Model-Based Systems Engineering for Internet of Things (IOT) Based Mesh Radio Network

By

Khawaja Fahad Shafi

00000361333

Session 2021-2023

Supervised by

Dr. Mian Ilyas Ahmad

A dissertation submitted to National University of Sciences &
Technology in partial fulfilment of the requirement for the degree of
MASTER OF SCIENCE
in
Systems Engineering

School of Interdisciplinary Engineering and Sciences (SINES)

National University of Sciences and Technology (NUST)

H-12, Islamabad 44000, Pakistan

February, 2024

Dedication

With boundless gratitude, I dedicate this thesis to my unwavering pillars of strength – my parents, cherished brother, and beloved sisters. Your incessant support, constant presence, and guiding light have illuminated my journey.

Declaration of Authorship

I, Khawaja Fahad Shafi, solemnly affirm that this thesis entitled "Model-Based Systems Engineering for Internet of Things (IoT) Based Mesh Radio Network" and the endeavours encapsulated herein are a manifestation of my own intellect and diligence. I affirm the following:

- This endeavour was undertaken entirely during my tenure as a research degree candidate at NUST.
- Any previous submission of portions of this thesis for a degree or other qualifications, whether at NUST or any other institution, has been explicitly acknowledged.
- In instances where I have drawn insights from the published works of others, due attribution has been meticulously accorded.
- In cases where I have cited the words of others, proper citation has been dutifully provided. This thesis, apart from such citations, is an embodiment of my individual work.
- I have gratefully acknowledged all significant sources of assistance that have guided me throughout this academic pursuit.
- In scenarios where this thesis draws upon collaborative work with others, I have unambiguously demarcated the contributions of others from my own.

With a sincere commitment to intellectual integrity, I affix my endorsement to this testament.

Signature: ______________________

Date: _________________________

Copyright Notice

Abstract

Model-Based Systems Engineering (MBSE) has emerged as a paradigm shift in the development of complex systems, providing a holistic and integrated approach that transcends the limitations of conventional document-centric Systems Engineering (SE) methodologies. The literature identifies the traditional system development models like V-model, waterfall, and spiral models, document-centric models, leading to disintegrated development, increased costs, and project delays. This study focusses on employing ARCADIA, an integrated MBSE approach, to engineer an IoT-based mesh radio network system utilizing Software-Defined Radio (SDR) technology. ARCADIA a MBSE framework addresses the challenges of traditional document centric methods in system engineering. First phase, operational analysis, was conducted to capture and define stakeholder needs, operational environment, and conditions in which the system will operate. System analysis built upon operational analysis to formalize system requirements and articulate the dynamic behavior of the system. The third step involved defining the logical architecture, treating the system as a white box. The final step involved the development of the physical architecture, specifying how the system will be built, by defining system components, detailing interfaces, and conducting a final tradeoff analysis, ultimately leading to the final architecture of the system. ARCADIA implementation resulted in addressing the complexities of SDR system through enhanced communication, collaboration, and integrity of people, processes, and product. Its efficiency in requirement management, change management and seamless verification and validation resulted in efficient system development as per the user requirements. The cost benefit analysis identified the significant return on investment (ROI) of 10% on overall project costs. The impact of this research extends beyond the present, setting a path for future applications in various sectors like unmanned arial vehicle, aerospace, telecommunication, energy systems, smart cities, healthcare, and transportation. This study, implementation of MBSE for IoT based mesh radio network, opens doors for a transformative shift in approach to develop complex system across commercial and defense landscapes.

Keywords: Systems Engineering (SE), Model-based systems engineering (MBSE), ARCADIA, Capella, Software defined radio (SDR)

Annex A to NUST Letter No.
0972/102/Exams/Thesis-Cert
dated 23 Dec 16.

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS/MPhil thesis written by Mr/Ms Khawaja Fahad Shafi Registration No. 361333 of **SINES** has been vetted by undersigned, found complete in all aspects as per NUST Statutes/Regulations, is free of plagiarism, errors, and mistakes and is accepted as partial fulfillment for award of MS/MPhil degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the said thesis.

Signature with stamp:

Dr. Mian Ilyas Ahmad
HoD Engineering
Professor
SINES - NUST, Sector H-12
Islamabad

Name of Supervisor: Dr. Mian Ilyas Ahmad
Date: 13/02/2024

Signature of HoD with stamp:

Dr. Mian Ilyas Ahmad
HoD Engineering
Professor
SINES - NUST, Sector H-12
Islamabad

Date: 13/02/2024

Countersign by

Signature (Dean/Principal):

Dr. Hammad M. Cheema
Principal & Dean
SINES - NUST, Sector H-12
Islamabad

Date: 27/02/2024

Acknowledgements

In the name of Allah, the Most Gracious, the Most Merciful. All praises are due to Allah Almighty, the Guiding Light who illuminated my path through this journey, revealing windows of insight when darkness loomed ahead. I offer gratitude to Allah Almighty for bestowing upon me the gifts of consistency and diligence, which paved the way for the completion of this endeavour. My heart brims with gratitude to Allah Almighty for endowing me with awareness, perception, and an unceasing flow of ideas. These blessings infused me with the courage, fortitude, and patience necessary to embark on and triumph in this odyssey.

Foremost, I extend my profound appreciation to my parents and siblings, whose unwavering encouragement and unwavering belief in me propelled me forward through every phase of life. Their boundless support, unshakable trust, and unparalleled encouragement have been the wind beneath my wings, consistently inspiring me to strive earnestly towards my aspirations.

I am indebted to my thesis supervisor, Dr. Mian Ilyas Ahmad, for his unwavering support and invaluable guidance throughout my research journey. His mentorship transcended boundaries, offering not only technical insight but also lessons in humility, kindness, and inclusivity. His encouragement to uphold high standards, regardless of the complexity of the challenges, illuminated my path and fortified my resolve. In times of adversity, his guidance became a beacon of hope, illuminating the way forward.

To the esteemed members of the General Appraisal Committee - Dr. Shahzad Rasool, and Dr. Ammar Mushtaq - I extend my heartfelt gratitude for your invaluable contributions and unwavering support throughout this project.

A special acknowledgment is reserved for Engr. Syed Salman Khalid, who entrusted me with an opportunity to showcase my potential, igniting the flame of creativity within me.

Numerous individuals have stood beside me throughout my master's journey, guiding me and revealing new horizons. I extend heartfelt appreciation to each one.

Lastly, my friends, you have been my pillars of strength throughout this journey. Without your camaraderie, life at NUST would have been a solitary path. The memories we've created and the bonds we've formed are treasures I shall forever cherish.

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction

This chapter initiates a comprehensive discussion by introducing the background of Model-based systems engineering (MBSE) and software-defined radio (SDR). It proceeds to outline the problem statement, research objectives, motivation, and the potential areas of application for Model-Based Systems Engineering and software-defined radio. Finally, the chapter provides an overview of the thesis layout in its concluding section.

1.1 Background

Model-based Systems Engineering (MBSE) is a system engineering approach that uses models to represent the different aspects of a system, such as its architecture, design, behavior, and requirements. The models are used to capture, analyze, and communicate the design and behavior of the system. MBSE is a holistic approach that allows engineers to consider the entire system, rather than just its individual components, and to evaluate the system's behavior and performance over its entire life cycle.

The origins of Model-based Systems Engineering (MBSE) can be traced back to the early days of systems engineering in the 1950s and 1960s, when the systems engineering process began to be formalized and standardized. The early systems engineers recognized the need for a more structured and systematic approach to the design and development of complex systems and began to use models to represent different aspects of the system, such as its architecture, design, and requirements.

In the 1970s, the field of computer-aided design (CAD) began to emerge, and systems engineers began to use computer-based tools to create, analyze, and manage system models. This led to the development of the first computer-aided systems engineering (CASE) tools, which were used to automate and streamline the systems engineering process. In the 1980s and 1990s, the use of object-oriented modeling languages, such as the Unified Modeling Language (UML), began to gain popularity in the software engineering community. This led to the development of the first MBSE tools that used UML to represent the system's architecture, design, and behavior. In the 2000s, the Systems Modeling Language (SysML) was developed specifically for systems engineering, and it quickly became the de facto standard for MBSE. SysML is an extension of

UML, and it provides a set of specific modeling elements and diagrams that are well-suited for representing the system's architecture, design, and behavior.

MBSE typically involves the use of formal modeling languages, such as the Systems Modeling Language (SysML) or the Unified Modeling Language (UML), to represent the system's architecture, design, and behavior. These models can be used to generate code, simulations, and other outputs that can be used to test and validate the system's design. One of the key advantages of MBSE is that it allows engineers to simulate and analyze the system's behavior before it is built, which can help to identify and resolve issues early in the development process. This can lead to a more efficient and cost-effective development process and can improve the quality of the final product.

Applying MBSE for the development of the SDR allows a comprehensive understanding of the radio system, from its architecture to its behavior. The Software Defined Radio (SDR) is a communication device utilized in commercial and defense to facilitate short-range communication. Its primary purpose is to maintain seamless communication within short range and with command headquarters. As a lightweight and portable radio system, the SDR is designed to be conveniently worn on the individual, equipment, typically affixed to their body or vest. By enabling secure and effective communication in the field, the SDR significantly enhances situational awareness, coordination, and overall command and control capabilities.

The specific capabilities and features of the Personnel Role Radio can vary depending on the manufacturer and model. The systems capabilities are summarized in the Figure 1.

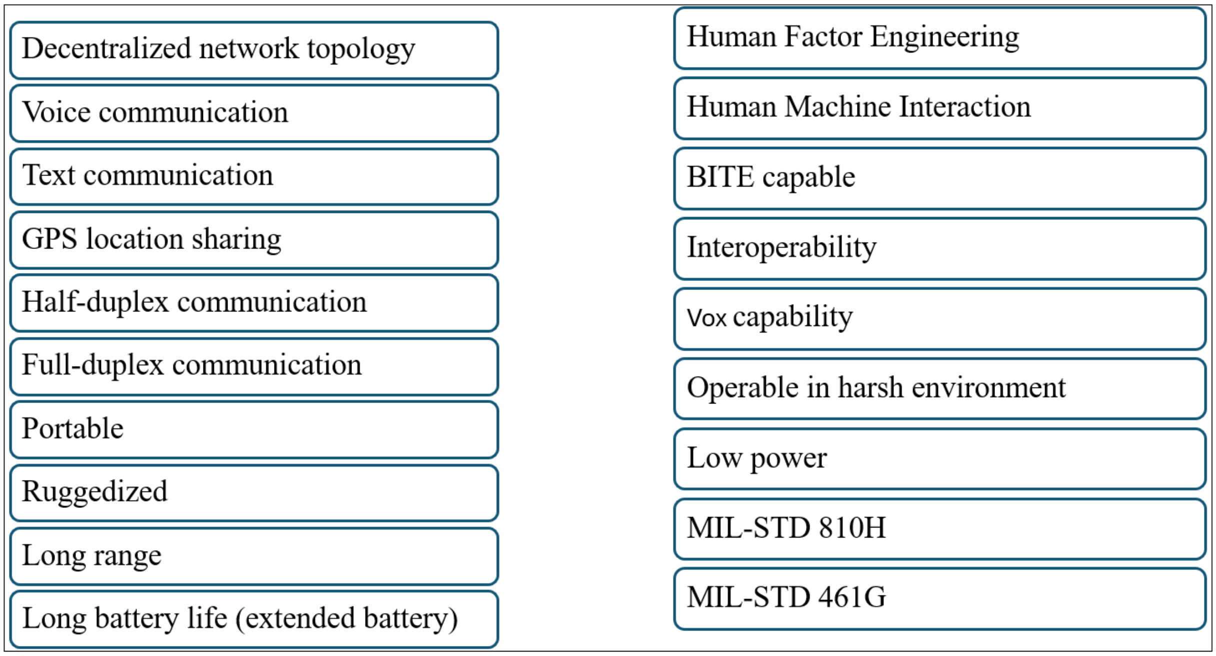


Figure 1: Software Defined Radio Capabilities

1.2 Problem Statement

Centralized radio systems are prone to failures from central device malfunctions. To achieve uninterrupted communication in dynamic environments, an independent radio system is necessary. Model-based systems engineering offers a comprehensive and integrated approach for developing IoT-based mesh radio networks and addressing these challenges. key reason for the need of IoT based mesh radio network and use of MBSE are for the development of IoT based mesh radio network are listed herein:

- Centralized radio communication systems are susceptible to failure in the event of malfunctioning of the central device.
- In dynamic and challenging environments necessitate uninterrupted communication, which can be achieved through the development of a radio system independent of a central device.
- Currently used systems engineering techniques and methodologies lack the essential element of integrated development.
- Model-based systems engineering emerges as a promises a comprehensive and integrated methodology capable of developing and managing complex systems.

- The primary objective of this research is to implement the Arcadia methodology, an integrated approach, for the development of the architecture of an IoT-based mesh radio network.

1.3 Research Objectives

The research objectives are listed herein.

1) Implementation of Model-based Systems Engineering in Capella for IOT-based Mesh Radio Network.
    - Implementation of Operational Analysis, System Analysis, Logical Architecture and Physical Architecture.
    - Integration of Arcadia phases.
2) Comprehensive comparison of Systems Engineering and Model-Based Systems Engineering
- Parameters: Economic Analysis, Integration, and verification & validation.
3) Qualitative Analysis of Model-based Systems Engineering.

1.4 Motivation

The growth of the Internet of Things (IoT) has resulted in the emergence of complex and interconnected systems, such as mesh radio networks, which require efficient and effective systems engineering methodologies for their design and development. Model-Based Systems Engineering (MBSE) has gained popularity as an approach that emphasizes the use of formal models to capture and communicate system information throughout the development lifecycle. Capella, an open-source MBSE tool, has become prominent in recent years due to its capabilities. This study aims to explore and evaluate the implementation of MBSE with Capella for the design and development of IoT-based mesh radio network systems. The research objective is to enhance system quality, reduce development time and cost, and improve collaboration among stakeholders. This study seeks to contribute to the body of knowledge on MBSE and its applicability in the field of systems engineering. The findings of this research can potentially guide practitioners and decision-makers in the effective design and development of IoT-based mesh radio network systems.

A mesh based SDR network refers to a communication network where multiple SDR devices are interconnected in a mesh topology to facilitate communication among group of users. In a mesh network, each SDR device serves as a node that can transmit and receive messages, and the nodes cooperate to relay messages to reach their intended destination. In a mesh based SDR network, the individual SDR devices communicate with each other in a decentralized manner. This means that each SDR device acts as both a transmitter and a relay station, allowing messages to be dynamically routed through the network. This mesh topology provides several advantages, including increased network coverage, improved resiliency, and the ability to establish multiple communication paths.

Key characteristics and benefits of a mesh based SDR network are listed herein:

- Increased Coverage
- Resilient Communication
- Ad hoc Connectivity
- Redundancy and Reliability
- Scalability
- Enhanced Situational Awareness

The aforementioned capabilities presented in these sections serve to effectively mitigate the existing issues identified in the personal role radios, as depicted in Figure 2.

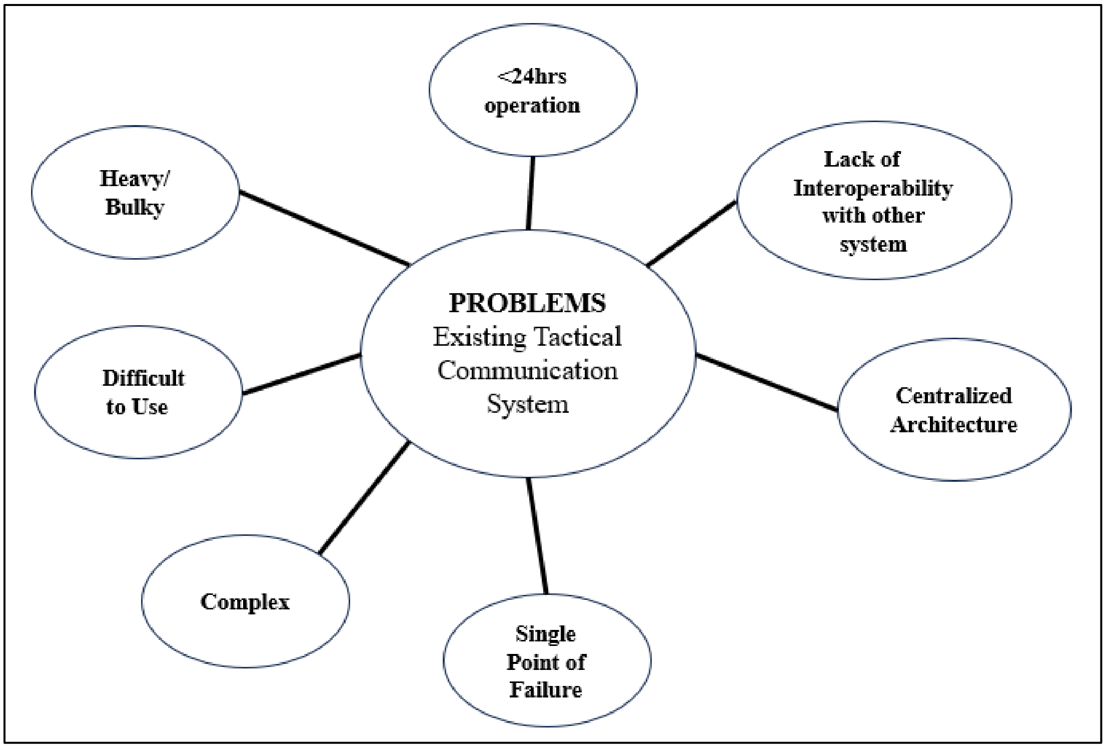


Figure 2: Existing SDR Systems Problems

1.5 Area of Application

Model-Based Systems Engineering (MBSE) and IoT-based mesh radio networks, specifically personnel role radios, find extensive applications across various industries. The utilization of MBSE and personnel role radios is prevalent in numerous sectors. The area of application for MBSE and personnel role radios is independent to each other.

Area of application for MBSE are listed herein:

- Aerospace and Defense
- Automotive Industry
- Telecommunication
- Healthcare Systems
- Healthcare Systems
- Energy Systems
- Transportation Systems
- Robotics and Automation

Application area for the personal role radio listed herein:

- Emergency Response and Disaster Management
- Border Patrol and Surveillance
- Search and Rescue Missions
- Military Operations
- Off-grid Communication
- Law Enforcement
- Security Services
- Public Events

1.6 Thesis Layout

Chapter 1, Introduction, sets the groundwork by precisely defining the problem statement and articulating the research objectives. In Chapter 2, Literature Review, a comprehensive examination of model-based systems engineering, and IoT-based mesh radio networks is presented. Chapter 3, Methodology, explains the implementation of the Arcadia methodology for the development of the personal role radio's architecture. Chapter 4, Results and Discussion, an exhaustive analysis of the research outcomes in relation to the predetermined objectives is provided. Lastly, Chapter 5, Conclusion and Future Work, presents a comprehensive consolidation of the thesis, along with pertinent suggestions for potential path of future research.

Chapter 2 Literature Review

This chapter specifies an in-depth analysis of Model-Based Systems Engineering (MBSE) and Software-Defined Radio (SDR). Structured into three main sections, the first section explores the fundamental concepts and principles governing MBSE, offering a comprehensive comparative assessment of MBSE and traditional Systems Engineering, and ultimately elucidating the benefits derived from MBSE implementation. The second section introduces Software-Defined Radio, highlighting its advantages, and delves into the practical applications of Model-Based Systems Engineering within the realm of SDR technology. The chapter concludes with discussing the research gap in the final section.

