data, against which the Anomaly Detection block is validated. At the time of writing, this validation data is not yet available. As a result, the implementation of this block would be the topic for future work.

Various techniques could be used to implement the function g (or the Anomaly Detection Block). For one, the equations 6.3, 6.5 and 6.7 could be used together to specify a Conditional Random Field (CRF). In this technique, all the $(\boldsymbol{x}, \boldsymbol{p})$ points make together a $n + m$-dimension graph (or a grid). The goal is to specify the z values of all vertices of the graph. Each z value at each vertex has a probability that it satisfies all the constraints. The probability is specified by the relation between its current z values, a desirable z_0 as the starting point, and its neighbors zs. On the one hand, the equation 6.3 could be used to specify desirable starting point z_0 for all zs. The g specified by the CRF should heuristically be close to this desirable point, i.e. $g : \max_g P(z = z_0)$. On the other hand, the constraints 6.5 and 6.7 specified the relations of one vertex and its neighbors. During one training run, all the vertices are scanned and z values are adjusted to maximize the weighted sum of all the probabilities from all the vertices:

$$g : \max_g \sum_{i=0}^{V} (\lambda_0 P(z_i = z_{i,0}) + \lambda_1 P(z_i \geq z_k) + \lambda_2 P(z_i = z_l)) \tag{6.8}$$

where V is the total number of vertices, $\lambda_{1,2,3}$ are weights. Vertices k correspond to data points that are chronologically ordered before that of vertex i, and vertices l correspond to data points that are concurrent with that of vertex i.

In fact, the equation above could be used as the objective function for a second possible approach to implement the Anomaly detection block. In this approach, a neural network could be trained, from which z could be calculated from data and operation parameters. In both cases, the user-defined thresholds and device-dependent starting values are implicitly included in the objective function. Thus, both approaches required retraining of the model for each new deployment. Both approaches have their relative advantages and disadvantages. The CRF approach needs to train more parameters and is not accurate for input that falls between the nodes in the grid. It has however better explainability. The neural network approach could produce output anywhere inside the specified region. However, care should be taken for outputs outside the "expected" region, and the model is less explainable. These considerations should be explored when the Block is implemented as the next step of this work, once the validation data become available.

6.1.2.2 The “Trend Prediction” Block

In the second step of the process, the prediction value y is calculated based on the observed z values until current time $z_{t=0..t_{now}}$, which could be described mathematically as follow:

$$y : \max_{y} P(y|z_{t=0..t_{now}}) \tag{6.9}$$

in which, the predicted value y is the most probable value given the history of z_t until the current time. As discussed in Section 2.3, there are various possibilities to construct the probability function $P(y|z_{t=0..t_{now}})$. In the first type of approach, one could construct directly a function or a model which outputs directly the y value. The most common method for this approach includes machine learning methods, which use ML or MAP methods for training model parameters. In the second type of approach, a model is then constructed to successively predict the next z values (i.e. $z_{t+1} = argmaxP(z_{t+1}|z_{0..t})$) until a certain failure condition (e.g. $z_t > 1$). From that, the desired prediction quantity (e.g. remaining useful lifetime) is calculated backwards.

Both approaches required a large amount of training data to function well. At the time of writing, there is only a very limited amount of data available, namely three data trajectories for 2 types of failure. As a result, the approach for developing the Trend Prediction Block has not been finalized. Nevertheless, experimental approaches were developed to show that it is possible to predict the desired quantity (e.g. RUL) given the collected data. The experimental approaches will be discussed in detail in the next section.

6.1.2.3 A Transitory Approach For Continual Improvement Of Prediction Model

Between the current experimental approach and the final machine-learning-based approach, a transitory approach could be developed so that a prediction of reasonable quality could be delivered when an additional amount of data becomes available. The transitory approach partly alleviates the lack of data problem by performing the model training not only during algorithm development, but also during initial calibration, and during the daily operation of the devices. In other words, the transitory approach proposes a modification to the traditional model lifecycle. In the traditional life cycle, the best model parameters $M : \max_M P(M|X_{train}$ is trained only during development time, and only prediction $y = \max_y P(y|M)$ is done during deployment time. This implies that, in this approach, the model must be good enough to account for all variances between devices and the different ways in which the devices were used. This requires more data for training. In the transitory approach, the model is still trained from the more limited available data. But this time, not only one best model but a range of possible model candidates should be delivered. This range can be represented by the probability function:

$$P(M|X_{train}) = P(X_{train}|M)P(M) \tag{6.10}$$

where X_{train} represents all the data available for training, $P(M)$ is the prior estimation for possible candidates of models, and normalization of the probability function is implicitly done. The next training step happens during initial calibration right after the production is finished, and before the device is delivered to the customer. At this stage, more device-specific data $X_{initial}$ become available, which also include data from a sweep test. The range of model candidates is again refined:

$$P(M|X_{train}, X_{initial}) = P(X_{initial}|M)P(M|X_{train}) \tag{6.11}$$

After being deployed at the customer site, the sensor and operational data of the device X_{deploy} was collected. This can be used to further fine-tune the selection of model candidates. As the model is later used for prediction, here, both ML and MAP approaches are suitable. For the MAP approach, model M is found by:

$$\begin{aligned} & M : \max_M P(M|X_{train}, X_{initial}, X_{deploy}) \\ where \quad & P(M|X_{train}, X_{initial}, X_{deploy}) = P(X_{deploy}|M)P(M|X_{train}, X_{initial}) \end{aligned} \tag{6.12}$$

Mathematically, these steps can be seen as successively applying Bayesian Interference to refine the selection of models as more evidence become available.

Finally, once a device broke down, all the history data from the device would be collected back. This would in turn be used to improve the selection range of models for all other devices, which is represented by prior distribution $P(M)$. The loop is thus closed and ensures that, with the transitory approach, data is used effectively to improve the performance at every stage, before a large enough quantity of data is collected for a final machine-learning-based approach. The whole process is illustrated in Figure 6.1. The characteristics of the transitory approach and the final machine learning approach are compared in Table 6.1. In light of these comparisons, in the long term, it is still recommended that the machine learning approach (or a derivative of it with some ideas borrowed from the transitory approach) is adopted due to higher accuracy.

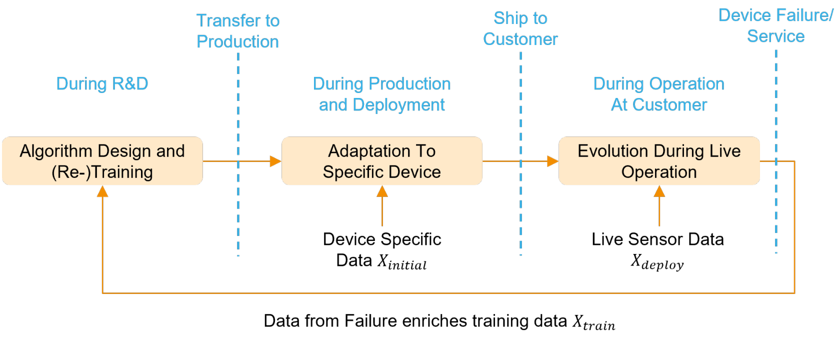

Transfer to Production
Ship to Customer
Device Failure/ Service
During R&D
During Production and Deployment
During Operation At Customer
Algorithm Design and (Re-)Training
Adaptation To Specific Device
Evolution During Live Operation
Device Specific Data $X_{initial}$
Live Sensor Data X_{deploy}
Data from Failure enriches training data X_{train}

Figure 6.1: In the proposed transitory approach, the prediction model is continually improved by efficient usage of all available data at all stages.

One should also keep in mind that this discussion is based on the edge-based deployment of the predictive maintenance algorithm. In a cloud-based solution, new data always become available for training. And with the availability of large centralized computational power, the model could be continuously improved. On the other hand, the model deployed based on the edge-based approach could also

Table 6.1: Comparion of the Long term Deep Learning based approach and the proposed Transitory Approach

Deep Learning Approach	Proposed Transitory Approach
The unknown underlying degradation models are learned from the data	Degradation models are hand-crafted from expert knowledge, data insights, and physical models
Advantage: • Can learn and represent very complex models • Can accurately map between each data point and the corresponding health state • Can implicitly account for variability between machines	Advantage: • Require little data • Can deliver reasonable results for cases, for which the model applies • The model still evolves to improve accuracy during operation at the customer, at a low computation cost
Disadvantage: • Require large dataset • High training cost in the beginning • Very costly to adapt live during operation at customer	Disadvantage: • Accuracy depends on the applicability of the hand-crafted model • Requires a lot of input from experts and assumptions

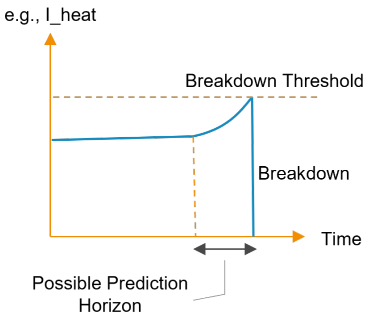
enjoy improvement if an Over-the-air update of the algorithm is possible. This will be discussed in Chapter 7.

6.1.2.4 Applying The Two-Block Algorithm Design For Detector

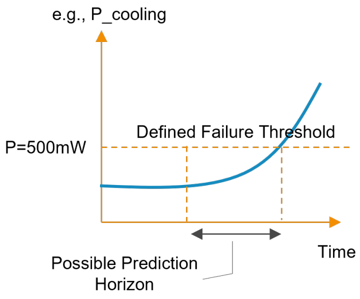
The transitory approach was developed with the X-ray tube in mind. However, the principle is general enough, so that it is also applicable to failure prediction for detectors. In fact, it is even simpler for the detector than the tube. This is due to a few differences:

- In the detector, there is only one operation condition, the detector was cooled down to a set temperature and function in that condition. In contrast, the tube can be set to various high voltage and anode current conditions.
- The measurement sensor in the detector could measure quantities, which are direct indications of the health status of the detector. The cooling current and voltages can be used to directly calculate the cooling power, which is dependent on the level of vacuum in the encapsulation. The actual temperature in the vacuum is also measurable, and any deviation from the set temperature is also indicative of a malfunction.
- For the detector, the failure threshold could be easily defined based on the manufacturer data sheet. The maximum supplied cooling power is one of them. The maximum allowable temperature deviation (i.e. the actual temperature should not be more than 1 degree off) could be used as another threshold.
- In the detector, the failure mechanisms manifest in the same data behavior. Either due to gradual or sudden loss in vacuum, the cooling power will generally be rising. The only spurious factor to this trend is due the ambient temperature, which could be eliminated by having an external temperature sensor. This is in direct contrast with the data from the X-ray tube, in which some quantities may rise or fall. When these reached a not yet well-defined threshold, breakdown occurs. In the aftermath, due to high voltage failure,

all these parameters assumed a signal level which is not relevant to its value before the failure. The situation with detectors is different. The signal first will stay relatively stable for a long time (same for the X-ray tube), during which no specific prediction could be given outside based on fleet-based statistics. Then the signal will rise up to and crosses a well-defined threshold as mentioned above. An illustration of the generalized behavior of the tube and detector up to and after the threshold can be found in Figure 6.2.



(a) X-Ray Tube



(b) Detector

Figure 6.2: General behavior of component data signals from Source and Detector in the time before, leading to, and after the point of failure.

All this behavior means that, for the detector, the requirements for the two Blocks in the algorithm are lower. The first Anomaly Detection Block does not need to account for the various operation conditions anymore. The threshold values required by this block are also very well defined without actually having to collect any data for it. The trend prediction block now has to deal with only similar trends. And the better measurement sensor embedded in the detector also enhances the accuracy of the trend and trend prediction.

6.2 Experimental Implementations Based On Initial Data

As mentioned in the previous section, with only a few data history runs to work with (3 history runs for 2 types of failure), a full realization was not attempted. In this section, some realization possibilities for the Trend Prediction block were explored. The goal is to show that a prediction of failure based on the collected data was feasible. For that purpose, two methods have been put to experiment, one for each type of failure. Both methods are classical (non-machine learning) methods, as they can be developed with very few to zero training data. The goal is then to use as much data as possible for validation. The meaning of the words "train" and "validation" in this work is different than that in machine learning literature. More specifically, due to the specificity of the classical approach, the model development process consists of 3 phases:

- In the "training" phase, the initial parameters of the model should be found from either one or zero full history run. For example, in the first following experimental method, the whole history (from $t = 0$ to $t = t_{failed}$) was used for finding initial parameters in the parametric curve fitting method. In the second experimental method, except for insight from the data, no training data was actually used.
- In the "inference" phase, only data from time $t = 0$ to a certain $t = t_{current} < t_{failed}$ was available. The data originates either from the very same data used in the training or other history run with very similar data. Only that this time, the model should be able to find out the failure time t_{failed} or the remaining useful lifetime without being able to "peek" into the "future" between $t_{current}$ and t_{failed}. This is also similar to real-life deployment conditions, where no data after the current time point could be used. In this way, the model demonstrates its ability to "recall" and provide prediction to failures with similar data "fingerprints".

- In the "validation" phase, the data is available up to a current time point ($t = 0$ to $t = t_{current}$) from **other** history run of the same failure mode. The model has never seen the data before. The goal is to verify the generality of the model. This is similar to the "test" phase and not the "validation" phase in machine learning literature.

For the failure related to the "Vacuum Leakage" failure mode, the prediction method based on fitting the Bezier curve was experimented with. For the "Burn-out Filament" failure mode, due to the nature of the data, exponential curve fitting was used to provide prediction with no training phase. The specifics of the methods will be discussed in the two following subsections.

6.2.1 Predicting Tube Failure Due To Vacuum Leakage

As discussed in Section 2.1.4, in the Vacuum Leakage failure mode, the X-ray tube itself becomes un-tight. It is suspected that air molecules get in the vacuum either obstructing the electron current from the cathode to anode or preventing the emission of electrons from the filament. To keep the same outputting x-ray intensity, the electronics would compensate by increasing the current through the filament. After a certain time, the compensation is no longer sufficient and failure happens.

For this failure mode, one history run was obtained. The data was already shown in Section 4.4.3. The analysis of data showed the behavior as described above with a rising heating current. This is the only quantity that shows changes relative early before the failure. Other quantities stay very stable, up to the very close to the failure point. With this observation, the trajectory of heating current data was thus used as the input to the algorithm.

When this trajectory is observed, it is clear that it comprises two portions. In the first one, the curve is flat and the data stays at the same level for a very long time. In the second portion, the data rises, or deviates, away from the flat level. There are generally a few explanations for the flat portion:

- The curve may have risen very slowly, but the measurement accuracy (8-bit in this case) was unable to record it
- During the flat portion, internal structural damage/fatigue or corrosion could have progressed. But as there was no vacuum leak yet, it did not lead to changes in data.
- The leakage may have been triggered by a random event (mechanical shock, thermal shock).

