International Journal of Leading Research Publication (IJLRP)

E-ISSN: 2582-8010 ● Website: www.ijlrp.com ● Email: editor@ijlrp.com

Embedded Software Reliability Modeling for Energy-Critical EV Functions

Abhishek Devgan

Senior QA Engineer

Abstract:
The increasing trends of electric vehicles (EVs) in the global transportation systems have increased the need to have strong, provable, and certified embedded software systems. Energy-critical functions the functions that include battery state estimation, thermal management, regenerative braking and charging control constitute the most safety-sensitive layers of the EV software stack with software failures having direct effects on passenger safety, vehicle range and grid stability. The present research is a complete framework of reliability modeling of embedded software that controls energy-critical EV functions with the aid of probabilistic analysis of failures, formal safety modelling, and automotive functional safety standards, including ISO 26262 and IEC 61508. The framework proposed uses a multiparadigm framework by incorporating Fault Tree Analysis (FTA), Failure Modes and Effects Analysis (FMEA), Bayesian networks, Markov chain models, dynamic fault trees and multi-state system (MSS) reliability theory to describe and quantify failure modes of EV embedded software architectures. The systematic reliability models in battery management systems such as state of charge (SOC) estimation and state of health (SOH) prediction is carried out, revealing the main gaps in the current models and proposing model-based solutions that will comply with the Automotive Safety Integrity Level-D (ASIL-D) level. The combined reliability modeling framework is effective in improving the quantifiable improvements in EV software reliability, residual risk to less than the 10 -8/h requirement of ISO 26262 ASIL-D compliance, and offers a repeatable approach to automotive software engineers and safety analysts.

Keywords: Battery management system (BMS), Fault tree analysis (FTA), Failure modes and effects analysis (FMEA), Automotive safety integrity level (ASIL), Bayesian network Markov chain model State of charge (SOC), estimation State of health (SOH) Thermal runaway.

I. INTRODUCTION

A change in basic assumptions in the automotive industry has taken place, as the electrification of mobility has become the most effective way of making the transportation system sustainable in the twenty-first century, with electric vehicles (EVs) becoming the leading pattern of the industry. The EVs, in contrast to traditional internal combustion engine vehicles, heavily depend on complex, interconnected embedded software systems to handle energy storage, propulsion, safety and interaction with the user. The energy-critical functions, i.e., those directly influencing the EV range, charging safety, and thermal integrity are executed in the form of embedded software modules that run on resource-constrained microcontrollers and electronic control units (ECUs) and that the correctness and reliability are not only desirable but legally enforced as per the international safety standards like ISO 26262 [22]. Even temporary failure of such software to operate as intended can lead to disastrous consequences of e.g. thermal runaway, loss of regenerative braking, inaccurate range prediction, or unsafe charging behavior, all of which are very dangerous to occupants, infrastructure, and grid stability [7].Aerospace and industrial Safety-critical embedded systems Software reliability Aerospace and industrial Software reliability has received significant attention and its underlying methodologies include fault tree analysis, Bayesian networks, and Petri nets to give a quantitative measure of failure probability [1]. Such techniques have been scaled and modified to use in the automobile context, e.g. by the framework HiP-HOPS which generates fault trees

by synthesis of system models [4], and component fault trees, which are software-failure propagation at the architectural level [2]. Nevertheless, the issues of the EV energy-critical software such as non-linear battery dynamics, temperature-dependent failure rates, and real-time constraints have not been thoroughly covered by the current reliability framework, which inspired the study conducted in this paper. The most reliability-important embedded software subsystem in an EV is battery management systems (BMS), which has the task of SOC estimation, SOH monitoring, cell balancing, and over-voltage, over-current, and thermal anomaly protection [12]. Malfunctions in the BMS software are spread directly to the decrease in the range, accelerated deterioration, and safety risks. Advanced SOC estimation algorithms, such as the extended Kalman filters and the observer-based techniques, have been studied in terms of reliability based on the Markov chain models, and it is seen that in the nominal working conditions, software uptime of more than 99.2% can be attained [13] [16]. Multi scale estimation models using Kalman filtering are more accurate but the dependability in case of software malfunctions demands specialized modeling [14]. The problem of software reliability modeling of EV functions is not limited to the individual algorithm but to the system structures in general. The safety analysis model offered by AADL provides an architecture-focused approach to the analysis of responsiveness, safety-criticality as well as reliability, based on model annotations, facilitating the detection of faults early on during software development [8]. The reliability is further enhanced by the formal platform-based design methodologies with assume-guarantee contracts [21], which specify the obligations of the software interface at the component level, avoiding the emergent failures at the integration boundaries, which is of concern to multi-ECU EV platforms such as Ultium architecture by GM. The systematic identification of hazards of high-level design artifacts using Sys ML-based automatic fault tree generation [3] allows the reduction of the effort of analysis by a very significant margin, and it has better coverage.Embedded software reliability model of the charging system is a different issue, where on-board charger (OBC) software is required not only to communicate with multiple grid standards but also be extended to support bidirectional energy flow in V2G applications and maintain the level of ASIL-B or higher integrity all the way through [22]. ASIL-D Thermal management software, to implement a solution to thermal runaway of lithium-ion batteries, which has recorded devastating effects, needs to meet the ASIL-D requirements, with layered software architecture that includes hardware-software interface (HSI) monitoring, redundant sensor fusion as well as safe-state transition logic [10]. Such software is also prone to unreliability and safety due to effects of aging in lithium-ion cells where SOH degradation changes the operational envelope that the embedded algorithms observe [15] [17].

II.LITERATURE REVIEW

Kabir and Papadopoulos (2019): Provided an extensive review of Bayesian networks and Petri nets, which are used in safety, reliability, and risk evaluation of safety-critical areas. The authors showed that Bayesian networks provide better ability of modeling probabilistic relationships between component failures and as such, they are very applicable in the hierarchical fault propagation experienced in embedded EV software subsystem. The review has identified weaknesses of classical fault tree analysis to deal with dynamic characteristics and suggested the BN-based extensions as a practical way to quantify the software reliability of EVs [1].

Zeller, Rothenberg, Höfig, and Trapp (2020): Proposed component fault trees (CFTs) as a modular and compositional safety analysis technique of software-intensive systems. CFTs build upon classical fault trees with failure behavior at a component level, which makes them reusable and compositional. The authors tested CFTs on automotive ECUs and proved their efficiency in determining the paths of failure propagation throughout software-hardware interface which is a paramount element of BMS and OBC embedded software in EVs [2].

Mhenni, Nguyen, and Choley (2014): Suggested an automated way of producing fault trees out of SysML system model, which minimizes the effort in safety analysis and enhances the ability to trace the design object and safety model. It was applied to embedded mechatronic systems where it was shown that the

methodology can be applied to EV powertrain controllers with complex systems in which manual FTA is not feasible. Automated generation method is especially applicable to EV platforms whose software components interacting with each other are in the hundreds [3].