2.1 Model-Based System Engineering (MBSE)

2.1.1 Introduction

Engineers have been using models in various forms for a long time, and it has been an essential part of their profession for decades. However, as systems became larger and more complex, engineers needed a new approach to system development. This led to the emergence of model-based systems engineering (MBSE) [1], a design process that revolves around using models as the core of system development. MBSE focuses on using models throughout a system's life cycle for tasks like requirement gathering, trade studies, design, analysis, and verification and validation. It aims to bring greater rigor and effectiveness to the development of complex systems. In MBSE, the "model" is the central source of truth, capturing multiple perspectives that answer various stakeholder questions. Unlike traditional engineering with models, where multiple models with different assumptions and semantics are used, MBSE employs a single model that stores all system-related information in a central repository. This feature enables the interconnection of model elements, efficient information retrieval, and systematic reasoning about the system. It also allows for automatic propagation of design changes, consistency checks, and error identification, which are crucial advantages of MBSE. As MBSE continues to mature and gain wider application, it proves to be a valuable approach in managing the complexities of developing sophisticated systems.

INCOSE defines the Systems Engineering [2] as: “*Systems Engineering is a transdisciplinary and integrative approach to enable the successful realization, use, and* *retirement of engineered systems, using systems principles and concepts***,** *and scientific, technological, and management methods".*

INCOSE define Model-Based Systems Engineering as: "*The formalized application of modeling to support system requirements, design, analysis, verification, and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases".*

2.1.2 Systems Engineering Evolution

The origins of SE can be traced back to the 1950s and 1960s [3], with the development of large-scale, complex systems such as aerospace and defense projects. SE emerged as a discipline to manage the complexities of these systems and ensure their successful development. In the 1970s and 1990s [4], SE practices primarily relied on documents to capture and communicate system information. These documents included requirements specifications, design documents, and test plans. In the 1990s, the limitations of traditional document-based SE became apparent as systems grew increasingly complex. MBSE emerged as a response to these limitations, emphasizing the use of models to represent system information [5]. The 2000s and onwards have seen the maturation of MBSE, with the development of standardized modeling languages, tools, and methodologies. MBSE has gained widespread adoption across various industries, including aerospace, automotive, and healthcare [6].

2.1.3 Model Based Systems Engineering Languages

The 1960s marked the early stages of MBSE, characterized by the emergence of data flow diagrams (DFDs) and entity-relationship diagrams (ERDs) as primary modeling tools. These languages focused on representing system data structures and relationships, laying the foundation for more advanced MBSE languages [7], [8]. The 1980s witnessed the introduction of the Unified Modeling Language (UML), a significant advancement in MBSE languages. UML provided a comprehensive set of diagrams for representing system requirements, design, behavior, and deployment. Its versatility and comprehensiveness made UML a dominant MBSE language for various system types [9]. The 1990s saw the development of SysML (Systems Modeling Language) [10], a specialized MBSE language tailored for modeling complex systems.

SysML addressed the limitations of UML in handling the intricacies of systems modeling, providing a more comprehensive and domain-specific language for systems development.

2.1.4 Advancements in Model-Based Systems Engineering (MBSE)

The modern world is full of complex systems involving individuals, tools, software, data, procedures, and physical structures. Because of the way they interact, these systems need a comprehensive strategy to be fully understood and developed. More people and groups participate in these systems as they become more complex and vaster, bringing with them a variety of viewpoints, abilities, duties, and interests. The field of systems engineering (SE) focuses on handling these complex systems. By offering the skills and techniques required to properly design and implement these systems, it seeks to comprehend and manage the entire sociotechnical system. Different definitions of systems engineering (SE) appeared in the 1970s as the subject began to develop. Although these definitions may differ, they all share certain similar concepts. These concepts consist of viewing systems, considering how various system components interact, and managing the full system life cycle with an engineering perspective. Early definitions from the 1970s put more of an emphasis on converting requirements into designs. The definitions from the 1990s and 2000s, on the other hand, grew to accommodate a wider viewpoint. They emphasized a more comprehensive viewpoint, considering how systems interact with social and technical factors as well as their emergent qualities [11].

MBSE is a way to design complex systems that is more organized and reliable. It uses interconnected models to represent and study the systems at every stage of development. MBSE focuses on two important aspects of creating reliable systems: Resilience Contracts (RC) and simulation-based testing methods [12].

The paper discusses the transition from traditional document-based approaches to model-centric ones in systems engineering [11]. It explores various aspects of Model-Based Systems Engineering (MBSE) and their integration for system success. The current state of MBSE, encompassing standards, formalisms, modeling languages, methodologies, and applications, is examined. Special attention is given to three key formalisms: a semantic glossary and model for SE concepts, an information model for system design, and a mathematical model for SE and MBSE. The paper employs literature review and analysis to provide insights into the evolving MBSE paradigm, which is expected to become a standard practice in systems engineering. It underscores MBSE's leadership in handling modern, complex interdisciplinary systems-of-systems (SoS) and advocates for standardization to establish best practices across diverse domains, such as health management, manufacturing, defense, and aerospace industries.

The paper investigates the current state and future prospects of model-based systems engineering (MBSE), emphasizing its application, influencing factors, and survey-based insights across industry, academia, and government. Findings indicate organizational hurdles to MBSE adoption, notably a lack of clear structures and managerial understanding. Survey respondents reported significant improvements in systems engineering tasks (50-75%). MBSE's historical evolution from the customization of the Unified Modeling Language (UML) for systems engineering since 2001 is noted. MBSE addresses the challenges posed by complex systems development with shortened timelines and reused components. It enhances quality, efficiency, and communication but faces challenges like cultural barriers, skills availability, and management support. The paper advocates for widely adopted methodologies and standards, such as SysML, to enhance systems engineering evaluation and verification [13].

The paper [14] and [15] addresses the adoption of model-based systems engineering (MBSE) in aerospace, automotive, vehicle modular kits, and defense organizations to reduce complexity. It proposes a methodological framework to analyze the costs and benefits of MBSE implementation, emphasizing the challenges organizations face, such as technical feasibility, financial viability and framework for reusing product models, unifying partial models, and supporting future product development. MBSE in modular kit development should align with Product Generation Engineering (PGE), depicting kits, modules, and configurable products. Model reusability is vital, requiring interdisciplinary collaboration. The challenge of tracking and communicating the aggregate risk of large, complex projects. It proposes a methodology that utilizes the data linkage capabilities of the Model-Based Systems Engineering (MBSE) model to tie risks to project items, creating meaningful aggregation of risk that can better inform the project when making decisions about risk acceptance, prioritization of mitigation activities, or engineering trade studies [16].

NASA's Human Research Program's Exploration Medical Capability (ExMC) Element has embraced Systems Engineering principles and tools, specifically Model-Based System Engineering (MBSE) and the Systems Modeling Language (SysML), to establish an initial architecture and requirements for an advanced exploration medical system [17]. MBSE and SysML were instrumental in translating clinical medical requirements into a language comprehensible to the engineering community, facilitating the integration of medical system requirements into exploration mission designs. The MBSE methodology, along with SysML, enabled the team to perform functional decomposition analysis and develop an initial set of requirements, with the architecture captured using SysML's structural elements to complement the behavioral aspects. The MBSE approach has created a functional model that aids in communicating the architecture, infrastructure, and requirements of the medical system required for future exploration missions.

Model-Based System Engineering (MBSE) can be used to assess the potential of electric aircraft for different tasks. Further research is needed to develop MBSE methods for the analysis of low emission propulsion systems for aviation [18]. Disaster preparedness is crucial for resource-limited rural communities. Model-Based Systems Engineering (MBSE) methods, including computer simulations, offer valuable insights for disaster planning and resource utilization. The study conducted in paper [19], discusses the application of MBSE and computer simulations in disaster preparedness for rural healthcare systems. It highlights the scarcity of research in this area and emphasizes the potential of MBSE as a novel approach to optimize disaster planning resources for rural communities.

The management of complexity has emerged as a critical factor influencing project success. The proposed approach combines qualitative and quantitative methods to assess system complexity within MBSE. It introduces the GOPPRR method [20], a novel formula for calculating structural complexity, and a toolchain based on the OSLC standard for visualizing and analyzing model complexity. A case study demonstrates the approach's effectiveness in supporting product trade-offs through complexity management. The approach aids in formalizing complex systems and offers quantitative insights for decision-making in system solutions. It contributes to complexity management in product development.

The two emerging concepts in systems engineering, model-based systems engineering (MBSE) and mission engineering (ME), integration enhance mission systems architecture development and mission analysis. It proposes an approach that identifies system requirements, aligns them with operational and synthesis models, and conducts trade space analysis to refine requirements.

This ensures consistency between operational and synthesis models for more robust early-stage system development [21]. Digital twins [22], introduced in 2002, are increasingly relevant in model-based systems engineering (MBSE). Unlike virtual prototypes, they are dynamic digital replicas of physical systems, continuously updated with real-time data. This paper advocates integrating digital twin technology into MBSE, highlighting benefits such as improved system simulation and IoT integration.

Table 1: Key Literature Considerations

SR	**Research Paper**	**Year**	**Author (Journal)**	**Contributions**	**Issues**
1	Towards a Domain-Specific Approach Enabling Tool-Supported Model-Based Systems Engineering of Complex Industrial Internet-of-Things Applications	2021	Christoph Binder et al. (Systems)	• MBSE implementation on dynamically changing systems in IIOT • Reference Architecture Model Industry 4.0	• Domain specific approach • New methodology • Focus on system level integrated development. • Do not specify the detailed development in SDLC
2	The Integration of Reliability, Availability, and Maintainability into Model-Based Systems Engineering	2022	Kyle Diatte et al. (Systems)	• Integration of RAM analysis in early stages of MBSE • RAM integration in resulted Cost saving and improved system performance	• New Methodology • Ignored aspects such as, Requirements, integration, Design integration, Interface integration
3	Economic Analysis of Model-Based Systems Engineering	2019	Azad M. Madni et al. (Systems)	• Investment and expected gains of MBSE depend largely on the system's intrinsic	• Economic analysis for specific domain systems. • Potential gaps in the framework, like areas

				characteristics. • Highlight the benefits of MBSE such as enabling informed decision-making and resource allocation.	not covered or assumptions made.
4	Model-Based Systems Engineering for Ship Design: An Economic Analysis	2019	J. Yang et al. (Elsevier)	• Quantitative framework: Cost benefit analysis • Provides insights into the cost-benefit equation of MBSE for this specific domain	• Limited empirical data • The research might not fully address the long-term economic effects of MBSE, such as its impact on maintenance and lifecycle costs.
5	A Dual-Radio Hybrid Mesh Topology for Multi-Hop Industrial IoT Networks in Harsh Environment	2022	Hasari Celebi (Journal of Electrical and Computer Engineering)	• Hybrid mesh radio topology network. • Multi hop communication in radio. • Supporting Wireless mesh radio network efficiency over wired mesh radio network in harsh environment.	• The performance evaluation is based on specific scenarios and assumptions. • Practical implementation limitations in industrial settings, such as cost, scalability, and security considerations • Limited Range of radio network.

2.1.5 Benefits of Using MBSE

The traditional document-centered approach to systems engineering (SE) processes often results in extended engineering phases and higher project costs. Model-Based Systems Engineering (MBSE) has been proposed as a solution, emphasizing system models over textual documents for information exchange among engineers. The paper [23] discusses the potential of using patterns to formalize reuse within MBSE, with an emphasis on achieving maturity. Expected benefits include improved quality of reused modeling artifacts. However, challenges like cultural resistance and implementation support hinder MBSE adoption. The paper suggests developing MBSE software tools for pattern management, creating shared knowledge repositories. Challenges related to intellectual property and model interoperability persist. Future efforts should focus on maturity scales to assess MBSE advancement through patterns.

This paper outlines [24] the benefits of MBSE itemized herein:

- Improved System Understanding
- Early Error Detection and Mitigation
- Enhanced Collaboration and Communication
- Efficient Requirements Management
- Streamlined System Integration
- Configuration Control and Versioning
- Reduced Development Time and Costs.

2.1.6 Arcadia Framework

The Arcadia (Architecture Analysis and Design Integrated Approach) Methodology [25], is a systems engineering methodology developed by Thales. It provides a structured and model-driven approach to the design and analysis of complex systems. The Arcadia Methodology is based on the principles of Model-Based Systems Engineering (MBSE) and focuses on the early and continuous consideration of system architecture throughout the development process. It aims to ensure that the architecture of a system is aligned with its requirements and that the system design is traceable back to those requirements. Arcadia is a structured five-phase methodology that begins with operational analysis, followed by system analysis, logical architecture, physical architecture, and concluding with the end product breakdown structure, as illustrated in the Figure 3.

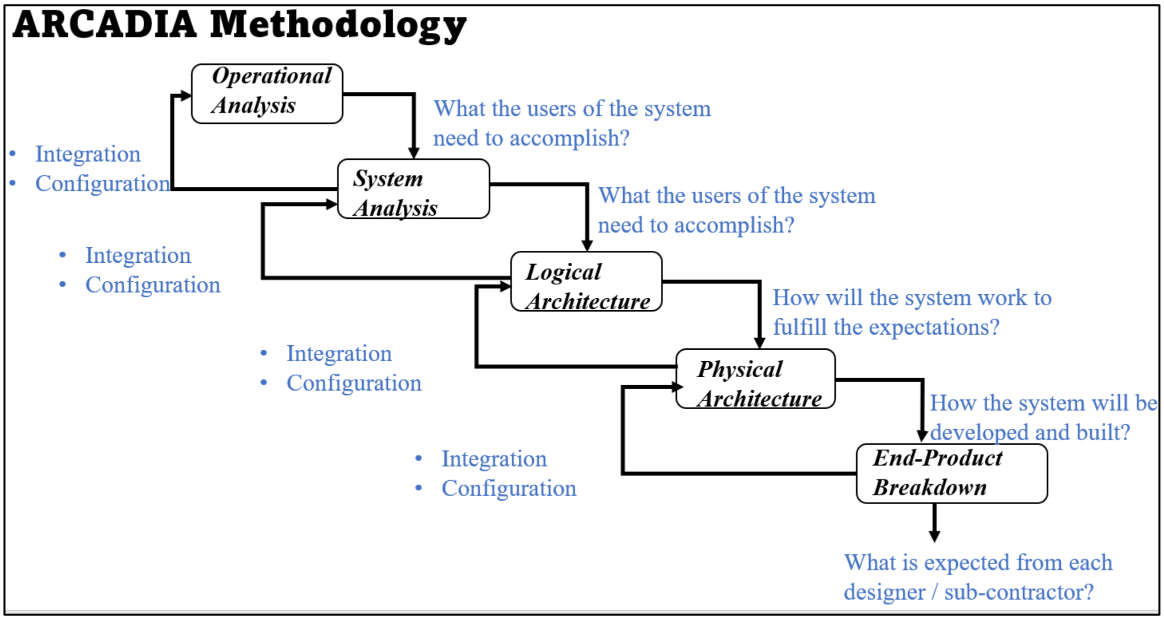


Figure 3: ARCDIA Methodology

2.2 Software Defined Radio (SDR)

2.2.1 Introduction

Software Defined Radio (SDR) is a technology paradigm that involves the utilization of software-based control mechanisms to govern the operational behavior of radio frequency hardware. Coined by Joseph Mitola [26], during the 1980s, an era characterized by his affiliation with E-Systems (presently Raytheon), the concept of "software radio" has garnered significant attention and engagement within the realm of research, notably in the domain of military investigations. The scholarly document underscores the fact that software radio has engendered a convergence of diverse studies and cooperative initiatives, leading to a proliferation of noteworthy technological progressions. Software Defined Radio (SDR) are characterized as a programmable wireless communication system wherein the execution of vital digital signal processing operations, encompassing tasks such as modulation and demodulation, encoding and decoding, error management, interleaving and deinterleaving, as well as scrambling and descrambling, is orchestrated through software frameworks, as opposed to the conventional reliance on hardware configurations pervasive in established radio communication infrastructures. This paradigmatic shift empowers the seamless integration of disparate waveform standards onto a singular platform, facilitating effortless transitions between them without any requisite modifications to underlying hardware constituents. In practice, a diverse array of hardware platforms serves as the bedrock for the software facet of SDR, prominently encompassing General Purpose Processors (GPPs), Digital Signal Processors (DSPs), Field Programmable Gate Arrays (FPGAs), and Application Specific Integrated Circuits (ASICs) [27].

The paper [28] discusses the tactical radio systems, including mesh topology and software-defined radios, play a critical role in defense and disaster response operations, offering secure and reliable communication channels. Mesh topology, in particular, offers several advantages. The study discusses aligning hardware technology performance with software radio requirements and the transition from hardware-based to software-intensive radio systems. Software radio architecture migrates radio functionalities into software, covering signal generation, modulation, coding, and more, while hardware remains essential for RF conversion and power management.

The transition of radio systems from analog to digital and the implementation of more functions in software, leading towards the software radio, explores the capabilities, pitfalls, and projection of software radio architectures. The software radio can reduce hardware size, weight, and power through fewer radio units, manage the increased complexity of emerging radio network standards within affordable acquisition and maintenance budgets, and provide powerful tools for object management software technology. Software radios are now in the segment of the market dominated by the military, big business, and governments. Over time, the software radio will continue to move down and to the right as the size, power, and cost of general-purpose DSP chips, AID and DIA converters, and related interconnect and memory allow.

The practical implications of SDR discussed in [29] are as follows:

- The software radio architecture provides new levels of service quality and channel access flexibility.
- Software radios can reduce hardware size, weight, and power through fewer radio units, which is beneficial in applications where access to multiple bands with multiple radio access modes is a necessity.
- Software radios can manage the increased complexity of emerging radio network standards within affordable acquisition and maintenance budgets.

- Software radios can provide powerful tools for object management software technology.
- The software radio is a powerful architecture framework that helps deliver advanced radio services in a way that leverages the economics of contemporary microelectronics and software technologies.

2.2.2 SDR Development Platforms

Cutting-edge Software-defined Radio (SDR) platforms in the realm of wireless communication protocols are expounded upon, encompassing the blueprinting of SDR architecture, its foundational building blocks, and prominent trajectories in design progression, alongside key developmental tools as discussed in [30]. The overarching framework of Software-defined Radio (SDR) technology, rooted in software-defined wireless protocols, endows a numerous features and functionalities. Diverse design strategies and architectures that underlie the maturation of SDR encompass, among other approaches, GPP, GPU, DSP, FPGA, and co-design-driven methodologies.

2.2.2.1 GPP Based Platforms.

The Kansas University Agile Radio (KUAR) [31], is a potent GPP based software-defined radio development platform that empowers advanced research in wireless radio networks, dynamic spectrum access, and cognitive radios. The KUAR platform offers a versatile RF frontend, wide transmission bandwidths, and ample center frequency ranges, along with a highly configurable design that provides developers with a comprehensive suite of hardware and software tools tailored to their specific expertise. Sora [32], a software radio platform that combines the benefits of both hardware and general-purpose processor (GPP) SDR platforms. Sora uses hardware and software techniques to address the challenges of using PC architectures for high-speed SDR. The Universal Software Radio Peripheral (USRP) [33], SDR Device represents a tunable transceiver meticulously engineered for the purpose of conceiving, prototyping, and implementing radio communication systems. This device serves as an optimal candidate for the prototyping of wireless communication solutions, the formulation of applications pertaining to Electronic Warfare (EW) and Signals Intelligence (SIGINT), as well as the deployment of diverse wireless systems.

2.2.2.2 DSP Based Platforms

Atomix [34], stands as a modular software framework tailored for constructing applications within wireless infrastructure. The framework highlights the viability of constructing modular digital signal processing (DSP) software through the utilization of fixed-timing computations termed "atoms." The author advocates for the exploration and refinement of Atomix with the aim of shaping it into a malleable data plane for a software-defined radio access network (SDRAN).

2.2.2.3 FPGA Based Platforms

Airblue is an FPGA-driven software radio platform [35] enabling the pliable transmission of data across layers and on-the-fly configuration adjustments. It rigorously upholds these core design tenets and facilitates streamlined modifications. The platform successfully fulfills the performance criteria stipulated by contemporary wireless protocols, affording the capability for on-packet feedback transmission, integration of novel decoding algorithms, and dynamic pipeline reconfigurations, all accomplished without compromising adherence to 802.11 timing stipulations.

2.2.3 Advantages of SDR

In [36] several advantages of software radio were identified; key advantages are listed herein:

- The ability to use a single hardware platform for multiple wireless standards, reducing costs and increasing flexibility.
- The ability to upgrade and modify radio systems through software updates, rather than requiring hardware changes.
- The ability to implement complex signal processing algorithms in software, allowing for more advanced features and improved performance.
- The ability to use SDR’s in a wide range of applications, from military communications, disaster response, security operations to commercial wireless networks.