Thus, data in this flat portion could be considered as the "normal" state, which could last indefinitely. This means the data could not be used yet for failure prediction relevant to this specific device. However, as mentioned earlier, the prediction during the flat normal state might still be given based on failure statistics from multiple machines. Nevertheless, the data for which is not available in-house. As a result, such prediction in the normal state would not be carried out in the algorithm. When the data starts deviating from the normal states, only then pattern matching could be done on the data from this specific machine and gives a machine-specific prediction. A curve is fitted to the observed rising trajectory of data to determine the time point when the value of the heating current exceeds a threshold value. A visual illustration of the process is shown in Figure 6.3.

In the training phase, full training data (from $t = 0$ to $t = t_{failure}$) is available for finding curve parameters. This fitted "meta-curve" serves as a general pattern expected from the data. In the inference phase, this general pattern has to be continuously adapted to the observed data (from $t = 0$ to $t = t_{current}$) in deployment time to obtain the specific curve for this data run. from which the failure time is found and the remaining useful lifetime was calculated back. For the validation phase, it is not easy to find another dataset of the same failure type. The reason is, this is sporadic failure (happens by random chance). As a result, we opt to verify the generality of the approach, not the exact model. We applied the same steps of the approach to another dataset. This "water filtration" data set

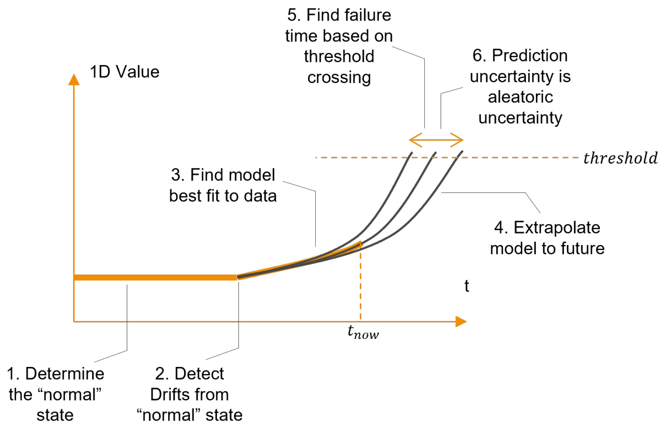

5. Find failure time based on threshold crossing
1D Value
6. Prediction uncertainty is aleatoric uncertainty
$threshold$
3. Find model best fit to data
4. Extrapolate model to future
t
t_{now}
1. Determine the “normal” state
2. Detect Drifts from “normal” state

Figure 6.3: General process for finding the time point of failure using curve fitting method on available data.

shares similar nature with the observed data from this failure mode. The meta-curve was trained from only one data run and should demonstrate its capability for prediction in other unseen data runs.

6.2.1.1 Approach

After the data deviates from the normal state, pattern matching could be done on the portion of data which is rising. The goal is then to predict the trajectory of the rise and to determine when it crosses the failure threshold. This corresponds to the second type of approach for realizing the Trend Prediction Block, mentioned in Section 6.1.2.2.

6.2.1.1.1 Bezier Curve Mathematically, the rising trajectory of data is usually described by either individual or a mixture of polynomial functions, exponential-logarithmic functions, and possible trigonometrical functions. However, for complex trajectories, there are a few problems with this approach. First, the composition of the mix of functions must be known in advance. Only the parameters of the functions may be optimized during the fitting process. Secondly, the composition of the function mixture may not be the most "economical way" to represent the data curve. This means that the mixture results in too many parameters to optimize, some of which may even cancel the effect of each other out. This leads to an ineffective optimization process or even an unbound process. Thirdly, some numerical issues may occur, especially when an exponential function is in the mix. The exponential function e.g. $y = ae^{bx} + c$ may have parameters that are many orders of magnitude apart, which cause trouble in the optimization/fitting process.

To better "encode" a curve, a Bezier curve is utilized here [79]. This is a type of parametric curve widely used in computer graphics. A parametric curve (in comparison to e.g. polynomial curve) does not define directly the relation $y = f(x)$ but relies on a parameter t to find the $(x(t), y(t))$ point corresponding to that t. The Bezier curve is used to represent a wide range of smooth curving trajectories in an economical way. A Bezier curve is defined by a certain number of control points. As t increases in the range $[0; 1]$, the point corresponding to that t will trace the trajectory of the curve. This trajectory spans from the first, towards the next control points, and ends at the last control points.

In our approach, the cubic (third-order) Bezier curve was used to approximate the trajectory of the portion of data towards the end of the lifetime. Such a cubic curve is defined by the control points P_0 to P_3. The point $B(t)$ on the curve corresponding to the value of parameter t can be found as:

$$B(t) = (1-t)^3 P_0 + 3(1-t)^2 t P_1 + 3(1-t) t^2 P_2 + t^3 P_3 \tag{6.13}$$

Notice that for $t = 0$, $B(t) = P_0$. As t increases, the point $B(t)$ moves closer to the other control points before it reaches $B(t) = P_3$ at $t = 1$. In practice, for the calculation, it is much more efficient to use the matrix form [80]:

$$B(t) = (x(t), y(t)) = TMP \tag{6.14}$$

where

$$\begin{aligned}
T &= \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \\
M &= \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \\
P &= \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}
\end{aligned} \tag{6.15}$$

Bezier curves provide a great advantage in flexibility. Normally, before curve fitting can commence, one has to specify the equation form of the model first (e.g. exponential $y = ae^x + b$, polynomial $y = ax^2 + bx + c$, and so on). This form requires manual programming and does not change during the fitting process. Instead of being able to specify only one equation form, the Bezier curve can be used as a "meta-model", being able to reproduce other curves by just varying the control points. This is illustrated in Figure 6.4.

In curve fitting algorithms, it is necessary to find points on the curves which correspond to each data point. The distance between the corresponding point is the quantity to be minimized. On a non-parametric closed-form curve (e.g. polynomial), if the data point is (x_d, y_d), the corresponding point would be $(x_d, f(x_d))$. From which the distance $|y_d - f(x_d)|$ is minimized. On a parametric curve

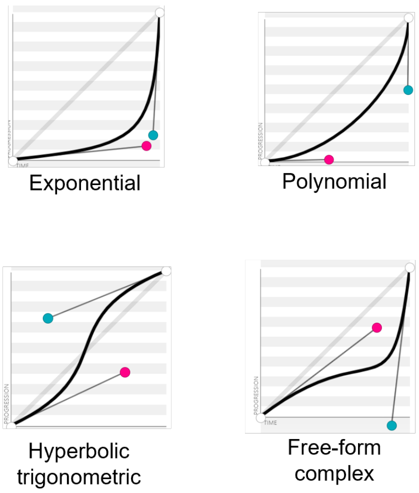


Figure 6.4: A cubic Bezier curve with four control points could reproduce a wide range of data trajectories [81].

such as the Bezier curve, the steps are more involved. First, the parameter t corresponding to x_d should be found by solving the third-order equation:

$$x_d = TMP_x$$
$$P_x = \begin{bmatrix} P_{0,x} \\ P_{1,x} \\ P_{2,x} \\ P_{3,x} \end{bmatrix} \tag{6.16}$$

From that, the y-coordinate of the point on the curve could be found by substituting t into:

$$y_B = TMP_y$$
$$P_y = \begin{bmatrix} P_{0,y} \\ P_{1,y} \\ P_{2,y} \\ P_{3,y} \end{bmatrix} \tag{6.17}$$

which defined the distance to be minimized as $|y_d - y_B|$. Note that the parameter t on the curve is different than the time t of each data point.

6.2.1.1.2 Training Phase The goal of the training phase is to find the best Bezier curve to fit the training data in the non-normal portion leading up to failure. For the cubic Bezier curve, this means determining the coordinates of 4 control points. With access to all data from $t = 0$ to $t = t_{failure}$, the last control point is easily determined at the data point of threshold crossing. The beginning point is determined by the point where the data starts deviating from the normal state and not going back. Due to the noisy nature of data, this requires a scanning routine which will be discussed next. The remaining task is the two remaining Bezier points (in the middle) so that the curve fits best to the data. In other words, the sum of squared distances between data points and the Bezier curve should be minimized. This quantity is denoted as the loss. The training objective would then be rephrased into an optimizing task: find the x-y coordinates of two Bezier points so that the loss is minimized. For this minimization problem, we used the SGD method and propagates the loss backward to adjust the coordinates of the points. All the relevant steps for the main training routine and necessary sub-routines are presented next.

For Preprocessing of data, the following steps are applied:

1. Smoothing/de-noising with Hamming windows

2. Down-sampling

For the training of the Bezier curve on known data run, the following steps are carried out:

1. Determine the characteristics (mean, std) of the flat/normal portion
2. Detect when data deviates from normal, trace back to the beginning of the deviation
3. Find the endpoint (data point (time, value) pair when the threshold is reached
4. Find a best-fit model by finding the two remaining Bezier points
    a) Set up SGD parameters, set up the coordinates of the two points as parameters to be optimized
    b) Finding the optimal parameters:
        i. Find the loss = error between data and Bezier curve
        ii. Backpropagation of loss
        iii. Update the parameter to improve loss
        iv. End when loss does not improve anymore

Step 2 basically performs a scanning routine to find the start point of the deviation portion of data. Due to the noise, the changes in signal could be falsely identified as the start of the rising portion. This means that the data rising trend has to continue rising above a threshold much higher than the noise level before it could be established that the data has deviated from the normal state and toward failure. On the other hand, when this threshold for rising is crossed, the starting data point of the rising trajectory was already behind in time and buried inside the noise. As a result, the approach here consists of two substeps: forward scanning to detect deviation from the normal state, and backward scanning to find the starting data point of the deviation. The algorithm for step 2 consists of the following substeps:

1. In a rolling window of data points moving forward in time:
    a) Calculate the mean error between data points z and characteristics of normal portion N: $\mu_z = \frac{1}{N}\Sigma_{i=0}^{N} \frac{y_i - \mu_N}{\sigma_N}$
    b) If the mean error is larger than a threshold in several consecutive rolling windows:
    c) Register the time point when the rising trend is detected
2. In a rolling window of data point moving **backwards**, starting from the time point found in step 1
    a) Calculate the mean and standard deviation of errors between data points and characteristics of normal portion $\mu_z = \frac{1}{N}\Sigma_{i=0}^{N} \frac{y_i - \mu_N}{\sigma_N}$ and $\sigma_z = \frac{1}{N}\Sigma_{i=0}^{N} \frac{(y_i - \mu_N)^2}{\sigma_N^2}$
    b) If mean and standard deviations of error are **smaller** than certain threshold values:
    c) Register the point as the starting point of the rising portion of data

In step 4(b)i, to find the deviation of the observed data points from the Bezier curve (that is to calculate the loss value), the following algorithm is applied:

1. Initiate control points of the Bezier curve
2. Find the data point with the highest y value
3. Find parameter t_{max} corresponding to this point
4. Sample 100 ts up to this t_{max}
5. Calculate (x, y) points on the curve corresponding to these ts. This yields the Bezier curve samples.
6. Find Bezier curve sample corresponding to each data point with the following routine:
    a) For each data point:

   b) For each Bezier curve points from the array above:

   c) If y of Bezier points > y of data points

   d) Two points are now "corresponding". The square of differences between them is accumulated.

7. Calculate the average of squares of differences by dividing the cumulative sum by the number of points

6.2.1.1.3 Inference Phase With the general pattern identified from the training stage, the inference phase would adapt this pattern to fit to the partial data which was available up to the current time point. The establishment of the normal and rising portion of data must be done without a full view of all history.

In the earlier stages, as only a few data points are available, the noise is comparatively large compared to the amount of data. Additionally, there could be also transient effects at the beginning. This makes determining the characteristics (mean and standard deviation) of the normal state unreliable. The inference module should wait until a certain time has passed until there is enough data to allow a normal state to be established.

After this, the module will scan the new incoming data for signs of deviating from the normal state. With the trajectory of data deviating more and more from the normal state towards the failure threshold, the curve fitting could then be done.

The difference in comparison to the training phase is that only a portion of the rising data is available. However, as the general pattern for the curve is already available from the training result, the meta-curve is then adapted to this partial data to find the specific curve. This is then used for prediction. The adaptation of the meta-curve to the curve for a specific device is based on the observation of other existing data sets. In which the data trajectories of devices/runs (of the same operational parameters) have very similar patterns, and differ only by an affine transformation. This will be illustrated later in the validation section of the method using the Water Filtration dataset. Based on this observation, the goal of the training module, during the curve fitting phase, is to find an affine transformation of the trained Bezier curve, so that the transformed curve best fits the partially-observed data. In this experimental realization, the scaling transformation in the x-direction was used for this fitting/adaptation stage.

In summary, the inference module acts as a simplified state machine with 3 states:

1. Waiting to establish the normal state
2. Observing the data and looking for the deviation from the normal state
3. After deviation is observed: fitting the curve and producing the prediction

For which the steps for the inference module are dependent on the state. The module is invoked every time new data is available. The module will check the data, determine its own state, and carry out the corresponding steps. In State 1, the steps are:

1. Check whether the length of the data array is long enough and the statistics of the data points are stable enough
2. If yes, determine the statistical characteristics of the normal data and then transition to State 2

In State 2, the steps are:

1. Detect when data deviates from normal
2. If yes, trace back the beginning of the deviation, transition to State 3

In State 3, the steps are:

1. Extract the partial portion of data starting from the beginning point of deviation
2. Find a best-fit model by finding the affine transformation of the Bezier curve obtained from the training phase:

   a) Set up SGD parameters, set up the affine transformation parameter as the parameter to be optimized
   b) Optimize the parameters:
      i. Find the loss by calculating the error between data points and Bezier curve
      ii. Backpropagate the loss
      iii. Update the parameters to minimize the loss
      iv. End when the loss does not improve anymore
3. Determine the aleatoric uncertainty of the fit
4. Find the time when the curves cross the failure threshold and produce the prediction

After step 2, the best-fit curve has been determined. Step 3 determines how uncertain the fit was, as a result of the noise in the data. This will be discussed next. Finally, in step 4, the threshold crossing is determined by an algorithm similar to the algorithm to find the y value given an x on the Bezier curve, only that in the other direction (namely finding x for a given y). The resulting x represents the time t at which the threshold value is reached, this is the time value of failure point t_{fail}.

6.2.1.2 Producing the Probability of Prediction Based On Aleatoric Uncertainty

Uncertainty from prediction can come from many sources and can be put into two categories:

- Aleatoric uncertainty arises from the noise or measurement error in data. In the XRF device, they are mostly due to the low bit resolution, and also due to the variation in signal when the device switches between operational states (e.g. during boot-up). To reduce this kind of uncertainty, one needs better measurement techniques (i.e. better sensors, alternative measurement methods, and so on).

