



Vesa Merinen

Use of AI in System Administration

Metropolia University of Applied Sciences

Bachelor of Engineering

Information and Communications Technology

Bachelor's Thesis

27.10.2024

ABSTRACT

Author:	Vesa Merinen
Title :	Use of AI in System Administration
Number Of Pages:	35 pages
Date:	27.10.2024
Degree:	Bachelor of Engineering
Degree Programme:	Information and Communications Technology
Professional Major:	Computer science
Supervisor:	Janne Salonen, Lecturer

The main goal of this thesis is to explore the practical application of Support AI, specifically OpenAI's GPT-based Tukiäly, in enhancing the maintenance, management, and troubleshooting of radio network analytics systems at DNA. In system administration and maintenance, one must work with a wide variety of networking software and functionalities. It cannot be expected that an expert would fully master every aspect, which makes problem identification and background research a time-consuming task. Support AI offers speed and clarity when dealing with detailed issues across different areas. The study focuses on key areas of operations tasks like : Linux environment management, database management, and TCP/IP network management. The purpose is to understand how Tukiäly can be effectively integrated into these daily activities to improve operational efficiency and saving time on system administration tasks. The research investigates how Tukiäly can assist in daily system management by providing real-world examples of its role in problem resolution and optimizing routine maintenance. The study also explores how Tukiäly can be used in error correction, and support data monitoring and enrichment processes. the thesis examines how Tukiäly can contribute to network monitoring, anomaly detection, and performance optimization, demonstrating its usefulness in resolving network-related issues more efficiently. The research process is rooted in practical experimentation and iterative development within the DNA environment, where Tukiäly is deployed. Through this process, the study provides real-use case examples and insights into how Tukiäly was used to achieve more speed in workflow and system management tasks within the company. The results highlight significant benefits in fault management, log interpretation, and the creation of automated scripts tailored to specific issues. While Support AI has proven valuable, the study emphasizes the importance of having a solid understanding of system administration to ensure the appropriate use of Tukiäly. Interacting with Tukiäly, known as "prompting," involves structured communication between the user and the AI, where each session is independent. This means users must focus on one topic per conversation to achieve the best results. Tukiäly has proven to be an effective tool, offering insightful suggestions and improving troubleshooting speed. However, the simplicity and small-scale nature of the scripts and automation generated by Tukiäly must be carefully managed to prevent system overloads. When used effectively, Tukiäly can streamline small-scale software development, automating sysadmin tasks and delivering significant time savings. Overall, the thesis provides practical insights into how Tukiäly has transformed system administration at DNA, improving operational efficiency in Linux environments, databases, and network infrastructure. The findings offer concrete recommendations for further implementation, showcasing how Tukiäly can be an essential tool for maintaining and optimizing technical systems.

Keywords: Tukiäly, Prompting, Server Management, Network Management, Troubleshooting

TIIVISTELMÄ

Tekijä:	Vesa Merinen
Otsikko:	Tekoälyn Käyttö Järjestelmänhallinnassa
Sivumäärä:	35 pages
Aika:	27.10 2024
Tutkinto:	Insinööri (AMK)
Tutkinto-ohjelma:	Tieto- ja viestintätekniikka
Ammatillinen Pääaine:	Ohjelmistotekniikka
Ohjaaja:	Janne Salonen

Tämän opinnäytetyön päätavoitteena on tutkia Tukiälyn (OpenAI GPT-pohjaisen tekoälypalvelun) käytännön soveltamista radioverkon analytiikkajärjestelmien ylläpidossa, hallinnassa ja vianetsinnässä DNA:lla. Järjestelmän hallinnassa ja ylläpidossa joutuu työskentelemään mitä erilaisimpien tietoliikenne ohjelmistojen ja toimintojen parissa. tällöin ei voi olettaa että asiantuntija osaisi täydellisesti jokaisen osa-alueen ja niinpä ongelmien kartoitus ja taustoittaminen on aikaa vievää puuhaa. Tukiäly tarjoaa nopeutta ja selkeyttä eri alueiden detaljitason ongelmiin Tutkimus keskittyy keskeisiin operatiivisiin alueisiin, kuten Linux-ympäristön hallintaan, tietokantojen hallintaan sekä TCP/IP-verkon hallintaan. Tavoitteena on ymmärtää, kuinka Tukiälyä voidaan käyttää tiettyihin päivittäisiin tehtäviin parantamaan operatiivista tehokkuutta ja säästämään aikaa järjestelmänhallinnan tehtävissä. Tutkimuksessa tarkastellaan, kuinka Tukiäly voi auttaa päivittäisessä järjestelmänhallinnassa tarjoamalla käytännön esimerkkejä sen roolista ongelmanratkaisussa ja vianetsinnän optimoinnissa. Lisäksi tutkimus selvittää, kuinka Tukiälyn apua voidaan käyttää virheenkorjauksessa sekä tietokannan käytössä. Opinnäytetyössä tutkitaan myös, kuinka Tukiäly voi edistää verkon valvontaa, poikkeamien tunnistamista ja suorituskyvyn optimointia ja osoittaa saavutettu hyöty ylläpitoon liittyvien ongelmien ratkaisussa. Tutkimusprosessi perustuu käytännön kokeiluihin ja iteratiiviseen kehitykseen DNA:n Radioverkon analytiikan ympäristössä, jossa Tukiäly on otettu käyttöön. Tämän prosessin kautta tutkimus tarjoaa käytännön esimerkkejä ja näkemyksiä siitä, kuinka Tukiälyä on käytetty työprosessien ja järjestelmänhallintatehtävien nopeuttamiseen yrityksessä. Tulokset osoittavat etuja vianhallinnassa, lokitietojen tulkinnassa ja automatisoitujen, tiettyihin ongelmiin räätälöityjen skriptien luomisessa. Vaikka Tukiäly on osoittautunut arvokkaaksi työkaluksi, tutkimus painottaa, että vahva järjestelmänhallinnan osaaminen on välttämätöntä sen asianmukaisessa käytössä. Vuorovaikutus Tukiälyn kanssa, jota kutsutaan "promptingiksi", tarkoittaa rakenteellista teksti -pohjaista kommunikointia käyttäjän ja tekoälyn välillä, jossa jokainen kommunikointi -istunto on itsenäinen yksikönsä. Tutkimus osoitti hyötyä siinä, että käyttäjät keskittyvät yhteen aiheeseen per keskustelu saadakseen parhaat mahdolliset tulokset. Tukiäly on osoittautunut tehokkaaksi työkaluksi, joka tarjoaa oivaltavia ehdotuksia ja nopeuttaa vianetsintää. Kuitenkin Tukiälyn tuottamien skriptien ja vaikutus on ymmärrettävä tarkasti, jotta järjestelmän ylikuormittuminen vältetään. Oikein käytettynä Tukiäly tehostaa pienimuotoista ohjelmistokehitystä, mahdollistaa automatisoituja järjestelmänhallinnan tehtäviä ja tuottaa selkeää ajansäästöä. Kokonaisuudessaan opinnäytetyö tarjoaa käytännön näkemyksiä siitä, kuinka Tukiäly on muuttanut järjestelmänhallintaa DNA:lla, parantaen operatiivista tehokkuutta Linux-ympäristöissä, tietokannoissa ja verkon infrastruktuurissa. Tulokset osoittavat hyödyllisyyden ja rohkaisevat Tukiälyn käytön jatkekehitykseen ja osoittavat, kuinka Tukiäly voi olla olennainen lisäapu teknisten järjestelmien ylläpidossa ja optimoinnissa.