2.2.4 Applications

Software-defined radio (SDR) assumes a crucial role within the domain of emergency communication for organizations dedicated to public protection and disaster relief (PPDR). In this context, the proposal introduces a demonstrative system hinged on the incorporation of SDR technology and software communication architecture (SCA), designed to bolster PPDR operations, with particular emphasis on furnishing satellite communication capabilities [37]. SDR platforms find application in the realization of radio frequency (RF)-based drone detection and defense systems, executed through the utilization of SDR platforms [38]. The utilization of SDR systems extends beyond their primary communication objectives to encompass threat detection and adversarial data gathering as discussed in [39]. This underscores the latent capacity of SDR systems to function as force multipliers, amplifying both intelligence capabilities and combat effectiveness, thereby bolstering warfighter survivability. SDR systems offer novel prospects for distributed signal acquisition and analysis within military and defense contexts.

Software Defined Radio (SDR) finds numerous military applications discussed in [40], such as:

- Providing flexible, upgradeable, and longer lifetime radio equipment for military and civilian wireless communications infrastructure.
- Providing more flexible and possibly cheaper multistranded terminals for end users.
- Serving as a convenient base technology for the future context-sensitive, adaptive, and learning radio units referred to as cognitive radios.

2.3 Research Gap

The current landscape of system development, especially at the local level, inadequately utilizes the potential of Model-Based Systems Engineering (MBSE), despite its prominence in technological advancements. This discrepancy highlights a noteworthy gap where the untapped potential of MBSE persists. Within this context, this research endeavors to rectify this deficit by concentrating on the incorporation of MBSE principles into Software-Defined Radio (SDR) development. SDR holds a fundamental role in contemporary communication systems; however, the underexplored implementation of MBSE in SDR development, particularly within local industries, emphasizes the pressing need for research aimed at explaining the prospective advantages, challenges, and optimal methodologies associated with MBSE integration in SDR development. This research aims to bridge the current disparity, simultaneously contributing to the progress of MBSE and SDR technology, ultimately furnishing an enhanced and more effective framework for complex system development.

Chapter 3 Methodology

The chapter delves into the methodology employed for system development, commencing with an introduction to the Arcadia methodology and a system overview. It then extensively addresses the utilization of Model-Based Systems Engineering (MBSE) in the intricate process of software-defined radio development, meticulously dissecting each phase of the Arcadia methodology and charting the evolution of the system from its initial requirements to its architectural realization. The chapter culminates with the presentation of the ultimate architecture for the software-defined radio, thereby encapsulating the entire journey of its development under the framework of the Arcadia methodology.

3.1 System Overview

This section discusses the system that aims to address the identified research gap within the existing literature, focusing on the implementation of model-based systems engineering to advance Software-Defined Radio (SDR). SDR is a crucial communication device with applications in both commercial and defense sectors, specifically tailored for short-range communication. Its usage is widespread among individuals and groups operating on the ground, facilitating effective communication within their operational domains as shown in Figure 4. SDR enables the group of individuals to communicate in short-range utilizing the decentralized network. SDR is characterized by its lightweight and portable design, intended for wear by individuals, typically attached to their body armor or vests, enhancing secure and efficient field communication, and contributing to situational awareness, coordination, and command and control. Key features of the Personnel Role Radio encompass its short-range communication, secure encryption, reliability, and support for text and voice communication channels, enabling real-time information exchange. SDR seamlessly integrates with various communication and command systems, whether used in commercial or defense operations, ensuring enhanced interoperability and coordination across different operational echelons.

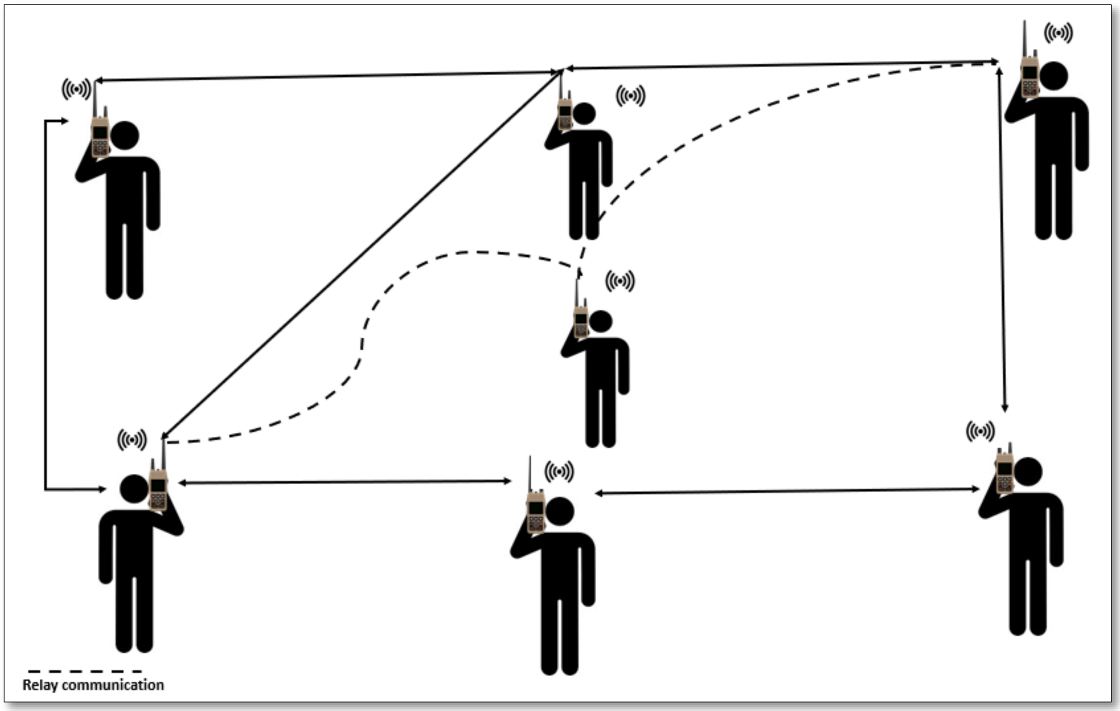


Figure 4: Group Communication Using SDR.

3.2 Operational Analysis

3.2.1 Stakeholder Needs and Environment

Operational analysis is a pivotal phase in the development of a software-defined radio (SDR) system. It involves the comprehensive identification of stakeholder needs and the examination of the operational context within which the system will operate. This encompasses the consolidation and capturing of operational requirements from stakeholders, understanding users' objectives and goals, and defining the entities, actors, roles, activities, and concepts associated with the system's operation. The primary goal of this phase is to gain a thorough understanding of the operational requirements for the SDR system and to address specific scenarios and communication challenges faced by stakeholders, such as disaster response organizations, fire fighters, defense personnel, commanders, and operational staff. Operational analysis seeks to define the objectives that users of the system must achieve, including ensuring reliable and secure communication, enhancing situational awareness, facilitating personnel coordination, and improving resilience in dynamic operational environments. The activates to be performed at the operational analysis level are shown in Figure 5. Identifying the pertinent entities, actors, roles, activities, and concepts is also crucial for comprehending the operational procedures and requirements. This foundational information serves as the basis for subsequent phases, including system analysis, logical architecture design, and physical architecture implementation, ensuring that the mesh based SDR system is tailored to meet the specific needs and objectives of its users.

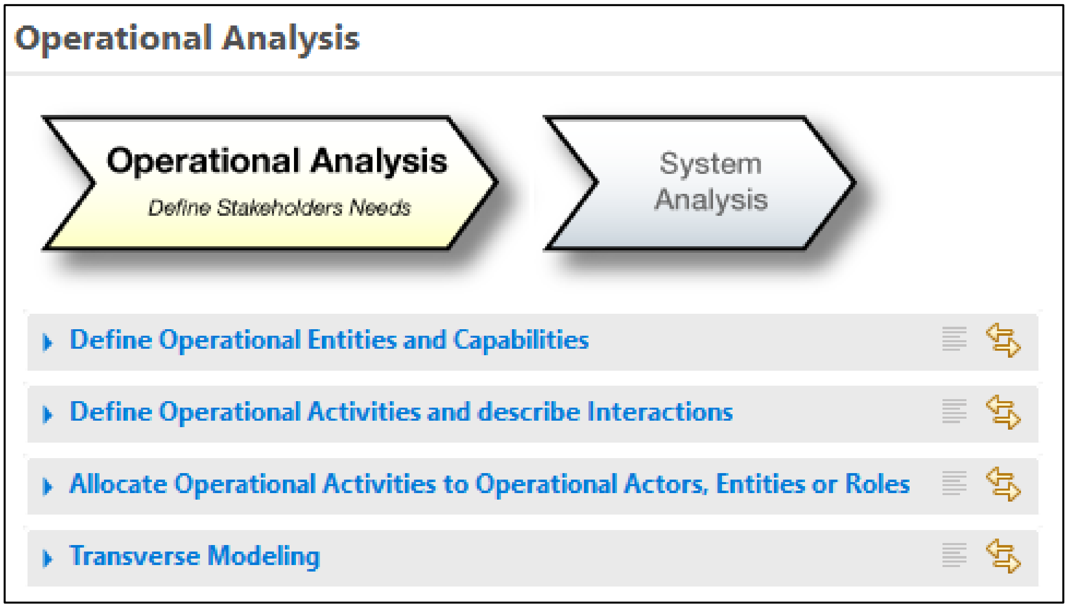


Figure 5: Operational Analysis Methodological Activities

This level of study will not delve into the details of the Transverse Modeling group, which encompasses the Mode and State diagrams, as well as the Class diagrams for data modeling. This methodological activity is present at each level of the Arcadia framework, excluding EPBS, and is therefore referred to as Transverse Modeling. System Analysis will leverage the phase to explore this aspect further. Arcadia is a method; the modeling process is completely flexible. The methodological activities presented in the Activity Explorer are nearly all optional and can be carried out in any order, and this is also the case for the diagrams.

This case study will consider the following diagrams shown in Figure 6, form the operational analysis phase for the software defined radio.

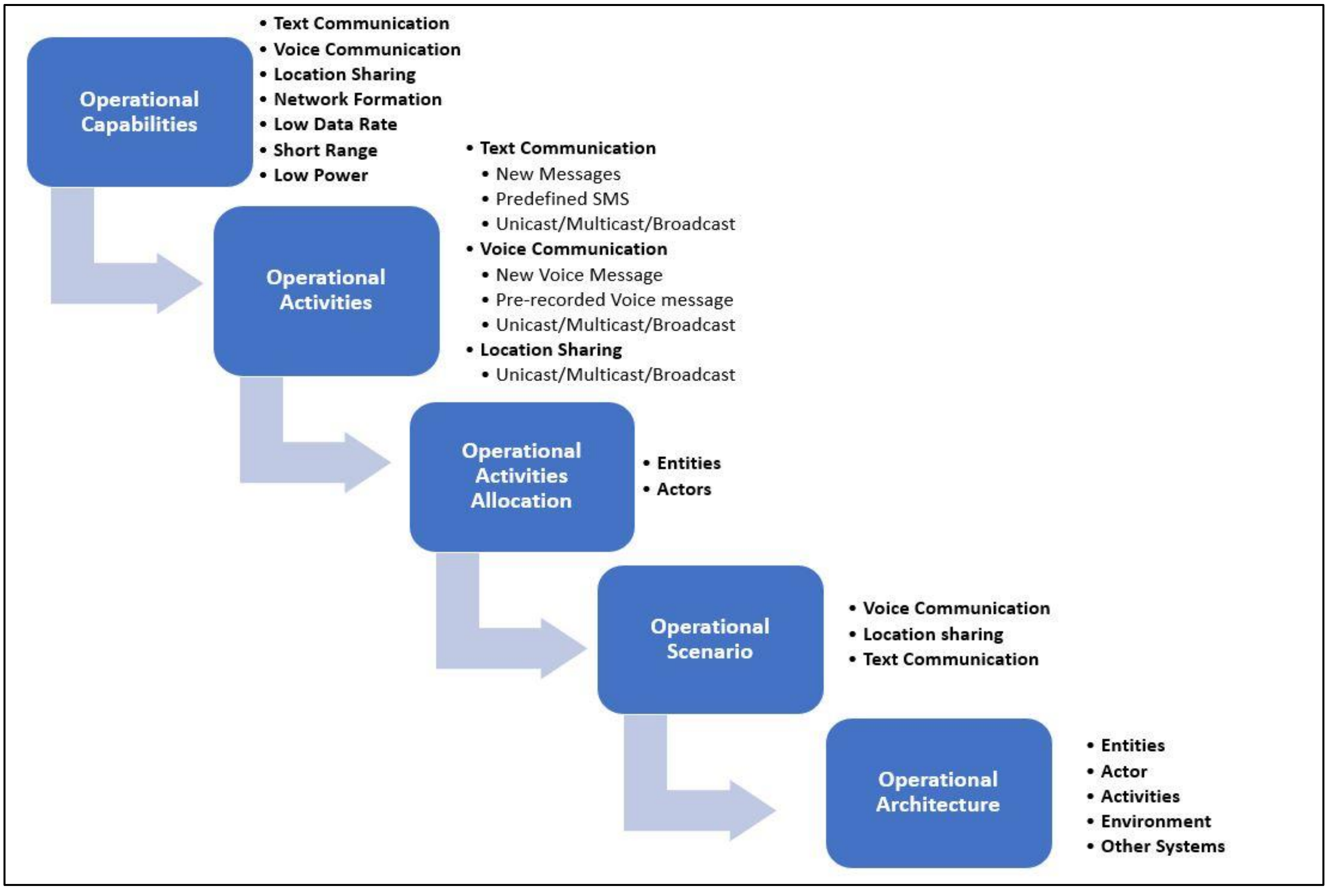


Figure 6: Operational Analysis

The operational analysis phase, a fundamental aspect of the Arcadia methodology, involves constructing diagrams to define and elaborate upon Operational Capabilities, such as voice and text communication, and location sharing, tailored for short-range and low-power operations. These capabilities are intricately linked through a network of Operational Activities, each contributing to the fulfillment of overarching objectives. The phase culminates with the allocation of these Activities to Operational Entities, creating an Operational Architecture Blank Diagram. This systematic examination serves to comprehensively understand operational needs and goals, ensuring alignment with desired outcomes and setting a robust foundation for subsequent system development and design stages, ultimately leading to an efficient and optimal SDR system solution.

3.2.2 Operational Capabilities

In this section, operational capabilities are meticulously defined using a structured approach that begins with the creation of an operational capabilities blank diagram, visually representing key capabilities and their interconnections. This diagram is instrumental in identifying crucial stakeholders, including individuals, groups, and customers, and their connections with these capabilities. This structured analysis provides a comprehensive understanding of the system's operational capabilities and their relationships, enabling informed decision-making and establishing a strong foundation for subsequent system development phases. It plays a pivotal role in meeting the system's operational requirements and objectives.

3.2.3 Operational Activities

This section discusses the definition of operational activities, building upon the foundation of operational capabilities. The operational activities diagram is developed to identify and define the essential activities needed to realize the SDR system's operational capabilities, including voice communication, text communication, and location sharing. Each capability is further decomposed into specific operational activities, offering a detailed breakdown of tasks and interactions required for efficient fulfillment as shown in Figure 6. This systematic approach is crucial for meeting the system's operational capabilities effectively.

3.2.4 Operational Architecture

In the operational analysis process, the next pivotal step involves allocating operational activities to the previously defined actors and entities, establishing a structured operational architecture for the SDR system. This allocation is facilitated using the operational architecture blank diagram, created through the "create new operational architecture diagram" command in Capella. Capella automatically integrates the defined actors and entities, aiding in the selection and placement of operational actors and entities within the architecture diagram. Subsequently, operational activities are assigned to their respective operational actors and entities, outlining their roles and responsibilities in executing these activities. This allocation process ensures a clear depiction of interactions and relationships between activities and operational entities, contributing to a well-structured operational framework as shown in Figure 7.

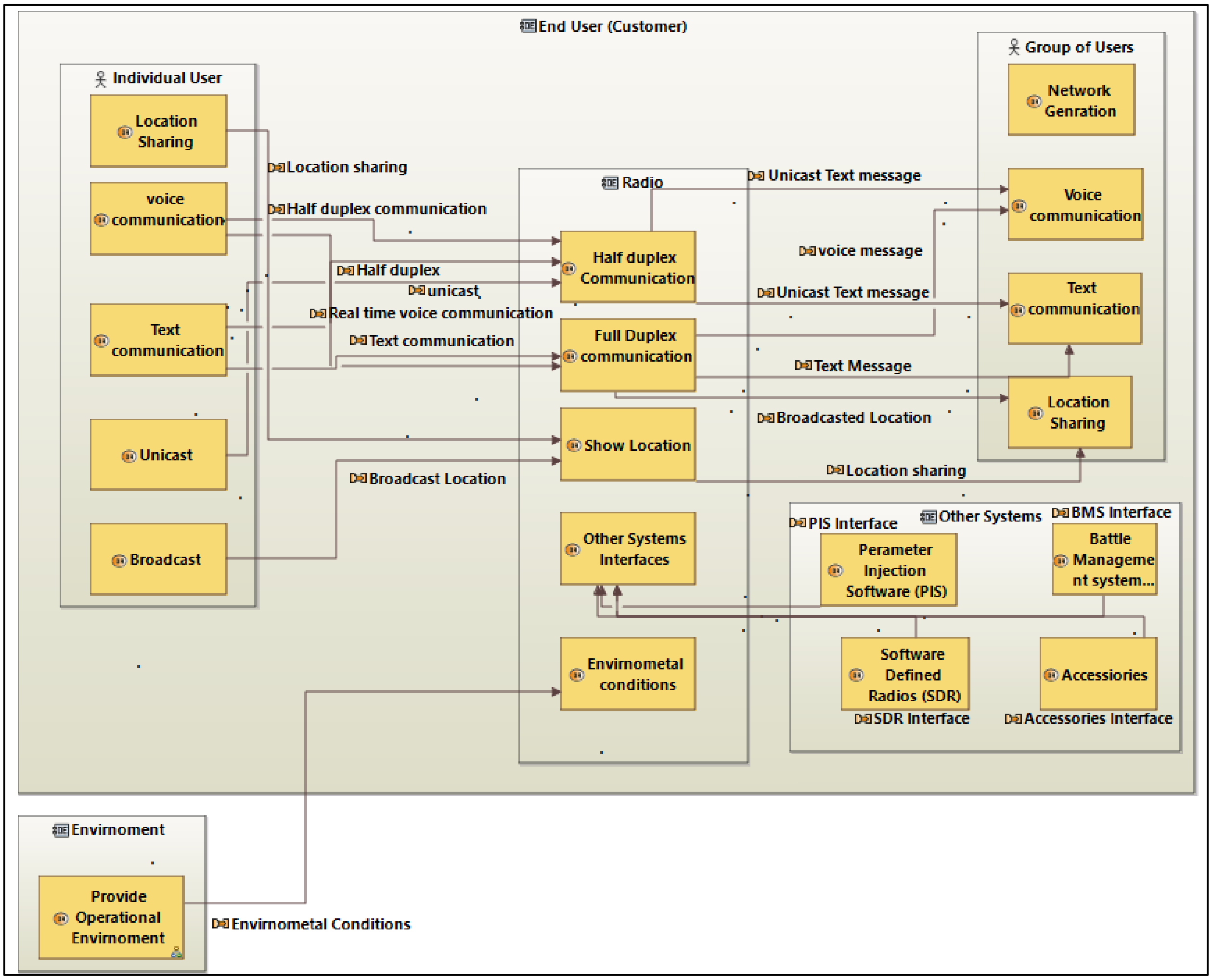


Figure 7: Operational Architecture

The operational architecture blank diagram for the SDR system reveals three primary operational entities: End User, Radio/Other Systems, and Individual and group of individuals. The system is designed to enable secure communication and location sharing between soldiers and squads using the radio. The architecture diagram also identifies other systems, such as software-defined radios, the Battle Management System (BMS), Parameter Injection Software (PIS), and accessories, which play a crucial role in supporting the overall system functionality. Additionally, environmental conditions are considered and incorporated into the architecture diagram, as they can have an impact on the performance and communication capabilities of the radio system.