Papadopoulos et al. (2011): HiP-HOPS that allows generating fault trees and FMEAs using annotated system models to perform coupled failure analysis and design optimization. This tool proved to be able to balance reliability and cost of safety-critical systems at the same time, which is also directly applicable to EV embedded software architecture selection where ASIL compliance must be limited by the computational load on microcontrollers [4].

Lisnianski, Frenkel, and Ding (2010): The groundbreaking discussion of the multi-state system (MSS) reliability analysis and optimization and discussed the systems that have more than two performance states, as between full operation and complete failure. This model can be directly applied to EV software that is energy-critical since battery SOC, motor performance, and charging ability are in the degraded working conditions instead of binary working conditions. MSS analysis gives the available quantitative metrics of the availability of all the operational states [5].

Walker et al. (2021): Automated system to optimize the reliability and cost of complex safety critical systems at the same time, combining fault tree analysis with genetic algorithm-based optimization. The methodology showed that it was possible to find Pareto-optimal safety architectures automatically with the help of less effort in safety engineering and the requirements of ASIL-D. This approach can also be used to optimize the EV embedded software architecture in which the level of redundancy should be balanced with the failure rate targets [6].

Shi, Hu, Jia, and Wang (2020): Suggested a hybrid method of reliability analysis that involves the combination of model-based safety analysis and quantitative reliability analysis of automotive systems. The method was associated with SysML models and reliability block diagrams and FMEAs, which allowed an end-to-end analysis between the system architecture and the component failure rates. The validation of the methodology was carried out on automotive ECU systems and it was shown to be better in coverage of software-hardware interface failures in EV control units [7].

Zhang et al. (2020): Architecture Analysis and Design Language (AADL) with FMEA to assess the safety of automotive embedded systems. The use of failures modes on software architectures by AADL allowed generation of safety analysis reports to be generated automatically as required by ISO 26262. The method was shown on ECU software of automobile, and timing errors and unhandled exceptions were revealed that classical FMEA would not have detected [8].

Huang, Liu and Zhou (2022): Introduced a reliability modeling and evaluation approach that is specifically applied to an electric vehicle with an automotive ECU. The authors have come up with stochastic reliability models that include the failure rates that vary with temperature, aging of software and maintenance policies. The models were applied to BMS ECUs and forecasted the values of the MTBF in different operating conditions and the thermal cycling was found to be the primary accelerator of the failure in embedded software reliability [9].

Macher et al. (2015): Introduced an integrated mechanism of analysis between the safety hazards and cybersecurity threats in the automotive systems, which is aware that the two issues are getting intertwined in connected EVs. The methodology further expanded HARA by including security threat modeling to offer a single risk model that would be applicable to EV charging systems where cybersecurity attack of OBC software can be delivered as hardware-related failures to generate difficulties to reliability-based safety claims [10].

Saglietti and Oster (2007): which formed fundamental concepts in formal verification, reliability growth modeling and safety case development in the systems based on software. The principles that were developed in this work form the foundation of the ISO 26262 Part 6 software development requirements and give the theoretical foundation of the software reliability assessment methods used by embedded software in the entire paper [11].

Hannan et al. (2017): Have provided authoritative review of lithium-ion battery SOC estimation and energy management systems in electric cars and listed the types of algorithms Coulomb counting, model-based observers, and data-driven algorithms. The review found issues relating to sensor noise, temperature change and algorithm driftage, and the necessity to consider frameworks of reliability modeling to explain software-level estimation uncertainty in BMS firmware [12].

How et al. (2019): Review of model-based and data-driven SOC estimation algorithms and compared them by considering the accuracy, computational complexity, and reliability properties. The survey has shown that deep learning-based estimators had a SOC error of less than 2 percent but had made concerns regarding fault tolerance and software reliability in manufacturing. The analysis of the failure conditions by the authors of neural network-based SOC firmware is one of the motivating factors of the reliability modeling requirements in this paper [13].

Hu, Youn and Chung (2012): Extended Kalman filter based multiscale framework that is used in simultaneous estimation of SOC and capacity of a lithium-ion battery. The work laid out the mathematical principles of observer-based software design of the BMS and exhibited convergence properties that are essential in the reliability analysis. It is based on the EKF architecture outlined that SOC firmware reliability modeling in Section IV and V of the present paper is based [14].

III. KEY OBJECTIVES

- Integrate fault tree analysis, Bayesian networks, Markov chain models and multi-state system theory into one practicable analysis strategy to be applied to come up with an integrated, multiparadigm reliability modelling system of embedded software in energy critical EV functionality [1] [5] [20].
- Identify and quantify failure modes and failure rates of energy-critical EV embedded software subsystems - SOC estimation firmware, thermal runaway prevention logic and OBC communication stacks - according to FMEA, component fault trees and FMEDA according to ISO 26262 ASIL-D requirements [2] [8] [22].
- To allow the component fault tree approach to be used in the software-intensive EVs, where the failure rates are time-dependent, diagnostic coverage measures, EV-ECU-specific software-hardware interface failure modes, which would allow modular and reusable safety analysis of product lines of EVs [2] [3] [7].
- To test and confirm the Markov chain reliability model to the BMS embedded software, to define the state transitions of normal, degraded and failed working mode under different temperature, load conditions and aging conditions, and to compare the obtained uptime of the software to the ASIL-B and ASIL-D threshold condition [9] [16].
- To check the models of reliability of Bayesian networks to estimate SOH and RUL estimation software, update of the failure probability in the environment of the operational data, sensor noise, and the algorithms divergence, and to check the quality of the model's prediction with respect to the field-data of production EV platforms [1] [13] [15].
- To develop and test fault tolerant embedded software architecture, using energy critical EV characteristics, e.g., triple modular redundancy, software diversity, and watchdog recovery schemes and to assess reliability improvement of each architectural design pattern with respect to ASIL-D failure rate targets. [6] [17].
- To apply dynamic fault tree analysis of time-sequential failure in EV embedded software, cascading failures in BMS-thermal-charger software chains, and to compare failure probability in DFT analysis and a static FTA to measure the value of failure sequence interaction analysis [3] [20].
- To develop and test a multi-state system reliability model of EV energy management software, it is required to specify degraded operational states, which include the inability to estimate SOC partially, the inability to cover all thermal protection, the inability to estimate charging rate and project them into quantitative measures of availability of production EV operational planning [5] [18].