Avainsanat: Tukiäly, Prompting, Serveri-hallinta, Verkonhallinta, Vianetsintä

Sisällys

Abbreviations

1	Introduction	1
2	Introduction to chat GPT	2
2.2	OpenAI	3
2.3	Chat GPT	3
3	Introduction to Research Environment	5
3.1	System Administration at DNA	5
3.2	Linux	5
3.3	SQL	7
3.4	Mobile Networks	8
3.4.1	Structure and Function of Mobile Networks	8
3.4.2	Mobile Network Technologies	9
3.4.4	Technologies Used in Mobile Networks	9
3.4.5	Radio Network Analytics and Performance Monitoring	10
4	ENIQ	11
4.1	ENIQ OS System Monitoring	14
4.4	ENIQ System administration	14
4.5	ENIQ Databases	16
5	Utilizing Tukiäly to troubleshooting	17
5.1	Prompting	17
5.2	Tasks examples for redhat linux	18
5.3.	Database work on ENIQ reports	22
5.4	Log search shell scripts	24
5.5.	network issues on ENIQ	25
6	Conclusion	33
	Sources	35

List of Abbreviations

ORM:	<i>Object-relational mapping</i> . Ohjelmiston oliomallin mukaisen esityksen kuvaaminen relaatiotietokantamallin mukaiseksi esitykseksi ja kääntäen.
TKHJ:	Tietokannan hallintajärjestelmä. Ohjelmisto tiedon tehokkaan hakemisen, säilyttämisen ja päivittämisen toteuttamiseksi. SQL (Structured Query Language): A standardized language used to manage and manipulate relational databases through operations such as querying, updating, and managing database structures.
TCP/IP:	(Transmission Control Protocol/Internet Protocol): A set of communication protocols used to interconnect network devices on the internet. It defines how data should be packetized, addressed, transmitted, routed, and received.
Shell:	A command-line interpreter in Unix and Linux systems that allows users to execute commands, scripts, and programs to perform various tasks, such as file manipulation and program execution.
RHEL	(Red Hat Enterprise Linux): A commercial Linux distribution developed by Red Hat, designed for enterprise environments with features like long-term support, security updates, and stability.
SE Linux	(Security-Enhanced Linux): A security module integrated into the Linux kernel that provides a mechanism for enforcing access control policies to enhance the security of the system.
ETL	(ETL Processes): Refers to the Extract, Transform, Load process, which involves extracting data from various sources, transforming it into a suitable format, and loading it into a database or data warehouse.
SAP BI:	(SAP Business Intelligence): A suite of tools and applications from SAP that allows for data warehousing, reporting, and analytics to support business decision-making processes.
2G, 3G, 4G, 5G:	Generations of mobile network technology, each providing increased data transmission speeds, capacity, and improved connectivity features. 5G, the latest generation, supports faster data speeds and lower latency for advanced applications like IoT and autonomous vehicles.
SMS:	(Short Message Service) / Text Messaging: A service used for sending short text messages between mobile devices over the cellular network.

GSM:	(Global System for Mobile Communications): A standard developed for mobile networks that enables mobile devices to communicate over cellular networks for voice and data services.
LTE:	(Long-Term Evolution): A standard for wireless broadband communication, commonly referred to as 4G, providing higher data speeds and improved network efficiency compared to previous technologies.
CDMA:	(Code Division Multiple Access): A channel access method used by various communication technologies to allow multiple signals to occupy a single transmission channel, optimizing bandwidth usage.
WCDMA:	(Wideband Code Division Multiple Access): A 3G technology that enhances data transfer speeds and capacity by using wider radio bandwidths compared to CDMA.
RAN:	(Radio Access Network): A component of a mobile network that connects mobile devices to the core network via radio links. It includes base stations and antennas.
RSRP:	(Reference Signal Received Power): A measure of the power level of a received signal in a mobile network, indicating the strength of the signal received from a base station.
SINR:	(Signal to Interference plus Noise Ratio): A measure of signal quality that compares the level of a desired signal to the level of background noise and interference.
ASCII:	(American Standard Code for Information Interchange): A character encoding standard used for representing text in computers and communication equipment.
XML:	(Extensible Markup Language): A flexible, text-based format used to store and transport data, designed to be both human-readable and machine-readable.
ASN.1:	(Abstract Syntax Notation One): A standard interface description language used to describe data structures for representing, encoding, transmitting, and decoding data in telecommunications and computer networking.
Disk I/O:	(Disk Input/Output): Refers to the read and write operations performed on a disk storage device. High disk I/O can indicate intensive data processing activities.

CPU: (Central Processing Unit): The primary component of a computer that performs most of the processing inside a system, executing instructions from programs and managing other hardware components.

UDP: (User Datagram Protocol): A communication protocol used for transmitting data with minimal overhead, often used in applications where speed is more critical than reliability, such as video streaming and gaming

1 Introduction

We explore In this thesis, how Tukiäly has been integrated into the daily workflow of DNA's telecommunication experts and what potential benefits and drawbacks it may have in this context. Telecommunication operators have begun investigating how to leverage AI across all their operations. The Operations Department has started utilizing this technology, and this work focuses on the various ways in which Tukiäly can be beneficial in the system management of operations.

This research was conducted as part of system administrators regular work through practical experiments, solving emerging fault situations and completing daily tasks. The aim was to identify the advantages, disadvantages, and potential pitfalls of using Tukiäly, with special attention to security and data protection considerations.

The scope of this thesis is limited to the practical troubleshooting and problem-solving tasks in system management within the operations context. Although AI and algorithms can be used for various automated tasks and commands, this study does not cover those applications. Tukiäly in this study is not integrated into the direct execution of operational tasks, nor is it modified to process data or perform autonomous actions in this thesis, we explore how Tukiäly has been integrated into the daily workflow of DNA's telecommunication experts and examine the potential benefits and drawbacks it may have in this context. Telecommunication operators have begun investigating how to leverage AI across all aspects of their operations. The Operations Department has already started utilizing this technology, and this work focuses on the various ways in which Tukiäly can be advantageous in the system management of operations.

Tukiäly, in its various forms, has made significant breakthroughs in telecommunication operators and data center monitoring, as well as in data processing and operational automation. It has proven useful in monitoring different error logs, gathering valuable data from them, and providing insights. In system administration environments, which are composed of numerous systems and software platforms, managing the variety of faults and problems that arise can be overwhelming for a single operator. New systems are constantly being introduced, and older software is frequently updated, creating a complex web of interdependencies where an update in one area can cause issues elsewhere.

Tukiäly steps in to help with these emerging situations by performing preliminary error investigations and, in some cases, even creating situational reports and suggesting potential actions. This capability is especially beneficial to staff who understand the broader system or functionality but may not be fully familiar with new databases or software at a detailed level.

This research was conducted as part of system administrators' regular work through practical experiments, solving emerging fault situations and completing daily tasks. The aim was to identify the advantages, disadvantages, and potential pitfalls of using Tukiäly, with special attention to security and data protection considerations.

The scope of this thesis is limited to practical troubleshooting and problem-solving tasks in system management within the operations context. Although AI and algorithms can be used for various automated tasks and commands, this study does not cover those applications. In this study, Tukiäly is not integrated into the direct execution of operational tasks, nor is it modified to process data or perform autonomous actions.

2 Introduction to chat GPT

2.1 Language Models

A language model is a type of artificial intelligence model designed to understand and generate human language. The development of language models dates back to the 1950s, starting with simple rule-based systems and evolving into more sophisticated statistical models like n-grams. In the late 20th century, the field made significant strides with the introduction of probabilistic models and early forms of machine learning.

The real breakthrough came with the development of neural network-based models. Recurrent Neural Networks (RNNs) and Long Short-Term Memory networks (LSTMs) allowed for better handling of sequential data, like language, by maintaining context over longer sequences. However, these models had limitations, particularly in their ability to capture long-range dependencies.

The advent of the Transformer architecture in 2017 revolutionized the field of Natural Language Processing (NLP). It introduced a new way to process data in parallel, making training on large datasets more efficient. This led to the development of models like BERT (Bidirectional Encoder Representations from Transformers), GPT (Generative Pre-trained Transformer), and later, more advanced versions such as GPT-2 and GPT-3. Today, language models are widely used in applications ranging from chatbots and machine translation to automated content creation and text summarization (Kapronczay 2022).

2.2 OpenAI

OpenAI is an AI research organization founded in December 2015 by a group of entrepreneurs, including Elon Musk, Sam Altman, Greg Brockman, and Ilya Sutskever, among others. The mission of OpenAI is to ensure that artificial general intelligence (AGI) benefits all of humanity. The organization aims to advance digital intelligence in the way that is most likely to benefit society as a whole.

Initially, OpenAI focused on a broad range of AI research topics, including robotics and game playing. However, its research into NLP has become one of its most prominent contributions to the field. In 2018, OpenAI released GPT-1, the first version of its Generative Pre-trained Transformer model, which marked a significant step forward in language modeling. This was followed by GPT-2 and GPT-3, each demonstrating increasingly powerful language generation capabilities (SpringerLink)

2.3 Chat GPT

ChatGPT is a specialized implementation of the GPT-3 model, developed by OpenAI. It was introduced as a conversational agent capable of understanding and generating text in a contextually coherent manner. The early development of ChatGPT began with research in generative language models and pre-training on a diverse dataset of internet text. The model was trained to predict the next word in a sentence, learning from patterns in the data to generate coherent and contextually relevant responses (Roldós 2020)..

Technical Overview: ChatGPT is based on the Transformer architecture, which uses self-attention mechanisms to process input data in parallel, rather than sequentially. This allows it to handle long-range dependencies and understand complex relationships within the text. The model undergoes two main phases: pre-training and fine-tuning. During pre-training, it learns general language patterns from a massive dataset. In the fine-tuning phase, it is trained on a narrower dataset with human feedback to improve its conversational abilities and reduce biases (Roldós 2020).