In summary, the operational analysis phase provides a comprehensive understanding of the operational requirements, capabilities, and activities of the SDR system. It enables the development team to design and develop a robust IoT-based mesh radio network that meets the specific operational needs and objectives. The operational capabilities, activities, and their allocation to the relevant actors and entities form a foundation for subsequent stages, including system analysis, logical and physical architecture design, and implementation.

The operational analysis also offers the other diagrams notably operational scenario, operational scenarios allow to create the scenarios to understand the use of the system in different situation. The scenario defines the activities take place during the scenario in a sequence with the time constraints. The scenarios defined for the personnel role radio includes text communication between the solder and the squad of a soldier, where the soldier communicate with the squad via text by broadcasting the message to the squad of soldiers. Similarly, the other scenarios include voice communication and location sharing.

3.3 System Analysis

This section focuses on the system analysis phase, which constitutes the second stage of the Arcadia methodology. During this phase, the aim is to establish an appropriate level of abstraction from the system in order to elicit the genuine needs of the stakeholders. System analysis, which is an integral part of arcadia methodology, encompasses activities that answer fundamental questions about the system, such as its intended functionality and the identification of external interfaces.

Methodological activities at system level are shown in the Figure 8.

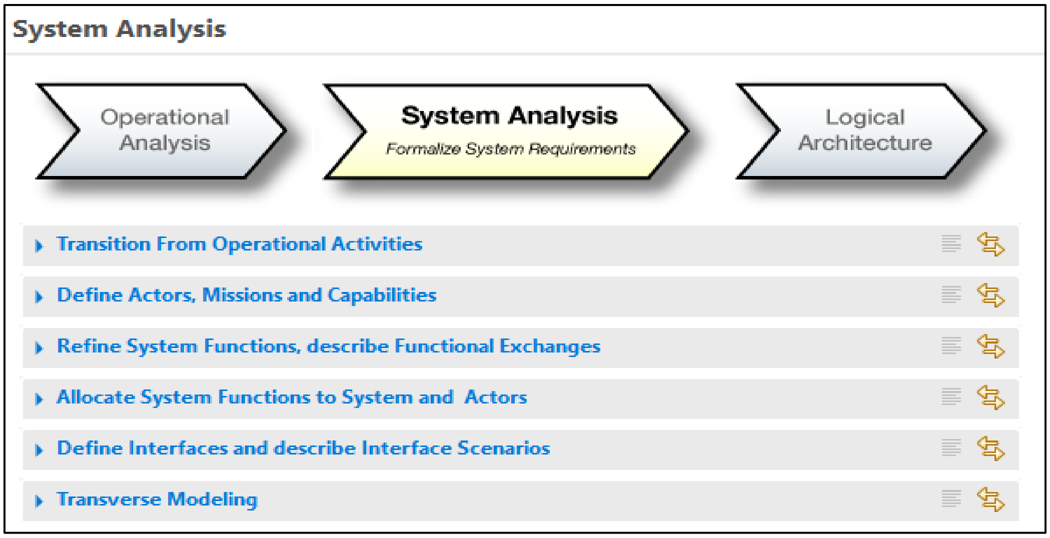


Figure 8: Methodological Activities of the "System Analysis" Level

The initial step in system analysis involves delineating the system's boundaries and consolidating requirements. This encompasses defining the scope, context, and gathering both functional and non-functional requirements from stakeholders. The activities to be performed at the system analysis level referring to the case study are shown in Figure 9. To model the system's functional aspects and dynamic behavior, data flow diagrams are utilized. A pivotal aspect of this phase is the creation of the system architecture blank diagram, which serves as a functional blueprint, capturing requirements, identifying system actors, and defining interfaces. System scenarios and state and mode diagrams are also developed to depict the system's behavior in different operational situations. This systematic approach in the Arcadia framework plays a critical role in comprehending system requirements, establishing its functional architecture, interface definitions, and behavior under various conditions. It provides a foundational framework for subsequent phases like logical and physical architecture.

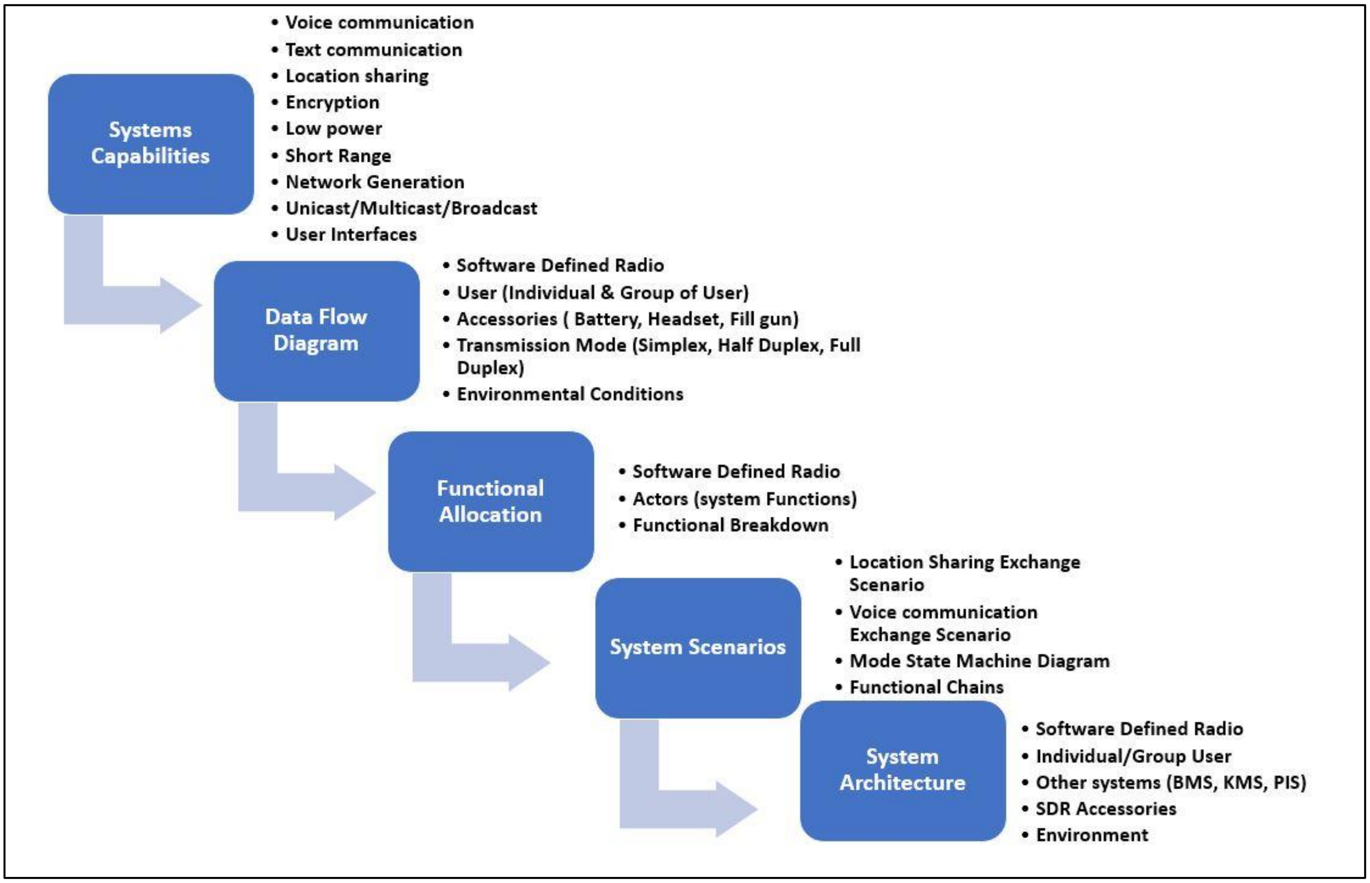


Figure 9: System Analysis

3.3.1 Transition form system Analysis

This section discusses the transition from operational analysis to system analysis, facilitated by Capella's features. An essential tool in this transition is the "Perform automated transition of operational activities " command, which automatically converts operational activities into system-level functions. These functions encapsulate the operational activities identified in operational analysis, particularly those related to the system's communication capabilities. Notably, this transition process also includes the integration of ports and links associated with these operational capabilities. This seamless transition aids in evolving from a high-level operational perspective to a more detailed system analysis, ensuring the continuity and integrity of the system development process.

3.3.2 System Capabilities

Operational analysis involved creating the domain model, independently of the system to be realized. Operational analysis provides a level of abstraction to focus the real needs of the stakeholders. The system capabilities at the system level were initiated from transitioning the system capabilities from operational analysis level. To recover the system capabilities, form the operational analysis activity explore “Contextually create new system capability or mission from the operational capability” transition command was executed, for each operational capability one system capability were created.

Once the system capabilities and functions are defined including voice communication, text communication, location sharing, encryption, low power, network generation, user interface and unicast, multicast and broadcast capabilities. the operational capabilities were linked with the respective system actors like voice communication text communication were linked to the individual and the group of users. This is accomplished by creating a contextual diagram which is a “system data flow blank” for complex/broadly defined capabilities, which facilitates a more detailed breakdown of the function. The transition from operational analysis to system analysis involves the modeling of system-level functions and the utilization of predefined operational capabilities.

3.3.3 Function Analysis

This section delves into system-level functional analysis, a crucial foundation for system requirements. During the transition phase, each capability was broken down into functions through the use of contextual blank diagrams. These functions were then incorporated into the diagram, creating a comprehensive breakdown of system functionality. This encompassed not only system requirements but also the prerequisites for meeting actor requirements, accounting for factors like environmental conditions and communication with individuals or groups. System data flow blank diagrams were generated, as shown in Figure 11, for each capability, capturing system requirements. The subsequent step involved the creation of a global data flow blank diagram that included all functions. In cases where requirements were undefined, new requirements were introduced by creating new functions. The global diagram provides an overview of requirements and functions of the system at this stage in system analysis. This systematic approach, utilizing contextual and global data flow diagrams, ensured a comprehensive system-level functional analysis.

3.3.4 Functional Chains

This section delves into the system's behavioral aspects using the functional chains capability. Functional chains, derived from the functional data flow, highlight interdependencies among system functions, offering insights into expected system behavior within specific contexts. This feature is particularly valuable for verification and validation tests, allowing the expression of non-functional constraints like latency, criticality, confidentiality, and redundancy along functional paths. Functional chains are crucial for comprehending and managing complex system behaviors, supporting in-depth analysis and evaluation. Several functional chains were created for the system, covering aspects like voice communication, location sharing, location display at battle management systems, and network injection parameters software as shown in Figure 10. Within each functional chain, source and destination functions were specified, and intermediate functions were represented by colored lines. Different colors and color boxes were used to distinguish multiple functional chains. Capella's "functional chain description" diagram allowed for easy modifications and tool access, enhancing the creation and management of functional chains.

3.3.5 System Architecture

This section discussed the development of the system functional architecture blank diagram in the context of function analysis and allocation of functions to the system and actors. An empty diagram was created, with a box representing the system, which was renamed as SDR (Personal role radio) to depict the system. The transition allowed the reuse of system-predefined actors, including individual and group users, other systems, the environment, accessories, and other SDRs in the system analysis phase. The system architecture blank diagram emphasized the identification of external interfaces of the system. System architecture enabled consideration of the system interfaces and planning for the development of the system interfaces at the early stage of the system development, as shown in Figure 10. Following the establishment of system actors, the function allocation process was initiated.

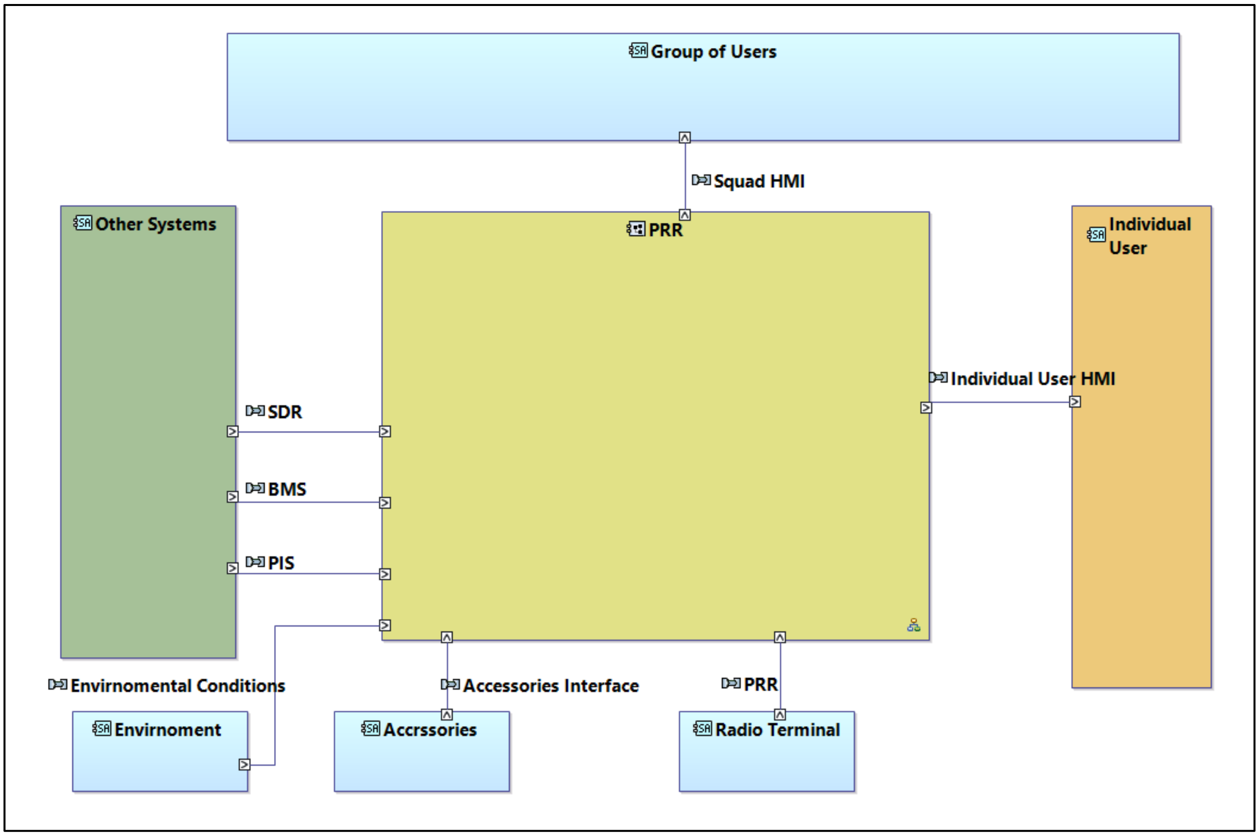


Figure 10: System Architecture Blank Diagram

System architecture blank diagram was created focusing on the allocation of functions to both the system and its external actors. Functions, such as voice, text communication, and location sharing, were allocated to the system, while additional functions, like microphone, speaker, and secure communication via advanced encryption, were defined at the system level. The system's capability for external interfaces and power management was also established. Following this function allocation to the system, functions were allocated to the system actors. All functions captured the system's requirements, ensuring specific requirements were allocated to both the system and its actors. Once functions were assigned to the system and actors, the next step involved defining functional and component exchanges, which illustrated data flow between the system and its external actors, labeled according to their nature. Dotted lines indicated interactions between the system and actor functions utilizing component exchanges for data flow, as shown in Figure 11.

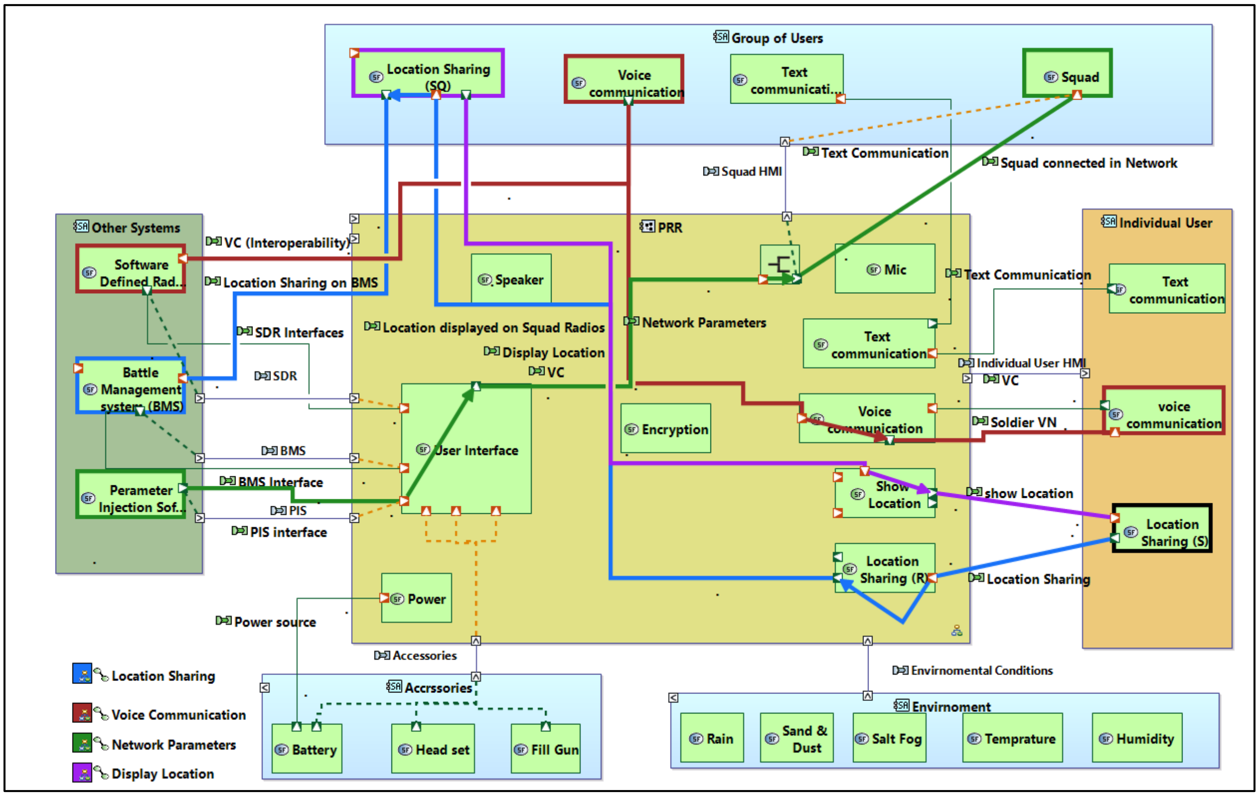


Figure 11: Function Allocation to System Architecture

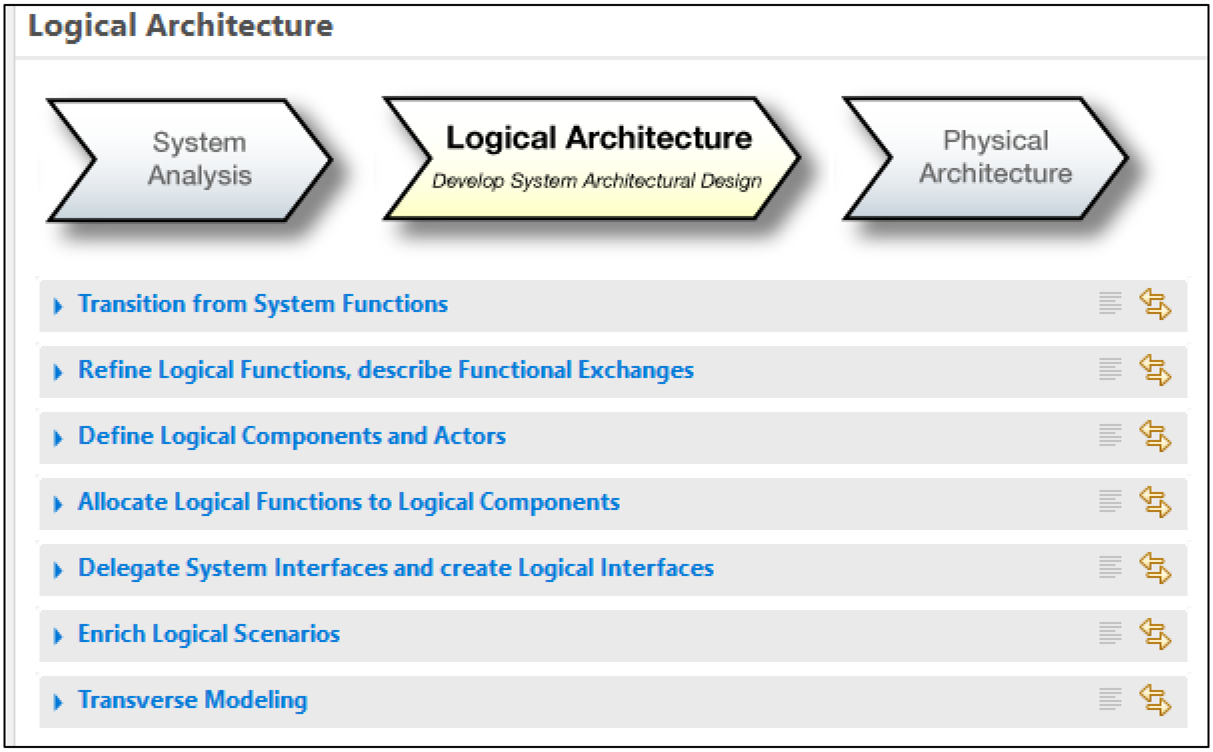
To simplify the diagram, multiple views were created to observe various aspects of the system-level architecture. These views included system architecture blanks and interactions with system actors, enhancing comprehension of external interfaces. After completing the interfaces and functional allocation between the system and actors, a functional breakdown diagram was generated. This diagram encompassed all allocated functions to both the system and actors, distinguished by green and blue function colors. Blue functions indicated allocation to actors,while green functions represented allocation to the system. These views and diagrams provided a structured understanding of the system's architecture and the roles of functions within it.