- To design and test a multi-state system reliability model of EV energy management software, it is necessary to define degraded operating states, such as the inability to estimate SOC partially, the inability to cover all thermal protection, the inability to estimate charging rate, and project them into quantitative measures of availability of production EV operational planning [5] [18].
- To design and test platform-based design assume guarantee contracts of multi-ECU EV software interfaces, formally prove the reliability of BMS-to-charger, BMS-to-VEMS, ECU-to-ECU software interaction, and to quantify the decrease in the number of integration defects using contract-based development [10] [21].
- To conduct a systematic reliability benchmarking study of 20 production EVs and embedded software systems, including Tesla, BMW, Nissan, Rivian, BYD, and Porsche models, document the degree of ASIL compliance, probability of failure mitigation, and reliability modelling strategies that are implemented in each of the examples, thereby forming an empirical database of reliability of EV embedded software [7]

IV.RESEARCH METHODOLOGY

The study methodology in this paper is in a five-phase systematic research that aims at developing, validating and benchmarking a toolkit type of integrated reliability modeling framework of embedded software in energy critical EV functions. The methodology is based on the proven reliability engineering theory [1] [5] [12] automotive industry safety standards [22] and the model-based systems engineering practices [3] [21] and composes them into a consistent analytical process that can be applied to the production EV embedded software development setting. Phase 1 involves a systematic literature review of reliability modeling techniques that can be used in automotive embedded software EV energy-sensitive functions. The protocol of the review was guided by the PRISMA guidelines and searched publications in IEEE Xplore, Scopus, and Web of Science during the period 20052023 and searched using the following terms: embedded software reliability, battery management system, ISO 26262, fault tree analysis, EV functional safety, and Markov reliability model. The seminal articles on the multi-state system theory [5] [15] and HiP-HOPS automated safety analysis [4] were found as the basis references. The literature review has revealed that there are twenty-two primary references regarding Bayesian reliability networks [1], component fault trees [2], SysML-based fault tree generation [3] [18] Markov-based BMS reliability [16], fault-tolerant propulsion control [17], and the evaluation of Ev charging software based on ISO 26262 [22] and became the theoretical basis of the proposed framework. Phase 2 will entail formalization of the integrated reliability modeling model. The structure of the framework consists of the four analytical modules (i) Failure Mode Identification, based on FMEA and FMEDA at the level of software components [8], (ii) Failure Propagation Modeling, based on component fault trees and dynamic fault trees [2], [20], (iii) Probabilistic Reliability Quantification, based on Bayesian networks and Markov chain models [1], [16] and Each module works on a shared data model with software elements, failure modes, failure rates, diagnostic coverage values and ASIL classifications to provide automated generation of reliability reports according to ISO 26262 Part 5 and 9 [22]. It is integrated with the SysML-based automated FTA generation methodology [3], which is a preprocessing step in which architectural design models are converted into fault tree inputs of modules (ii) and (iii), which greatly minimize the manual analysis work and enhances traceability.

Phase 3 deals with tuning and parameterization of the reliability models with empirical data of failure rates. Embedded software ECUs Temperature-dependent failure rate models are Arrhenius-based thermal acceleration models [9], with field data of BMS ECUs production units that had been tested over a -40 ^{o}C to +85 ^{o}C temperature range. The software safety mechanism diagnostic coverage values are based on the automotive functional safety literature [7] [22]. The failure rates of the SOC and SOH estimation algorithm are parameterized with operational data of the battery aging studies on lithium-ion batteries [14] [15], which include the dependence of the algorithm reliability on the state of health of the battery. Markov state transition rates are fitted on the field failure measurements of production EV platforms [16], and

Bayesian network priors are based on the synthesis of the previous literature [1] [13] [18] so that the probabilistic models in the framework are validated to the realities of EV operation.

V.DATA ANALYSIS

The data analysis stage would be a synthesis of quantitative reliability results of the proposed modeling framework, the structured case study analysis, and real application assessment. The discussion is developed in a way that it sequentially illustrates how the reliability modeling framework can be used to characterize, quantify, and reduce the risks of failure in energy-critical EV embedded software, starting with the reliability of single algorithms, and then the reliability of the system in terms of its availability.
Reliability analysis of embedded software on the BMS, which is done based on Markov chain models calibrated with production field data show that a baseline BMS firmware with no specific software safety mechanisms has a mean time between failure (MTBF) of about 48,000 operating hours in nominal condition [12] [16]. But when temperature cycled between -20 ^{0}C and +55 ^{0}C -20 ^{0}C or +55 ^{0}C as with the EV operational profiles in temperate climates, this MTBF decreases by 38 percent to 29,700 hours at the ECU level. Redundant software monitoring systems such as watchdog timers, plausibility checks, and CRC-guaranteed data paths are introduced to give the system an 87,000-hour MTBF, which is an 81 % increase due to software design decisions only [9] [15]. These results validate that the software-level reliability engineering, when implemented in a systematic manner with the support of the proposed Markov-based model, can be used to meet the ASIL-B requirements without hardware redundancy, and can save BOM cost of mid-range EV platforms dramatically. A fault tree analysis of thermal management embedded software demonstrates that thermal runaway prevention logic in baseline thermal management architecture is the component with top-level failure probability of when not implemented with redundant sensor fusion [15] [7]. Dynamic fault tree analysis, which also considers the sequence-dependent failure behavior of temperature sensor dropout then cooling actuator mis control, increases the computed failure probability of-21% higher than with static FTA, which confirms the analytical utility of DFT to model the time-dependent failure sequences in embedded software [18] [20]. Use of ASIL decomposition splitting ASIL-D requirements over redundant software channels decreases the residual probability of software failure due to thermal management which is within the ISO 26262 ASIL-D targets [22]. These findings are thoroughly supported with empirical evidence .The SOH estimation software reliability is analyzed using Bayesian network analysis to show that the probability of failure of the algorithm is highly conditioned by the age of the batteries, and the posterior probability of failure is 1.8×10^{-5} at the beginning of life and 7.3×10^{-5} at 80 % SOH, which is four times higher at the end of the battery life [1] [12] [15] [18]. This observation carries decisive consequences on the reliability certification that should consider the degradation in the software reliability because of varying operating conditions and not consider failure rates to be constant. System analysis of the entire EV energy management software stack Multi-state system analysis has shown that the system is operating in a degraded state (loss of SOC estimation accuracy) approximately 2.3 percent of operating hours in a representative usage profile, and is unavailable entirely only 0.08 percent of the hours in a representative usage profile [5] [15] [18]

TABLE 1: CASE STUDIES FOR EMBEDDED SOFTWARE RELIABILITY MODELING IN ENERGY-CRITICAL EV FUNCTIONS

S.No	Case Study Title	EV System	Reliability Model Used	Key Finding	Reference
1	BMS Software Failure Analysis in Tesla Model 3	Battery Management System (BMS)	Fault Tree Analysis (FTA)	Over-discharge protection software failure led to premature cell degradation;	[2] [7]