Common Use Cases: Today, ChatGPT is used in a variety of applications, including:

- **Customer Support:** Automating responses to frequently asked questions and assisting customers with common issues.
- **Educational Tools:** Providing explanations and tutoring in various subjects.
- **Content Creation:** Assisting in writing articles, generating creative content, and drafting emails.
- **Programming Help:** Offering code suggestions and debugging assistance to developers.
- **Healthcare:** Assisting with patient queries and providing information on common medical conditions (with appropriate safeguards).

ChatGPT's versatility and ability to generate human-like responses have made it a valuable tool across multiple industries, transforming how organizations interact with their customers and manage information (SpringerLink)

3 Introduction to Research Environment

3.1 System Administration at DNA

At DNA, system administration work encompassed a broad range of responsibilities related to the management and maintenance of the Ericsson-supplied ENIQ system, which was integral to radio network analytics. This role involved comprehensive system administration tasks such as system maintenance, software updates, fault management, and continuous monitoring of the ENIQ system's performance. The infrastructure was built on a Linux server environment, with three distinct databases handling the data collected through IP connections from various radio and mobile network systems. This data was crucial for generating Business Intelligence (BI) reports, which were used by radio network planners to optimize network performance.

The system administration duties required proficiency in Unix scripting and expertise in troubleshooting ETL (Extract, Transform, Load) processes. This included diagnosing issues during ETL jobs, ensuring smooth data extraction and transformation workflows, and escalating problems to the vendor, Ericsson, when issues arose within their applications or when the system did not function as expected. Additionally, the role involved testing vendor-supplied upgrades, modifications, and security updates to ensure they were implemented correctly and did not negatively impact system stability or data integrity.

System administrators needed to possess strong skills in Linux system maintenance and troubleshooting, as well as the ability to manage IP traffic and maintain secure and efficient database connections. They were also responsible for customizing database reports using SQL or BI tools to meet the specific requirements of the radio network planning team. The ability to effectively manage escalations with Ericsson and verify the proper implementation of vendor-provided updates and security changes was critical for maintaining the reliability and security of the ENIQ system, ensuring that it continued to deliver accurate and timely data for network analysis and decision-making.

3.2 Linux

Linux is a robust and versatile open-source operating system widely used in various computing environments, from personal computers to servers and supercomputers. Its modular design and powerful command-line interface make it particularly suited for system management, maintenance, and troubleshooting tasks. Linux offers extensive support for scripting and automation through shell scripts and tools like “cron” for scheduling, which are

essential for automating routine maintenance tasks and responding to system alerts. Linux has been a fundamental component in the telecommunications industry for many years, largely due to its stability, flexibility, and cost-efficiency. Its open-source nature allows telecom operators to modify and tailor the operating system to meet their specific needs, which is particularly important in an industry that requires high levels of performance and reliability. By enabling custom kernel modifications and optimizations, Linux provides the ability to fine-tune system performance and resource management, crucial for handling the large-scale network infrastructures typical in telecommunications (Dustin 2019).

One of the key advantages of Linux in this field is its scalability. Telecom operators manage vast networks with high volumes of data traffic, and Linux's architecture is well-suited to handle such demanding environments. Its extensive support for scripting and automation through tools like shell scripts, cron jobs, and Ansible, among others, allows for efficient system management and the automation of repetitive tasks. This is essential for maintaining service uptime and quickly responding to any network issues that may arise.

In addition to scalability, cost-effectiveness is another significant benefit. As an open-source operating system, Linux eliminates the need for costly software licenses, which is particularly beneficial for telecom operators running extensive and complex infrastructure. This reduction in operational costs enables companies to allocate resources to other critical areas, such as network expansion or customer service improvements.

However, the use of Linux in telecommunications also presents certain challenges. Managing a Linux-based environment requires specialized skills and knowledge, which can be a barrier to entry for companies that lack experienced system administrators. The complexity of Linux, particularly in configuring and maintaining network services, means that extensive training and expertise are necessary to fully leverage its capabilities. Furthermore, while Linux supports a wide range of hardware, some proprietary telecom equipment may not have complete driver support, leading to potential compatibility issues that need to be addressed through custom development or third-party solutions.

Red Hat Enterprise Linux (RHEL) has become a preferred choice for many telecom operators due to its enterprise-level support and stability. RHEL offers comprehensive support services, including security updates, patches, and expert assistance, which are critical for maintaining the reliability and security of telecom systems. The stability and long-term support provided by RHEL are particularly valuable in telecommunications, where systems must operate continuously with minimal downtime. This reliability, combined with

performance optimizations specific to enterprise environments, makes RHEL an ideal choice for tasks such as low-latency networking and real-time data processing.

Security is another crucial aspect where RHEL excels. With advanced features such as Security-Enhanced Linux (SE Linux), RHEL provides robust mechanisms for managing access controls and protecting sensitive data. This level of security is essential for telecom operators who must ensure compliance with strict regulatory requirements and protect their infrastructure from potential threats (Dustin 2019).

3.3 SQL

SQL (Structured Query Language) is a standardized programming language widely used for managing and manipulating relational databases. It is indispensable for querying, inserting, updating, and deleting data from databases. Additionally, SQL allows for defining and modifying database structures, controlling access, and handling large datasets efficiently (Addison-Wesley, 2003). Given its robust capabilities in managing structured data, SQL is a key technology across many industries, including financial systems and telecommunication networks.

In the context of data pipelines, such as ETL (Extract, Transform, Load) processes, SQL plays a crucial role in transforming and enriching raw data before it is stored in a database. The ETL process begins by extracting data from multiple sources, including log files, APIs, or other databases. Once the data is extracted, SQL operations are applied during the transformation stage to clean, aggregate, and organize the data. SQL commands such as JOIN, GROUP BY, and WHERE are essential for combining data from various sources, applying business rules, and structuring the data in a way that is suitable for analysis or reporting. This transformation step ensures that data is processed into a format that maximizes its value (Addison-Wesley, 2003).

After the transformation phase, SQL is used to load the processed data into a target database, such as an Oracle database. In this step, SQL commands like INSERT INTO populate the enriched data into relevant tables, while transaction controls such as COMMIT and ROLLBACK ensure data integrity and consistency throughout the process. By utilizing SQL, the final stage of the pipeline ensures that data is efficiently stored, indexed, and made accessible for further analysis or reporting (Addison-Wesley, 2003).

In radio network analytics, the data processed through the ETL pipeline often includes critical performance metrics that are vital for monitoring and optimizing network operations. Examples of such metrics include the call drop rate, which measures the percentage of calls

that terminate unexpectedly, and busy hour statistics, which track network performance during peak traffic periods. These and related metrics are collected from various radio network components such as base stations, network controllers, and switches.

Once extracted, this raw data undergoes SQL-based operations to aggregate, filter, and enrich it. For example, data from different radio network components is combined to provide a comprehensive view of the network's performance across regions or during specific time frames. SQL commands like GROUP BY and JOIN are often employed to merge multiple data sources, calculate averages, or generate detailed reports on key metrics such as call drop rates and busy hour performance. Finally, this enriched data is loaded into an Oracle database, allowing telecom operators to monitor network health, optimize resource allocation, and ensure high-quality service for their customers. This integration of SQL within the ETL pipeline supports effective decision-making and continuous improvement of network performance.

Additionally, a system administrator plays a critical role in ensuring the quality and integrity of the data in the database. The administrator can directly access the database through the command line or use graphical tools such as Oracle SQL Developer. Using SQL queries, the administrator checks the data quality in the tables that serve as the source for analytics reports. In the event of errors or data discrepancies, the administrator can manually inspect, correct, and update the data in these tables. This proactive maintenance ensures that the analytical reports generated from the system are accurate and reliable, minimizing the impact of any issues on network performance monitoring and decision-making (Addison-Wesley, 2003).

3.4 Mobile Networks

Mobile networks are the backbone of modern wireless communication, enabling millions of people around the world to connect through voice calls, text messages, and internet services. They are composed of a complex infrastructure that allows seamless communication between mobile devices and the broader telecommunication network. Mobile networks have evolved significantly over the past few decades, transitioning from basic voice communication systems to high-speed data networks that support a wide range of applications, including video streaming, IoT (Internet of Things), and real-time navigation.

3.4.1 Structure and Function of Mobile Networks

A mobile network consists of several key components, including base stations, core network elements, and user equipment. The base stations, often referred to as cell towers, provide wireless coverage and facilitate communication between mobile devices and the network. These base stations are interconnected with the core network, which manages call routing, data transfer, and user authentication. The core network connects to external networks, such as the internet or other mobile networks, to enable global communication.