3.3.6 System Scenarios

This section focuses on system scenarios that illustrate the system's behavior during various operations. These scenarios are explored after the development of the system architecture, offering insights into the dynamic aspects of the system. A functional scenario was created to depict the location-sharing behavior of the system, showing the chronological data flow during this process, and defining constraints. Scenarios also allowed for specifying delays between activities. Exchange scenarios, involving entities or actors, were generated for location sharing and voice communication. These scenarios presented functions on vertical lines, providing a unique perspective for analyzing system behavior and concluding the system analysis phase. They visually conveyed how the system functions in different situations, aiding in a comprehensive understanding of its behavior.

3.4 Logical Architecture

This section will explore the progression of logical architecture, which constitutes the third phase in the Arcadia methodology. Logical architecture signifies the point at which the system's is opened as a box, departing from its treatment as a black box as in the system analysis phase. One of the primary goals in the development of logical architecture is to refrain from making technical decisions and instead focus on defining and constructing the system's structural elements. By adhering to this approach, the logical architecture ensures that a wide array of methodological activities can be employed, the methodological activities at logical architecture level are shown in the Figure 12.



Logical Architecture

- Transition from System Functions
- Refine Logical Functions, describe Functional Exchanges
- Define Logical Components and Actors
- Allocate Logical Functions to Logical Components
- Delegate System Interfaces and create Logical Interfaces
- Enrich Logical Scenarios
- Transverse Modeling

Figure 12: Methodological activities at Logical Architecture Level

In the logical architecture phase, optional activities were provided, and a deliberate yet justified decision was made regarding the creation of diagrams. The first step involved automatically recovering Functions and Actors from the System level to initiate the Logical Architecture level. Within the System, enduring Logical Components were established, which remained independent of any technological choices. Subsequently, Functions at the Logical level were allocated to these components, resulting in the breakdown or completion of Functions derived from the System level. Furthermore, attention was given to Ports and Component Exchanges derived from the System level, which were assigned to the Logical Components, and internal Component Exchanges were introduced as needed. The methodology for developing the logical architecture of the system is illustrated in the Figure 13.

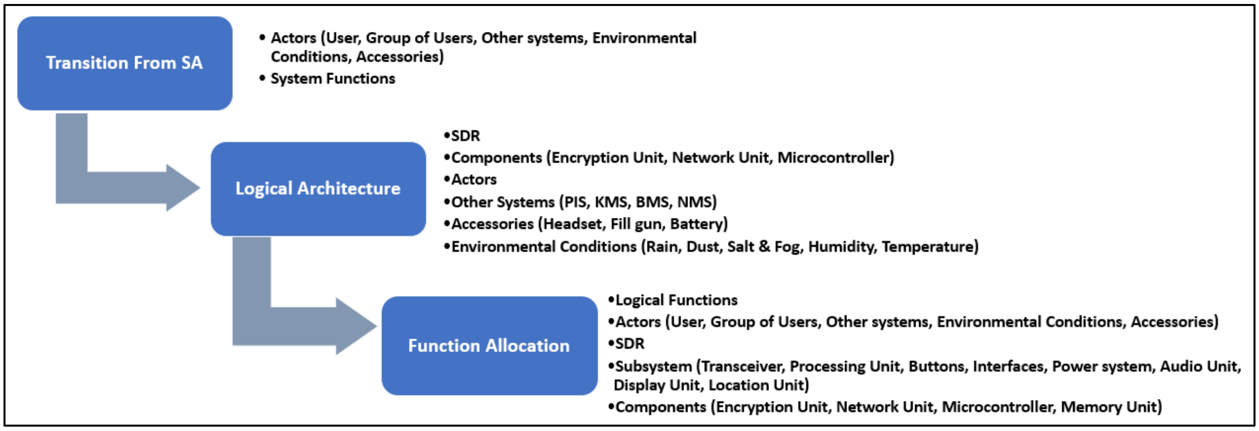


Figure 13: Logical Architecture

3.4.1 Transition from SA to LA

This section discusses the transition from system analysis to the logical architecture phase, with the main aim of identifying the structural elements of the system without delving into technical specifics. The logical architecture phase initiates this transition by converting system functions and actors into logical functions and actors, consolidating system actors into single actors in the logical framework. This phase allowed for the creation of new logical elements and functions as required. The logical architecture encompasses various entities, including users, groups of users, radio terminals, end users, other systems, and system accessories. The transition process included information about component ports, actor interactions, functional exchanges, and component exchanges. The focus is on structurally defining the system's components and their interactions in the logical architecture.

3.4.2 Logical Architecture

This section highlights the development of the logical architecture blank diagram and the transition from system analysis to the logical architecture phase. It commenced with the creation of the logical architecture diagram, which encapsulates a box representing the logical system. This facilitated the identification of component ports and exchanges during the transition process. Subsystems and components including encryption unit, network unit, microcontroller, transceiver, interfaces, power system, location and audio units were defined at the logical level in alignment with system requirements. Logical components were introduced for each logical subsystem, and logical actors were integrated into the diagram to encompass all defined actors at the system level within the logical architecture as shown in Figure 14. Component exchanges were automatically generated through this transition, portraying the system's interactions with actors. Logical components are pivotal, allowing for the decomposition of subsystems and the allocation of functions. This phase delineated the logical architecture of the system and its actors, providing a foundation for subsequent function allocation.

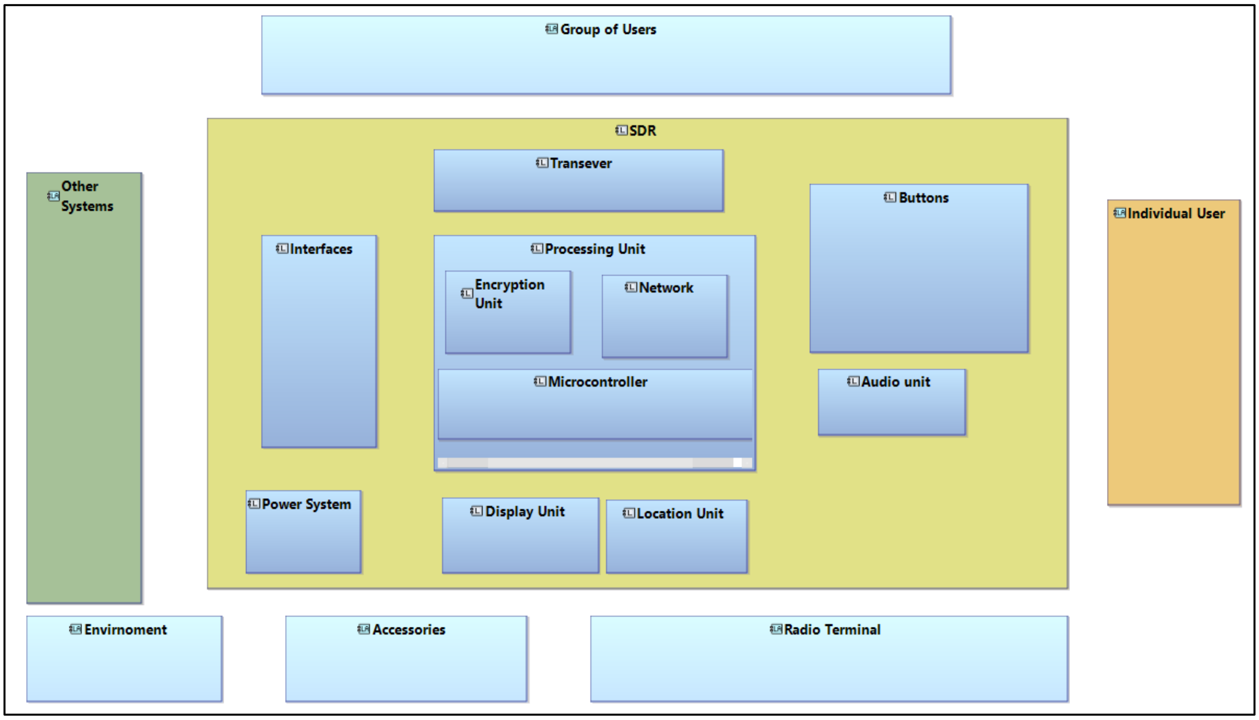


Figure 14: Logical Architecture Blank Diagram

3.4.3 Function Allocation to LA

This section is dedicated to the allocation of functions to the logical subsystems and actors within the system as shown in Figure 15. Once the system's components and actor functions are clearly defined, the subsequent step involves allocating functions to these specific subsystems and actors. This allocation process is conducted methodically, with each function of the system being allocated one by one to the respective subsystems. As functions are allocated, they begin to manifest as functional exchanges, which subsequently transform into component exchanges. These exchanges depict the flow of functions between source and destination functions allocated to a given subsystem. For the sake of visual clarity and a better understanding of the diagram, functions or components can be repositioned within the diagram, thus improving the overall comprehensibility of the allocation of functions.

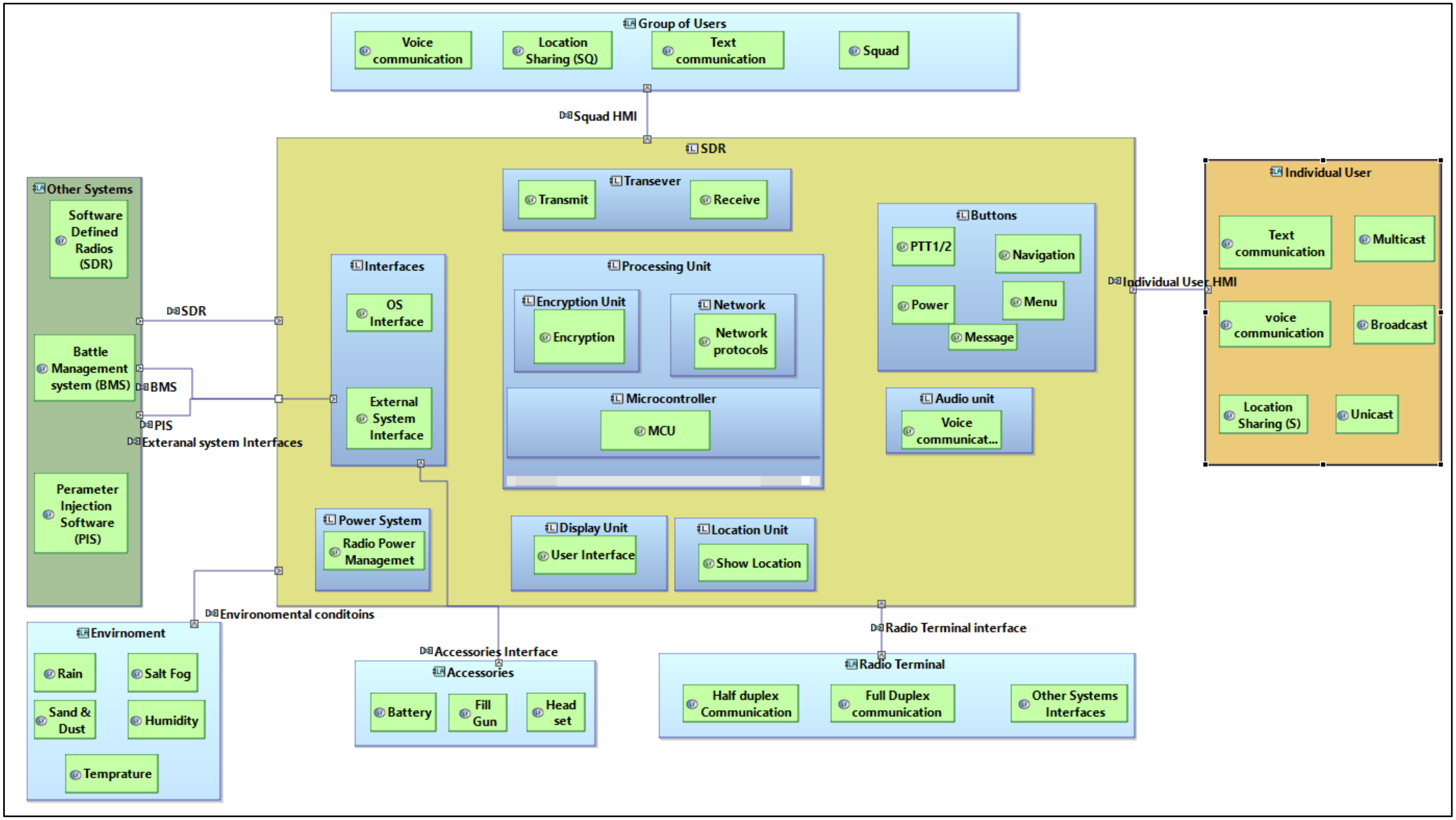


Figure 15: Function Allocation to Logical Architecture

3.5 Physical Architecture

This section focuses on the development of physical architecture phase of arcadia methodology. The physical architecture phase of the Arcadia methodology empowers the modeler to make informed technical decisions when developing a system architecture. This phase delves into the intricate details of the system by opening it up to the parts level, thereby defining all the structural elements. Within the physical architecture phase, multiple diagrams are provided, offering a range of options. These diagrams, however, are mostly optional, allowing the modeler to exercise discretion and choose the most appropriate ones for their specific architectural development. The accompanying Figure 16 showcases the methodological activities available during the physical architecture phase.

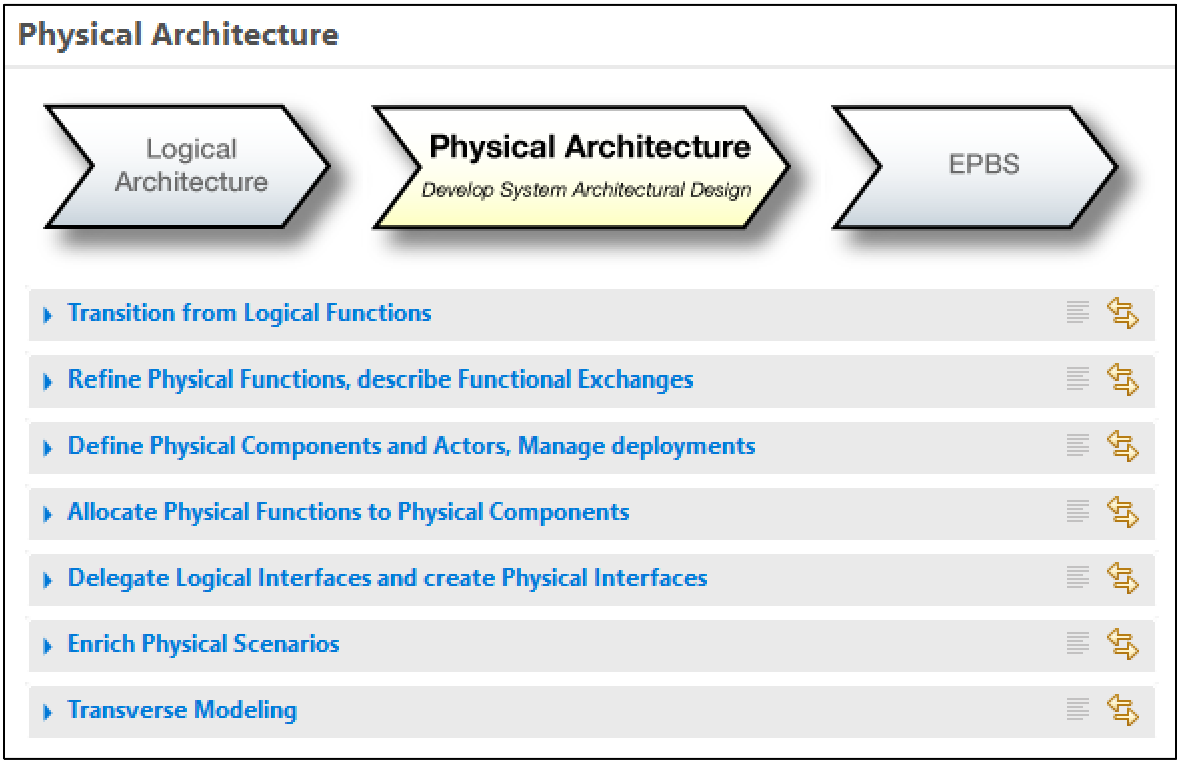


Figure 16: Methodological Activities From the "Physical Architecture" Level

The development of the physical architecture commenced with the selection of diagrams based on the modeler's discretion. In this phase, the Functions and Actors from the Logical level were retrieved and incorporated into the process. Concrete Physical components were introduced within the system, guided by deliberate technological choices. Following the establishment of the physical components, the allocation of Physical level Functions to these components was implemented within the physical architecture blank diagram. The workflow for the development of the physical architecture is illustrated in the accompanying Figure 17.