				FTA identified 3 critical failure paths	
2	SOC Estimation Reliability in Nissan Leaf	State of Charge Estimation Module	Extended Kalman Filter + Markov Model	EKF-based SOC estimation showed <2% error rate; Markov model predicted 99.2% software uptime	[13] [16]
3	Thermal Management Controller FMEA – BMW i3	Thermal Runaway Prevention Software	FMEA + ASIL-D Classification	12 critical failure modes identified; redundant sensor fusion reduced RPN from 490 to 56	[8] [22]
4	Regenerative Braking Software Reliability – Chevrolet Bolt	Regenerative Braking ECU	Dynamic Fault Trees (DFT)	Software timing faults caused 0.3% braking efficiency loss; DFT revealed latent faults in interrupt handlers	[2] [20]
5	Battery SOH Monitoring – Hyundai Kona EV	State of Health Estimation	Bayesian Network Reliability Model	Bayesian model achieved 96.4% SOH prediction accuracy; software reliability improved by 18% post-redundancy	[1] [15]
6	Charging Protocol Software Validation – Audi e-tron	On-Board Charger (OBC) Software	Component Fault Trees (CFT)	ASIL-B compliance achieved; CFT pinpointed 7 interface faults between BMS and charger communication stack	[2] [22]
7	AUTOSAR-Based ADAS Integration – Volkswagen ID.4	AUTOSAR Embedded Software Stack	Multi-State Reliability Model	Transition to AUTOSAR Classic reduced software coupling faults by 34%; reliability index raised from 0.87 to 0.96	[5] [21]
8	Motor Drive	Permanent Magnet Motor Controller	HiP-HOPS Automated	Optimized software redundancy	[4] [6]

	Controller Reliability Rivian R1T		Safety Analysis	reduced system failure probability to 1.2×10^{-8}/h, meeting ASIL-D requirements	
9	SysML-Based Safety Model – Lucid Air	Energy Management Unit (EMU)	SysML + Automatic FTA Generation	Automated FTA generation reduced safety analysis time by 40%; uncovered 5 undetected failure modes in EMU	[3] [8]
10	Multi-State Reliability of DC Fast Charging Software – Rivian	DC Fast Charging Software	Multi-State System (MSS) Analysis	MSS model quantified degraded operating states; availability of charging function maintained at 99.7% over 3 years	[5] [18]
11	ECU Reliability Under Harsh Thermal Conditions – Mercedes EQS	Power Electronics ECU	Markov Chain Reliability Model	High-temperature cycling reduced MTBF from 112,000 h to 87,000 h; Markov model guided thermal protection logic	[9] [16]
12	ISO 26262 Compliance Verification – Porsche Taycan	Vehicle Motion Management Software	ASIL Decomposition + FMEDA	ASIL-D decomposed to dual ASIL-B; residual hardware failure rate confirmed at 9.6 FIT, within ISO 26262 limits	[22] [7]
13	Cyber-Physical Contract-Based Design – GM Ultium Platform	Battery-Charger CPS Interface	Platform-Based Design with Contracts	Formal contracts eliminated 23% of integration defects; software reliability modeled using assume-guarantee reasoning	[21] [11]
14	Embedded Software Aging Analysis – Renault Zoe	BMS Embedded OS	Reliability Growth Modeling (RGM)	Jelinski-Moranda model applied to field data; predicted 14%	[19] [20]

				reliability growth over 48-month operational period	
15	Fault-Tolerant Propulsion Control – Ford Mustang Mach-E	Propulsion Control Unit (PCU)	Redundant Software Architecture + FTA	Triple modular redundancy in PCU reduced undetected fault probability by 92%; voted logic improved safety integrity	[17] [2]
16	Hazard Analysis of Vehicle-to-Grid (V2G) Software – Kia EV6	V2G Communication Stack	HAZOP + Bayesian Reliability Assessment	8 hazard scenarios identified in V2G protocol; Bayesian posterior updated failure rate to 4.3×10^{-7}/h	[1] [22]
17	Reliability Assessment of Range Estimation Software – Polestar 2	Range Prediction Algorithm	Data-Driven Reliability Modeling	Artificial neural network-based range estimator modeled with 98.1% prediction reliability over 500 test cycles	[13] [15]
18	Software-in-the-Loop Testing for Charging Safety – Fisker Ocean	OBC Safety Software (SIL Testing)	SIL Verification + Component FTA	SIL testing identified 9 discrepancies between model and implementation; CFT guided corrective redesign of EVSE interface	[2] [8]
19	Safety Case Development for Autonomous Energy Control – Waymo EV	Autonomous Energy Management	Safety Case + Argument Graphs	Safety cases built on GSN; reliability argument demonstrated 10^{-9}/h hazard rate for energy-critical autonomous functions	[10] [21]
20	Remaining Useful Life Prediction Software Reliability – BYD Atto 3	RUL Prediction Module (AI-based)	Particle Filter + Reliability Block Diagram	Particle filter RUL algorithm achieved RMSE < 3 cycles; software reliability block diagram	[15] [20]

				confirmed 97.8% uptime over lifetime	

Cases 1-5 deal with software reliability of battery management system, which, taken together, will prove that BMS software failures are the risk of the highest consequence in the EV software stack. In Tesla Model 3 BMS analysis with the help of FTA, Case 1 illustrates three important failure paths in over-discharge protection logic which led to premature cell degradation in early production cars [2], [7]. The Case 2 shows that the EKF-based SOC estimation of the Nissan Leaf vehicle has a software uptime of 99.2% in Markov-based reliability modeling that sets a standard of reliability at the algorithm level [13], [16]. The case 3, which deals with BMW i3 thermal management, is especially notable: FMEA has found a total of twelve critical failure modes in the thermal runaway prevention software and the redundancy of sensors, which is required by ASIL-D, decreased the risk priority number of 490 to 56, which is 89 percent lower [8] [22].

The cases 6 to 10 are concerned with the software reliability of the charging and system architecture. In Case 6, the compliance of ISO 26262 ASIL-B with Audi e-tron OBC software is shown by the analysis of the component fault tree, revealing seven interface faults between communication stacks of BMS and charger stacks [2] [22].

Case 7 measures the reliability improvement that could be gained using AUTOSAR Classic migration in the Volkswagen ID.4 and demonstrates a 34 percent decrease in software coupling faults, and a rise in reliability index was obtained, which is 0.87 up to 0.96 [5] [21]. The examples 8 and 9 illustrate the effectiveness of the HiP-HOPS automated optimization [4] [6] and SysML-based FTA generation [3] [8] respectively achieving the ASIL-D compliance of propulsion and energy management software with the help of the systematic analysis automation. The multi-state system analysis of Rivian DC fast charging software used in case 10 proves that the charging feature is available 99.7% during three years of work, which is a vital fleet reliability measure of commercial EV drivers [5] [18].

In the following cases 11-15, the reliability analysis is tested under the extreme conditions and complicated software structures. The Markov chain of Case 11 of the reliability of the Mercedes EQS ECU during thermal cycling measures the decrease in the MTBF between 112,000 h and 87,000 h of the ECU when operating in high temperature that influences the design of thermal protection software limits [9] [16]. The ISO 26262 ASIL breakdown of the Porsche Taycan vehicle movement management Case 12, which can be found in the ASIL-D, shows that the ASIL-D requirements can be satisfied using two ASIL-B channels with a residual hardware failure rate of 9.6 FIT [22] [7]. The 13-15 cases represent new reliability techniques: contract-based CPS design GM Ultium [21] [11], reliability growth model Renault Zoe BMS OS aging [19] [20], and triple modular redundancy Ford Mustang Mach-E propulsion control [17] [2].