- Epistemic uncertainty arises due to the lack of knowledge about the underlying model. Specifically, it comprises both the uncertainty in the form of the model (e.g. which type of mathematical equation) and the variances in the parameters of that form. To reduce this uncertainty, more data should be obtained to achieve a better approximation of the model.

With only one history run available for this failure mode, epistemic uncertainty could not be estimated (or is valued at infinity). The aleatoric uncertainty, however, can be estimated by how well the best curve fits the data points.

After the inference steps, the best parameters for the affine transformation have been obtained, which results in the curve which best fits the data points. Without loss of generality, these best parameters are denoted as one-dimensional α_0. To find the aleatoric uncertainty of the RUL, the aleatoric uncertainty of the parameter α of the curve must be found. To find this, the approach would answer the question: “What parameter α of the curve would lead to a curve, whose (y-) values is one (or k) standard deviation away from the best curve?” In other words, two αs should be found for two curves that have values 1 (or k) standard deviation(s) higher and lower than the best curve.

Answering the question requires one insight and one approximation. The important insight is that the Loss function used in the training/inference is related to the standard deviation. To demonstrate this, consider the best parameter α_0 with the lowest loss. This parameter generates a curve with points $\bar{z_i}$. The real data points are denoted z_i. The best MSE loss is then:

$$L_0 = \frac{1}{n}\Sigma_{i=0}^{n}(z_i - \bar{z_i})^2 = \sigma^2 \tag{6.18}$$

the best MSE loss L_0 (from α_0) is the square of standard deviation σ between data points and the best fit curve. The goal is to find α_+ which results in a curve with points $z_+ = \bar{z_i} + k\sigma$ and α_- resulting in points $z_- = \bar{z_i} - k\sigma$. Then the two curves resulting from α_+ and α_- will then be k standard deviation from the best curves. The loss value corresponding to the new curves with values $\bar{z}_i \pm k\sigma$ would be:

$$\begin{aligned} L_\pm &= \frac{1}{n}\Sigma_{i=0}^{n}(z_i - \bar{z}_i \mp k\sigma)^2 \\ &= \frac{1}{n}\Sigma_{i=0}^{n}(z_i - \bar{z}_i)^2 \mp \frac{2}{n}k\sigma\Sigma_{i=0}^{n}(z_i - \bar{z}_i) + k^2\sigma^2 \\ &\approx \sigma^2 \mp 0 + k^2\sigma^2 \\ &= (1+k^2)\sigma^2 = (1+k^2)L_0 \end{aligned} \tag{6.19}$$

For the curve which is one standard deviation from the best curve ($k = 1$), its loss value would be $L_\pm = 2L_0$.

Next, the parameters $\alpha_\pm$ which leads to losses $L_\pm$ should be found. For this, the loss value as a function of parameters represented by a curve on the $\alpha - L$ plane must be considered. Consider the curve in the local region near α_0, the curve lowest point will be (α_0, L_0). Other α parameters in the local region near α_0 lead to higher loss values $L > L_0$. Here, an approximation was made. The curve in the local region near α_0 was represented by a second-order curve:

$$L = a\alpha^2 + b\alpha + c \tag{6.20}$$

With this representation, the symmetric parabola has its lowest point at:

$$(\alpha_0, L_0) = \left(\frac{-b}{2a}, c - \frac{b^2}{4a}\right) \tag{6.21}$$

Resulting from this equation and the equation 6.19, the question could be rewritten as: finding $\alpha_\pm$ so that $L_\pm = (1+k^2)(c - \frac{b^2}{4a})$. The solution for $\alpha_\pm$ is:

$$\begin{aligned} \alpha_\pm &= \frac{-b \pm k\sqrt{4ac - b^2}}{2a} \\ &= \alpha_0 \pm \frac{k\sqrt{4ac - b^2}}{2a} \end{aligned} \tag{6.22}$$

Finally, one needs to find the a, b, and c coefficients of the local $\alpha - L$ curve. They would be found by numerical method. Starting from the best parameter α_0 obtained from the inference phase, a parameter scan is carried out. For each value in the array of $\alpha_\epsilon = \alpha_0 \pm j\epsilon$, a curve will be generated. From which the loss values L_ϵ corresponding to those will be calculated similarly to the training/inference subsection above. A second-order polynomial fit routine would take $(\alpha_\epsilon, L_\epsilon)$ pairs as inputs and return the a, b, and c coefficients. Finally, the steps in finding aleatoric uncertainty of RUL can be summarized in the following steps:

1. Starting from best parameter α_0
2. Generate array of $\alpha_\epsilon = \alpha_0 \pm j\epsilon$ in the local region of α_0
3. Calculate loss values L_ϵ corresponding to the array
4. Fit a second-order polynomial to the $(\alpha_\epsilon, L_\epsilon)$ values
5. Return a, b and c coefficients from the fit
6. Calculate $\alpha_\pm = \alpha_0 \pm \frac{k\sqrt{4ac-b^2}}{2a}$
7. For the parameter α_+ (similar steps applied for α_-)
8. Create a curve with parameter α_+
9. Find the time points where the curve crosses the failure threshold
10. Calculate the RUL = failure time - current time

The resulting $RUL_\pm$ would be the upper and lower bound of the uncertainty region.

6.2.1.3 Evaluation of Algorithm

In this section, the approach is evaluated with data set from real physical systems. Recalling the introduction to Section 6.2, evaluation could be carried out either as "inference" or as "validation". In both cases, the approach has access only to (real) data up to a certain time point and has to produce a prediction. The time point is then increased in the next steps and the algorithm yields the next prediction. The loop continues and all the prediction values are recorded and compared with the true failure point. The difference between "inference" and "validation" is that, in validation, the algorithm has to work with data unseen before to verify the generality of the approach.

For the vacuum leakage failure mode, there was only one data run. This run was already used to "train" the Bezier curve in the approach. Consequently, only "inference" is possible on this collected data run. To evaluate the generality, the algorithm was applied to another dataset available from the academic, which shares some similarities with the data set from the XRF device.

To evaluate the resulting prediction, it is important to consider the solution requirements as mentioned in Section 3.2.1. From those, the two relevant requirements are:

- The Prediction horizon should be early enough for the communication and planning of the customer and service team
- The prediction accuracy should be good, especially within the prediction horizon.

There are two possibilities to define the prediction horizon. It could either be defined as related to the time point, after which a prediction could be made. This is, however, very dependent on the data itself, when the data starts to be recorded, and especially on the data resolution. A more impartial metric could be the time point, after which the algorithm could produce the prediction better than a certain accuracy. In other words, a better approach should produce an acceptable prediction starting from an earlier time. The earlier the prediction, the earlier communication with the service team could take place. So that the service team can make a plan to visit the customer to carry out the repair before failure happen.

For the evaluation of the prediction horizon, the following metrics are evaluated:

M1 **Duration before failure, where prediction error < 8 hours**: within this time frame, planning could be done with an uncertainty of $\pm$ 1 working day

M2 **Duration before failure, where prediction error < 40 hours**: planning could be done with uncertainty of $\pm$ 1 working week

For the requirements of prediction accuracy, the metric often used in the Art is the average absolute prediction error (so-called "L1" error). Knowing the exact time point of failure, the customer can replan production to take into account the sudden decrease in capacity. The time duration, in which the error is calculated, still needs to be determined. This is dependent on the planning process of the customer. For the discussion here, a one to two-week time window before the failure was chosen. This results in the following metrics:

M3 **Average absolute prediction error from -40 hours to failure**: how accurate is the prediction within 1 working week before failure

M4 **Average absolute prediction error from -80 hours to failure**: how accurate is the prediction within 2 working weeks before the failure

6.2.1.3.1 On Collected Data

For the training and inference on the XRF data set, the signal for the filament current was used. As discussed in Section 6.2.1, the signal starts off with a flat portion, in which data-based prediction is not given. After the flat portion, the signal starts rising and the failure occurred as the current reached a threshold.

During the inference, the data up to a certain point is available. This is shown in Figure 6.5. The algorithm extracted only the rising signal portion. From this, an affine transform was found, so that the pre-learned Bezier curve fits the best to the data. Figure 6.6 visually demonstrates this process. The noisy data and the Bezier curve with 4 control points (dots) are shown. The end of the curve marks the point of threshold crossing (and thus failure). The bar at the endpoints shows the uncertainty of prediction resulting from the noise in data.

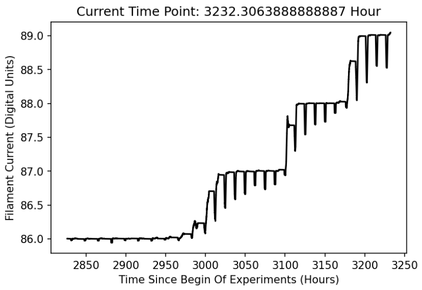


Figure 6.5: Available data (filament current values) up to a certain time point. Notice that it starts with a flat portion, which is the signal level in a healthy normal state. The prediction process could start once the signal starts deviating from the normal state.

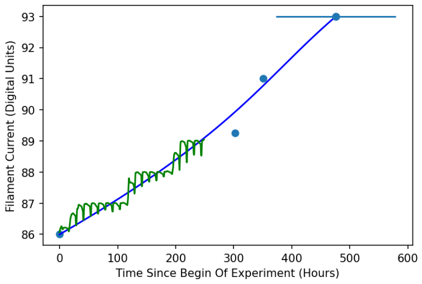


Figure 6.6: The portion of signal deviating from the normal state (green) is extracted and used for predicting the signal trajectory (blue). The dots are control points of the Bezier curve. The highest dots represent the failure time point. The horizontal line across the dot represents the prediction uncertainty.

After the prediction is found, the current time point is increased, more data become available and the next prediction is produced. Figure 6.7 shows how the prediction changes as the current time point advances in time. The horizontal axis marks the current time point (i.e. data is available up to this point). The failure prediction for the current time point is the dot, whose value on the y-axis shows the predicted failure time point. The horizontal line shows the true failure time point. The vertical bars around each dot show the prediction uncertainty at that time.

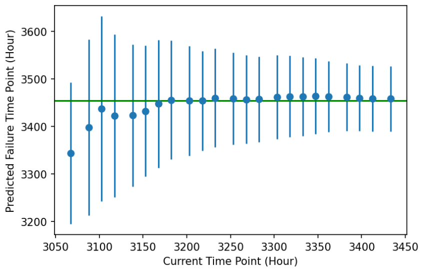


Figure 6.7: Each dot represents a prediction of failure time point (y-axis), given the availability of data up to a certain time point (x-axis). The vertical lines represent prediction uncertainty. The horizontal line (green) is the true failure time point. As time advances, more data is available and the failure time point could be predicted more accurately and more precisely.

From the graph, the approach was demonstrated to be able to produce a reasonable prediction for the failure time point. The accuracy of the prediction has a tendency of increasing over time as more data becomes available. The uncertainty also decreases with an increasing amount of data.

Table 6.2: Prediction performance for the Vacuum Leakage failure mode

Metric	Duration before failure, where prediction error		Average absolute prediction error, from X hours to failure	
	<8 hours (**M1**)	<40 hours (**M2**)	X = -40 hr (**M3**)	X= -80 hr (**M4**)
	higher is better		*lower is better*	
Result (hours)	50	330	4.98	6.94

The Table 6.2 show the evaluation of the inference using the metrics. Based on the results and the assumption of an 8-hour working day, and a 40-hour working week (which is applicable for 80% of the customer cases), the results could be interpreted as follows:

- From **M1**: Around one week before the failure, the day of failure could be predicted.

- From **M2**: The week in which failure happen could be predicted around two months ahead.
- From **M3** and **M4**: Within one or two weeks before failure, the exact half-day, in which the failure happen, could be predicted.

With regards to the requirements (Section 3.2.1), this performance should be just enough for the customer to plan their production around the failure. It should also be enough to contact Service and arrange a prophylactic repair before the actual failure. The experimental implementation show that the Predictive Maintenance could address the domain's problem, as discussed in Section 1.2.3.

It is important to note that, the performance here represents the best-case scenario as the data was seen during the training. However, due to the availability of only one data run, no further evaluation could be done. However, the generality of the approach could be verified using another dataset with similar characteristics. In this case, the Water Filtration data set was used.

6.2.1.3.2 On Transfer Dataset To evaluate the generality of the approach, the Water Filtration Data set was used. The approach is carried out in a similar way as for the XRF data. The difference now is the availability of multiple data runs for each failure mode. For this evaluation, one data run from one failure mode was selected for the training. In the "validation" phase, the algorithm uses the "trained" curve, processes the available data up to a current time point, and produces a prediction for another run, which it has not seen before. As the time scale in this data set is different than that of XRF data, the metrics M1 to M4 defined above would not be used. It also does not make sense to use them in this context. Instead, as the generality of the approach is the focus, only the prediction accuracy is used. Specifically, the prediction accuracy for the unseen data runs is compared to the inference accuracy of the data run used during the training.

The Water Filtration dataset originates from the Data Challenge of the European Conference of Prognostics Health Management 2020. A water circuit was built with a pump, reservoir, and water filter. The water contains particles of different particle sizes and in different concentrations. The pressure drop across the filter was recorded by sensors over time. As particle accumulates on the filter, the flow of water becomes obstructed, and the pressure drop increases. The failure threshold is defined at the time point when the pressure drop rises beyond 20 psi. This dataset has many properties which make it suitable to be used for the validation:

- Similar behavior as XRF data: in the data trajectory, there is also a flat portion at the beginning, after which the signal rises following a monotonic trajectory before crossing the threshold.
- Simple reduction to 1 feature: even though the dataset provides reading from many sensors, the prediction can largely be done from a single feature which is the pressure drop across the filter.
- Well-defined failure threshold: the failure threshold is a fixed value at 20psi pressure drop, regardless of whether the filter becomes completely obstructed or not.
- The rising portion in signal from the same failure mode has a similar signature. There are three different sizes for the contaminating particles. Each of them results in different shapes for the data curve, and can thus be considered as different "failure modes". However, for one particle size profile (even at different particle concentrations), the shapes of the curve are similar.

From the dataset, the data runs numbered 1 to 12 of the same "failure mode" are selected. The pressure drop over time for some selected data runs is shown in Figure 6.8. For the training, run number 3, which has the largest rising data portion, was selected.

During the evaluation, the available data up to a current time point was used as input into the algorithm to produce a prediction. The current time point was increased in increments of 10s. This increment value is selected by the dataset authors, however, any other increment would be suitable. With this size of

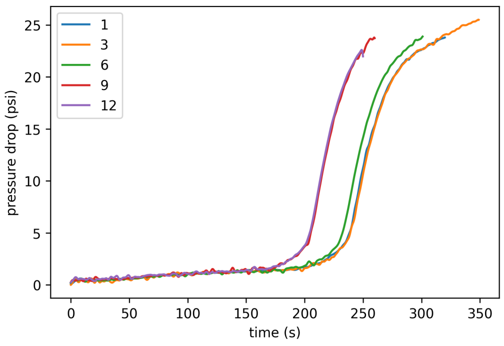


Figure 6.8: Exemplary data trajectories for different test runs of the same condition from the Water Filtration Data set. The curves are from runs 1,3,6,9, and 12 respectively. Notice the curves has similar general shape.

increment and the timescale of the data runs (each has a length of around 200 seconds with the rising portion lasting around 50 seconds), only 4 to 5 predictions were produced per data run.