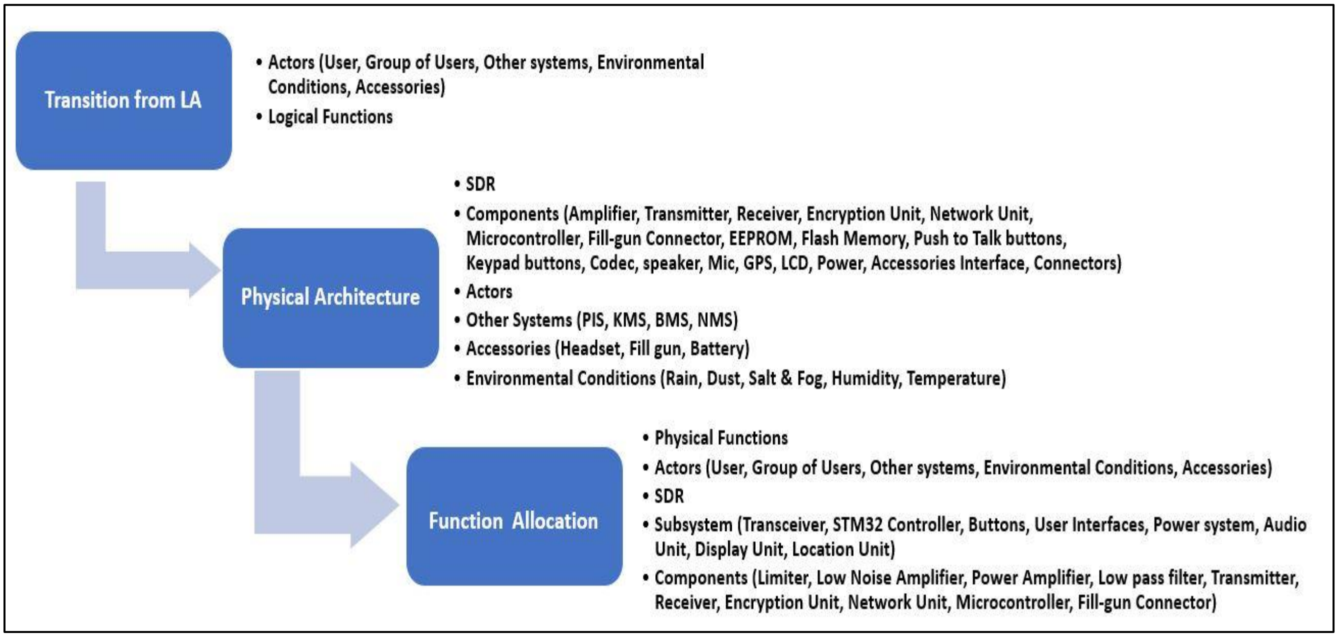


Figure 17: Physical Architecture

3.5.1 Transition from LA to PA

The transition from logical architecture to physical architecture involves unveiling the inner workings of the system and identifying its tangible structural elements. The objective at the physical architecture level is to define the actual, concrete components that form the system. The transition was initiated by defining the physical functions, which were created based on the logical functions. Each logical function at the logical level was transformed into a corresponding physical function, establishing a direct mapping between the two levels keeping the traceability and integrity intact. Subsequent step involved generating the external actors, on transition physical actor was generated for each corresponding logical actor. During the transition, the flexibility to include or exclude the transfer of component ports and exchanges enable the modeler to choose whether to transfer these elements or not, completing the transition.

3.5.2 Physical Architecture

This section presents the process of developing the physical architecture blank diagram, which holds significant importance within the physical architecture phase and aligns with the overarching goal of the Arcadia methodology. The physical architecture stands as the ultimate and conclusive portrayal of the system's architecture. The blank diagram for physical architecture was initiated, distinct from upper levels where the system is typically represented as a box; in this case, the initial physical architecture diagram was void. This specific diagram within the physical architecture palette is comprised of three types of concepts.

The palette of the physical architecture is complex of all the diagrams proposed by Capella. As a matter of fact, in this type of diagram, three types of concepts were utilized:

- Node Physical Components (yellow rectangles), which contained other Node Components
- Behavior Physical Components (blue rectangles), which were deployed on the Node Physical Components
- Physical Functions (green rectangles), which were allocated to the Behavior Physical Components

The initial step in developing the physical architecture was to establish the system as the central node physical component, encompassing a range of subsystems and functions. These subsystems and functions were further represented as additional node physical components and behavior physical components within the system. The node components were categorized based on their specific attributes, including hardware, software, and others; in this case, the system was categorized as a hardware component. The RF subsystem was defined, consisting of receiver and transmitter as sub-node physical components, along with other subsystems like STM controller, GPS, Audio subsystem, Buttons, interfaces, Memory, Programming HDR, PTT HDR, and KMS HDR. The following step involved defining the remaining node physical components and specifying their properties as hardware or software. These components, detailed in Figure 18, were integral to the overall physical architecture. Once all node physical components were defined, the subsequent task involved defining corresponding behavior components and assigning them to their respective node physical components. This allowed the allocation of functions to the relevant physical components, thereby defining the system's behavior in executing various operations, illustrated in Figure 18.

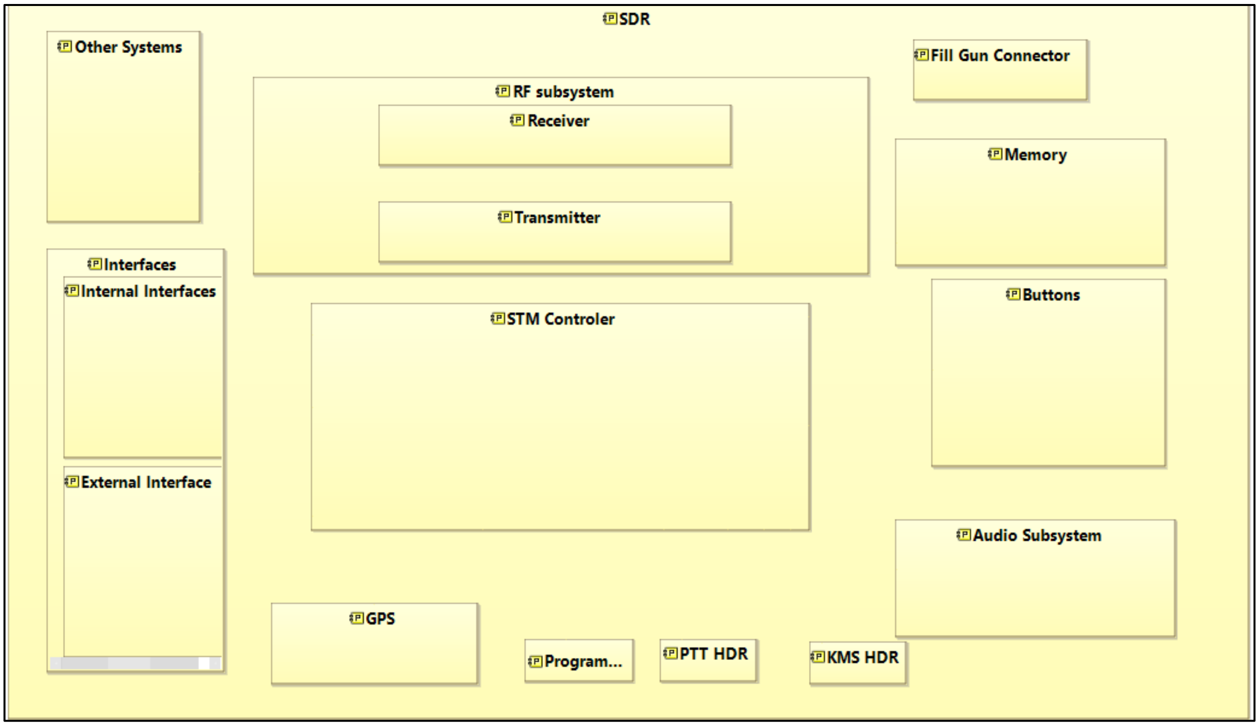


Figure 18: Physical Architecture Blank Diagram

To finalize the physical architecture blank diagram, the physical actors were automatically inserted. This ensured the inclusion of relevant actors within the diagram, aligning with the established physical architecture. The completed physical architecture blank diagram is illustrated in the accompanying Figure 19, demonstrating the successful integration of node physical components, behavior components, and actors in the physical architecture blank diagram.

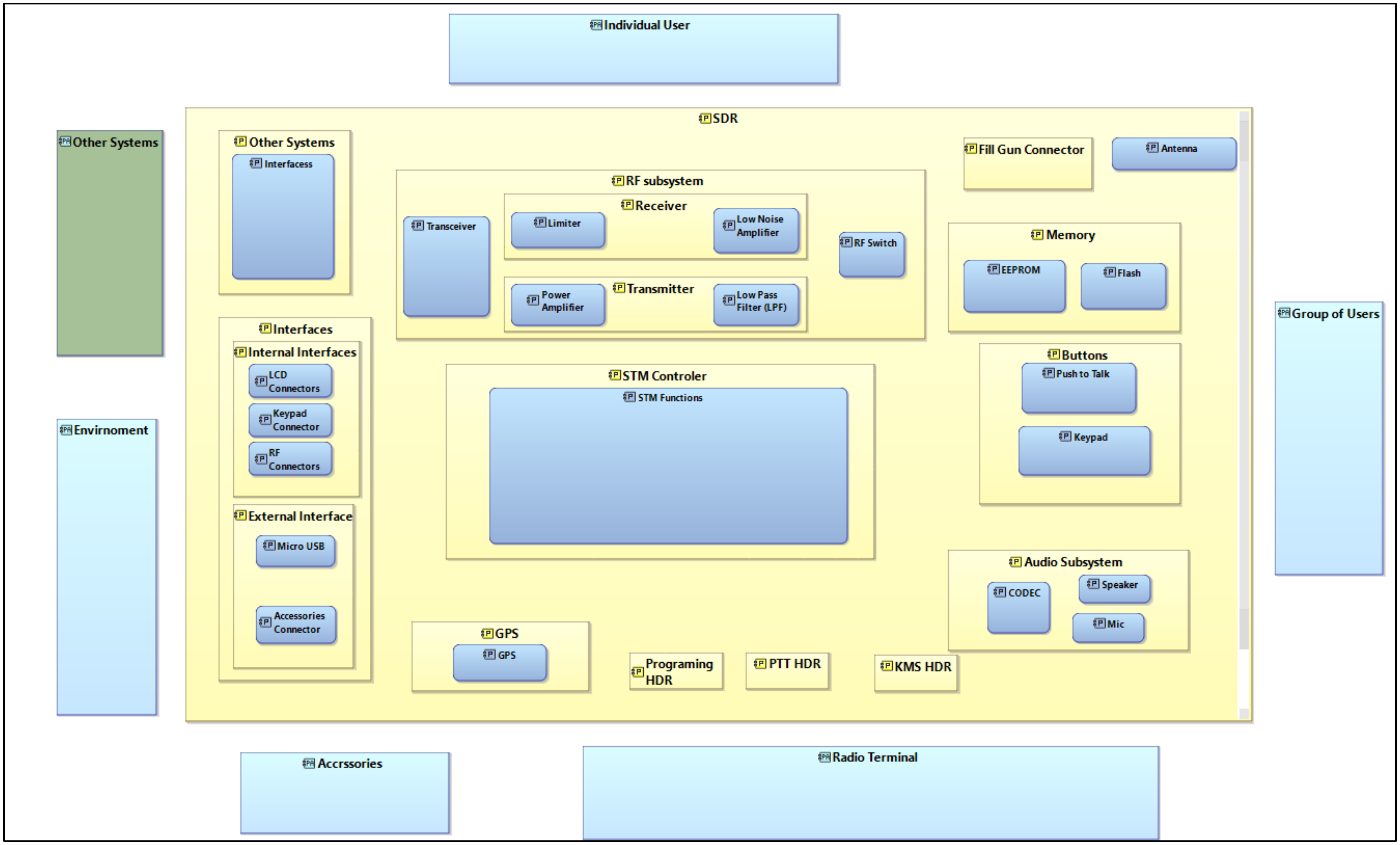


Figure 19: Physical Component Allocation to PA

3.5.3 Function Allocation

This section explains the allocation of functions within the physical architecture blank diagram. Upon the completion of the physical architecture blank diagram, the initial step involved allocating functions to the actors. To enhance the diagram's visual clarity, minor adjustments were made. Following this actor-function allocation, functions were assigned to the behavior components of the system. This allocation included functions like voice communication to codec, location sharing to GPS, transmit and receive to transceiver, encryption, and main control to STM controller, and similar assignments for other system behavior components. As functions were allocated, functional exchanges and ports emerged to represent connections between source and destination ports of the functions. To streamline the diagram for further work, the "Collapse component ports" filter was employed to hide component ports. Once all functions were allocated, the physical architecture blank diagram was considered complete, as depicted in Figure 20.

Function allocation to the behavior component is a powerful feature of Capella. It enables the derivation of an architecture based on the system's needs, which are represented by the functions defined at different levels of the methodology. During the allocation process, if a system function lacks an appropriate system architecture, it indicates that the existing architecture fails to meet the user's requirements. This feature compels the modeler to develop an architecture that aligns with the end user's needs. Capella prompts the addition of structural elements if the function allocation remains incomplete. This capability allows the modeler to identify the necessary and sufficient architecture components required to fulfill the user's needs and achieve a user-centric architecture.

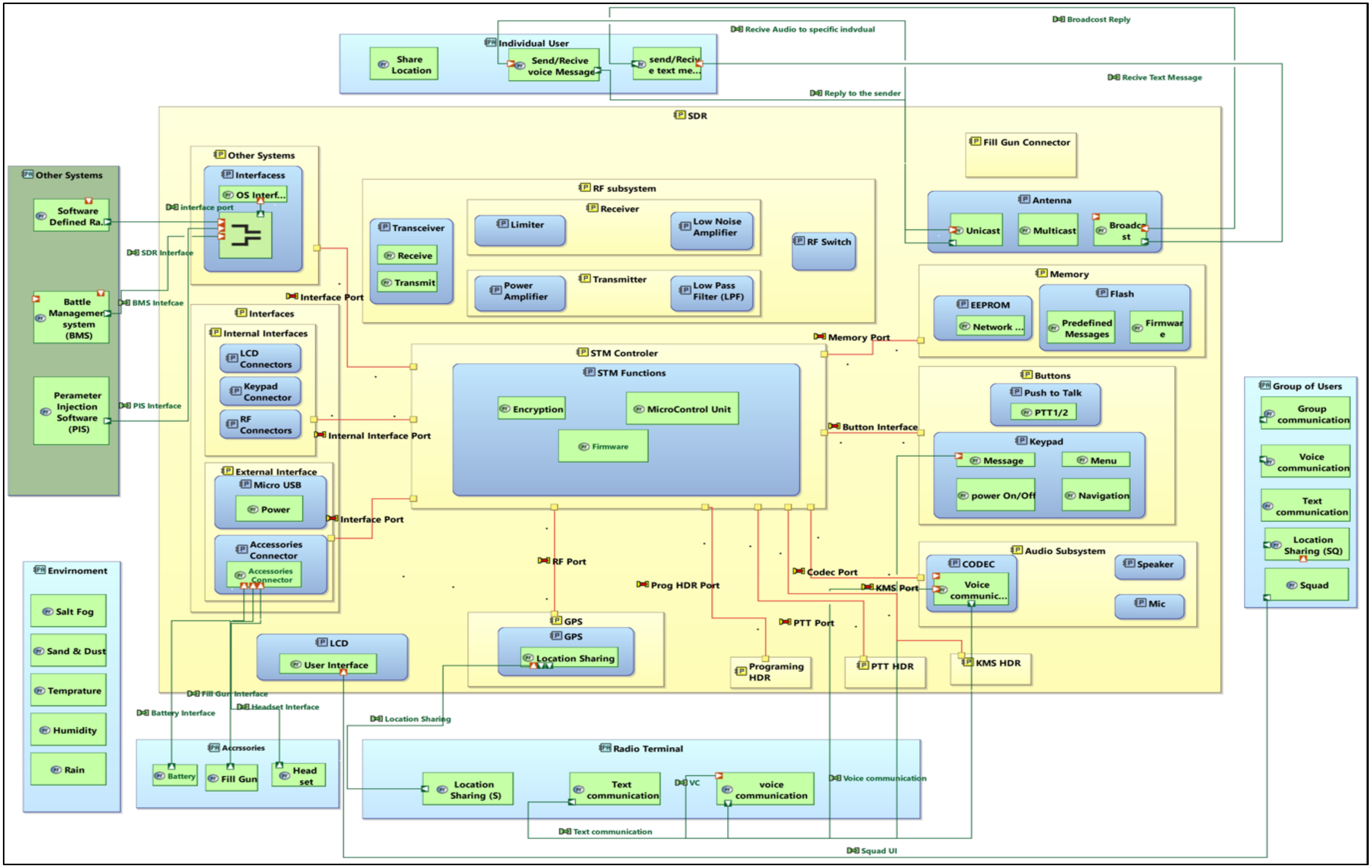


Figure 20: Function Allocation to PA

Chapter 4 Results and Discussion

This chapter delves into the results derived from the implementation of Model-Based Systems Engineering (MBSE) for development of MBSE. These results are systematically discussed within the framework of the predefined objectives. The initial section inspects the integrity of people, processes, and products for the development of the system. Subsequently, the following section comprehensively addresses the subject of verification and validation within the scope of Model-Based Systems Engineering. The concluding section explores the economic advantages and benefits associated with the adoption of Model-Based Systems Engineering.

4.1 Integrity in MBSE

This section provides an overview of the integrity aspect within the Arcadia methodology, a model-based systems engineering approach. Arcadia focuses on three fundamental aspects of integrity: process, product, and people integration as shown in Figure 21, all crucial for successful system development. Process integrity entails the coordinated and integrated development of the system, following a defined process for system architecture development. Product integration involves the incorporation of various elements, including system actors, system breakdowns, and system behavior. People integration encompasses stakeholder involvement throughout the Arcadia methodology, ensuring their active participation in architecture development, the integration involves the system integration. Detailed discussions on process, product, and people integration will be presented in the subsequent sections, exploring their significance in achieving overall system integrity.

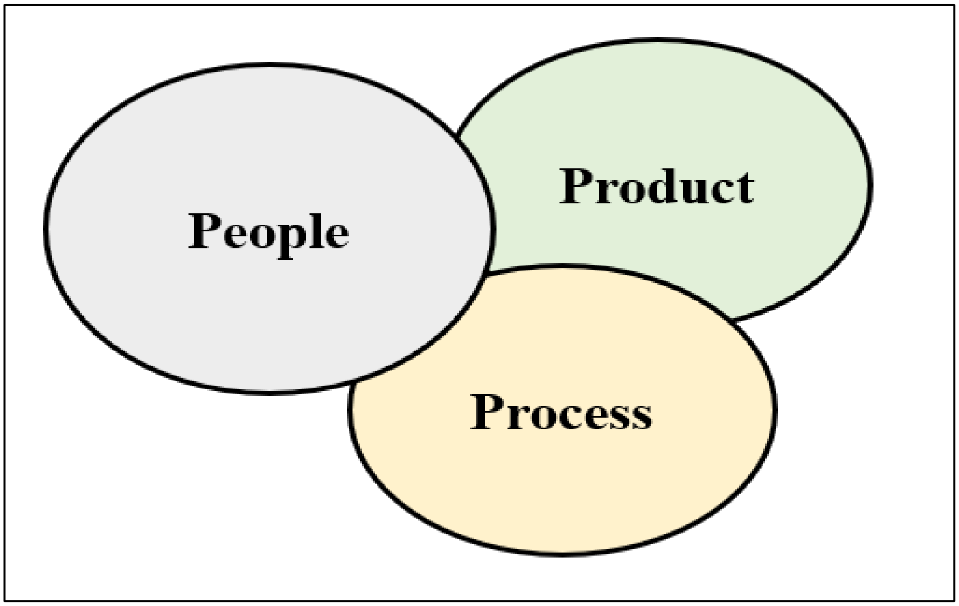


Figure 21: 3P Integration in MBSE

4.1.1 People

Model-Based Systems Engineering (MBSE), when coupled with the Arcadia methodology, places a strong emphasis on the integration of people as shown in Figure 22, highlighting their pivotal role in shaping the architecture of a given system. Within the well-structured framework of the Arcadia methodology, numerous facets of people integration are seamlessly included in the system development. To initiate this process, Arcadia encompasses the comprehensive integration of actors and stakeholders within the system, giving meticulous attention to their distinct perspectives and actively involving them in the development of the system's needs and requirements. The actors for the software defined radio as shown in Figure 22 affects the systems behavior and design as arcadia enforces the consideration of these actor which will interact with the system. This approach enriches the system development process, resulting in heightened comprehensiveness and efficacy. Arcadia effectively facilitates the inclusion of system actors in the intricate task of system architecture development. This collaborative engagement extends across the operational and systems analysis phases, where Arcadia actively consider people integration. Developers, end-users, and a diverse array of stakeholders actively participate in defining the system's precise needs and requirements, collectively wielding a substantial influence over the system's requisites, functionality, and overarching architectural design throughout the system development lifecycle. By emphasizing people integration, not only does

Arcadia enable the deliberate incorporation of human factors into the definition of the system's architecture, but it also significantly influences the system's design. This consideration fosters the development of an ergonomic system design, ensuring that the system is not only functionally efficient but also human-centered and user-friendly.