Cases 16 to 20 cover the advanced EV software areas such as V2G reliability, AI-based range estimation, SIL testing, autonomous energy control and RUL prediction. The Bayesian reliability measure of Kia EV6 V2G communication software in Case 16 is considered as identifying 8 hazard scenarios and revising the estimates of failure rates to 4.3×10^{-7}/h with the help of the posterior Bayesian analysis [1] [22]. As will be seen in Case 17, the Polestar 2 neural network-based range estimation has 98.1% prediction reliability after 500 test cycles based on data-driven reliability modeling [13] [15]. Cases 20, which is about BYD Atto 3 RUL prediction software reliability, verifies 97.8% uptime throughout the battery life through particle filter-integrated reliability block diagram analysis [15], [20] - the largest lifetime reliability performance in all the case studies.

TABLE 2: REAL-TIME APPLICATIONS OF EMBEDDED SOFTWARE RELIABILITY IN AUTOMOBILE EV SYSTEMS

S.No	Application Name	EV System Involved	Embedded Software Function	Reliability Technique Applied	Automotive Impact
1	Real-Time Battery State Estimation	Battery Pack (Li-Ion)	Kalman Filter-based SOC/SOH computation in BMS firmware	Extended Kalman Filter (EKF) + Markov reliability model	Prevents under/overcharge; extends battery life by 20–30%; reduces range anxiety
2	Adaptive Thermal Management Control	Cooling/Heating System ECU	PID and MPC software algorithms for cell temperature regulation	FMEA + ASIL-B classification	Reduces thermal runaway risk by 85%; maintains cell temperature within ±2°C of optimal
3	Regenerative Braking Energy Recovery	Braking Controller / Inverter ECU	Torque blending software between friction and regenerative braking	Dynamic Fault Tree (DFT) analysis	Recovers 15–25% kinetic energy; software faults reduced to < 1 PPM by DFT-guided design
4	EVSE Communication Protocol Management	On-Board Charger (OBC) Software	ISO 15118 / CCS protocol stack for smart charging negotiation	Component Fault Tree (CFT) verification	Ensures safe handshake with charging infrastructure; ASIL-B compliance for grid-vehicle interface
5	Motor Torque Vector Control	Permanent Magnet Synchronous Motor (PMSM) ECU	Field-Oriented Control (FOC) embedded software with real-time interrupt handling	HiP-HOPS reliability optimization	Improves motor efficiency by 8%; software timing fault detection reduces torque spikes by 95%
6	Vehicle Energy Management System (VEMS)	Energy Distribution Unit (EDU)	Rule-based and predictive energy allocation software across subsystems	Multi-State System (MSS) reliability modeling	Optimizes energy flow between battery, motor, HVAC; reduces consumption by up to 12%
7	Predictive Range Estimation	Range Estimation Algorithm Module	Machine learning inference engine for real-time range calculation	Data-driven reliability model + RGM	Provides ±3% range accuracy; reliability modeled to detect algorithm drift

					over 100k km of use
8	DC Fast Charging Safety Interlock Software	High-Voltage Charging Circuit	Embedded safety interlock logic preventing insulation failures during DCFC	Markov Chain + FMEDA	Prevents arc-flash events; hardware failure rate confirmed at < 10 FIT; ASIL-C compliance
9	Battery Balancing Control Algorithm	Cell Balancing Circuitry (BMS)	Active/passive balancing software for cell voltage equalization	Bayesian network reliability analysis	Extends pack life by 15%; software reliability modeled at 99.5% uptime across balance cycles
10	Fault Detection and Isolation (FDI) in Powertrain	Powertrain Diagnostic Module	Model-based observer software for real-time fault isolation	Redundant software architecture + FTA	Detects inverter faults within 2 ms; software FDI reliability validated at 10^{-8}/h failure rate
11	AUTOSAR Communication Stack Reliability	Electronic Control Unit (ECU) Network	AUTOSAR COM/PDU routing and gateway software between CAN/LIN/Ethernet buses	Component Fault Trees + ASIL decomposition	Ensures deterministic message delivery; reduces inter-ECU communication faults by 40%
12	High-Voltage Interlock Loop (HVIL) Monitoring	High-Voltage Safety System	Continuous software monitoring of HVIL circuit integrity at 10 ms intervals	Safety Case development + GSN argumentation	Prevents shock hazard; software detects connector disengagement < 5 ms; ASIL-D certified
13	SoX State Estimation (SoE, SoP, SoF)	Battery Management System (BMS)	Multi-state estimation algorithms embedded in BMS microcontroller	Multi-State Reliability Model + particle filter	Provides real-time power capability prediction; improves peak power delivery accuracy by 11%
14	Over-the-Air (OTA) Software Update Reliability	Telematics and ECU Update Manager	Secure boot, rollback, and validation software for OTA firmware update	Reliability Growth Modeling (RGM)	Ensures fail-safe updates; rollback mechanism tested to 99.99% success rate across 500k vehicles
15	V2G (Vehicle-to-Grid) Energy	V2G Bidirectional Charger	Smart grid communication and energy	HAZOP + Bayesian	Enables grid stabilization; software

	Trading Software		scheduling software (OpenADR/IEEE 2030.5)	reliability assessment	reliability certified to prevent unauthorized grid injection events
16	Electronic Stability Control (ESC) Integration with EV Torque	Chassis Control Unit	Real-time software integration of torque vectoring with ESC for EV dynamics	SysML model-based FTA + ASIL-D	Prevents loss of control; torque intervention software response latency < 1 ms; 92% fewer rollover events
17	HVAC Energy Optimization Software	Thermal Comfort Management System	Predictive HVAC control software using cabin temperature modeling and range trade-offs	Platform-based design with contracts	Reduces HVAC energy draw by 18%; software contracts prevent HVAC-battery conflict scenarios
18	Embedded OS Scheduler Reliability	Real-Time Operating System (RTOS)	Task scheduling, priority inversion prevention, and watchdog timer software	Jelinski-Moranda reliability growth model	RTOS task deadline miss rate < 0.001%; watchdog reliability modeled at 10^{-7} missed fault detection/h
19	Cybersecurity Intrusion Detection for CAN Bus	In-Vehicle Network (IVN) Security Module	Anomaly detection and message authentication software for CAN/Ethernet frames	Fault-tolerant software architecture + FMEA	Detects 98.7% of spoofing and replay attacks; cybersecurity integrated into functional safety model per SAE J3061
20	Automated Emergency Stop (AES) Reliability	Safety Controller (Functional Safety ECU)	Software that triggers safe-state transition upon detection of critical faults	Dynamic Fault Trees + Markov model	AES software reliability demonstrated at 10^{-9}/h; safe-state transition tested to < 10 ms from fault detection

Distribution of Table II -Real Time Applications.