For the evaluation, the task of "inference" was then conducted on data run 3 to serve as a baseline. For the "validation", the other data runs from 1 to 12 except 3 were then used as input into the algorithm. The average absolute prediction error was calculated for each case and the results are presented in Table 6.3.

The results from Table 6.3 have illustrated the generalization ability of our approach. The "meta-curve" was trained using only a single data run number 3. It was able to provide an acceptable level of performance for all other unseen data runs (so-called "test set"). It could even provide very good performance in some cases. In some other cases, the error was large. For this, the reason could be the incorrect determination of the starting point of the curve (i.e. the point at which data deviates from normal behavior). Clearly, the neural network shows superior performance to our approach. However, more data was available for the training

[1] The Neural Network was trained on a subset consisting of 25% of all data runs on the whole WF Data Set, including all different operation conditions. The results show the Mean Absolute Error when the network is tested on the whole data set.

Table 6.3: Prediction accuracy for the individual test runs. The accuracy from a state-of-the-art neural network (with much more data for training) is included for reference [82].

Data run number	Mean Absolute Prediction Error (seconds)
Data Runs Seen During Training	
3	0.52
Data Runs Unseen Before (Test Set)	
1	1.64
2	1.55
4	3.14
5	7.31
6	4.62
7	0.98
8	6.40
9	4.46
10	4.46
11	5.07
12	5.86
Our Approach Average	4.14
Neural Network Average[1]	1.169

of the neural network. In summary, our approach could provide an acceptable transitory solution before enough data could be obtained to train a neural network.

6.2.2 Predicting Tube Failure Due To Burned-Out Filament

As discussed in Section 2.1.4, and extensively in Section 5.1.2 in the Burned-Out Filament failure mode, the evaporation of tungsten material creates hot spots on the filament of the X-ray tube. The efficiency of thermionic emissions from those points is improved, but the filament gets quickly thinned out. This means the filament current required for the same anode current (and thus x-ray intensity) **reduces** with time. When the diameter gets small enough, the filament breaks at those points, preventing the filament current from flowing, at which the failure occurs.

For this value mode, two history runs were obtained. The data were already presented in Section 4.4.3. Those histories, as mentioned, were obtained not from normal usage of the devices by the customer. The tubes were already used for 61 and 53 thousand hours before being installed in the test setup. The device was modified to operate in the condition to accelerate the thinning-out process of the filament. Analysis of data and experiments in this case aimed to demonstrate that a failure prediction could be made for this failure mode.

The analysis of data, presented in the first subsection, shows that it is possible to predict the failure in the two collected runs without having to train a model. Details of this approach will be discussed in the second subsection. The actual evaluation of the approach on the two data runs is presented in the third subsection.

6.2.2.1 Analysis Of Data

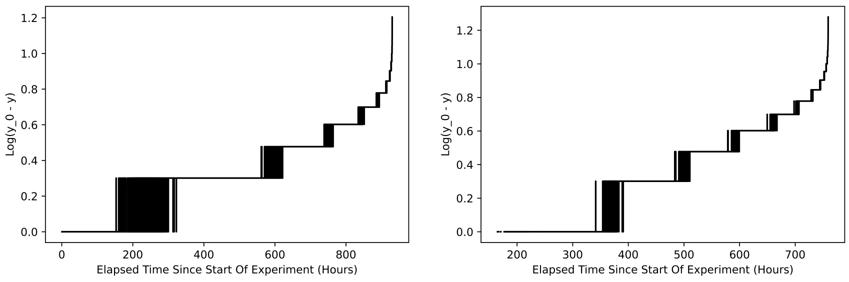
The analysis of data shown in Section 4.4.3 demonstrates the behavior of falling heating current. Similar to the previous failure mode, the heating current is the only quantity that shows changes relatively early before the failure. Other quantities stayed stable up to the point of failure. As a result, in this case, the trajectory of heating current data was also used as the input to the algorithm.

When the trajectories in Figure 4.4 and 4.5 are observed, it is visible that the falling rate of the heating current accelerates with time. Analysis and experiments were conducted on the two data trajectories. This delivered 5 key observations, of which the last one is the basis for the implementation of the prediction method for this failure mode.

The first observation showed the difference in signal levels between tubes, and the low resolution of the signal. During the test, one tube has the digital value of anode current decreased from 85 to 72, while the other from 87 to 76. This means that the tube-to-tube variant is large enough so that failure criteria defined by a threshold of signal value would not be suitable. It also means that there are only around 13 (and 11 respectively) digits in the difference between the start and failure point. This is the equivalent of $\log_2 13 = 3.7$ bits in digital resolution, which would further reduce the accuracy of the prediction, and delay the time point from which a prediction could be made reliably.

The second observation showed that, due to the nature of the ADC being used to sample the signal, it is difficult to determine the exact pre-sampled signal. When the signal is closely observed, it has a staircase shape with a series of horizontal landings and vertical edges. On the one hand, all the points on the landings have the same value, which was certainly not the case. It is thus hard to determine exactly which of those points at which time has this value. The exact determination of data points is more important in this failure mode than in the previous failure mode. This is due to the fact that in this failure mode, the signal showed exponential changes (rather than polynomial or linear changes). In this case, variances in data could result in vastly different prediction outcomes. On the other hand, the edges of the staircase shape were not clear-cut. They were composed of a series of spikes of data transition from one signal level to the next signal level. There was a hypothesis that, in certain ADC, the probability of signal level transitions (aka. bit flip) is proportional to the real signal levels. For example, if the real signal was 82.2, a series of 100 consecutive data points would contain around 80 readings with a signal level of 82, and 20 readings with a signal level of 83. In this case, the real signal can be reconstructed simply by using a moving averaging filter. However, this was not the case for the ADC being used. In fact, as evident from the very wide landings (with no spikes) and very narrow edges, when the real signal was near either end of the digits (e.g. $[82; 82.3]$ and $[82.7; 83]$), level transitions (or bit flips) did not happen with this ADC. However, around the edges, the transition densities get higher up to a certain point, before reducing as the signal settles in the new level. It is logical to assume that, at the point of the highest transition densities, the value of the real signal actually lies exactly between the digital levels (e.g. at 82.5). This means that these data points between the digital levels (call "edge points" could be determined with high temporal accuracy, and thus vital in determining the (exponential) trajectory of signal towards failure.

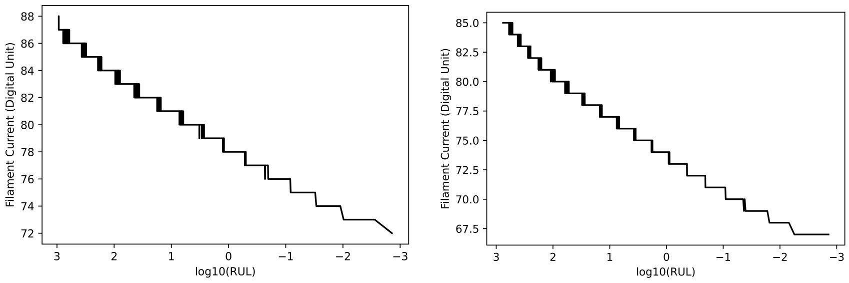
The third observation showed that the signal level did not fall exponentially with time. Normally, one would suspect that near the signal may change exponentially



(a) Tube with 53000 working hours before test **(b)** Tube with 61000 working hours before test

Figure 6.9: Against the expectation, plotting demonstrates that there is no linear dependence between $\log(I_0 - I)$ and t.

towards the point of failure according to the relation $I_0 - I = a \exp t + b$. To evaluate this hypothesis, the logarithm of the signal was plotted against time. If the hypothesis were true, then there would be a linear dependency between $\log(I_0 - I)$ and t, and the plot would be a line. However, as shown in Figure 6.9. This was not the case for both data trajectories.

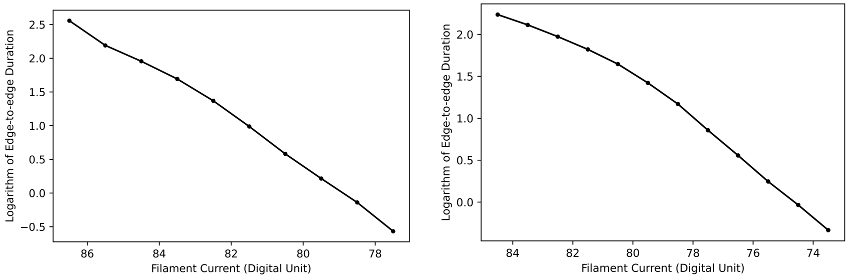


(a) For 53k Tube **(b)** For 61k Tube

Figure 6.10: Plotting confirms the linear dependency between I and $\log(t - t_{failure})$.

In fact, the fourth observation showed that the opposite of the hypothesis above was true. The remaining useful lifetime falls nearly exponentially with the signal, or $\exp I$ linearly dependent on (l.d.o.) $t - t_{failure}$. This is equivalent to I l.d.o. $\log(t - t_{failure})$. To evaluate the hypothesis, the input signal was plotted against the logarithm of RUL, as shown in Figure 6.10. The resulting signal is nearly linear. At this point, one idea of the realization of the algorithm would be to perform (linear) regression with $log(RUL)$ on the x-axis and anode current's digital value on the y-axis. However, this approach has one important problem, the x values depend on $t_{failure}$, which is itself an unknown. This approach would then need to be reformulated into an optimization problem of the form: find the parameters α, β and $t_{failure}$ to minimize loss $L = \frac{1}{N}\Sigma_{i=0}^{N} I_i - (\alpha \log(t_i - t_{failure}) + \beta)$. The optimization routine would take pairs of (I, t) data and try to adjust the parameters towards the points where the value of L is minimum. The optimization could be carried out with the SGD algorithm. This approach was tested and brought good results. However, in comparison to the final selected approach, it has a few disadvantages:

- It is less flexible as the model's function, namely $\alpha \log(t_i - t_{failure}) + \beta$, has to be defined beforehand
- It is more computationally expensive due to all the necessary steps of the optimization algorithm
- Similar to all other optimization routines, the convergence is dependent on actual values of hyper-parameters



(a) For 53k Tube **(b)** For 61k Tube

Figure 6.11: The plots confirms the linear dependence between I and $\log(\Delta t)$, which is utilized for failure prediction.

The final approach, which is simpler, more flexible, and delivers more accuracy, is based on another observation. When the logarithm of the time duration between edge points (e.g. $\log(\Delta t)$ between I =82.5 and 81.5) was plotted against the values of the edge points (e.g. 82.5, 81.5, 80.5 and so on), the graph shows nearly linear dependency. This could easily be shown mathematically. As shown in the last observation, $\exp I$ is l.d.o. $t - t_{failure}$. This results in $\frac{\delta \exp I}{\delta I}$ is l.d.o. $\frac{\delta(t-t_{failure})}{\delta I}$. It follows $\exp I$ is l.d.o. $\frac{\Delta t}{\Delta I}$. For the time duration between edges, the change ΔI in signal level is always the same (namely equals to 1). Thus $\exp I$ is l.d.o. Δt or I is l.d.o. $\log(\Delta t)$. This observation results in the simple approach of regression with the data points of $(I, \log(\Delta t))$. The key advantage of this method compared to the method from the previous observation is: the time duration Δt between edges does not require knowledge of the unknown $t_{failure}$. As a result, regression could be done directly. Regression is much more computationally efficient than optimization, with a deterministic number of linear algebra steps. On the other hand, if the regression of higher order is needed, it could be flexibly applied without changing the model. In the method, up to second-order regression was applied when possible, which could account for some non-linearity specific to each trajectory. In this case, flexibility also translates to higher accuracy of prediction.

Finally, a failure criterion (or threshold) must be defined for this approach, due to the (deliberate) explicit absence $t_{failure}$ from the mathematical basis. From the output of the regression, a series of $(I, \log(\Delta t))$ points in the future could be predicted. In this work, the failure criteria is defined as:

$$\log(\Delta t) = 0 \tag{6.23}$$

which means the edge-to-edge duration is less than 1 hour. Due to the exponentially decrease nature of Δt, the next edge-edge duration will be a lot less than 1hr and keep decreasing exponentially. This means that the actual physical failure time is very close to the time defined by this failure criterion. The most important benefit of this approach is that it is based on time and not the threshold of filament current values, which might be very different for different filaments.

6.2.2.2 Approach

Resulting from the above analysis of data, the approach basically performs regression on the $(I, \log(\Delta t))$ of the edge data points. The approach is thus composed of two subroutines. One to reliably identify edge data points, at which the value is exactly between the digital levels. The other subroutine performs the regression and predicts the time when the failure criterion is met.

The first routine should find the approximate temporal location (i.e. time) of the transition from one value to the next lower value. Essentially, it should register all the transition events, and find the location when the transition density is the highest.

1. Set up starting level = stable digital value of current signal
2. For incoming data point:
    a) If incoming data value = previous data value - 1
    b) Add the time point to an array
3. If stopping condition is encountered:
4. Find the mean of all time points in the array $\longrightarrow$ edge time
5. Starting level = current signal level

In step 4, the edge time could be reliably determined after a stopping condition was met and the array of transition time points was finalized. Due to the exponential nature of the curve, the edge points must be determined with high accuracy. Thus the array must also be well-determined. In this implementation, the stopping condition is when the signal starts to sink 2 digits below the starting level. In this way, all possible transition between the starting level to 1 digit below is definitely and completely captured. Too early stopping would lead to missing later transition time points and skewing the edge time. Too late stopping would make the predictor react late to the changing signal level, which could be quite fast towards the end of the device's lifetime. Note that the alternative implementation of using a rolling filter to find the edge point, which is based on fixed window size, would encounter difficulty. This is due to the exponential nature of the data, which requires the rolling windows to be exponentially shortened over time. The level of noise is another difficulty, which results in a "false positive" determination of edge points.

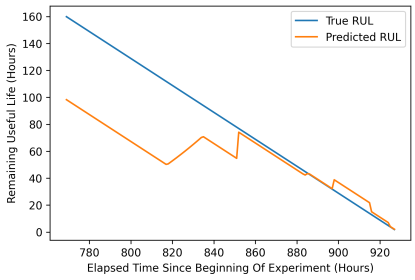
To start the regression, at least two edge-to-edge durations are required, which means at least three edges must be already detected. After which the prediction subroutine could be started.