Mobile networks operate through a process called cellular technology, where the coverage area is divided into small geographical areas known as cells. Each cell is served by a base station, and as a user moves from one cell to another, their connection is handed over to the next base station seamlessly. This structure allows for efficient use of radio frequencies and enables large-scale mobility for users.

3.4.2 Mobile Network Technologies

The evolution of mobile networks can be categorized into different generations, each representing a significant technological advancement:

1G (First Generation): Introduced in the 1980s, 1G networks were analog systems that provided basic voice communication without any data capabilities.

2G (Second Generation): Launched in the 1990s, 2G networks introduced digital communication, enabling text messaging (SMS) and basic data services. Technologies like GSM (Global System for Mobile Communications) became widely adopted.

3G (Third Generation): The early 2000s saw the advent of 3G networks, which provided faster data rates and enabled mobile internet access. It supported multimedia services like video calls and mobile TV.

4G (Fourth Generation): With the introduction of 4G networks, data speeds increased significantly, allowing for high-definition video streaming, online gaming, and advanced mobile applications. LTE (Long-Term Evolution) is the most widely used 4G technology.

5G (Fifth Generation): The latest generation, 5G, offers even higher data rates, lower latency, and the ability to connect a massive number of devices simultaneously. It supports advanced applications such as autonomous driving, smart cities, and enhanced IoT connectivity.

3.4.4 Technologies Used in Mobile Networks

Mobile networks rely on a variety of technologies to function efficiently:

Radio Access Network (RAN): This component consists of base stations and antennas that facilitate the wireless connection between mobile devices and the network. Technologies like CDMA, WCDMA, and LTE are used in different generations of RAN.

Core Network: The core network handles the management of user connections, data transfer, and communication with external networks. It uses technologies like IP Multimedia

Subsystem (IMS) and Network Functions Virtualization (NFV) to support a variety of services.

Backhaul Network: This part of the network connects the RAN to the core network and is typically composed of high capacity wired or wireless links. Technologies such as fiber optics and microwave transmission are commonly used for backhauling.

Spectrum: Mobile networks operate over specific frequency bands, known as the spectrum. Efficient spectrum management is crucial for optimizing network performance and ensuring coverage and capacity (Holma et al 2014).

3.4.5 Radio Network Analytics and Performance Monitoring

For mobile operators, ensuring optimal network performance and user experience is critical. This is where radio network analytics comes into play. Operators continuously monitor and analyze various parameters to maintain and improve the quality of service. Some of the key metrics and parameters measured include:

Signal Strength and Quality: Operators monitor signal strength (RSRP) and signal quality (SINR) to ensure that users have a strong and clear connection. Poor signal strength can lead to dropped calls and slow data speeds.

Call Drop Rate (CDR): This metric indicates the percentage of calls that are disconnected unexpectedly. A high CDR can indicate issues with network coverage or capacity.

Handover Success Rate: This measures the success of transitions between cells as users move. A low handover success rate can result in dropped calls and interrupted services.

Data Throughput: This metric measures the speed at which data is transmitted and received by users. High data throughput is essential for activities such as video streaming and online gaming.

Latency: Latency measures the time it takes for data to travel from the user device to the network and back. Low latency is crucial for real-time applications like online gaming and video conferencing.

Capacity and Congestion: Operators monitor network capacity and congestion levels to ensure that the network can handle the number of users and the volume of data traffic without degradation in performance.

By analyzing these metrics, operators can identify potential issues, optimize network performance, and enhance the overall user experience. Advanced analytics tools and machine

learning algorithms are increasingly being used to predict and prevent network problems before they impact users, making mobile networks more reliable and efficient. At DNA, radio network analytics is collected using the ENIQ (Ericsson Network Integrated Quality) system. ENIQ is a sophisticated platform used for managing and analyzing performance data in mobile networks. The system collects detailed metrics from the radio network, which are then stored in a database for further processing and analysis.

This data repository allows the network operations team to generate various reports that are crucial for multiple aspects of network management. These reports serve three primary purposes:

Fault Management: By analyzing the performance data, operators can detect anomalies and identify potential issues within the network. This helps in proactive fault detection and quick resolution of problems, ensuring minimal service disruption.

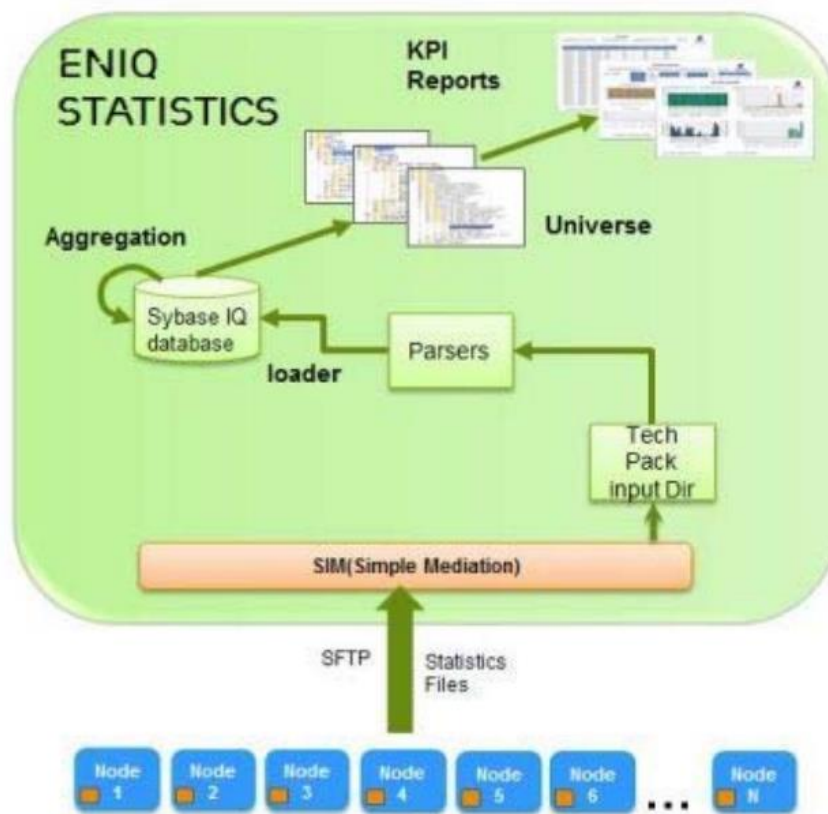
Network Optimization: ENIQ enables detailed analysis of network performance indicators, such as signal strength, data throughput, and call drop rates. This information is used to fine-tune the network parameters and improve overall network efficiency, ensuring a better user experience.

Radio Network Planning and Modifications: The collected data provides insights into how the network is being utilized, highlighting areas with high traffic demand or potential coverage gaps. This helps in making informed decisions about network expansion, reconfiguration of existing resources, and implementing changes to the radio network to accommodate growing demand or improve coverage

4 ENIQ

Ericsson Network IQ Statistics (ENIQ) is an advanced performance management solution designed to monitor and optimize multi-vendor and multi-technology telecommunications networks. As mobile networks evolve and face increasing traffic demands, maintaining high service quality is essential for service providers. ENIQ plays a critical role by offering a complete view of network performance, enabling proactive network management and

optimization.



Picture 1 Data processing workflow

Picture 1 illustrates the data processing workflow from **Node 1** through **Node 9**, where the raw data is transferred via **SFTP** to the **mediation** layer. The mediation layer passes the data through various **parsers** before loading it into the **Sybase IQ database**. Once in the database, the data undergoes **aggregation** and is organized into a **universe** structure, ultimately producing the final **KPI reports**.

The ENIQ platform is built on Linux Red Hat and operates on HP hardware, providing a robust infrastructure for managing vast amounts of network data. This architecture offers high performance, scalability, and reliability, essential for handling the enormous data volumes generated by modern mobile networks. Additionally, the system comes with a comprehensive backup solution, including both standard backups and tape backups, ensuring multiple layers of data protection and redundancy.

The system is geographically distributed, allowing data to be collected and processed from multiple locations. This is crucial for large-scale, multi-region telecom operators, ensuring real-time analysis and management of network performance across various regions. The ENIQ system integrates ETL (Extract, Transform, Load) processes, where performance data from radio network components such as base stations, nodes, and cells are collected in real-time. These data are delivered to ENIQ in file formats, enriched, and then loaded

into the system's database through data lakes. This real-time data flow allows operators to maintain a close watch on network performance and take timely action when needed.

ENIQ collects performance data from a variety of sources, supporting interfaces such as ASCII, XML, and ASN.1. This flexibility enables seamless integration into diverse network environments and technologies. Data is typically transferred through file interfaces or via a network element manager like Ericsson Network Manager (ENM). Once collected, the enriched data is stored in SAP databases, which handle the storage and further processing. Ericsson delivers this end-to-end solution, providing service providers with a holistic performance management tool that covers data collection, processing, and analysis.