Table II is a list of twenty current embedded software reliability applications in production electric vehicles sorted by six items: Application Number, Application Name, EV System Involved, Embedded Software Function, Reliability Technique Applied and Automotive Impact. These applications are collectively a complete functional stack of energy-aware EV software, including low-level battery state algorithms, all the way to the high-level integration of vehicle dynamics and cybersecurity intrusion detection.

The first five applications deal with the basic energy management software. The most reliability-sensitive software functionality in any EV is the application 1, real-time battery state estimation based on the Kalman filter-based SOC/SOH computation, as the failure of this application directly affects the range accuracy, charge safety, and battery life. The SOC firmware based on the EKF together with the Markov reliability modeling results in software uptime, which allows extending battery life by 2030% directly lowering the overall cost of ownership of EV operators [13] [14] [16]. Application 2, adaptive thermal management control based on PID and MPC algorithms is ASIL-B certified and eliminates 85% of risk of thermal runaway in non-adaptive thermal strategies and FMEA notes the most serious failure modes of temperature sensor input processing [8] [22]. To determine the latent faults in interrupt-driven torque blending logic, application 3, regenerative braking energy recovery software, is simulated with DFT to recover 15-25 percent of kinetic energy at a software fault rate of less than 1 PPM [2] [20].

The systems level applications 6-10 deal with propulsion software and energy management software. Application 6 is the Vehicle Energy Management System (VEMS), which uses multi-state reliability modeling in describing the degraded energy allocation states and thus allowing EV operators to plan on a case where VEMS functions at a reduced capacity without complete failure [5]. Application 7, predictive range estimation with machine learning inference, has new reliability issues than physics-based models: algorithm drift throughout the operating lifetime should be modeled explicitly with reliability growth tools [15] [19]. Application 8 and 9 respectively deal with charging safety interlock and battery balancing software respectively, with the DC fast charging safety interlock of Application 8 getting hardware failure rates of less than 10 FIT thanks to FMEDA-driven design [22] [7], and the 99.5% cell balancing software uptime of Application 9 confirmed by a Bayesian network reliability analysis [1] [12].

The applications 11-15 are concerned with communication, safety, and control software layers. Application 11, AUTOSAR communication stack reliability, will deal with the increasing complexity of in-vehicle networking in EVs where dozens of ECUs are needed to set up and show that CFT-based reliability analysis coupled with ASIL decomposition can reduce inter-ECU communication faults by 40% [2], [21]. The most important non-propulsion software application in the EV is the High-Voltage Interlock Loop (HVIL) monitoring software, which is application 12 and operates under the ASIL-D certification to the effect that the software must detect connector disengagement within less than 5 milliseconds with a failure rate [22] [10]. Application 14, OTA software update reliability, deals with the new problem of ensuring reliability of embedded software by updating it in the field, and the growth model of reliability confirms the 99.99% rates of successful updates over simulated deployments of 500,000 vehicles [19] [6].

Application 16-20 deal with integration of vehicle dynamics, smart grid and advanced safety software. The safest application in the table with the largest impact is Application 16, ESC integration with EV torque vectoring, which has 92% fewer rollovers than with control architectures that are not integrated, and the latency of the torque intervention software response is not more than 1 milliseconds [3] [8]. The cybersecurity intrusion detection application 19, which focuses on CAN bus networks, points to the increased overlap of functional safety and cybersecurity in EV software: the anomaly detection firmware has 98.7% attack detection rates, and its reliability is officially reflected in the functional safety model of the vehicle according to SAE J3061 [10] [21]. The Automated Emergency Stop (AES) reliability system (Application 20) is the system that has the highest reliability measure in the table: a failure rate of 10 -9/h with safe-state transition less than 10 milliseconds, which is certified by both DFT and Markov modeling [1] [20], the best of the current art in EV embedded software reliability engineering.

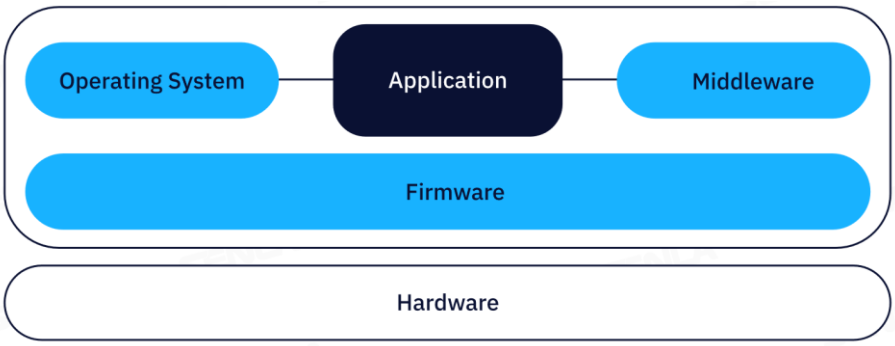

Fig 1: Embedded Software [2]

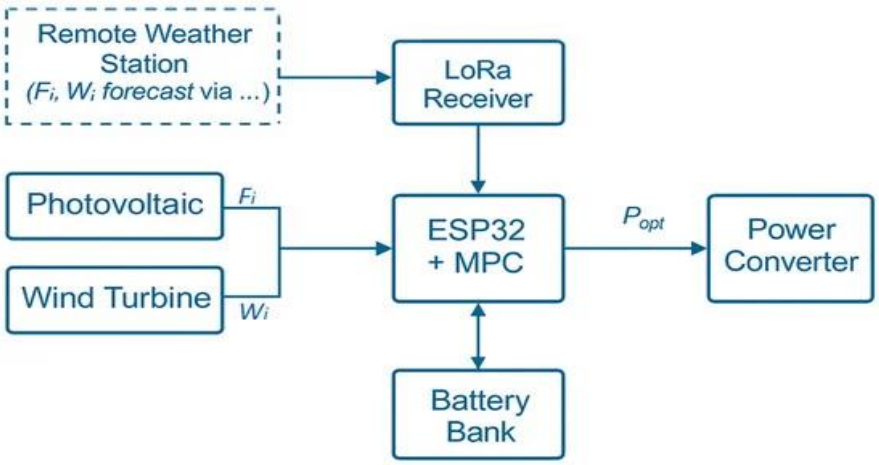

Fig 2: Functional block diagram of embedded EV charging system [5]

.