1. Calculate array of edge-to-edge durations by subtracting detected edge time points
2. Calculate the logarithm of the durations $\log(\Delta t)$
3. If there are two durations available perform linear regression
4. else perform second-order regression
5. Until $\log(\Delta t) < 0$:
    a) Using the regression coefficients, find successively the next $\log(\Delta t)$ s
    b) Append the $\log(\Delta t)$ values to an array
6. From the array, calculate $RUL = \Sigma_i 10^{\log(\Delta t)_i}$

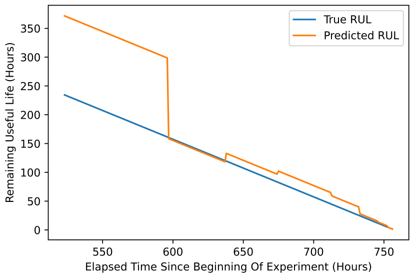
In steps 2 and 3, depending on the amount of available data, the best regression order could be flexibly determined. In this implementation, second-order regression was used whenever the data permits. This demonstrates an improvement in flexibility and accuracy over the approach mentioned in the last section (i.e. optimizing $L = \frac{1}{N}\Sigma_{i=0}^{N} I_i - (\alpha \log(t_i - t_{failure}) + \beta)$ using SGD). Most importantly, the algorithm above does not need a training phase, nor it requires any starting parameters in order to work.

6.2.2.3 Evaluation of Algorithm

In the approach, since there are no specific parameters that need to be adjusted, there is no "training" phase. As a result, when the approach is applied to the two available data runs (from tubes already with 53 and 61 thousand hours), all the results are basically "validation" results.



(a) For 53k Tube



(b) For 61k Tube

Figure 6.12: Plot of predicted remaining lifetime (orange) versus true remaining lifetime (blue).

Figures 6.12 show the Remaining Useful Lifetime (RUL) predicted by the approach against the true RUL. As the current time point increases, the true RUL decreases linearly [2]. The predicted RUL, shown in the same picture, should try to get close to the true RUL as possible. From the figures, a few observations could be made and explained as follows.

- The predicted RUL line has jumps: The jumps correspond to the time when an edge (transition in digital value) is reliably determined. With the newly detected edge, the edge time is used as new input to the algorithm so that

[2] in the so-called "linear RUL assignment scheme", there is also the "piece-wise linear RUL assignment scheme" [83], which is beyond the scope of this discussion

it could update the prediction. Between the edges, the signal stays at only one value and provides thus only limited information content.[3].

- The predicted RUL has improved accuracy with time: As more and more edges become determined, more points could be used for regression and prediction of the next edge. In the beginning, a bare minimum of 2 edge-to-edge durations were available, which makes only linear regression available. This combined with the logarithmic time scale would amplify the error when the final prediction is produced in units of hours (and not of log-hours). As more and more edges become available as input, the regression could be done in higher order, producing a more accurate prediction.

For comparison, two other approaches were also developed which also do not use the two available data runs. One of them is already mentioned above, in which the parameters α, β and $t_{failure}$ were optimized to minimize loss $L = \frac{1}{N}\Sigma_{i=0}^{N} y_i - (\alpha \log(t_i - t_{failure}) + \beta)$, where (t_i, y_i)s are pairs of edge time and edge value. This approach will be referred to as "Direct failure time determination".

An alternative approach utilizes a Long Short-Term Memory (LSTM) model to predict the next edge-to-edge time, given the previous edge-to-edge times [84]. LSTM can take in sequences of various lengths as inputs, and produce as output the next element of the sequence. LSTM is a variant of a Recursive Neural Network (RNN) but improves the dependence between the output element and the earliest elements in the input sequence (the so-called "Vanishing Gradient Problem" of RNN). To train the LSTM, we generate multiple artificial sequences of Δt_is using linear dependence $log(\Delta t_i) = \alpha - i\beta$ with multiple different (α, β) pairs and with i indexing the sequence. These training sequences approximate the sequence of edge-to-edge durations possibly encountered in real data. After being trained, the LSTM model is put into the main approach and replaces the regression step. The prediction is produced in a similar way to the main approach.

[3] The fact that the signal stays at a single level could be used for prediction updates in one scenario. In which, the algorithm has predicted an edge to happen at a certain time. When the time point is reached and the edge has not happened, the algorithm will take it into account. This correction behavior can be seen between hours 820 and hours 840 in the sub-figure on the left of Figure 6.12

Table 6.4: Comparing the different approaches for failure prediction

Approach	Dataset	Duration before failure, where prediction error <8 hours (M1)	Duration before failure, where prediction error <40 hours (M2)	Average absolute prediction error, from X hours to failure X = -40 hr (M3)	Average absolute prediction error, from X hours to failure X= -80 hr (M4)
		higher is better		*lower is better*	
Our Approach	53k Tube	76	100	4.2	4.4
	61k Tube	24	160	7.3	13.2
LSTM-based Approach	53k Tube	12	30	20.6	22.2
	61k Tube	11	44	12.1	19.6
Direct RUL Determination	53k Tube	3	76	14.8	12.8
	61k Tube	0	24	32	36

Table 6.4 shows the comparison between the approaches using the metrics M1 to M4 defined in Section 6.2.1.3. From the tables, a few interpretations could be made:

- The main approach of linear/higher-order regression produced the best results across the board. It is followed by the LSTM-based solution.
- The variance of prediction performances between the two tubes is non-negligible. As a result, it is not simple yet to make a definitive statement about the performance of the algorithm. Nevertheless, some semi-quantitative remarks could be made. For once, around 2 to 4 working weeks (40 hours) before the failure, the customer gets informed about the week in which the failure can happen. Within 1 to 2 working weeks before the failure, the exact failure day could be predicted. This information could be just enough for the customer to plan, and communicate with the service team for repair.

The better performance of the main method could be explained by its flexibility. For once, the order of the model could be dynamically adapted to the available data and shape of the data trajectory. The "Direct failure time determination" approach approximates first-order regression under the hood. To improve it with second-order regression, a new loss function $L = \frac{1}{N}\Sigma_{i=0}^{N} I_i - (\alpha \log(t_i - t_{failure}) \log(t_i - t_{failure}) + \beta \log(t_i - t_{failure}) + \gamma)$ has to be constructed. This results in a massive penalty in performance on the one hand. On the order hand, convergence issues and numerical issues may arise from complex transcendental loss functions. The LSTM-based approach has other issues related to all the (α, β) pairs that define the training set. Basically, the model only produces good results if the real data corresponds to the (α, β) it has seen before. Even for known (α, β), the performance may also be affected by many factors, such as the length of the noise in the input sequences. The biggest issue, however, is the behavior of the LSTM model on data outside the training range. The produced prediction could be completely off and could not be explained. All of the mentioned issues leave room for improvement in this approach.

6.3 Conclusion and Next Steps

In this chapter, an approach and experimental development of the algorithms for predicting failure were presented. The results demonstrate both the technical and business feasibility of predictive maintenance on the XRF device.

First, an approach and an architecture for the algorithm were presented to address the problem resulting from the lack of data. The approach is an transitory solution, in which each newly available data (from failure, from device-specific calibration, during operation) could contribute directly to improving the prediction performance. This continuous improvement is enabled by the algorithm architecture consisting of two blocks. The first block addresses the difference between individual devices and different operation conditions, and condenses the data into an "Anomaly Score". The second block observes and predicts the trends of this variable with time, so as to predict the failure point. This approach serves as an interim solution until enough data is collected for a full deep-learning approach.

With the limited amount of data available, two experimental algorithms were developed to address the Vacuum Leakage and Filament Burnout failure modes. Both can be assigned to the second block in the architecture above. In essence, they fit a curve to known data points and find out the time point when a failure threshold is reached to predict the failure.

For the Vacuum Leakage failure, the Bezier curve was utilized as a type of "meta-model" which can represent many types of models (exponential, polynomial, and so on) in a very economical way. To validate the generalization ability of the approach, it was tested on the Water Filtration data set and gives a decent performance on unseen test data.

For the Filament Burnout failure modes, data analysis has shown the unexpected linear dependence between signal value and the logarithm of the remaining life-time, i.e. between I and $\log(t_{fail} - t)$ (instead of the ordinary exponential relation between I and $\exp t$). As a result of that, the time duration from one bit flip to the next was used for failure prediction. The chosen approach utilizes regression of varying orders to find the best-fit curve to those durations. The result shows the chosen approach has better adaptability than more complex approaches and could deliver the required performance.

Even though the full algorithm has not been developed, the discussion in this section has demonstrated the possibility of doing Predictive Maintenance for the XRF device based on component data. The demonstrated performance could deliver enough prediction horizon and accuracy to address the domain issues of unexpected failure and efficient service.

However, there is still a long way to transfer these results to practice. The first block of the architecture still needs to be realized to address the varying operation conditions. For the second block, the chosen approaches should be adapted when the data of higher bit resolution become available in the future. In the far future, a deep learning-based solution should be considered when more data become available.

7 Integration of Predictive Maintenance In Product

In the last Chapter, we established that predictive maintenance is possible for the most common failure case of the X-ray tube. After further data become available in the future, the full predictive algorithm will be developed. Then it will need to be integrated into a product so that it can bring actual value to customers.

To offer Predictive Maintenance features to customers, there are two possibilities for centralized or distributed deployment:

- In a centralized deployment, the processing of data and production of results were done centrally, most probably on a cloud server. There is only one instance of the program. But as it processes different data from different devices, the result would be different. Nevertheless, it has access to data from all the available devices to further improve its performance
- In a distributed deployment, the processing of data and calculation of results were done at the local computer (also called the Edge). In this case, the data must not be sent through the Internet. Local processing also means that the availability of service is guaranteed and not dependent on internet connection or server uptime. The downside is that data from all devices are not sent back and thus do not contribute to improving the algorithm. The collection of data would require another system similar to that as presented in Chapter 4.

Multiple rounds of discussion with internal stakeholders were conducted and many possible features were presented. Based on the feedback of the stakeholders and the vision for the product portfolio, it is decided that predictive maintenance should be deployed as a value-added module inside the measurement software of the next generation. In other words, a distributed deployment was preferred. This is due to the capabilities and usage habits of the customer. Nevertheless, a centralized, cloud deployment was entirely possible in the future. In the meantime, the decision resulted in the architecture variant, in which the predictive maintenance features were closely integrated into the main measurement software.

This chapter discusses the concrete Predictive maintenance features in the software, their possible realization, and their integration into the main software. The first section describes the features which should be offered by the PM Module. The second section describes a proposal for software architecture to enable those features. In the final section, the design and partial implementation of the main components in this architecture will be described.

7.1 Predictive Maintenance Features In Software

To determine which features should be offered by the Predictive Maintenance Module in the main measurement software, many rounds of discussions and validations were conducted with the stakeholders including product management and service. In the end, the features are organized into two main value propositions, each corresponding to the descriptive or predictive aspect of the information related to the device. The "Measurement Reliability Indicator" value proposition focuses on the description, presenting information related to the current health status of the device fleet. The "Machine Lifetime Prediction" value proposition presents predictive information, offering foresight of the health status of the components. The two value propositions and their features are presented in Table 7.1. A concise description of the benefit of the features is also offered, whereby the benefits are described from the viewpoint of the end customer. The list of benefits for the service team would follow the discussion in Section 1.2.3. However, further formalization and validation are required as a part of the future work.

Table 7.1: Value Propositions and Features For The Predictive Maintenance Module In The Main Measurement Software And Proposed Benefits To End Customers

Value Proposition	Features - Short Description	Proposed End-Customer Benefit
Measurement reliability indicator	Asset Management Dashboard - overview of the status of all machines Reliability Index - overview of each machine's reliability in red-yellow-green lights	A quick, visual way to manage multiple machines and make sure they are all fully functional
	Component Health - detailed health status of the X-ray source and Detector	Customer can decide to exchange the components early to maintain certain asset availability (e.g., 99% uptime)
	Calculated Health Indicators – compare current value vs Ordinary value, serve as the basis for Health index	Customer can be shown clearly the basis of health calculation, based on real data from their machine and not made-up numbers
Machine Lifetime Prediction	Breakdown Probability Prediction - within next X thousand working hours	With knowledge of breakdown, Customer can decide to exchange component to avoid an unexpected stop in production
	History of Breakdown Probability – plotting probability against time	With the history of breakdown probability, customer can decide to exchange now or wait until the next regular service

7.2 Software Architecture Of The Predictive Maintenance Module

This section presents the concept for the architecture of the Predictive Maintenance Module which would enable the aforementioned features. In the first subsection, the architecture concepts of the main software were very shortly presented as a basic guideline for Predictive Maintenance Module. In the second subsection, some additional requirements for the design of the Module will be discussed. This is followed by a discussion of the proposal for the architecture and an explanation of the design choices.

7.2.1 Short Description Of The Architecture Of The Main Software

The Predictive Maintenance Features should be integrated into, and become a part of the main measurement software. It is worth noting that the integration target is the next generation (in development) of the main software, which is different than the main measurement being used during data collection in the Chapter 4. This new main software utilizes next-generation web technologies for development. With web technology, it is possible that the (web) user interface could be accessible via all devices in the network, even via smartphone, which allows multiple types of users to use it in multiple ways. Some users are interested in actually carrying out the measurement, other users may be just interested in checking the results with the smartphone.

Modern web technology also allows a component-based architecture. On the one hand, the software is developed in components. This allows a separation of presentation, logic, and data components, which brings about the advantage of the MVC design pattern such as improving the reusability, testability, and maintainability of the individual components [85]. On the other hand, during the actual starting process, the software can dynamically load a subset of suitable components, instead of just everything as is the case for monolithic architecture. In this way, the software can adapt to changes in software and hardware configurations.

7.2.2 The Predictive Maintenance Module(s)

As mentioned in the chapter introduction, the distributed deployment of predictive maintenance features was prioritized. This means that, in each copy of the main software, the PM Module should have the capability to do the whole process from data capture, processing, and result presentation.

The concept for the design of the Module should satisfy the following requirements:

- Similar architectural idea as the main software, i.e. component-based design, separation of components for front-end, back-end, and data access.
- Simple integration into the code base of main software: just copy the code base of PM to the main software, with minimal change in the original code base.
- Making use of the infrastructure of the main code: device communication, database models.
- Easy update of the algorithm without reinstalling the main program, which opens up the possibility for the over-the-air update.

Over-the-air update is desirable in the context of predictive maintenance algorithm. In the beginning, the algorithm which was put out to the market may be trained with less data. As more and more data become available, a better algorithm may be developed. In a centralized deployment, only one instance of the algorithm on the cloud should be updated and every device would enjoy the improved performance. However, due to the nature of the selected distributed deployment, the single instances of the PM algorithm on separate devices are different. The devices delivered earlier to customers would have older versions of the algorithm. They require to also enjoy the benefit of the latest algorithm. An Over-the-air update delivery process also does not require any time from the service technician or any human intervention at all.