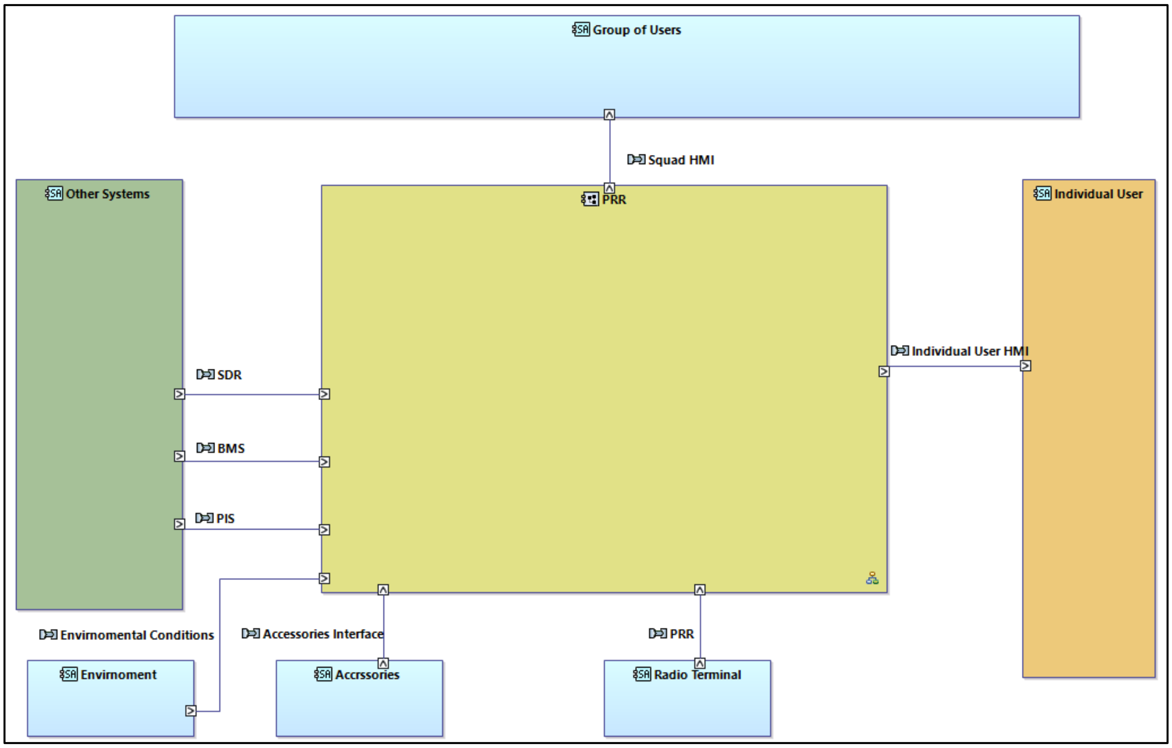


Figure 22: People's Integration

4.1.2 Process

Model-Based Systems Engineering (MBSE) serves as the linchpin for upholding the integrity of the system architecture development process. Within this context, MBSE is harnessed via the Arcadia methodology, a comprehensive framework that seamlessly integrates processes vital to architecture development. This method is distinguished by its five discernible phases: operational analysis, system analysis, logical architecture, physical architecture, and end product breakdown structure, illustrated in Figure 3. These sequential phases provide a structured process, guiding the trajectory of architecture development. The implementation of the Arcadia methodology is facilitated through the Capella tool, furnishing an integrated environment that fortifies the efficiency and uniformity of the development process. Essential to the concept of process integrity in system development is the preservation of correctness, reliability, and consistency throughout the entire developmental lifecycle. Adhering steadfastly to these foundational principles paves the way for the dependable and cohesive development of the system. Significantly, within the realm of MBSE applied to personal role radio development, the integration of the Arcadia methodology confers several noteworthy facets. The Arcadia methodology defines well-structured processes encompassing requirements engineering, system analysis, logical and physical architecture development, verification & validation, change management, stakeholder management, and communication. it champions a model-driven approach, enabling the comprehensive capture and delineation of the myriad system behaviors, facilitating a profound understanding of the system's intricate behaviors and interactions. The methodology's dedication to a need-driven approach ensures that activities and processes harmoniously align with the system's prerequisites, guaranteeing congruence between the architecture and its intended purpose. It also accentuates the establishment of a dependable and efficient architecture, leveraging established practices and techniques to enhance the efficiency of the architecture development process. The inclusion of a well-defined verification and validation process ensures impeccable alignment of the system architecture with specified requirements and intended performance criteria. Furthermore, the methodology underscores consistency by providing a structured approach to system development, ensuring that all processes and activities are executed uniformly, culminating in a coherent and dependable architecture. Notably, it offers flexibility, accommodating the dynamics of process improvement and system development, amenable to adjustments and enhancements rooted in evolving needs and lessons accrued throughout the developmental lifecycle.

4.1.3 Product

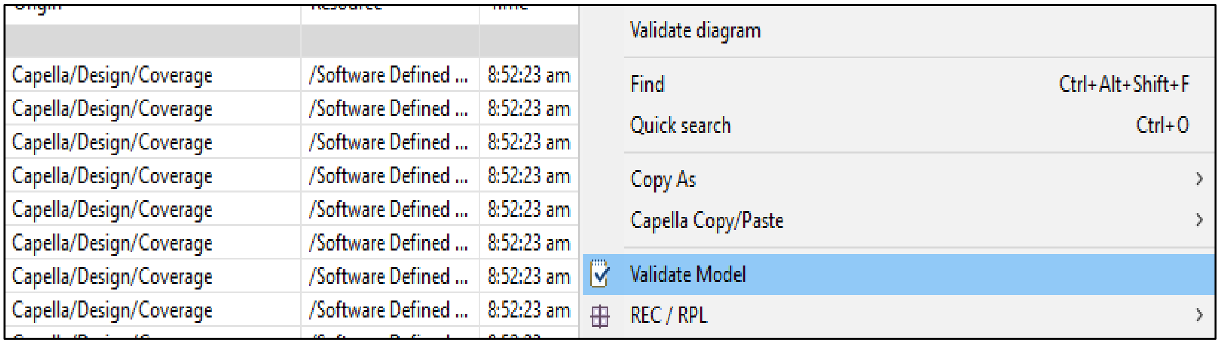
Product integration plays a pivotal role in the realm of model-based systems engineering (MBSE), serving as a critical element in preserving the overall integrity of the end product. Within the expansive framework of MBSE, the Arcadia methodology assumes a central and essential role in endorsing the development of systems architecture, finely tuned to meet the precise needs and requirements of stakeholders. Arcadia encompasses several facets that are directly relevant to the process of product integration. Initially, it empowers modelers by providing them with the capability to meticulously define system elements, encompassing everything from the overarching system to its subsystems, components, and individual parts.

This delineation is firmly grounded in the requirements and needs articulated by stakeholders, resulting in a comprehensive representation that enhances the structural clarity of the product. Furthermore, Arcadia adeptly facilitates the consideration of external systems and actors that interact with the focal system as shown in Figure 20, thereby fostering a seamless integration of the product within its operational context.

Arcadia adheres to a top-down approach to breakdown as discussed in detail in chapter 3, skillfully harmonizing stakeholder needs. This approach ensures the creation of an architecture that not only aligns with requirements but also encompasses all pertinent aspects of the system. This comprehensive perspective necessitates the inclusion of all indispensable parts, components, and subsystems, ultimately culminating in an integration that encompasses the system's vital components. Arcadia's feature of change integrity is particularly noteworthy as it enables architectural modifications at any stage while upholding coherence and consistency within the integrated product. Additionally, Arcadia underscores the enhanced integration of system behavior within the architecture, enabling a more efficient and effective system architecture. Consistency in change management remains a steadfast principle throughout the Arcadia methodology, with modifications seamlessly propagating across the model. This approach effectively mitigates any potential inconsistencies or conflicts during the integration process. Notably, the methodology is firmly rooted in a need-driven architectural approach, unwaveringly prioritizing the satisfaction of stakeholder needs throughout the entirety of the system development process.

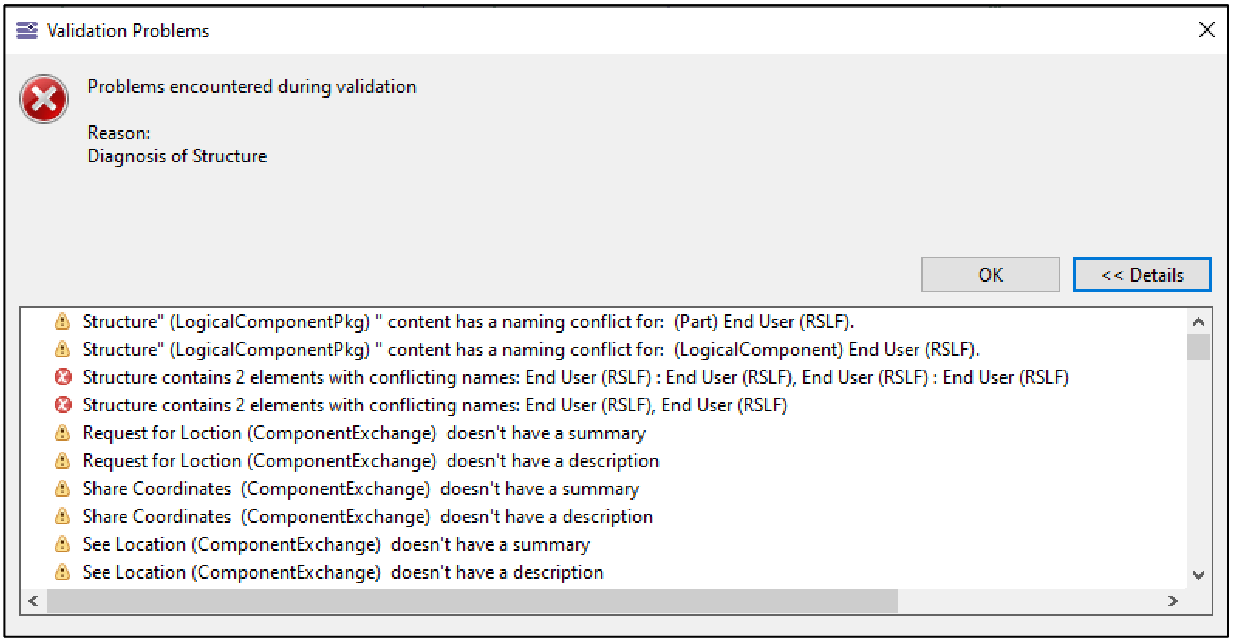
4.2 Verification and Validation

The Arcadia methodology offers a range of approaches for the verification and validation of system models, each contributing to the assurance of model correctness and integrity. One method involves utilizing Capella's "model verification" command as shown in Figure 23, which assesses model consistency and concurrency, ensuring compliance with predefined rules and standards. This command, as illustrated in Figure 24, displays the number of errors in the model, which diminish as errors are rectified, ultimately leading to a successful model verification, as shown in the Figure 25.



Capella/Design/Coverage	/Software Defined ...	8:52:23 am
Capella/Design/Coverage	/Software Defined ...	8:52:23 am
Capella/Design/Coverage	/Software Defined ...	8:52:23 am
Capella/Design/Coverage	/Software Defined ...	8:52:23 am
Capella/Design/Coverage	/Software Defined ...	8:52:23 am
Capella/Design/Coverage	/Software Defined ...	8:52:23 am
Capella/Design/Coverage	/Software Defined ...	8:52:23 am
Capella/Design/Coverage	/Software Defined ...	8:52:23 am

Figure 23: Model Verification & Validation



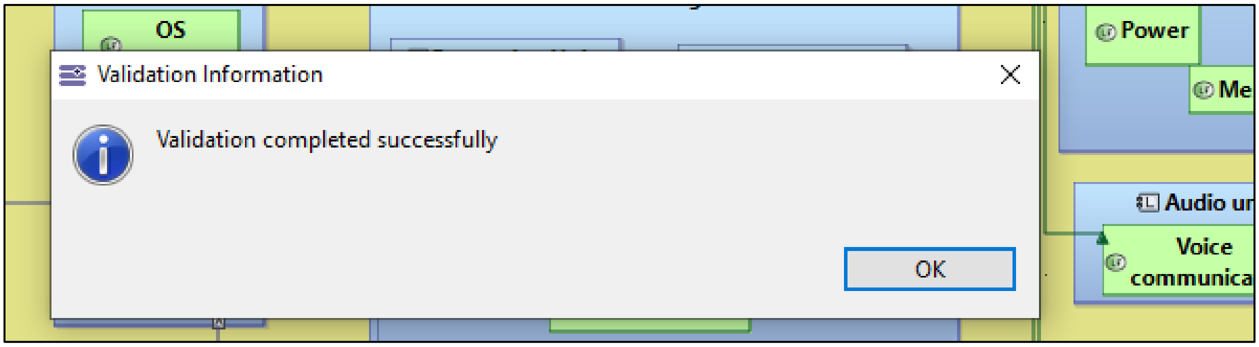
Validation Problems

Problems encountered during validation

Reason:
Diagnosis of Structure

Structure" (LogicalComponentPkg) " content has a naming conflict for: (Part) End User (RSLF).
Structure" (LogicalComponentPkg) " content has a naming conflict for: (LogicalComponent) End User (RSLF).
Structure contains 2 elements with conflicting names: End User (RSLF) : End User (RSLF), End User (RSLF) : End User (RSLF)
Structure contains 2 elements with conflicting names: End User (RSLF), End User (RSLF)
Request for Loction (ComponentExchange) doesn't have a summary
Request for Loction (ComponentExchange) doesn't have a description
Share Coordinates (ComponentExchange) doesn't have a summary
Share Coordinates (ComponentExchange) doesn't have a description
See Location (ComponentExchange) doesn't have a summary
See Location (ComponentExchange) doesn't have a description

Figure 24: Model Verification & Validation (Errors)



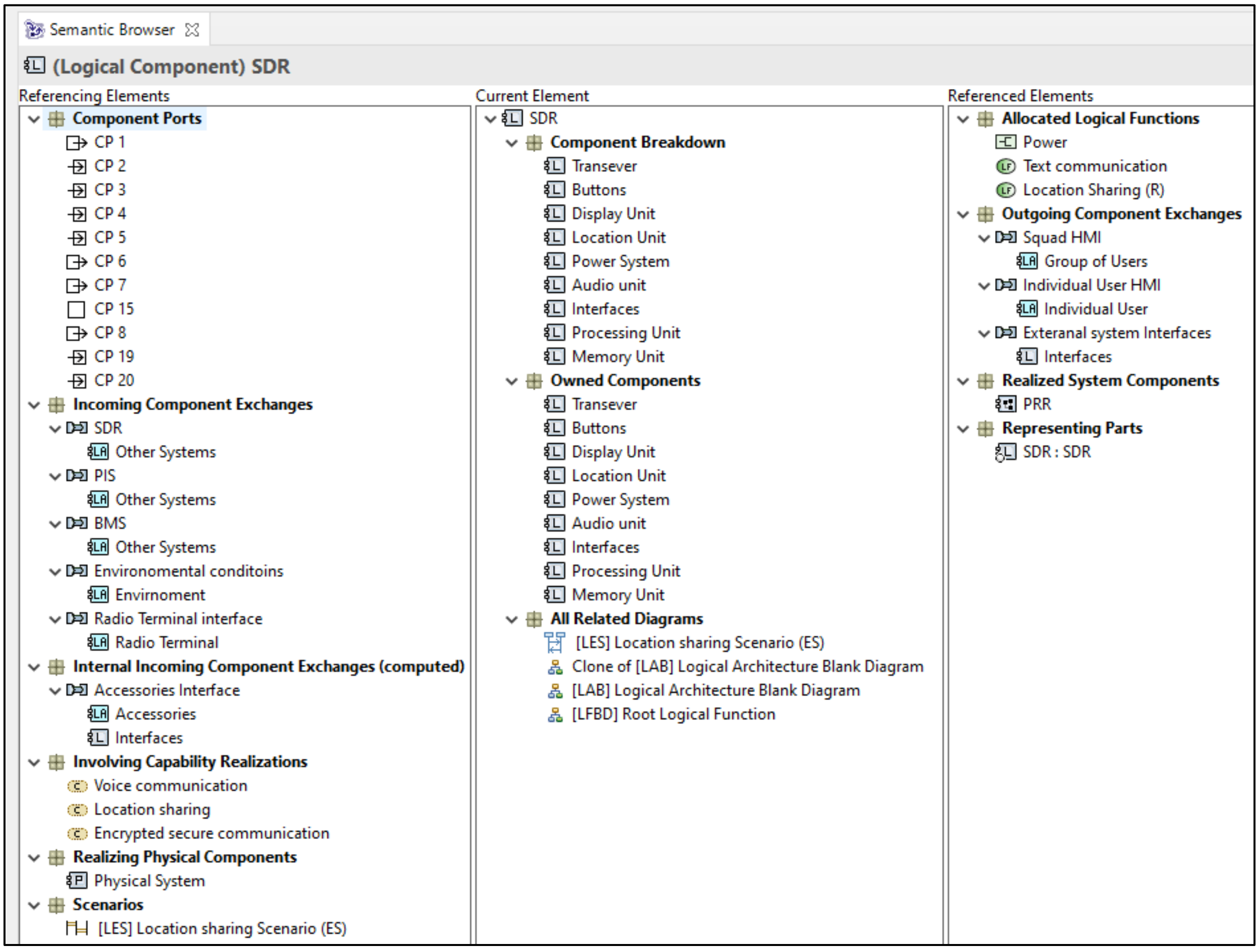
Validation Information

Validation completed successfully

Figure 25: Successful Model Validation

Additionally, the model verification command within Capella can be leveraged to validate the accuracy and integrity of individual elements across the entire model, confirming that each element aligns accurately with the overarching system architecture.

The Arcadia methodology further benefits from the inclusion of a robust command known as the "semantic browser," which aids in verifying and tracing model elements. This tool empowers modelers to discern how specific elements or entities are employed within the architecture, thereby ensuring their appropriate utilization, and sustaining overall consistency as shown in Figure 26.