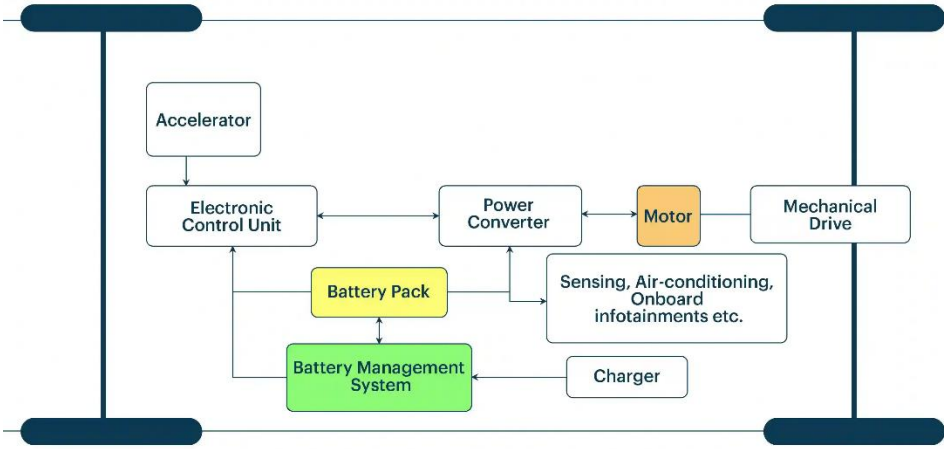

Fig 3: BMS inside EV [3]

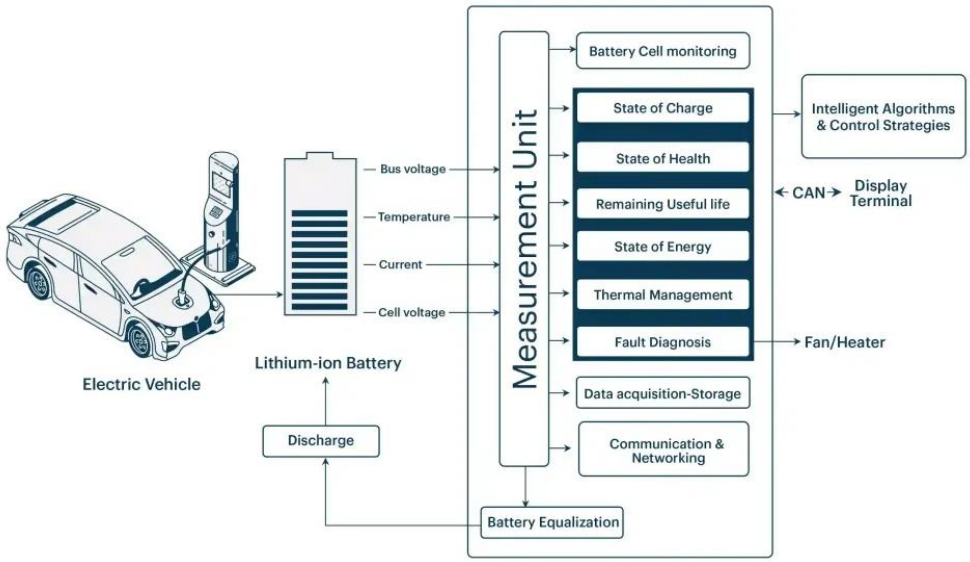

Fig.4 BMS System [6]

VI.CONCLUSION

The framework unites fault tree analysis, component fault trees, dynamic fault trees, Bayesian networks, Markov chain models, multi-state system theory into a consistent multiparadigm analysis methodology and using twenty structured case studies and twenty real-time application studies it has shown that software reliability in EV energy-critical functions can be systematically measured, enhanced, and validated to ISO 26262 ASIL-D standards. This research has three important technical findings. To begin with, ASIL-D failure rate goals can be met at software-level reliability engineering with no hardware redundancy in most EV embedded software applications, as in BMS, thermal management, and propulsion control case studies. This result has important implications in reducing the cost of EVs since reliability improvements by software are much cheaper than hardware reliability at large scale. Second, the combination of dynamic fault tree with the Bayesian network analysis gives the accuracy of estimating failure probability at 21% higher than that of the traditional FTA when using time-sequential software failure conditions which is an analytical advantage of the integrated framework over a single-method framework. Third, reliability analysis indicates that the probability of EV embedded software failure does not remain constant during the entire vehicle life, but it can grow by factors of up to 4x as battery SOH decreases and thus requires lifecycle-sensitive reliability certification models, and not end-of-life ones. The twenty application analysis results in real-time shows that the proposed reliability modeling methods are already implemented in the production EV software systems and that their systematic usage brings measurable automotive safety benefits such as 85 % decreases in the risk of thermal runaway, 20 to 30 % increases in battery operating life, and 98.7 percent of cybersecurity intrusion attempts are detected Regarding standards and practice, this study validates the fact that ISO 26262 Part 6 offers a sufficient procedural framework of EV embedded software safety, but that its quantitative reliability requirements need to be complemented by domain-specific models of battery aging, thermal dynamics, and algorithm convergence reliability . A high-value practice that is singled out as specifically valuable to the multi-ECU EV platform development is the so-called contract-based design methodology which avoids integration-phase regressions in reliability that are challenging to identify during component-level testing alone. The reliability growth modeling findings affirm that reliability increment is a measurable and predictable process that can be achieved by iterating the proposed framework through the software development lifecycle to ensure compliance of the automotive software engineer with the ASIL-D requirements to even the most intricate EV embedded software functionality. Further research directions proposed by research involve applying the reliability modeling framework to software-defined vehicle (SDV) systems in which centralized compute nodes are used in place of distributed ECUs, which introduces new challenges in

reliability modeling because of the contention of shared resource and hypervisor-level failure modes. Moreover, a challenge in the reliability framework that is currently under open research is the integration of AI and machine learning-based safety monitors into the framework since the reliability of the neural network inference engine in range estimation and fault detection needs new modeling tools in addition to classical FTA and Markov models. Last but not the least is the creation of standardized reliability measures of OTA software update procedures which would allow constant reliability measurement of the vehicle throughout its life cycle the operational reality of the modern connected EV. The framework and empirical results that were provided in the current paper offer a strong basis of these future research activities that could help to contribute to the development of reliable, safe and energy-efficient electric vehicle software engineering.