With these considerations in mind, the Predictive Maintenance Module was also designed to be a collection of components. As illustrated in Figure 7.1, there are components inside the main software and one component outside the main software running as a separate process. The components will be discussed next in the order which follows the flow of data and information.

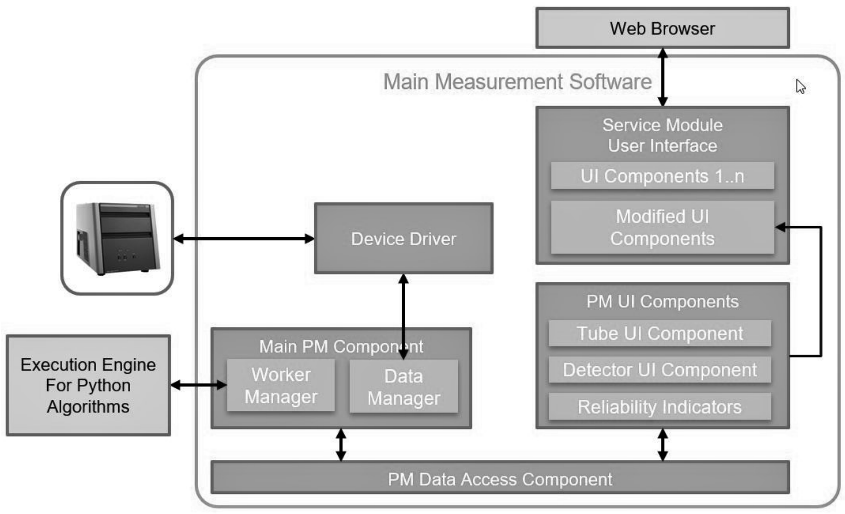


Figure 7.1: Component-based view of the proposed architecture of the Predictive Maintenance Module, embedded inside the main measurement software. The PM Module reuses the Device Driver and Service Module User Interface components, which were implemented by the Main Software. The Execution Engine component runs in a separate process from the main software and is thus drawn outside the "Main Measurement Software" frame.

The main PM component is first responsible for getting the data from the device. It does that by utilizing another existing component in the main software, which is the device driver. The PM component then sends the data to the external

Execution Engine of the Algorithm. The returning results and the collected data will be persisted in the local database. It does this by utilizing services offered by the PM Database access component. This component basically defines multiple database models for different types of component data and results (including remaining lifetime, reliability index, and so on). It also offers services for other PM-related components to save and access these data.

To present the results, the UI components first query the results from the database, again using the PM data access component. It then renders the UI elements with their results. Web technology allows the PM UI component to be embedded into other UI components of the main software. For example, the webview for the information related to the X-ray tube could additionally have one more indicator for its health status. This embedding operation can be done very easily during development time and does not require heavy modification to the UI of the main software. Nor it will crash the main software during runtime if the PM components do not function properly.

A special feature of this architecture is a separate external component for algorithm execution. On the one hand, it massively simplifies the process of integrating the developed algorithm into the main software. The main software is normally written in the programming language associated with the Web framework being used, for example, C# was used for the .NET framework. While Python has become an excellent choice for developing data science and data engineering-related algorithms [86]. This is due to the excellent capabilities in data processing and machine learning, and support from third-party libraries. An algorithm written in Python (possibly including some machine learning algorithm) could be integrated into the rest of the components on the main software side in three ways:

- Translation of the algorithm to the new language: tightest integration but require a lot of translation work. The target framework may not support the capabilities needed by the algorithm. This may require extensive re-implementation of many sub-algorithms and components or is just not possible.

- Tight coupling of the Python algorithm to the target code. In this case, the lines of code of the Python algorithm are embedded into the code of the main framework. This requires a third-party extension from the framework, which in many cases only supports a subset of the capabilities offered by Python and its libraries. The management of resources (memory, thread) must also be modified to fit into the target framework. All in all, the extensive re-implementation may be avoided but the integration work remains significant. A recompilation of the relevant component is also needed once the algorithm needs modification.
- Loose coupling to the code of target framework: in this case, a separate Python process is started to execute the algorithm. The algorithm can run unmodified with all capabilities enabled. Nevertheless, the transfer of data and results between processes requires additional communication, which may slow down the whole process, but real-time performance is not a requirement here. On the other hand, the code for communication needs only to be implemented once and is simply copied or embedded as a header into the next version of the algorithm file.

Only the third option enables the ideal scenario, in which the developed algorithm is handed over without any extensive re-implementation or modification to the code on the Python side and target platform side.

On the other hand, this architecture design choice also enables the possibility of an over-the-air update of the algorithm. This is facilitated by the fact that Python is an interpreted language and not a compiled language:

- In a compiled language, the source code is compiled into assembly or machine code before being delivered to the target platform for execution. This normally results in better performance related to interpreted code since the compiling steps are already done, and more optimizations could be done in compilation time.
- In interpreted language, an interpreter program will read, translate and execute the source code during run time. In this way, the source code can

be delivered to any of the target platforms, and the specificity of executing the code on the platform is taken care of by the interpreter program. This provides more flexibility and clarity in debugging but is also slower than (pre-)compiled code.

With a separate Python component for algorithm execution, a new version of the algorithm may be distributed/delivered in a form similar to a text file to the target platform. The text file may be encrypted for security and intellectual property protection. The distribution can be done en masse in the following way. The improved algorithm code could be put on a server. The local copy of the software would initiate the connection and pull the updated code to local storage. Without any further steps, the new features are now available. As a side note, pulling the update, whereby the local device initiates the connection from inside the firewall, is much easier than updating distribution by pushing. The reason is that, in the context of most customers, most firewalls will simply block a server trying to initiate a connection to a device inside the local network.

The architecture choice of a separate algorithm execution component also enables further extension in the future. In this scenario, the two processes for the PM main component and execution component may run on different computers. The sampling of data and long-term storage still take place on the local computer. In the meantime, the execution can be done on a dedicated server, or on the cloud, where more computational power, or acceleration hardware such as GPU is available. In the scope of this work, both processes are implemented on the same computer.

7.3 Component design

In this section, some important components of the Predictive Maintenance Module architecture will be discussed in more detail. The first subsection presents the software-side components with emphasis on the PM main components. The second subsection discusses the algorithm execution component. The inter-process communication between those two components will be the topic of the third subsection.

7.3.1 Software-side Component Design

On the main software side, the PM main component, the PM Data access component, and the PM UI components were designed and partly implemented.

The UI components will get the data and results from the database and present it to the user via a web interface. The exact "look and feel" of the components was conceptualized with the help of a UI specialist. The views were then presented to stakeholders and feedback was incorporated into the design. The implementation of those designs was facilitated by the web framework used by the main software. Modern web technologies enable a UI component to be embedded into another component, allowing a hierarchy of components to be presented as a unified UI. The designed views were broken down into sub-components, for example, a page could contain many indicator components, each of which comprises a number or text and a progress bar. Sub-components also consist of a code part, which defined the look of the component, and a code part, which gets the data and defines the logic of presenting the component. For example, the fill color of the indicator will turn from green to red if a component reaches a critical state. The sub-components will then be embedded into larger components such as pages or tabs. The PM UI modules at the end offer these larger components, which can then be embedded into the main program.

The PM Data access component provides data persistence for data and measurement results by defining data models. These are basically classes that represent a certain collection of data (e.g. tube data reading) or results (e.g. remaining lifetime, health status). The fields in the classes are individual parameters (e.g. voltage, current, and timestamp). During the application-building process, the routines offered by the web framework will parse those classes and create corresponding tables inside the database. One table corresponds to one class, and the table columns correspond to class fields. The component also makes use of other existing components in the main software to provide CRUD (create, read, update, delete) capabilities to other PM components which work with these classes.

The PM main component, as the center of the PM module, performs two roles: it coordinates the flow of data and results, and it manages the execution of the algorithm. During the start of the program, the component will be initialized. Upon which these preparation tasks are carried out:

- The module will make use of the Device Driver component from the main software to query for connected devices. It also sets up the interface from which the component data can be read out.
- A new process will be configured and started, which is the external algorithm execution component
- A channel will be set up for the communication of data and command between the processes. Further details about the communication will be discussed in Section 7.3.3.
- The communication endpoint, on the side of the PM main component, is set up. After that, it will attempt to establish a connection to the algorithm execution component, which should also have been started and been ready to connect at this point.

After the preparation steps, the PM main component will start three concurrent tasks. Each task implements a loop, in which:

- The sensor reading loop will query the component from the device periodically for sensor values and operational parameters.
- The decision loop will check the conditions for executing the algorithm. Algorithms for predictive maintenance generally would not execute every time a new data sample is available. The reason is to minimize unnecessary computing load, and the slow degradation phenomenon as targeted by this work will not dramatically change within a time scale of seconds. If the condition is met, it will send a command to the algorithm execution

component to start the calculation. It will also handle sending the necessary data.

- The receiver loop will be attached to the communication endpoint. It will look for the responses from the execution component, such as acknowledgment of command, or return of algorithm results.

In many cases, waiting for the execution of the algorithm to finish may take a large amount of time. In the meantime, the receiver loop should not block other loops, especially the loop for periodical sensor reading. To make the operations not blocking another there are generally two implementation options in the software for concurrency:

- In thread-based concurrency, all the sub-programs are carried out (quasi-) at the same time. Execution time is distributed fairly to all threads through true hardware parallelism (multiple CPU cores) or apparent parallelism (the OS will take care of choosing which instruction from which thread to execute on one CPU core). Thread-based parallelism is suitable for the CPU-bound scenario, in which threads perform complex computation.
- In asynchronous concurrency, all the sub-programs are carried out inside a so-called event loop. Each program will take turns to execute until it reaches a waiting point, where it waits for a signal or a resource to become available before it can continue with the next instruction. At this point, the execution is yielded back to the event loop and the loop will continue the execution of another sub-routine. Asynchronous concurrency is particularly suitable for routines that are IO-bound. Those routines will do little calculation so parallel computation is not needed. However, they will spend a lot of time waiting for different time-consuming operations to finish such as reading from the hard drive or waiting communication response.

In the implementation of the components, both forms of concurrency are utilized. The three loops of the PM main component do not require heavy calculations but mostly wait for the sensor value to be read, for a certain condition to be met before sending a message, or for the return message. For this reason, the asynchronous concurrency form is applied. The scenario, when both types of concurrency are needed, happens at the algorithm execution component, which is the topic of the next subsection.

7.3.2 Algorithm-side Component Design

The algorithm execution component receives data from the main program, perform the Predictive Maintenance algorithms to produce prediction results, and sends them back to the main program side for storage and display. It runs on a separate process started by the PM main component. Upon starting, it will:

- set up a communication endpoint in the main channel created from the PM main component
- establish the connection with the PM main component by exchanging handshake messages
- set up another thread that will carry out the computation work when a command is received. The algorithm will perform heavy and CPU-bound computation. As a result, the thread-based concurrency mentioned in the last subsection would be applied here.
- start the receiving loop and wait for data and command from the PM main component. In this way, the algorithm execution component acts as a server (in the server-client model), to which the client (PM main component) sends requests. Upon which it will perform the task and return the result in the response.

After these preparation steps, the component awaits in the receiving loop. Once a message was received, it is parsed to obtain the instruction (and the corresponding data). The request was acknowledged. After that, the function, in which the predictive maintenance algorithm was implemented, was queued to the computation thread. After the result is available, the results are sent back to the PM main component.

7.3.3 Inter-Process Communication

As the PM component and the algorithm execution component run on two separate processes, an intra-process communication method would normally not work. Those include function calls, passing around variables, references, or pointers. For inter-process communication, other methods should be applied, which generally include:

- Shared memory or shared storage. The sender stores the content of the message in a shared region of memory or a file. It then notifies the sender about it. This method allows a large amount of data to be exchanged very efficiently. However, the shared region must be well managed, to avoid race conditions or being exploited by malicious agents. The method also limits the processes to be on the same computer.
- Message passing on a direct channel. The sender and receiver are connected on a direct channel, such as named pipes or sockets. The content is wrapped inside a message and passed around in the channel. This method is less efficient as it required more overhead, and the content to be copied from one process to another. However, this method allows the flexibility that processes being on a different computer, maybe even over the internet.
- Message passing using a queue or a broker. In this case, there is a third-party agent, with which both the sender and receiver exchange messages. This has similar advantages and drawbacks as the message passing on a channel. In addition, it allows more features such as multi-cast of message storage. However, it also requires that either the OS provides this third-party (e.g. a FIFO queue), or an external broker (e.g. MQTT) should be installed and maintained.

In the case of the communication between the PM main component and execution component, message passing on a direct channel is most suitable. The shared memory techniques do not allow the future planned feature of the process to run on different machines. And the flow of messages between the process is simple enough so that no third-party broker is needed.

From the implementation point of view, during the start of the components, the communication is set up following these general steps. First, a communication channel should be created. For example, the program can ask the OS to create a named pipe or ask for a free socket (or a local loopback interface if both processes are on the same computer). Then, endpoints should be created to read from or write to the channel. For the named pipe, normally a file handle is created, to which the data could be written to the pipe as if it was written to a file. For sockets, a TCP client object could be created, which opens a stream object for reading and writing. After this step, a connection should be established between the parties. This involves exchanging handshake messages to acknowledge the existence of other parties and exchanging details, such as authentication, message format, buffer capacity, and so on. Finally, both party enters a receiving loop. In the loop, each party waits for the incoming message. The incoming message could then be decrypted, verified for integrity, and parsed. During parsing, the message header is analyzed to get the exact requests, instructions, and the format of the content. With the message content analyzed, the work can be queued or executed. The receiving loop then sends an acknowledge message and if possible the results back. After that, it will come back to waiting for the next incoming message in the next iteration of the receiving loop.

One drawback of message-based communication is the reduced performance due to overheads and copying the message around. Those cannot be avoided but can be minimized with the help of efficient message serializers such as Protobuf [87]. A message serializer essentially converts an object into a series of bytes or characters so that it can be sent through a communication channel. At the receiver end, the series of bytes/characters is "de-serialized" and converted back to an equivalent object. Common serialization techniques include XML and JSON, both converted the object to a string representing a (nested) tree structure of key-value pairs. This allows readability and flexibility, as the data structure is embedded together with the data content in the serialized string. However, the inclusion of a similar data structure is in many cases redundant, as the same types of objects get exchanged, the same structure gets repeatedly communicated. On the other hand, string format might not be necessarily efficient for exchanging number-based data. Protobuf is a serialization library from Google. It converted objects directly to binary format using a very performant serializer, which also supports multiple languages. The data structure is not embedded in the content but must be declared prior in the so-called Proto-files and made available to both communication parties. This, however, does not completely hinders the flexibility, as optional fields and repeated fields can be included. In summary, the pre-defined format and binary encoding of messages reduce the message size and make copying data in IPC much more efficient. While the efficient serializer reduces the overhead computation needed. Both made the direct message-based communication between the PM component and the execution component much more efficient.