One of the key strengths of ENIQ is its ability to operate in complex, multi-vendor environments, making it highly effective for managing heterogeneous networks. ENIQ offers functionalities such as historical trend analysis, capacity planning, and the generation of both standard and customized reports. This versatility enables operators to monitor key performance indicators (KPIs), analyze network behavior, and make informed decisions on network optimizations and investments.

ENIQ's reporting capabilities are particularly valuable for managing large-scale, carrier-grade operations. The system can generate out-of-the-box standard reports on essential metrics, while also allowing operators to create custom reports that address specific business needs. This flexibility ensures that service providers can focus on critical areas of network performance that directly impact service quality.

In addition to its advanced data handling and reporting functions, ENIQ is designed with ease of use in mind. The platform includes a range of administrative tools, making tasks like backups, restores, and system management straightforward, without requiring deep database expertise. The system's multiple backup solutions, including the option for tape backups, provide operators with enhanced data protection, ensuring that performance data is securely stored and can be recovered in case of failure.

Overall, Ericsson Network IQ Statistics is a powerful tool for network performance management. By providing detailed, actionable insights into network operations, ENIQ helps operators maintain high service quality while efficiently managing network resources. The combination of Linux Red Hat infrastructure, HP hardware, SAP databases, and a geographically distributed architecture makes ENIQ an ideal solution for collecting, processing, and analyzing performance data across complex, large-scale telecom networks. With its real-time data collection and ETL processes, ENIQ ensures operators can meet the challenges posed by growing traffic demands and evolving network environments.

4.1 ENIQ OS System Monitoring

System monitoring in a data analytics server environment is a critical process that ensures the smooth operation and optimal performance of complex infrastructures, such as those involving Linux Red Hat servers, TCP/IP networks, databases, and ETL (Extract, Transform, Load) processes. In such an environment, the server is not only responsible for handling large volumes of data but also for maintaining the integrity and availability of that data for analytical purposes. This requires continuous monitoring of various system components to detect potential issues before they escalate into significant problems.

In the context of a Red Hat Linux server environment, system monitoring involves keeping track of key performance indicators (KPIs) such as CPU usage, memory consumption, disk I/O, and network traffic. These metrics provide insights into the server's health and capacity to handle data-intensive tasks. TCP/IP monitoring, on the other hand, focuses on the network performance, ensuring that data packets are transmitted efficiently between the server and other network entities. This includes monitoring connection states, packet loss, latency, and network interface errors, which are crucial for maintaining a stable and reliable communication framework for data transfer.

For database and ETL processes, monitoring extends to the performance of the database management system and the efficiency of data extraction, transformation, and loading activities. This includes tracking query performance, transaction rates, and potential bottlenecks in data processing pipelines. Monitoring tools help in identifying slow-running queries, lock contention issues, and data load failures, which can impact the timeliness and accuracy of the data being fed into the analytics systems.

4.4 ENIQ System administration

System administration is a dynamic and challenging role that requires a deep understanding of the operating system, networking, and security principles. Administrators must be prepared to respond quickly to unexpected issues while continuously improving the server environment to support the organization's needs effectively.

From a system administrator's perspective, managing ENIQ application involves a comprehensive set of tasks aimed at ensuring the stability, security, and performance of the server infrastructure. This begins with the initial configuration and installation of the operating system, where the administrator sets up network configurations, user permissions, and security policies to align with the organization's requirements. Regular tasks include updating the system with the latest security patches, managing disk space, and configuring various services such as web servers, databases, or file systems. Automation tools like shell scripting are often used to streamline repetitive tasks, allowing for efficient management of multiple servers simultaneously.

Maintenance of application environment also entails continuous performance monitoring and optimization. Administrators use a variety of commands, such as “top”, “htop”, and “iotop”, to monitor resource usage, including CPU, memory, and I/O operations. Network performance is also crucial, and tools like netstat and ss help in identifying active connections and potential network bottlenecks. Logs play a critical role in server maintenance; administrators regularly review system logs found in directories like “/var/log/” to identify any unusual activity or errors that could indicate a security threat or a system malfunction. Proactive maintenance practices, such as configuring automated alerts and performing regular system audits, help prevent issues before they impact system operations.

When it comes to troubleshooting, system administrators must be adept at diagnosing and resolving a wide range of issues that could affect server performance or availability. Troubleshooting begins with identifying the problem, which could range from software misconfigurations to hardware failures. Tools like “journalctl” for viewing system logs, “ping” and “traceroute” for network diagnostics, and “strace” for tracking system calls are essential for pinpointing the root cause of issues. Administrators also use backup and recovery tools to restore services quickly in case of data corruption or system failures. Effective troubleshooting requires not only technical skills but also a methodical approach to problem-solving, ensuring that the underlying issues are addressed without causing further disruption to the services running on the servers.

Command-line commands and shell scripts are fundamental tools for Linux server administration, providing system administrators with powerful means to automate, monitor, and troubleshoot server operations efficiently. The command line allows administrators to interact directly with the system's core functions, enabling precise control over server configurations and operations. Commands such as “ls” for listing directory contents, “cp” for copying files, “rm” for removing files, and “chmod” for modifying file permissions are just a few examples of basic yet essential commands for managing the file system and user access.

Shell scripts take this functionality to the next level by allowing administrators to automate repetitive tasks and complex workflows. A shell script is essentially a sequence of command-line instructions written in a script file, which the shell interpreter can execute. This is particularly useful for tasks like scheduled backups, batch processing of files, and automated updates, where the same set of commands needs to be executed repeatedly. By scripting these tasks, administrators can ensure consistency, reduce the risk of human error, and save time. Shell scripts also support control structures such as loops and conditionals, enabling the automation of more sophisticated processes that require decision-making logic.

The “tail” command is another critical tool for system administrators, particularly in the context of log file monitoring. By default, “tail” displays the last ten lines of a file, making it ideal for viewing the most recent entries in system logs located in directories such as “/var/log/”. The command can be used with the “-f” option (e.g., “tail -f /var/log/messages”) to

continuously follow new additions to a log file in real-time. This is invaluable for troubleshooting and monitoring system activity, as it allows administrators to observe the immediate impact of configuration changes or to detect issues as they occur. For example, when debugging a network issue, an administrator can monitor the system log and network service logs simultaneously to correlate events and identify the root cause of the problem.

Overall, the combination of command-line commands, shell scripting, and real-time monitoring with tools like tail empowers system administrators to manage, automate, and troubleshoot Linux servers effectively

4.5 ENIQ Databases

ENIQ uses Oracle's relational database system that is a robust and widely used platform designed for managing and storing large volumes of structured data. It supports complex queries, transactions, and high-availability configurations, making it suitable for enterprise environments that require reliable and scalable data management solutions. The system uses SQL for querying and managing data, and it provides features like data replication, partitioning, and advanced security options to ensure optimal performance and data integrity.

ETL (Extract, Transform, Load) is a process used to extract data from various sources, transform it into a suitable format, and load it into a target database or data warehouse. This process is essential for integrating data from Mobile Network and preparing it for analysis and reporting. ETL workflows involve cleaning and validating data, transforming it to meet business requirements, and ensuring that the data is consistently and accurately loaded into the target system.

A database administrator (DBA) working with Oracle databases and ETL processes has several critical responsibilities. They must ensure the performance, security, and availability of the database by regularly performing tasks such as backups, patching, and tuning queries. In addition, they design and manage ETL processes to ensure data is extracted from various sources, transformed according to business rules, and loaded efficiently into the database. The DBA is also responsible for troubleshooting any issues that arise, such as data inconsistencies or performance bottlenecks, ensuring that the database and ETL processes run smoothly and reliably

5 Utilizing Tukiäly to troubleshooting

5.1 Prompting

The interaction with Tukiäly was conducted through a text-based interface, where users would input queries and receive responses from the chatbot. To enable Tukiäly to deliver accurate and useful responses, particularly for solving Unix/Linux errors or generating efficient shell scripts, it is essential to provide clear and specific inputs. Detailed descriptions of the issue or requirements for a task significantly improve the relevance and quality of responses.

It is important to note that Tukiäly cannot save prompts or recall previous conversations, except for the current active chat session. Once a session ends, Tukiäly has no memory of past interactions. Therefore, when writing prompts, users should be mindful of this limitation and ensure they provide all necessary context for the current session. Tukiäly can only refer to the information given within the active chat and cannot access any prior chats or inputs, making it essential to re-provide relevant details for every new query.