Semantic Browser

(Logical Component) SDR

Referencing Elements

- **Component Ports**
  - CP 1
  - CP 2
  - CP 3
  - CP 4
  - CP 5
  - CP 6
  - CP 7
  - CP 15
  - CP 8
  - CP 19
  - CP 20
- **Incoming Component Exchanges**
  - SDR
    - Other Systems
  - PIS
    - Other Systems
  - BMS
    - Other Systems
  - Environomental conditoins
    - Envirnoment
  - Radio Terminal interface
    - Radio Terminal
- **Internal Incoming Component Exchanges (computed)**
  - Accessories Interface
    - Accessories
    - Interfaces
- **Involving Capability Realizations**
  - Voice communication
  - Location sharing
  - Encrypted secure communication
- **Realizing Physical Components**
  - Physical System
- **Scenarios**
  - [LES] Location sharing Scenario (ES)

Current Element

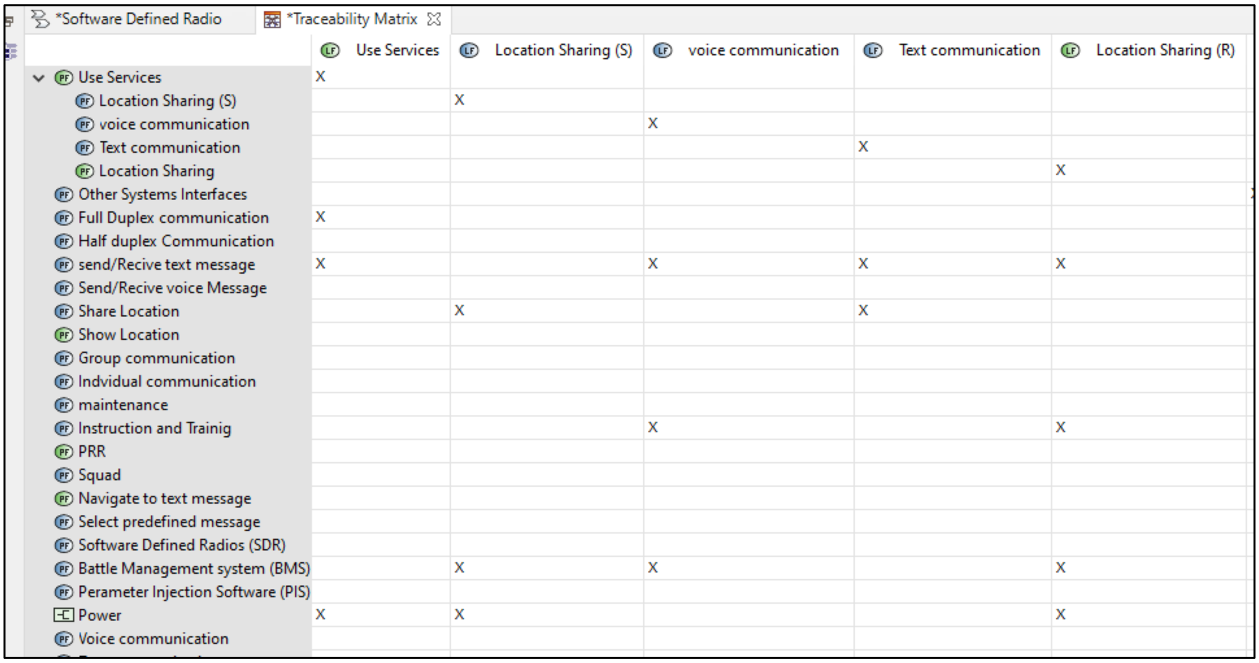
- SDR
  - **Component Breakdown**
    - Transever
    - Buttons
    - Display Unit
    - Location Unit
    - Power System
    - Audio unit
    - Interfaces
    - Processing Unit
    - Memory Unit
  - **Owned Components**
    - Transever
    - Buttons
    - Display Unit
    - Location Unit
    - Power System
    - Audio unit
    - Interfaces
    - Processing Unit
    - Memory Unit
  - **All Related Diagrams**
    - [LES] Location sharing Scenario (ES)
    - Clone of [LAB] Logical Architecture Blank Diagram
    - [LAB] Logical Architecture Blank Diagram
    - [LFBD] Root Logical Function

Referenced Elements

- **Allocated Logical Functions**
  - Power
  - Text communication
  - Location Sharing (R)
- **Outgoing Component Exchanges**
  - Squad HMI
    - Group of Users
  - Individual User HMI
    - Individual User
  - Exteranal system Interfaces
    - Interfaces
- **Realized System Components**
  - PRR
- **Representing Parts**
  - SDR : SDR

Figure 26: System Element Verification and Validation Using Semantic Browser

Capella augments the verification process with the provision of a valuable tool called the "Traceability Matrix", as shown in Figure 27. This tool serves to verify the allocation of system requirements throughout the system, offering a visual representation of the relationships between requirements and their allocation to the system, subsystems, and components, thus assuring comprehensive verification.



	Use Services	Location Sharing (S)	voice communication	Text communication	Location Sharing (R)
Use Services	X				
Location Sharing (S)		X			
voice communication			X		
Text communication				X	
Location Sharing					X
Other Systems Interfaces					
Full Duplex communication	X				
Half duplex Communication					
send/Recive text message	X		X	X	X
Send/Recive voice Message					
Share Location		X		X	
Show Location					
Group communication					
Indvidual communication					
maintenance					
Instruction and Trainig			X		X
PRR					
Squad					
Navigate to text message					
Select predefined message					
Software Defined Radios (SDR)					
Battle Management system (BMS)		X	X		X
Perameter Injection Software (PIS)					
Power	X	X			X
Voice communication					

Figure 27: Traceability Matrix

Furthermore, at each stage of the Arcadia methodology, a practice is enforced where function allocation to architecture blank diagrams is rigorously followed. This practice guarantees that functions are appropriately allocated and seamlessly integrated within the system model, facilitating effective verification and validation efforts. To further enhance the assurance of requirement allocation, both functional and non-functional, Capella provides the "manage function allocation" command, which furnishes insights into the allocation of functions to the system at each level, thereby supporting comprehensive verification and validation efforts. These verification and validation commands collectively contribute to ensuring the integrity of various aspects of the system model.

4.3 Economics Analysis of MBSE

This section delves into the economic outcomes resulting from the application of Model-Based Systems Engineering (MBSE) in the context of IoT-based mesh radio networks. The study focuses on how MBSE is used in the development of the need-driven architecture of personnel role radios. To evaluate the economic advantages, SDLC were considered, listed herein:

- Concept Development
- System Requirements Analysis
- System Design & Development
- Integration and Testing
- Verification and Validation

The development of a handheld radio follows a structured approach through key phases. In concept development, project objectives and feasibility are assessed, setting high-level goals. System requirements analysis involves detailed functional and non-functional requirements, defining behaviors and performance criteria. System design focuses on creating architectural layouts, prototypes, and system models. System development translates these designs into tangible hardware and software components, integrating subsystems for a cohesive system. The integration and testing phase integrates subsystems and rigorously tests the entire system for smooth operation. Verification and validation stages confirm compliance with requirements and user needs, including inspections and tests. System validation ensures the system's functionality in its intended environment, validating real-world performance for deployment.

The development of a handheld radio involves various cost-incurring activities throughout different phases. Throughout the handheld radio development phases, costs cover stakeholder engagement for expert insights, feasibility study expenses, and meeting organization in Concept Development. System Requirements Analysis involves cost of human resource, tools for documentation, and requirement elicitation costs. System Design incurs expenses for engineers, design software, and prototyping. System Development includes human resource, hardware/software purchases, and testing tool expenses. Integration and Testing involves personnel and equipment costs. Verification and Validation phase covers human resource, testing tools, and documentation expenses. System Validation includes field testing, operational costs, and user training expenses. These diverse costs are pivotal across the handheld radio development stages.

The implementation of Arcadia, an MBSE approach, on IoT-based mesh radio networks offered a range of economic benefits. In the concept development the MBSE helped to observe and define the system concept allowing tradeoff analysis, reduction in the number of human resource and time to conduct the concept development of the system. Arcadia effectively streamlined requirement management, reducing the dependance on extensive human resources and associated costs. Arcadia facilitated early identification of constraints, risks, and design issues within the system design and development process, thereby preventing unforeseen expenses and conserving resources. Arcadia model defines the integration of components and sub-systems, enabling modeler to define the interface and mitigating the associated risks such as, compatibility and communication protocols. The seamless verification and validation capabilities of Arcadia ensured precise alignment with stakeholder needs, minimizing rework and associated expenses. Arcadia enabled efficient documentation and lifecycle management, fostering clear communication among stakeholders and teams, leading to minimized change management costs, and ensuring system understanding across the project's lifecycle. This resulted in not only in optimized resource utilization but also contributed significantly to cost reduction across various factors of radio development, particularly in testing, prototyping, and lifecycle maintenance.

Cost-Benefit Analysis for MBSE

The economic analysis of the system is caried out by cost benefit analysis tool. Cost-benefit analysis (CBA) is a systematic approach to estimating the strengths and weaknesses of alternatives. It is used to determine options which provide the best approach to achieving benefits while preserving savings in. Parameters analyzed for the cost benefits analysis includes concept development, system requirement analysis, system design and development, integration and testing, and verification and validation.

The figures employed in this analysis are taken from actual project implementation with slight modification to ensure the integrity of intellectual property of the research funding organization. The project has a total budget of $20 million with the allocation of budget for the system development lifecycle phases listed in Table 1. The ensuing cost-benefit analysis is based on the outlined benefit parameters.

Table 2: Project Cost Distribution over the System Development Life cycle

System Development Lifecycle	**Cost with SE**	**Cost with MBSE**	**Reduction in Actual Cost**	**Reduction in Actual cost (in %)**
Concept Development	$2M	$1.5M	$0.5M	25%
System Requirements Analysis	$1M	$0.85M	$0.15M	15%
System Design & Development	$11M	$10.45M	$0.55M	5%
Integration and Testing	$3M	$2.5M	$0.5M	16.67%
Verification and Validation	$3M	$2.7M	$0.3M	10%

Total cost with SE: $20M

Total cost with MBSE: $18M

Cost Saving:

$$\$20M – \$18M = \$2M \text{ -------------------- } (1)$$

Return on Investment (ROI):

$$ROI = (Profit/Total\ Cost) \times 100$$

$$ROI = (2M/20M) \text{ x } 100$$

$$ROI = (0.1) \text{ x } 100$$

$$ROI = 10\% \qquad \text{------------------------- } (2)$$

The cost-benefit analysis shows that MBSE indicates the significant economic benefits for the development of the SDR system. The total estimated cost savings is $2 million, which represents return on investment of 10% cost of the project as shown in Equation (1) & (2).

Chapter 5 Conclusion and Future Work

This chapter serves as a platform for the exposition of concluding remarks and prospective areas of research within the domains of Model-Based Systems Engineering (MBSE) and Software-Defined Radios (SDR).

5.1 Conclusion

The study aimed to implement the Arcadia methodology, a Model-Based Systems Engineering (MBSE) approach, for the development of an IoT Based Mesh Radio Network device throughout the system lifecycle. The research explored the system hierarchy at four levels: system, sub-system, components, and parts, encompassing the system's journey from concept development to architecture development. Arcadia comprises of five phases, this study covering four of them.

The Operational Analysis phase discussed the system concept and the first-level system hierarchy, focusing on the system's operational requirements, including location sharing, text, and voice communication with the individual user and within a group of users. The system's ability to operate in various challenging conditions, such as high/low temperatures, rain, and dust, was also examined. System entities, actors, activities, and roles were identified. System Analysis further explored the first-level system hierarchy, defining dynamic behavior, system context, and eliciting requirements from the system's capabilities. It highlighted the system's operations, functional data flow, and interaction with actors during activities like location sharing, voice communication, and integration with other systems including Parameter Injection Software (PIS) and Battle Management System (BMS).

The Logical Architecture expanded the system as a "white box," addressing the second and third levels in the system hierarchy, defining subsystems and components without diving into technical considerations. It defined the basic structure of the system, specifying subsystems, components, and ensure that the systems will meet the stakeholder requirements. The components included the processing unit, power system, transceiver, location unit, display unit, audio processing unit, and external interfaces. The subsystems and components included the encryption unit, network protocols unit, microcontroller, and buttons. This exploration of system structure elements facilitated initial trade-off analysis for the subsystems and components. The subsequent phase, Physical Architecture, focused on defining system components and parts. It refined the components identified in the logical architecture and specified the actual structure of the system. Components such as the STM 32 controller, memory unit including Electrically Erasable Programmable Read-only Memory (EEPROM) and Flash Memory, codec for audio processing, GPS for location sharing, LCD for the user interface, transceiver, power amplifier, and low pass filter for radio frequency transmission were defined. The system's interfaces, both internal and external, including accessories, fill gun connector, headset, and wireless connections interfaces to other systems, were identified. During the definition of the system's physical architecture, a second-level trade-off analysis was performed, considering both internal and external system interfaces and addressing potential risks. This completed the architecture of the system and the implementation of Arcadia for the development of the IoT-based mesh radio network device.

Throughout the system's development, a comprehensive model verification and validation process was carried out in conjunction with each developmental phase, ensuring compliance to stakeholder requirements. This model Verification and Validation methods (model verification, diagram validation, validation using semantic browser, and traceability matrix) enhanced system development and supported efficient change management, allowing seamless configuration management at any stage of development. This methodological development resulted in seamless integration of people, product, and process, configuration management, seamless verification, and validation, and a 10% reduction in the actual cost of the project by minimizing potential risks and development time, as well as an efficient system design.

5.2 Future Work

The successful implementation of Model-Based Systems Engineering (MBSE), specifically utilizing the ARCADIA methodology, for the development of Software-Defined Radio (SDR) opens up avenues for intriguing future work and enhancements in this domain. These include expanding the application of MBSE to various complex systems like autonomous vehicles aerospace, space, information technology, and energy systems, thereby extending its benefits. There is potential for further integration and enhancement of change management strategies within the Arcadia framework, optimizing its capacity to adapt to evolving requirements. Exploring advanced risk mitigation techniques, such as predictive analytics and machine learning, may lead to more precise risk assessment, such as python for Capella add on can be utilized for predictive analysis. Detailed economic and cost modeling can be developed to quantify the potential savings and return on investment associated with the adoption of MBSE. Ensuring the interoperability and standardization of MBSE with other engineering and development standards is imperative. this research has established a strong foundation for MBSE, especially within the ARCADIA framework, in the development of complex systems, and the identified areas for future work aim to refine and extend MBSE's application, address challenges, and enhance its impact across various industries. These efforts will contribute to the efficiency of system development and have far-reaching implications for the future of engineering practices.

References

[1] A. M. Madni, "Model-based systems engineering: Motivation, current status,," *Systems Engineering,* vol. 21, pp. 172-190, 2018.

[2] INCOSE, "INCOSE," INCOSE, 1990. [Online]. Available: http://www.incose.org. [Accessed 2021].

[3] A. Hall, A methodology for systems Engineering, D. Van Nostrand Company , 1962.

[4] B. S. Blanchard, Systems Engineering and Analysis, Prentice Hall, 1981.

[5] N. Castela, J. Lopes, A. R. D. S. Tribolet and G. Arminda, "Businees Process Modeling with UML," in *International Confrenece on enterprise information systems,* 2002.

[6] T. Weikiens, Systems engineering with SysML: Modeling, analysis, verification, and design, Morgan Kaufman Publishers , 2008.

[7] L. L. Edward Yourdon, Structured design: Fundamentals of a discipline of computer program and systems design, New York: Yourdon Inc, 1978.

[8] P. P.-S. Chen, "The entity-relationship model: A framework for information retrieval and database," *ACM Transactions on Database Systems,* vol. 1, no. 0362-5915, pp. 9-36, 1976.

[9] G. Booch, The unified modeling language user guide, Adison - Wesley Professional , 1998.

[10] A. M. R. S. Sandford Friedenthal, A Practical Guide to SysML, Morgan Kaufmann, 2016.

[11] A. L. Ramos, "Model-Based Systems Engineering: An Emerging Approach for Modern Systems," in *IEEE TRANSACTIONS ON SYSTEMS*, Barcelona, 2012.

[12] J. D’Ambrosio, A. Adiththan, E. Ordoukhanian, P. Peranandam, S. Ramesh, A. M. Madni and P. Sundaram, "An MBSE Approach for Development of Resilient Automated Automotive Systems," *systems,* vol. 7, no. 1, 2019.

[13] T. Huldt and I. Stenius, "State-of-practice survey of model-based systems engineering," *Systems Engineering,* vol. 22, pp. 134-145, 2018.

[14] A. M. Madni, "Economic Analysis of Model-Based Systems Engineering," *systems,* vol. 7,

no. 12, 2019.

[15] A. Albers, N. Bursac, Hamburg, C. Birk, J. Powelske and S. Muschik, "Model-based systems engineering in modular design," *Design Science,* vol. 5, no. 17, 2019.

[16] S. B. T. M. H. e. P. a. K. H. S. Darpel, "Method for tracking and communicationg aggregate risk through the use of model-based systems Engineering tools," *Space Safety Engineering* , vol. 7, pp. 11-17, 2020.

[17] A. Hanson, J. Mindock, S. Okon, M. Hailey, K. McGuire, J. Bardina, H. Stewart, W. Toscano, S. Winther, T. Burba, Rubin and e. al, A Model-Based Systems Engineering Approach to Exploration Medical System Development, 2019.

[18] M. Peciak and W. Skarka, "Assessment of the Potential of Electric Propulsion for General Aviation Using Model-Based System Engineering (MBSE) Methodology," *aerospace,* vol. 9, no. 74, 2022.

[19] T. A. Berg, K. N. Marino and K. W. Kintziger, "The Application of Model-Based Systems Engineering to Rural Healthcare System Disaster Planning: A Scoping Review," *Disaster Risk Science,* vol. 14, pp. 357-368, 2023.

[20] Z. Hu, J. Lu, J. Chen, X. Zheng, D. Kyritsis and H. Zhang, "A Complexity Analysis Approach for Model-based System Engineering," Budapest, 2020.

[21] P. Beery and E. Paulo, "Application of Model-Based Systems Engineering Concepts to Support Mission Engineering," *Systems,* vol. 7, no. 44, 2019.

[22] A. M. Madni, M. Carla C and S.-D. Lucero, "Leveraging Digital Twin Technology in Model-Based Systems Engineering," *systems,* vol. 7, no. 7, 2019.

[23] Q. Wu, D. Gouyon, E. Levrat and S. Boudau, "Use of Patterns for Know-How Reuse in a Model-Based Systems Engineering Framework," *SYSTEMS,* vol. 14, no. 4, 2020.

[24] P. H. Feiler and D. P. Gluch, Model-Based Engineering with AADL: An Introduction to the SAE Architecture Analysis & Design Language., New York: Pearson Education Inc, 2013.

[25] Thales, "ARCADIA Method," Thales, 2005. [Online]. Available: hhtp://mbse-capella.org/arcadia.html. [Accessed 2021].

[26] M. Christophe and P. Jacques, "Software Radio: A Catalyst for Wireless Innovation," *IEEE*

Communications Magazine, 2015.

[27] N. Kassri, A. Ennouaary, S. Bah and B. Hajar, "A Review on SDR, Spectrum Sensing, and CR-based IoT in Cognitive Radio Networks," *International Journal of Advanced Computer Science and Applications,* vol. 12, no. 6, 2021.

[28] J. Mitola, "Software Radios Survey, Critical Evaluation and Future Directions," *[Proceedings] NTC-92: National Telesystems Conference, Washington, DC, USA,* vol. 10.1109/NTC.1992.267870, 1992.

[29] J. Mitola, "The software radio architecture," *EEE Communications Magazine,* vol. 33, pp. 26-38, 1995.

[30] R. Akeela and B. Dezfouli, "Software-defined Radios: Architecture, state-of-the-art, and challenges," *Computer Communications,* vol. 128, 2018.

[31] G. J. Minden, J. B. Evans, L. Searl, D. DePardo, V. R. Petty, R. Rajbanshi, T. Newman, Q. Chen, F. Weidling, J. Guffey, D. Datla, B. Barker, M. Peck, B. Cordill, A. M. Wyglinski and A. Agah, "KUAR: A Flexible Software-Defined Radio Development Platform," in *2nd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks,* 2007.

[32] K. Tan, J. Zhang, J. Fang, H. Liu, Y. Ye, S. Wan, Y. Zhang, H. Wu, W. Wang and G. M. Voelke, "Sora: high-performance software radio using general-purpose multi-core processors," *Communications of the ACM,* vol. 54, no. 1, 2011.

[33] E. research, "Ettus research - networked software defined radio (SDR)," *[Online],* no. https://www.ettus.com/..

[34] M. Bansal, A. Schulman and S. Katti, "Atomix: A Framework for Deploying Signal Processing Applications on Wireless Infrastructure," in *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI ’15)*, 2015.

[35] M. C. Ng, K. E. Fleming, M. Vutukuru, S. Goss, Arvind and B. Hari, "Airblue: A system for cross-layer wireless protocol development," in *ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS*, 2010.

[36] G. Baldini, T. Sturman, A. Dalode, A. Kropp and C. Sacchi, "An emergency communication system based on software-defined radio," *EURASIP Journal on Wireless Communications and Networking,* vol. 2014, no. 1, 2014.

[37] Florin-Lucian Chiper; Alexandru Martian; Calin Vladeanu; Ion Marghescu; Razvan Craciunescu; Octavian Fratu, "Drone Detection and Defense Systems: Survey and a," *Sensors,* vol. 22, no. 4, 2022.

[38] C. Bergstrom, S. Chuprun, S. Gifford and G. Maalouli, "Software defined radio (SDR) special military applications," *MILCOM 2002. Proceedings,* vol. 1, 2002.

[39] T. Ulversoy, "Software Defined Radio: Challenges; Opportunities," *IEEE Communications Surveys & Tutorials,* vol. 12, no. 4, 2010.

[40] R.-j. Yan, "Development and Missions of Unmanned Surface Vehicle," *Marine Science and Application* , vol. 9, pp. 451-457, 2010.