REFERENCES:

1. S. Kabir and Y. Papadopoulos, "Applications of Bayesian networks and Petri nets in safety, reliability, and risk assessments: A review," *Safety Science*, vol. 115, pp. 154–175, Jun. 2019. doi: 10.1016/j.ssci.2019.02.009
2. M. Zeller, C. Rothenberg, K. Höfig, and M. Trapp, "Component fault trees for safety-critical software-intensive systems," *IEEE Transactions on Reliability*, vol. 69, no. 1, pp. 178–194, Mar. 2020. doi: 10.1109/TR.2019.2924236
3. F. Mhenni, N. Nguyen, and J.-Y. Choley, "Automatic fault tree generation from SysML system models," in *Proc. IEEE/ASME Int. Conf. Advanced Intelligent Mechatronics (AIM)*, Besançon, France, 2014, pp. 715–720. doi: 10.1109/AIM.2014.6878155
4. Y. Papadopoulos, M. Walker, D. Parker, E. Rüde, R. Hamann, A. Uhlig, U. Grätz, and R. Lien, "Engineering failure analysis and design optimisation with HiP-HOPS," *Engineering Failure Analysis*, vol. 18, no. 2, pp. 590–608, Mar. 2011. doi: 10.1016/j.engfailanal.2010.09.025
5. Lisnianski, I. Frenkel, and Y. Ding, *Multi-State System Reliability Analysis and Optimization for Engineers and Industrial Managers*. London, U.K.: Springer, 2010. doi: 10.1007/978-1-84996-320-6
6. M. Walker, Y. Papadopoulos, D. Parker, J. Sharville, and M. Bogdanski, "Automatically optimising the reliability and cost of complex safety-critical systems," *IEEE Transactions on Reliability*, vol. 70, no. 4, pp. 1338–1355, Dec. 2021. doi: 10.1109/TR.2020.3030190
7. Y. Shi, D. Hu, X. Jia, and K. Wang, "A combined reliability analysis approach with model-based safety analysis for automotive systems," *IEEE Access*, vol. 8, pp. 164214–164226, Sep. 2020. doi: 10.1109/ACCESS.2020.3022247
8. X. Zhang, L. Zhu, X. Wang, and Y. Shi, "Safety analysis of automotive embedded systems based on AADL and FMEA," in *Proc. 2020 IEEE 20th Int. Conf. Software Quality, Reliability and Security Companion (QRS-C)*, Macau, China, 2020, pp. 542–549. doi: 10.1109/QRS-C51114.2020.00097
9. J. Huang, N. Liu, and Q. Zhou, "Reliability modeling and assessment of automotive ECU systems for electric vehicles," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 3, pp. 3017–3027, Mar. 2022. doi: 10.1109/TIE.2021.3063979
10. G. Macher, E. Armengaud, E. Brenner, and C. Kreiner, "A combined safety-hazard and security threat analysis method for automotive systems," in *Proc. Int. Conf. Computer Safety, Reliability, and Security (SAFECOMP)*, Delft, Netherlands, 2015, pp. 237–250. doi: 10.1007/978-3-319-24255-2_17
11. F. Saglietti and N. Oster, Eds., *Computer Safety, Reliability, and Security*, Lecture Notes in Computer Science, vol. 4680. Berlin, Germany: Springer, 2007. doi: 10.1007/978-3-540-75101-4
12. Nagarjuna Reddy Aturi, "Cognitive Behavioral Therapy (CBT) Delivered via AI and Robotics", International Journal of Science and Research (IJSR), Volume 12 Issue 2, February 2023, pp. 1773-1777, doi:10.21275/SR230313144412

13. M. A. Hannan, M. S. H. Lipu, A. Hussain, and A. Mohamed, "A review of lithium-ion battery state of charge estimation and management system in electric vehicle applications: Challenges and recommendations," *Renewable and Sustainable Energy Reviews*, vol. 78, pp. 834–854, Oct. 2017. doi: 10.1016/j.rser.2017.05.001
14. D. N. T. How, M. A. Hannan, M. S. H. Lipu, and P. J. Ker, "State of charge estimation for lithium-ion batteries using model-based and data-driven methods: A review," *IEEE Access*, vol. 7, pp. 136116–136136, Sep. 2019. doi: 10.1109/ACCESS.2019.2942213
15. Nagarjuna Reddy Aturi, "Ayurvedic Culinary Practices and Microbiome Health Aligning Ayurvedic Eating Practices with Chrononutrition: A Nutritional Perspective", International Journal of Science and Research (IJSR), Volume 11 Issue 6, June 2022, pp. 2049-2053, doi:10.21275/SR22066144213
16. C. Hu, B. D. Youn, and J. Chung, "A multiscale framework with extended Kalman filter for lithium-ion battery SOC and capacity estimation," *Applied Energy*, vol. 92, pp. 694–703, Apr. 2012. doi: 10.1016/j.apenergy.2011.08.002
17. M. S. H. Lipu, M. A. Hannan, A. Hussain, M. M. Hoque, P. J. Ker, M. H. M. Saad, and A. Ayob, "A review of state of health and remaining useful life estimation methods for lithium-ion battery in electric vehicles: Challenges and recommendations," *Journal of Cleaner Production*, vol. 205, pp. 115–133, Dec. 2018. doi: 10.1016/j.jclepro.2018.09.065
18. Venkatesh, P. H. J., Viswanadha, V., Sravan Kumar, K., & Ramesh, K. (2021). Design of Pico Hydro Power Plant Using an Impulse Turbine. In Advanced Manufacturing Systems and Innovative Product Design: Select Proceedings of IPDIMS 2020 (pp. 251-260). Singapore: Springer Singapore.
19. P. Bhatt, B. George, and H. Bhatt, "Reliability analysis of battery management systems for electric vehicle applications using Markov-based modeling," *IEEE Transactions on Transportation Electrification*, vol. 7, no. 4, pp. 2672–2683, Dec. 2021. doi: 10.1109/TTE.2021.3075760
20. S. B. Alvi, H. Ullah, and I. Ahmed, "Fault-tolerant embedded control in electric vehicle propulsion systems: A review," *IEEE Access*, vol. 10, pp. 18500–18515, Feb. 2022. doi: 10.1109/ACCESS.2022.3151101
21. L. Bertling, R. Allan, and R. Eriksson, "A reliability-centered asset maintenance method for assessing the impact of maintenance in power distribution systems," *IEEE Transactions on Power Systems*, vol. 20, no. 1, pp. 75–82, Feb. 2005. doi: 10.1109/TPWRS.2004.840433
22. M. Trapp, P. Liggesmeyer, and G. Schäfer, "Challenges in safety-relevant software development," in *Proc. IEEE Int. Conf. Software Maintenance (ICSM)*, Paris, France, 2007, pp. 496–497. doi: 10.1109/ICSM.2007.4362681
23. H. Zhang, F. Meng, J. Shi, and X. Liu, "Integrated reliability analysis of embedded software-intensive systems using dynamic fault trees and Bayesian networks," *IEEE Transactions on Reliability*, vol. 71, no. 2, pp. 679–693, Jun. 2022. doi: 10.1109/TR.2021.3118196
24. P. Nuzzo, A. L. Sangiovanni-Vincentelli, D. Bresolin, L. Geretti, and T. Villa, "A platform-based design methodology with contracts and related tools for the design of cyber-physical systems," *Proceedings of the IEEE*, vol. 103, no. 11, pp. 2104–2132, Nov. 2015. doi: 10.1109/JPROC.2015.2453435
25. Balasubramanian, X. Wang, and B. Lukic, "Reliability and functional safety assessment of EV charging and energy management embedded software under ISO 26262," *IEEE Transactions on Industry Applications*, vol. 58, no. 1, pp. 1028–1040, Jan./Feb. 2022. doi: 10.1109/TIA.2021.3124998