In technical troubleshooting scenarios, providing clarity involves including specific details such as the exact error message, the operating system version (e.g., Red Hat, Ubuntu), and any relevant system behavior leading to the issue. For example, when dealing with a "Permission Denied" error on a Red Hat server, specifying the location of the script and file permissions is crucial. Similarly, if a kernel panic occurs following a system upgrade, describing the system's inability to boot or any error messages seen during startup can guide the troubleshooting process more effectively.

When requesting shell scripts, specifying the task and all relevant details improves the quality of the results. For instance, stating that the script should monitor /var disk usage and send an alert if usage exceeds 80% is essential. Additionally, defining specific thresholds and alert mechanisms will help refine the solution. In cases where the best approach is unclear, asking for best practices allows for a structured and functional script to be generated.

Moreover, Tukiäly supports file uploads for enhanced interaction. Users can provide text files, Excel spreadsheets, SQL outputs, and other documents in various formats such as .txt, .csv, .xlsx, .sql, and .json for further analysis or processing. The maximum file size for uploads is typically 20MB, which is sufficient for most logs, reports, and datasets. This capability allows Tukiäly to process larger amounts of information efficiently and integrate it into troubleshooting or scripting tasks.

5.2 Tasks examples for redhat linux

In a typical scenario on a Red Hat Linux server, the system administrator may encounter an issue where the disk is filling up rapidly due to an ever-growing number of error log files. This situation often occurs when a process repeatedly logs the same error. In this case, assume that a certain log file in the “/var/log/ETL” directory is getting filled with error messages every minute, and the system keeps generating new log files.

The system administrator notices that the disk space on the server is running low, and upon investigation of the directories, specifically “/var/log/ETL”, they find that a new error log is created every minute. Each log file contains the same recurring error message, indicating that there is a process running out of control, logging errors due to a potential failure in the ETL pipeline.

Steps Taken by the System Administrator:

Initial Investigation: The sys admin inspects the “/var/log/ETL” directory and confirms that there are many error logs being generated every minute. This indicates that some process is continuously failing and logging these errors.

opening cmd tool and going to correct directory as System Administrator -user and checking the directory with command “ls -lh”

Figure 2 cmdline

```
cd /var/log/ETL
ls -lh
```

Figure 2 output:

```
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123000
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123002
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123004
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123006
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123008
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123010
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123012
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123014
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123016
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123018
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123020
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123022
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123024
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123026
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123028
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123030
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123032
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123034
-rw-r--r-- 1 root root 5.0K Oct 03 12:30 ERROR_ETL_connections_20241003_123036
```

These errors are so many that they fill up the diskspace

Using AI to Create a Diagnostic Script:

To track down which process is creating these error log files, the system administrator turns to AI (such as Tukiäly) to generate a script. The goal is to create a script that:

- Finds the **process ID (PID)** responsible for writing these logs.
- Identifies where the process was started from and which user initiated it.

Prompt to AI:

"Create a bash script that monitors which process is creating error logs in /var/log/ETL. The script should return the process ID, the user who started it, and the path where it was executed from."

Figure 3 AI generated script:

```
#!/bin/bash
LOG_DIR="/var/log/ETL"
ERROR_LOG=$(ls -t $LOG_DIR | head -n 1) # Get the latest log file
PID=$(lsof +D $LOG_DIR | grep "$ERROR_LOG" | awk '{print $2}') # Find PID writing

if [ -n "$PID" ]; then
    echo "Process ID: $PID"
    echo "User: $(ps -o user= -p $PID)"
    echo "Command Path: $(readlink /proc/$PID/exe)"
    echo "Started from: $(ps -o args= -p $PID)"
else
    echo "No active process found writing to logs."
fi
```

This script in Figure 3 does the following:

- Navigates to the /var/log/ETL directory and finds the most recently created log file.
- Uses the **lsof** command to determine which process is writing to this log file, extracts the process ID (PID), and returns the user who started the process.
- Outputs the command or path from which the process was executed and provides the complete arguments used to start the process.

Investigating the Root Cause: After running the script, the sys admin identifies the process causing the issue:

Figure 4 running Tukäly Shell script

```
./find_log_process.sh
```

Figure 5 findings

```

Process ID: 12345
User: etl_user
Command Path: /usr/bin/etl_process
Started from: /usr/bin/etl_process --source=db_conn

```

Figure 6 command to kill the faulted process

```
kill -9 12345
```

Cleaning Up Disk Space: After stopping the faulty process, the system administrator removes the unnecessary log files to free up disk space

Figure 7 cleaning of log file

```
rm -rf /var/log/ETL/*.log
```

to ensure that this error does not reoccur and cause the disk to fill up again, the system administrator can ask Tukiäly to create an automated alerting script. The goal is to monitor the creation of these log files and trigger an alert if they are generated too frequently. The normal expectation is that this specific error should occur no more than twice per hour. However, if more than 30 log files are created within one hour, particularly if they are being generated every two seconds over a 30-minute period, it indicates an abnormal situation.

The alerting system would be configured to monitor the /var/log/ETL directory, and if more than 30 error logs are created in an hour, a notification email will be sent to the system administrator's email address. This ensures that the sys admin is made aware of the issue before the disk becomes full, allowing for proactive resolution.

Here is the Tukiäly prompt for the alert script:

"Create a bash script that monitors /var/log/ETL for log files created every two seconds over a 30-minute period. If more than 30 log files are created in an hour, send an email alert to sysadmin@example.com."

The generated script could look like the output in Figure 6

Figure 8 Script created by Tukiäly

```
#!/bin/bash

LOG_DIR="/var/log/ETL"
THRESHOLD=30 # Max number of error logs allowed per hour
TIME_LIMIT=3600 # 1 hour (in seconds)
ALERT_EMAIL="sysadmin@example.com"
COUNT=$(find $LOG_DIR -type f -mmin -60 | wc -l) # Find logs created in the last hour

if [ "$COUNT" -gt "$THRESHOLD" ]; then
    echo "Alert: More than $THRESHOLD error logs created in the last hour." | mail -s 'ETL Error Log Alert' $ALERT_EMAIL
fi
```

This script seen in Figure 8 counts how many error logs are created in the last 60 minutes. If the number of logs exceeds 30, it sends an email notification to the system administrator. This early warning system helps ensure the sys admin is alerted before the logs consume excessive disk space, allowing them to take corrective action before the situation worsens.

5.3. Database work on ENIQ reports

The ENIQ SAP BI tool is used for day to day report creation. sometimes there are error situation due to upgrades , maintenance or changes to the database and some reports are affected. The most common Database errors are printed out as SQL errors or file permissions and they must be investigate by System Administrator Using Tukiäly it is possible to generate a bit more complex SQL commands and they can be made to be shells scripts or used via command line directly.

A typical error message in SAP BI (Business Intelligence) when there is a data type mismatch might happen during some feature change or upgrade project or when vendor is making regular software changes to the database. in this example there has been undocumented change such as changing a field from an integer to a text field. The BI reports fails and produces error that might look something like this:

"Error: Data type mismatch in table column. Field XYZ expected type INTEGER but found TEXT."

The error indicates that the data model or transformation logic needs to be updated to account for the new data type in the table or that the data is incorrectly formatted for its expected type.

in this example The AI will analyze the error and offer detailed information about its root cause. For example, a data type mismatch might occur when a field in the database expects an integer, but a text value is encountered. Understanding the error is the first step toward fixing it.

System administrator copy pastes the error to Tukiäly chat window and receives a response about the nature of error with some suggestions how to improve situation.

Once the nature of the error is clear, the AI is asked to generate a small shell script that can be executed from the command line. This script will inspect the properties of the affected database table, checking for issues like column data types or file permissions. For instance, the AI might generate a script that examines the structure of the database table.

user: "please make a shell script I can download and that will log in to database and finds these data columns an lists them in a downloadable file"

Example shell script for checking table properties:

```
```bash
```

```
#!/bin/bash
```

```
mysql -u user -p password -e "DESCRIBE XYZ;" > table_structure.txt
```

After examining the table's properties, the next step is to ask the AI to create an SQL script that can correct the error. This could involve changing a column's data type to match the expected format. For example, the AI can create an SQL command to modify the column data type and ensure it matches the required specification.

Example SQL correction script generated by the AI:

```
```bash
```

```
#!/bin/bash
```

```
mysql -u user -p password -e "ALTER TABLE XYZ MODIFY COLUMN field_name INTEGER;" > changes_log.txt
```

This SQL script can be run directly from the Linux command line. The changes made by the script are saved to a text file, so the user can review them later and verify what has been altered.

To ensure accuracy, the AI will generate a script that applies a single change first. This way, the user can verify the correctness of the modification before applying it more broadly. Once the initial change has been confirmed, the user can proceed with a more extensive modification, if necessary.