7.4 Conclusion

This chapter presented the concrete features of Predictive Maintenance in the main software and discussed their benefits to the end customer. To realize the feature, a concept for the architecture of the module was presented to allow effortless integration into the rest of the software, as well as enable Over-the-air update of the everchanging PM algorithm. Based on this, the detailed designs of the components were presented. Except for the algorithm itself, the majority of the software components (including Algorithm Execution, Main and UI Components) of the module were already implemented and have been presented internally as a proof-of-concept. However, the results would not be presented here in this work following Non-disclosure Agreement with the XRF device manufacturer.

8 Conclusion

8.1 Conclusion

In this work, a complete workflow for the development of a data-based Predictive Maintenance Solution for the X-Ray Fluorescence Measurement Machine from a Small-Medium Enterprise was presented. The workflow encompasses all the steps from data collection, data analysis, and algorithm development up to integration into the product. In each step, architecture for the technical systems was developed based on state-of-the-art techniques and real-life requirements. A partial realization of the technical systems for all the steps was carried out. The results demonstrated the possibility of failure prediction despite the low resolution of data. A technical method for improving the prediction performance was theoretically and practically demonstrated.

Chapter 4 discussed the technical and non-technical aspects of collecting the device failure data. For this, a Data Collection campaign was coordinated and carried out. The aim was to collect data from both customer devices and devices in the laboratories and showrooms. To enable the campaign, a Data Collection System was conceptualized and partially realized, which covers all the aspects of Data Engineering workflow from data generation, ingestion, storage, and transformation up to facilitation of data usage. To address the data protection concern, the system also includes the role of the service technician for the secured transfer of data from customer devices from other countries back to the central data server on-premise. Inspection of the failure data revealed that the Filament current was the key signal in predicting device failure. However, the signal was sampled with an 8-bit ADC, which leads to many difficulties such as a decreased prediction horizon and problems in determining the true signal level for failure prediction.

Chapter 5 introduced high-precision digital measurement as a method for improving the prediction performance. First, a physics-based simulation model was developed to reconstruct the signal over its entire lifetime at high fidelity. It was found that it is important to take into account not only the evaporation of tungsten from the surface but also the phenomenon of a local hot spot. Analysis shows that 16-bit measurement of component data would extend the prediction horizon by at least 20-fold and cover essentially the entire lifetime of the tube. Besides that, a 12-bit measurement could be a good cost-performance trade-off. Based on promising theoretical results, an experimental test setup was built. The hardware XRF device was extended with a high-precision 16-bit data logger. Software was developed to preprocess a huge amount of data. Even though failure has not happened yet to the test setup, collected data confirms the hypothesis by showing changes very early (even since the start) in the lifetime.

Chapter 6 discussed the algorithmic aspect of failure prediction for the X-Ray tube. First, an approach and an architecture for the algorithm were presented to address the problem resulting from the lack of data. The proposed algorithm was proposed to compose of the Anomaly Detection Blocks and the Trend Prediction Block. This enables reasonable prediction results with few failure data. More importantly, any new incoming data either from calibration, from normal usage, or from failures of other devices can be used very efficiently to improve the prediction performance. With the limited amount of data available, two experimental algorithms were developed to address the Vacuum Leakage and Filament Burnout failure modes. For the Vacuum Leakage failure, the Bezier curve was utilized as a type of "meta-model" which can represent many types of models (exponential, polynomial, and so on) in a very economical way. This can be fit to historical data and predict future data points, from which the time point of failure could be deducted. For the Filament Burnout failure modes, linear dependence was found between signal value and the logarithm of the remaining lifetime, i.e. between I and $\log\left(t_{fail} - t\right)$. Based on this insight, regression based on the time duration from one bit flip to the next was used for failure prediction. The chosen approach has better adaptability and performance than more complex approaches and could predict a failure within two weeks in advance. This leaves enough time for the customer to react and for the service team to schedule a repair before failure happens.

Chapter 7 presented the concrete features of the Predictive Maintenance in the main software and discussed their benefits to the end customer. A software architecture for integrating PM into the software, which enables the Over-the-air update of the algorithm, was also presented. The majority of the software components from the architecture were already implemented and have been presented internally as proof-of-concept.

All in all, the most significant contribution of this work was to lay out not only a solid foundation but also to make important first steps towards a full Predictive Maintenance Solution for the Product of the SME. The proof-of-concept showed that a Predictive Maintenance solution that can address the domain problem as mentioned in Chapter 1 was possible. The workflow, approach, and architecture could very well be reused by similar SMEs to develop their own PM offerings. In practice, the creation of such a data-based system will encounter various hurdles, most importantly the worry about data protection. Herein lies another significant contribution, this work has presented some suggestions for both technical systems and non-technical steps, which addresses this worry adequately, while enabling innovative data-based solutions to be developed.

8.2 Recommendation And Future Work

Based on the results of the work in Chapter 5, a clear recommendation could be formulated for incorporating measurement hardware of 12-bit precision or more in future devices. There is the possibility that with the new hardware (and thus new data resolution), the whole data collection campaign will need to start anew. However, it is expected that the actual behavior of the physical system will not change drastically in response to the change in electronics. As a result, the insights from the data analysis and prediction algorithm may well be reused for the new data.

A full realization of the algorithm still requires the realization of the Anomaly Detection Block. In order to properly realize this block for the X-Ray Tube, the behavior of the tube at any operation condition has to be captured during the whole lifetime of the tube. In practical terms, a continuous, repetitive sweeping of operational parameters and concurrent collection of data has to be conducted. Once this data is available, the techniques for realizing the first block, mentioned in Section 6.1.2.1, could be applied.

The failure prediction algorithm for Detector should also be realized. However, the Detector has essentially no wear-and-tear part, thus a naturally happening degradation is not a case, and failures happen mostly by chance. Artificially replication of failure circumstances should be attempted, so that the detector could be pushed into the failure scenario, and data could be captured. One method is to purposefully create a defect on the shell of the detector module, and operate the module in a humid and corrosive environment.

It is also recommended that the device failure statistics should be captured and stored for all components on the field. Such statistics are very helpful not only for internal departments (such as Quality Control) but also enables Predictive Maintenance solution to give a prediction in cases of the absence of any observable change in data. This could be done using techniques from the field of Reliability Engineering (e.g. Weibull method) or Survival Analysis (e.g. Cox method). The generated data could be easily ingested and stored in the Big Data server realized by this work.

Finally, the data collection architecture and the over-the-air updateable software module realized in this work could be used to serve new and innovative value propositions. It could assist in realizing the subscription pricing model or the cloud computation model. Further value propositions are possible considering the specific scenarios of the countries of the customers. All of those should be investigated.

List of Figures

List of Tables

List of Publications

Conference contributions

[Vu et al.(2020)Vu, Stock, Fan, and Stork] Con Tran Vu, Simon Stock, Lintao Toni Fan, and Wilhelm Stork. Highly parallelized rendering of the retinal image through a computer-simulated human eye for the design of virtual reality head-mounted displays. In Peter Schelkens and Tomasz Kozacki, editors, *Optics, Photonics and Digital Technologies for Imaging Applications VI*, volume 11353, page 1135316. International Society for Optics and Photonics, SPIE, 2020. doi: 10.1117/12.2555872. URL `https://doi.org/10.1117/12.2555872`.

[Vu et al.(2021)Vu, Chandra-Sekaran, and Stork] Con Tran Vu, Ashok Chandra-Sekaran, and Wilhelm Stork. A deep learning first approach to remaining useful lifetime prediction of filtration system with improved response to changing operational parameters using parameterized fully-connected layer. In *PHM Society European Conference. 6*, pages 439–447, June 2021. doi: 10.36001/phme.2021.v6i1.2823. URL `https://papers.phmsociety.org/index.php/phme/article/view/2823`.

Bibliography

[1] B. bmbf.de, “Industrie 4.0.” https://www.bmbf.de/bmbf/de/forschung/digitale-wirtschaft-und-gesellschaft/industrie-4-0/industrie-4-0, 2011. [Accessed 31-May-2023].

[2] IBM, “What is Industry 4.0 and how does it work? | IBM — ibm.com.” https://www.ibm.com/topics/industry-4-0, 2023. [Accessed 31-May-2023].

[3] i SCOOP, “Industry 4.0 and the fourth industrial revolution explained — i-scoop.eu.” https://www.i-scoop.eu/industry-4-0/, 2023. [Accessed 31-May-2023].

[4] M. . Company, “What are Industry 4.0, the Fourth Industrial Revolution, and 4IR? — mckinsey.com.” https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-are-industry-4-0-the-fourth-industrial-revolution-and-4ir, 04 2022. [Accessed 31-May-2023].

[5] H. Leurent and E. de Boer, “White Paper: The Next Economic Growth Engine - Scaling Fourth Industrial Revolution Technologies in Production,” tech. rep., World Economic Forum, 01 2018.

[6] K. Shaw, “802.11x: Wi-Fi standards and speeds explained — networkworld.com.” https://www.networkworld.com/article/3238664/80211x-wi-fi-standards-and-speeds-explained.html, 2022. [Accessed 31-May-2023].

[7] P. Sachithanandan, “From SaaS to StitchFix: A Look Back at the History of the Subscription Model | Upscribe — upscribe.io.” `https://www.upscribe.io/blog/from-saas-to-stitchfix-a-look-back-at-the-history-of-the-subscription-model/`, 2021. [Accessed 31-May-2023].

[8] T. McDonald, “The Industry Handbook: Software Industry — investopedia.com.” `https://www.investopedia.com/articles/markets/050416/industry-handbook-software-industry.asp`, 2022. [Accessed 31-May-2023].

[9] J. Frankenfield, “What is Cloud Computing? Pros and Cons of Different Types of Services — investopedia.com.” `https://www.investopedia.com/terms/c/cloud-computing.asp`, 2023. [Accessed 31-May-2023].

[10] M. Grant, “Software as a Service (SaaS): Definition and Examples — investopedia.com.” `https://www.investopedia.com/terms/s/software-as-a-service-saas.asp`, 2022. [Accessed 31-May-2023].

[11] A. Barseghian, “Council Post: What’s Behind The Rise Of The Subscription Model? — forbes.com.” `https://www.forbes.com/sites/forbestechcouncil/2019/08/12/whats-behind-the-rise-of-the-subscription-model/?sh=506fa48135c3`, 2019. [Accessed 31-May-2023].

[12] R. Iyengar, Y.-H. Park, and Q. Yu, “The impact of subscription programs on customer purchases,” *SSRN Electronic Journal*, 2020.

[13] G. Gruman, A. Morrison, and T. Retter, “White Paper: Software Pricing Trends - How Vendors Can Capitalize on the Shift to New Revenue Models,” tech. rep., PricewaterhouseCoopers, 01 2007.

[14] L. Ubbaja, “The Subscription Business Model: How to Sell Once, Earn Forever — uscreen.tv.” `https://www.uscreen.tv/blog/subscription-business-models/`, 2023. [Accessed 31-May-2023].

[15] P. Srivastava, “History, Present, and Future of the Subscription Business Model — startuptalky.com.” https://startuptalky.com/subscription-business-model/, 2022. [Accessed 31-May-2023].

[16] P. Mukhopadhyay, “Council Post: How A Subscription Business Can Increase Business Valuation — forbes.com.” https://www.forbes.com/sites/forbestechcouncil/2020/10/26/how-a-subscription-business-can-increase-business-valuation/?sh=3bd367b2e76d, 2020. [Accessed 31-May-2023].

[17] cleverific Order Editor, “The subscription business model: A how-to guide — Edit Order — editorder.net.” https://www.editorder.net/guide-to-subscription-business-model, 2023. [Accessed 31-May-2023].

[18] HeidelbergerDruckmaschinen, “Subscription Smart & Plus | HEIDELBERG — heidelberg.com.” https://www.heidelberg.com/global/de/services_and_consumables/print_site_contracts_1/subscription_agreements/subscription_1.jsp. [Accessed 08-Jun-2023].

[19] M. Choi, S. Moss, J. Nading, E. Reasor, and D. Remley, “Creating consumer—and business—value with subscriptions — mckinsey.com.” https://www.mckinsey.com/capabilities/growth-marketing-and-sales/our-insights/sign-up-now-creating-consumer-and-business-value-with-subscriptions, 05 2021. [Accessed 31-May-2023].

[20] eurostat, “EU small and medium-sized enterprises: an overview — ec.europa.eu.” https://ec.europa.eu/eurostat/web/products-eurostat-news/-/edn-20220627-1, 06 2022. [Accessed 31-May-2023].

[21] A. Tukker, “Eight types of product–service system: eight ways to sustainability? experiences from SusProNet,” *Business Strategy and the Environment*, vol. 13, pp. 246–260, jul 2004.

[22] C. Lerch and M. Gotsch, “Die rolle der digitalisierung bei der transformation vom produzenten zum produzierenden dienstleister,” *Die Unternehmung*, vol. 68, no. 4, pp. 250–267, 2014.

[23] V. Lugovsky, “Council Post: Industry 4.0: The Beneficial Trends And Challenges For SMEs — forbes.com.” `https://www.forbes.com/sites/forbesbusinesscouncil/2021/09/03/industry-40-the-beneficial-trends-and-challenges-for-smes/?sh=26615b4e293c`, 2021. [Accessed 31-May-2023].

[24] E. Bollhöfer, D. Buschak, and C. Moll, “Dienstleistungsbasierte geschäftsmodelle für industrie 4.0 – aktueller stand und potenziale für kmu,” vol. 3, p. 1287, 2016.

[25] J. M. Müller, “Assessing the barriers to industry 4.0 implementation from a workers’ perspective,” *IFAC-PapersOnLine*, vol. 52, no. 13, pp. 2189–2194, 2019.

[26] J. M. Müller, “Business model innovation in small- and medium-sized enterprises,” *Journal of Manufacturing Technology Management*, vol. 30, pp. 1127–1142, Dec. 2019.

[27] N. Asia, “Why is machine maintenance important? — nchasia.com.” `https://www.nchasia.com/en-au/nch-insights/why-is-machine-maintenance-important`, 2020. [Accessed 31-May-2023].

[28] B. Bither, “What is Machine Maintenance? Improving Equipment Health and Uptime — machinemetrics.com.” `https://www.machinemetrics.com/blog/machine-maintenance`, 04 2021. [Accessed 31-May-2023].

[29] K. Belgesi, “ISO 9001 maintenance procedure — sciencetr.com.” `https://www.sciencetr.com/en/iso-9001/iso-9001-prosedurleri/iso-9001-bakim-proseduru`, 2023. [Accessed 31-May-2023].