Finally, the AI is asked to generate another shell script to verify the data type change or any other modifications made to the database. This ensures that the user can confirm the change has been applied correctly and can check the data type or any other attribute.

Example verification script:

```
```bash
```

```
#!/bin/bash
```

```
mysql -u user -p password -e "DESCRIBE XYZ;" | grep field_name
```

All modifications made by these scripts will be logged in a text file (e.g., `changes\_log.txt`). This log allows the user to review all changes and track what the script has done, ensuring that the process is fully documented and traceable. IT is paramount that sys adm using AI has good knowledge of SLQ and the Database and ensured that backups are up to date in case of errors. it is good to understand the amount of damage that can be introduced with direct changes to database

## 5.4 Log search shell scripts

Oracle database is printing lot of error logs so it is feasible to check the alarm logs time to time. some alarms are monitored by the System administration tools and configured to send email or set to trigger alarm monitoring system which is alarming personnel to

promptly do actions as the flow of data might be compromised, once the actual fault has been cleared there will be root cause analysis and troubleshooter goes thru all the relevant backlogs and error files. this task is best done by shell -scripts that go thru the database alarms based on timestamps. any errors found on oracle logs are collected by

## 5.5. network issues on ENIQ

Network issues happen time to time due to the software changes or maintenance tasks or hardware failures In this scenario, a system administrator notices **time-out** issues on ENIQ **server** during network communications. The initial suspicion is that there might be a problem with the server's **network interface card (NIC)**, leading to packet loss or slow responses. After initial troubleshooting, it's suspected that the issue might be related to the **MTU size** (Maximum Transmission Unit) or a handshake problem between the server and the nearest switch.

For Identifying the Issue The system administrator provides the error details as text to Tukiäly, explaining the time-out issues, slow connections, or intermittent network performance degradation. **for example like this** : *"I'm facing network timeouts on a Linux Red Hat HP server. I suspect the issue may be related to the MTU size or handshake problems between the server and the switch. How can I investigate this further and check if MTU is causing the issue?"*

The AI analyzes the issue and suggests that the root cause could be an MTU mismatch or an improper TCP handshake between the Linux server and the nearest switch. The Tukiäly suggests checking the server's MTU configuration and verifying if it matches the network's expected MTU size.

The Tukiäly then generates a shell script or provides commands that the system administrator can use to check the current **MTU** size configured on the network interface card (NIC).

**Example AI Response:** *"To check the current MTU size on the server's NIC, use the following command to investigate the settings:"*

Figure 9

```
#!/bin/bash
Check MTU size of the network interface
ip link show eth0 | grep mtu
```



This command in figure 9 will display the MTU size configured on the NIC (assuming the interface is eth0). If the MTU size is different from the expected MTU size in the network (e.g., 1500 bytes), it might be causing packet fragmentation or timeouts

After discovering that the MTU size might be incorrect (for example, set to **9000** when the network requires **1500**), the system administrator asks the AI to help adjust the MTU setting and test if the problem is resolved.

*"The MTU size on my interface is currently set to 9000, but the network expects it to be 1500. Can you help me adjust it and test the connectivity?"*

**Example AI Response:** *"You can adjust the MTU size on the interface with the following command:"*

Figure 10

```
#!/bin/bash
Change MTU size to 1500 for interface eth0
sudo ip link set dev eth0 mtu 1500
```

The administrator runs this command (figure 10) , which sets the MTU size to **1500**, and then performs a **ping** test to verify connectivity:

Figure 11

```
ping -M do -s 1472 192.168.1.1
```

In this test in figure 11 , "-M do" ensures the ping packets don't fragment, and -s 1472 ensures that the total packet size (1472 bytes of data + 28 bytes for ICMP header) matches the new MTU size

If the issue persists after adjusting the MTU, the AI suggests checking for any **TCP handshake** issues between the server and the switch, especially if there's still time-out behavior. The AI generates a command to capture network traffic and analyze TCP SYN and ACK packets.

**Example AI Response:** *"To check for any TCP handshake issues, you can use tcpdump to capture network traffic and look for abnormal SYN or ACK packet behavior."*

Figure 12

```
#!/bin/bash
Capture TCP handshake packets on interface eth0
sudo tcpdump -i eth0 'tcp[tcpflags] & (tcp-syn|tcp-ack) != 0'
```

Command in figure 12 allows the system administrator to capture TCP traffic and monitor the SYN and ACK exchanges, which might reveal any packet loss, delayed acknowledgments, or handshake retries that are leading to time-outs.

Once the issue has been investigated and adjustments made, the AI recommends performing additional connectivity tests (such as **traceroute** or **ping** tests) to ensure that the server is now communicating properly with the switch.

**System Administrator asks** *"Please provide commands to perform connectivity tests after adjusting the MTU."*

**Response:** *"Use the following commands to test connectivity and verify if the timeout issue has been resolved:"*

Figure 13

```
Perform a ping test to check connectivity
ping -c 5 192.168.1.1

Use traceroute to check the route to the destination
traceroute 192.168.1.1
```

These tests mentioned in figure 13, help ensure that the changes have resolved the network time-out issue.

After completing the adjustments, the Tukiäly might suggest setting up a monitoring script that periodically checks the status of the network interface, logs any changes in MTU size, or captures packet loss over time.

**Example Monitoring Script:**

Figure 14

```
#!/bin/bash
Monitor MTU size and connectivity every 10 minutes
while true; do
 echo "Checking MTU size..."
 ip link show eth0 | grep mtu
 echo "Testing connectivity..."
 ping -c 5 192.168.1.1
 sleep 600 # Check every 10 minutes
done
```

By using this script in figure 14, the system administrator can continuously monitor the network interface and ensure the settings remain stable.

Key Benefits of Using AI in this Quick Troubleshooting was that Tukiäly helps diagnose the issue quickly, pointing out potential root causes like MTU mismatch or handshake failures, provides precise commands to check MTU size, capture traffic, and adjust settings, saving time. Also Tukiäly generates scripts to test and verify the fixes, ensuring that the problem is properly addressed. last but no least it helps automate monitoring, ensuring the problem doesn't recur without the system administrator's knowledge.

## 5.6. Problem in ETL Process

In an ETL (Extract, Transform, Load) data pipeline, data is continuously received, transformed, and passed along for further processing or storage. If an issue arises at any point in this pipeline, such as receiving an empty file or no data at all, it can create significant gaps in the dataset, leading to incomplete or inaccurate reporting. These issues may be caused by upstream problems, such as failed data extraction or transmission errors. When such anomalies occur, they need to be investigated and resolved promptly to avoid disruptions in the data flow and prevent missing or corrupted data from affecting downstream processes. Identifying and troubleshooting the root cause of such problems is critical to maintaining the integrity of the ETL pipeline and ensuring reliable data delivery this is an example of real life situation where the data pipeline process encounters an error. the error message is put as it is to Prompt for Tukiäly and the following correspondence is executed ;

```
"11.04 15:30:14 19249 WARNING etl.CUSTOM_DC_BTECH.Loader.Loader_CUS-
TOM_DC_BTECH_BLADE_POWER.1.PartitionedLoader : ZERO ROWS LOADED TO TA-
BLE CUSTOM_DC_BTECH_BLADE_POWER_RAW_03 from file(s)
'/ENIQ/data/etldata_/00/custom_dc_btech_blade_power/raw/CUS-
TOM_DC_BTECH_BLADE_POWER_1618144212894_b20_2021-
```

04-11.txt','/ENIQ/data/etldata\_/03/custom\_dc\_btech\_blade\_power/raw/CUSTOM\_DC\_BTECH\_BLADE\_POWER\_1618144212912\_b20\_2021-04-11.txt"

Figure 15

```
ls -lh /eniq/data/etldata_/00/custom_dc_btech_blade_power/raw/CUSTOM_DC_BTECH_BLADE_POWER_1618144212912_b20_2021-04-11.txt
ls -lh /eniq/data/etldata_/03/custom_dc_btech_blade_power/raw/CUSTOM_DC_BTECH_BLADE_POWER_1618144212912_b20_2021-04-11.txt
```

If the files are **0 bytes**, they are empty.

### Command to Preview File Contents:

Figure 16

```
head -n 10 /eniq/data/etldata_/00/custom_dc_btech_blade_power/raw/CUSTOM_DC_BTECH_BLADE_POWER_1618144212912_b20_2021-04-11.txt
head -n 10 /eniq/data/etldata_/03/custom_dc_btech_blade_power/raw/CUSTOM_DC_BTECH_BLADE_POWER_1618144212912_b20_2021-04-11.txt
```

This will show you the first 10 lines of the files. You can also use `cat` to view the entire file, but `head` is faster for large files.

## 2. Check for Schema Mismatch Between Data and Table

You can manually compare the source file structure with the table structure.

### Command to Check Table Structure:

Figure 17

```
mysql -u root -p -e "DESCRIBE CUSTOM_DC_BTECH_BLADE_POWER_RAW;" > table_structure.txt
cat table_structure.txt
```

Compare this output with the structure of your data in the source file. If there are any inconsistencies (e.g., field types), it might indicate a schema mismatch.

### Script to Check Field Mismatch (Example for CSV File):

If your files are in CSV format, you can use the following script to quickly check for mismatched columns

Figure 18

```

FILE="/eniq/data/etldata_/00/custom_dc_btech_blade_power/raw/CUSTOM_DC_BTECH_BLADE_POWER_RAW"
TABLE="CUSTOM_DC_BTECH_BLADE_POWER_RAW"

Get column names from the file (assuming CSV)
FILE_COLUMNS=$(head -n 1 $FILE | tr ',' '\n')

Get column names from the MySQL table
TABLE_COLUMNS=$(mysql -u root -p -e "DESCRIBE $TABLE;" | awk '{print $1}')

echo "File Columns:"
echo "$FILE_COLUMNS"

echo "Table Columns:"
echo "$TABLE_COLUMNS"

Compare columns
diff <(echo "$FILE_COLUMNS") <(echo "$TABLE_COLUMNS")

```

Run the script and check if there are differences between the file columns and the table columns.

### 3. Check Partitioning Issue

Check whether the data is being partitioned correctly. If you have partitioned tables, you can examine how the data is split.

#### Command to List Table Partitions:

Figure 19

```
mysql -u root -p -e "SHOW TABLE PARTITIONS LIKE 'CUSTOM_DC_BTECH_BLADE_POWER_RAW';"
```

This will display the partitions in the table. Compare the output with the expected partition structure to verify if there's an issue.

#### Script to Check Partitioning in Data Files:

If your source files are supposed to be split by partition (e.g., different files for different partitions), you can create a script to check the partition distribution in the files.

Figure 20

```
#!/bin/bash

FILE="/eniq/data/etldata_/00/custom_dc_btech_blade_power/raw/CUSTOM_DC_BTECH_BLADE_POWE

Check if file contains partition data
grep "partition" $FILE | sort | uniq -c
```

Run the script and verify if the partitions mentioned in the file match the actual partitions in the table.

#### 4. Check File Permissions

The ETL process may not have permission to write to the table or access the source files.

##### Command to Check File Permissions:

Figure 21

```
ls -l /eniq/data/etldata_/00/custom_dc_btech_blade_power/raw/CUSTOM_DC_BTECH_BLADE_POWE
ls -l /eniq/data/etldata_/03/custom_dc_btech_blade_power/raw/CUSTOM_DC_BTECH_BLADE_POWE
```

Ensure that the files have the correct permissions for the ETL process to read them.

##### Command to Check Table Permissions:

Figure 22

```
mysql -u root -p -e "SHOW GRANTS FOR 'etl_user'@'localhost';"
```

This command will show you the privileges that the ETL user has. Ensure that the user has **INSERT** privileges on the table.

#### 5. Retry the Load

Once you've investigated and resolved the issues, retry the data loading process.

##### Command to Re-run the ETL Loader (Example Command):

Figure 23

```
./run_etl_loader.sh
```

Ensure that the script or command used to load the data is correct and rerun the ETL process after fixing any identified issues.

### Bonus: Monitor the ETL Process for Errors

You can set up a script to monitor the ETL log files in real-time to catch errors as they occur.

#### Script to Monitor Log File:

Figure 24

```
#!/bin/bash

LOG_FILE="/eniq/data/etldata/logs/etl_loader.log"

tail -f $LOG_FILE | grep --line-buffered -E "ERROR|WARNING|ZERO ROWS LOADED"
```

This script continuously monitors the log file and highlights any errors or warnings, such as "ZERO ROWS LOADED." ChatGPT

In this interaction, the issue involved a Red Hat Linux system where an ETL process failed to load any rows into the target database table, prompting the need for investigation. The system administrator (Sys Admin) used ChatGPT to quickly identify several potential causes, such as empty or corrupt files, schema mismatches, partitioning issues, and permission problems. ChatGPT provided specific commands and shell scripts for each scenario, helping the Sys Admin to verify the integrity of the files, check the database schema, investigate partitioning configurations, and ensure proper file and user permissions. Additionally, commands to re-run the ETL process and monitor logs in real-time were suggested.

In this case, Tukiäly significantly accelerated the troubleshooting process by offering clear directions and appropriate shell commands that the Sys Admin could use immediately. The AI provided targeted solutions and actionable steps that would have otherwise required more manual research, effectively streamlining the diagnosis and resolution of the issue.

there are several key benefits and considerations to keep in mind. Tukiäly is most effective when prompts are specific and detailed, allowing it to deliver accurate and targeted suggestions for troubleshooting, diagnostics, or automation tasks. It serves as a valuable tool for quickly diagnosing system issues, generating shell scripts, and providing step-by-step guidance on complex problems.

At the same time, it's crucial to maintain security and privacy standards, ensuring that sensitive company information, such as IP addresses or internal passwords, is never directly shared. Instead, placeholders can be used in prompts to avoid exposing sensitive data. Tukiäly's ability to assist in the automation of routine tasks—such as log monitoring or

system backups—can greatly improve the efficiency of system administrators by reducing manual workloads.

Additionally, Tukiäly can be used for learning and documenting new tools and processes, aiding in both the personal development of system administrators and the creation of internal documentation. However, it is essential to validate any generated scripts or commands by testing them in a controlled environment before deploying them in production. While Tukiäly offers powerful support in daily system administration tasks, the responsibility of executing changes and ensuring system stability ultimately rests with the administrator.

By applying Tukiäly effectively within these guidelines, system administrators can streamline operations, enhance troubleshooting efficiency, and automate routine tasks, all while adhering to the organization's security and operational standards.

## 6 CONCLUSION

In this thesis, the focus was on the practical use of AI, OpenAI's GPT-based Tukiäly, in system administration for maintaining, managing, and troubleshooting DNA's radio network analytics systems. System administration involves working with a wide range of complex systems, databases, hardware and network technologies. While system administrators are expected to have broad expertise, it's unrealistic to expect anyone to master every tool and technology in detail. This is where Tukiäly becomes a valuable asset, assisting administrators in handling tasks they might not be fully familiar with and providing speed and clarity in diagnosing detailed issues across various systems.

A key takeaway from the research is that while Tukiäly can generate commands and scripts for system management, it is crucial that system administrators understand the processes before implementing them. Simply copying and running commands directly from Tukiäly chatbot is not recommended. Administrators must carefully review and test each command or script in a controlled environment to ensure they function as intended without causing further system issues. Tukiäly's strength lies in helping administrators create smaller test cases, allowing them to validate the logic before applying changes on a larger scale. This careful approach is essential to avoid system overloads or misconfigurations. Moreover, Tukiäly accelerates the troubleshooting process by providing relevant suggestions on the appropriate commands and scripts to use. For example, when parsing large log files, Tukiäly quickly delivers precise scripts, eliminating the need to reinvent solutions from scratch.



In conclusion, while Tukiäly offers significant advantages in assisting with system administration tasks, its effectiveness depends on the administrator's ability to understand and validate the AI-generated outputs. Tukiäly can greatly enhance efficiency by providing tailored troubleshooting steps and scripts, speeding up root cause analysis, and offering useful insights that reduce the time required to solve complex issues. It allows administrators to focus on fixing the problem without having to create solutions from the ground up. Tukiäly acts as a supportive tool, enhancing the administrator's capabilities without replacing the need for expertise and careful oversight.

## SOURCES

Kapronczay, Mór. 2022. A Beginner's Guide to Language Models. Online. BuiltIn.  
<<https://builtin.com/data-science/beginners-guide-language-models>>.

Roldós, Inés. 2020. NLP, Machine Learning & AI, Explained. Online. Mon-keyLearn.  
<<https://monkeylearn.com/blog/nlp-ai/>>. 9.6.2020. Accessed 25.5.2024

Vassiliadis, Panos. Springer, 2009 A Survey of Extract–Transform–Load Technology

Kirkland, Dustin.2016 Learn Linux in a Month of Lunches. Manning Publications, 2016

Covers essential Linux commands and scripting for effective system administration. Kirkland, Dustin. Learn Linux in a Month of Lunches. Manning Publications, 2016.

Holma, Harri, and Antti Toskala. LTE for UMTS: Evolution to LTE-Advanced. 2nd ed., Wiley, 2014.

Vassiliadis, Panos. A Survey of Extract–Transform–Load Technology. Springer, 2009.