[30] P. Group, “5 Key Standards for Maintenance Professionals | Prometheus Group — prometheusgroup.com.” `https://www.prometheusgroup.co`

m/posts/5-important-standards-maintenance-professionals-should-be-aware-of, 12 2022. [Accessed 31-May-2023].

[31] F. Software, “What is Machine Maintenance? Examples & Best Practices — fiixsoftware.com.” https://www.fiixsoftware.com/glossary/machine-maintenance/, 2023. [Accessed 31-May-2023].

[32] S. Bradbury, B. Carpizo, M. Gentzel, D. Horah, and J. Thibert, “Digitally enabled reliability: Beyond predictive maintenance — mckinsey.com.” https://www.mckinsey.com/capabilities/operations/our-insights/digitally-enabled-reliability-beyond-predictive-maintenance, 2018. [Accessed 31-May-2023].

[33] H. F. G. helmut fischer.com, “How XRF Measurement Works | X-ray Fluorescence Analyzers & Detectors | Thickness Measurement.” https://www.helmut-fischer.com/techniques/basics-of-xrf-x-ray-fluorescence-analysis, 2023. [Accessed 31-May-2023].

[34] S. Cornaby, A. Reyes-Mena, H. K. Pew, P. W. Moody, T. Hughes, A. Stradling, D. C. Turner, and L. V. Knight, “An XRD/XRF instrument for the microanalysis of rocks and minerals,” *Measurement Science and Technology*, vol. 12, pp. 676–683, May 2001.

[35] B. Beckhoff, *Handbook of practical X-ray fluorescence analysis*. Springer, 2006.

[36] A. Thompson, *X-RAY DATA BOOKLET*. Center for X-ray Optics and Advanced Light Source - Lawrence Berkeley National Laboratory, 10 2009.

[37] R. V. Grieken and A. Markowicz, eds., *Handbook of X-Ray Spectrometry*. CRC Press, Nov. 2001.

[38] R. Behling, *Modern Diagnostic X-Ray Sources: Technology, Manufacturing, Reliability*. CRC Press, 2015.

[39] G. F. KNOLL, *Radiation detection and measurement*. JOHN WILEY, 2020.

[40] AMETEK, *XR-100 FastSDD User Manual*, 2023.

[41] K. Thompson, “Technical Note: Silicon Drift Detectors,” tech. rep., Thermo Fischer Scientific, 2012.

[42] “ Common X-Ray Tube Failure Modes spellman® application note.” https://www.spellmanhv.com/en/Technical-Resources/Application-Notes-X-Ray-Generators/AN-02. Accessed: 2022-08-15.

[43] J. REIS, *Fundamentals of data engineering plan and build Robust Data Systems*. O’REILLY MEDIA, 2022.

[44] P. Lea, *IOT and Edge Computing for architects: Implementing edge and IOT systems from sensors to clouds with communication systems, analytics, and security*. Packt Publishing, 2020.

[45] MongoDB, “NoSQL Vs SQL Databases — mongodb.com.” https://www.mongodb.com/nosql-explained/nosql-vs-sql. [Accessed 08-Jun-2023].

[46] IBM, “What is the CAP Theorem? | IBM — ibm.com.” https://www.ibm.com/topics/cap-theorem, 2023. [Accessed 31-May-2023].

[47] B. Balusamy, N. A. R., and A. H. Gandomi, *Big Data: Concepts, technology and Architecture*. John Wiley and Sons, Inc., 2021.

[48] D. Zburivsky and L. Partner, *Designing Cloud Data Platforms*. Manning Publications Co., 2021.

[49] G. Walter and S. D. Flapper, “Condition-based maintenance for complex systems based on current component status and bayesian updating of component reliability,” *Reliability Engineering & System Safety*, vol. 168, pp. 227–239, Dec. 2017.

[50] U. upkeep.com, “What are some failure prediction models in predictive maintenance?.” https://www.upkeep.com/learning/failure-prediction-models-predictive-maintenance/, 2023. [Accessed 31-May-2023].

[51] A. Kızılersü, M. Kreer, and A. W. Thomas, “The weibull distribution,” *Significance*, vol. 15, pp. 10–11, Apr. 2018.

[52] D. Wilkins, “The Bathtub Curve and Product Failure Behavior (Part 1 of 2) — weibull.com.” `https://www.weibull.com/hotwire/issue21/hottopics21.htm`, 2002. [Accessed 31-May-2023].

[53] E. L. Kaplan and P. Meier, “Nonparametric estimation from incomplete observations,” *Journal of the American Statistical Association*, vol. 53, no. 282, pp. 457–481, 1958.

[54] D. R. Cox and D. Oakes, *Analysis of survival data.* Chapman & Hall/CRC Monographs on Statistics and Applied Probability, Philadelphia, PA: Chapman & Hall/CRC, June 1984.

[55] D. Collett, *Modelling survival data in medical research, second edition.* Chapman & Hall/CRC Texts in Statistical Science, Philadelphia, PA: Chapman & Hall/CRC, 2 ed., Mar. 2003.

[56] R. Chalapathy and S. Chawla, “Deep learning for anomaly detection: A survey,” 2019.

[57] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[58] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” 2014.

[59] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, pp. 533–536, Oct. 1986.

[60] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, vol. 9, pp. 1735–1780, 11 1997.

[61] T. tableau.com, “Time Series Forecasting: Definition, Applications, and Examples.” `https://www.tableau.com/learn/articles/time-series-forecasting`, 2023. [Accessed 31-May-2023].

[62] J. D. Hamilton, *Time series analysis*. Princeton, NJ: Princeton University Press, Jan. 1994.

[63] J. H. et al., “Darts: User-friendly modern machine learning for time series,” *Journal of Machine Learning Research*, vol. 23, no. 124, pp. 1–6, 2022.

[64] A. Saxena, K. Goebel, D. Simon, and N. Eklund, “Damage propagation modeling for aircraft engine run-to-failure simulation,” in *2008 International Conference on Prognostics and Health Management*, IEEE, Oct. 2008.

[65] W. A. Berezowitz, J. R. Breunissen, and D. M. Miesbauer, “X-ray tube life prediction method and apparatus,” Sep 2002.

[66] Y. Ma, “Modeling the filament condition for failure prediction of a filament in an x-ray tube,” master’s thesis, Technische Universiteit Eindhoven, Eindhoven, The Netherlands, October 2016. Available at `https://pure.tue.nl/ws/portalfiles/portal/48627719/Ma_2016.pdf`.

[67] G. Nalbantov, D. Todorov, N. Zografov, S. Georgiev, and N. Bojilova, “Predictive maintenance tool for non-intrusive inspection systems,” 2021.

[68] B. B. C. Devices and Solutions, *Datasheet: Cross Domain Development Kit | XDK*. Available at `https://www.bosch-connectivity.com/media/downloads/xdk/xdk_node_110_combined_datasheet.pdf`.

[69] J. P. Carzolio, “A Guide to Consistent Hashing | Toptal® — toptal.com.” `https://www.toptal.com/big-data/consistent-hashing`, 2023. [Accessed 31-May-2023].

[70] P. Jakkula, “HBase or cassandra? a comparative study of NoSQL database performance,” *International Journal of Scientific and Research Publications (IJSRP)*, vol. 10, p. p9999, Mar. 2020.

[71] ProjectPro, “HBase vs Cassandra-The Battle of the Best NoSQL Databases — projectpro.io.” `https://www.projectpro.io/article/hbase-vs-cassandra/485`, 2023. [Accessed 31-May-2023].

[72] J. Landis, “Implementing Hot-Warm-Cold in Elasticsearch with Index Lifecycle Management — elastic.co.” `https://www.elastic.co/blog/implementing-hot-warm-cold-in-elasticsearch-with-index-lifecycle-management`, 2019. [Accessed 31-May-2023].

[73] A. D. Wilson, “The evaporation rate of filament material from alternating current heated filaments,” *Appl. Opt.*, vol. 2, pp. 1055–1059, Oct 1963.

[74] J.-J. Jang, H.-L. Zhu, and H.-S. Mok, “Filament x-ray tube current control method using indirect filament temperature estimation,” *Applied Sciences*, vol. 11, no. 22, 2021.

[75] E. Suli and D. F. Mayers, *An introduction to numerical analysis*. Cambridge, England: Cambridge University Press, Aug. 2003.

[76] L. Bottou, “Online algorithms and stochastic approximations,” in *Online Learning and Neural Networks* (D. Saad, ed.), Cambridge, UK: Cambridge University Press, 1998. revised, oct 2012.

[77] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, Curran Associates, Inc., 2019.

[78] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014.

[79] M. E. Mortenson, *Mathematics for Computer Graphics Applications*. New York, NY: Industrial Press, 2 ed., Jan. 1999.

[80] J. Herold, “Best Fit Bezier Curve — Jim Herold — jimherold.com.” `https://www.jimherold.com/computer-science/best-fit-bezier-curve`. [Accessed 09-Jun-2023].

[81] L. Verou, “cubic-bezier.com.” `https://cubic-bezier.com/`. [Accessed 09-Jun-2023].

[82] C. T. Vu, A. Chandra-Sekaran, and W. Stork, “A deep learning first approach to remaining useful lifetime prediction of filtration system with improved response to changing operational parameters using parameterized fully-connected layer,” in *PHM Society European Conference. 6*, pp. 439–447, June 2021.

[83] K. Ince, E. Sirkeci, and Y. Genç, “Remaining useful life prediction for experimental filtration system: A data challenge,” *PHM Society European Conference*, vol. 5, no. 1.

[84] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, pp. 1735–1780, Nov. 1997.

[85] V. Sarcar, *Design patterns in C#*. New York, NY: APRESS, 1 ed., June 2018.

[86] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.

[87] Google, “Protocol buffers language guide,” (https://developers.google.com/protocol-buffers/docs/proto), 04 2016.

[88] P. Russo, *Handbook of X-ray imaging: Physics and technology*. Taylor and Francis, 2018.

[89] V. Nissen, D. Stelzer, S. Straßburger, and F. Danie, eds., *Multikonferenz wirtschaftsinformatik (MKWI) 2016*. Ilmenau, Germany: TU Ilmenau Universitätsbibliothek, Mar. 2016.

[90] R. BEHLING, *Modern diagnostic X-ray sources: Technology, manufacturing, Reliability*. CRC PRESS, 2023.

[91] G. Lutz, *Semiconductor radiation detectors device physics*. Springer, 2007.

[92] T. Systems, “The Importance and Benefits of Predictive and Preventive Maintenance — tmasystems.com.” `https://www.tmasystems.com/resources/the-importance-and-benefits-of-predictive-and-preventive-maintenance`, 2023. [Accessed 31-May-2023].

[93] PhysicsOpenLab, “Si-PIN Photodiode β Detector.” `https://physicsopenlab.org/2017/04/28/si-pin-photodiode-%CE%B2-detector/`, 2017. [Accessed 31-May-2023].

[94] Spyder, “Spyder ide,” 2022. [Online; accessed 15-August-2022].

[95] D. Ili, D. Mostić, E. Dolicanin, K. Stanković, and P. Osmokrovic, “Mechanisms of electrical berakdown in low vacuums,” 2011.

[96] R. P. Little and W. T. Whitney, “Electron emission preceding electrical breakdown in vacuum,” *Journal of Applied Physics*, vol. 34, no. 8, pp. 2430–2432, 1963.

[97] G. Djogo and J. Cross, “Dependence of gap voltage collapse during vacuum breakdown on geometry and plasma dynamics,” *IEEE Transactions on Dielectrics and Electrical Insulation*, vol. 4, no. 6, pp. 848–853, 1997.

[98] W. E. Forsythe and E. M. Watson, “Resistance and radiation of tungsten as a function of temperature,” *J. Opt. Soc. Am.*, vol. 24, pp. 114–118, Apr 1934.

[99] A. D. Wilson, “Tungsten filament life under constant-current heating,” *Journal of Applied Physics*, vol. 40, pp. 1956–1964, 1969.

[100] R. T. Fielding, “Chapter 5: Representational state transfer (rest),” in *Architectural Styles and the Design of Network-based Software Architectures*, 2000.

[101] R. Fielding, “Hypertext transfer protocol (http/1.1),” Internet Engineering Task Force (IETF), 06 2014.

[102] R. T. Fielding, “Chapter 2: Network-based application architectures,” in *Architectural Styles and the Design of Network-based Software Architectures*, 2000.

[103] R. Strahl, “Web assembly and blazor: Re-assembling the web,” 07 2000.

[104] J. Bishop, “C# 3.0 design patterns: Use the power of c# 3.0 to solve real-world problems,” in *C# Books from O’Reilly Media*, 05 2012.

[105] P. F. Tiako, “Formal modeling and specification of design patterns using rtpa,” in *Software Applications: Concepts, Methodologies, Tools, and Applications: Concepts, Methodologies, Tools, and Applications*, 03 2009.

[106] S. Wedyan, Fadi; Abufakher, “Impact of design patterns on software quality: a systematic literature review,” IET Software, 02 2020.

[107] W. E. Forsythe and E. M. Watson, “Resistance and radiation of tungsten as a function of temperature,” *Journal of the Optical Society of America*, vol. 24, p. 114, Apr. 1934.

[108] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, pp. 436–444, May 2015.

[109] J. Howard and S. Gugger, “Fastai: A layered API for deep learning,” *Information*, vol. 11, p. 108, Feb. 2020.

[110] R. Łomowski and S. Hummel, “A method to estimate the remaining useful life of a filter using a hybrid approach based on kernel regression and simple statistics,” *PHM Society European Conference*, vol. 5, no. 1.

[111] H. Beirami, D. Calzà, A. Cimatti, M. Islam, M. Roveri, and P. Svaizer, “A data-driven approach for rul prediction of an experimental filtration system,” *PHM Society European Conference*, vol. 5, no. 1.

[112] G. S. Babu, P. Zhao, and X.-L. Li, “Deep convolutional neural network based regression approach for estimation of remaining useful life,” in *Database Systems for Advanced Applications*, pp. 214–228, Springer International Publishing, 2016.

[113] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks.,” in *AISTATS* (Y. W. Teh and D. M. Titterington, eds.), vol. 9 of *JMLR Proceedings*, pp. 249–256, JMLR.org, 2010.

[114] R. Gouriveau, *From prognostics and health systems management to predictive maintenance 1 : monitoring and prognostics*. Hoboken, NJ: Wiley, 2016.

[115] R. Mobley, *An introduction to predictive maintenance*. Amsterdam New York: Butterworth-Heinemann, 2002.

[116] G. Vachtsevanos, *Intelligent fault diagnosis and prognosis for engineering systems*. Hoboken, N.J: Wiley, 2006.

[117] H. Li, D. Parikh, Q. He, B. Qian, Z. Li, D. Fang, and A. Hampapur, “Improving rail network velocity: A machine learning approach to predictive maintenance,” *Transportation Research Part C: Emerging Technologies*, vol. 45, pp. 17–26, Aug. 2014.