DESIGN THINKING

SUDARSHANA KARKALA | EV.ENGINEER | carsoftwaresystems@gmail.com | +91 9845561518 | LinkedIn | CAR SOFTWARE SYSTEMS

INTRODUCTION

What is Design Thinking?

Design Thinking is a problem-solving methodology that focuses on user-centric solutions through iterative processes. It is widely used in product development, engineering, and software design to ensure innovative, feasible, and scalable solutions.

In EV Battery Management System (BMS) Software Development, Design Thinking helps:

- Understand user needs (EV drivers, fleet operators, service providers).
- Create scalable and modular software architectures.
- Improve battery performance, safety, and energy efficiency.
- Integrate AI-driven diagnostics and predictive maintenance.

Design Thinking - Customer Centric Approach

Design Thinking revolves around understanding and prioritising the needs of the end-user.
It promotes a human-centred approach in software and hardware design by:

- Focusing on user empathy to identify real-world problems.
- Encouraging cross-disciplinary collaboration to generate holistic solutions.
- Leveraging rapid prototyping to validate solutions early in the development process.

Five-stage Design Thinking process in BMS Software Development

Design Thinking is an iterative, non-linear process that consists of five key phases:

- **Empathise** – Understanding user needs and real-world challenges.
- **Define** – Clearly defining software and hardware problems.
- **Ideate** – Generating innovative ideas for BMS solutions.
- **Prototype** – Developing working models for testing.
- **Test** – Validating with real-world user feedback.

STAGE 1: EMPATHIZE – UNDERSTANDING USER NEEDS IN EV BMS

Why Empathy Matters in BMS Software?

- Battery failures impact vehicle safety and longevity.
- EV users demand accurate State of Charge (SOC) and State of Health (SOH) estimations.
- Manufacturers need cost-effective, scalable BMS solutions.

User Research & Data Collection

- Driver Interviews – Understanding range anxiety, charging habits, and expectations.
- Fleet Operator Surveys – Evaluating large-scale fleet maintenance issues.
- Sensor Data Analysis – Collecting thermal, voltage, and current readings for real-world performance insights.

Example - User-Centric BMS Challenges:

- Problem: Drivers find SOC estimates inaccurate.
- Empathy-Driven Insight: Integrating AI-based learning models to enhance SOC accuracy based on driver habits.

STAGE 2: DEFINE – IDENTIFYING CORE BMS CHALLENGES

How to Define the Right BMS Problem?

- Analyse Software-Hardware Interaction – Identify bottlenecks in real-time communication between BMS and EV control units.
- Define Safety & Compliance Requirements – Ensure alignment with ISO 26262, IEC 61508, and ASPICE standards.

Defining Key Problem Statements in EV BMS:

- SOC Inaccuracy – Existing models fail in extreme temperatures.
- Thermal Management Limitations – Lack of proactive cooling system adjustments.
- Battery Swapping Integration – Software not optimised for modular battery packs.

Example - Well-Defined BMS Problem:

- Problem Statement: “EV users experience sudden drops in range due to inefficient SOC prediction. How can we enhance real-time SOC accuracy?”

STAGE 3: IDEATE – GENERATING INNOVATIVE BMS SOLUTIONS

Creative Techniques for BMS Software Innovation

- Brainstorming Sessions – AI-based diagnostics, cloud-based BMS analytics.
- SCAMPER Method – Modify existing SOC algorithms to incorporate real-time environmental factors.
- Benchmarking – Study Tesla, BYD, and Rivian BMS approaches for best practices.

Example - Ideation for SOC Prediction Improvement:

- Solution: Implementing a machine-learning model that adapts SOC calculations based on weather, terrain, and driving behaviour.

STAGE 4: PROTOTYPE – DEVELOPING BMS SOFTWARE MODELS

Types of Prototyping in BMS Software:

- Digital Twin Simulation – Testing BMS algorithms in a simulated EV environment.
- Hardware-in-the-Loop (HIL) Testing – Real-time validation using an actual BMS controller.
- Rapid UI Mockups – Designing intuitive interfaces for driver interaction with battery analytics.

Example - BMS Prototyping Process:

- Develop AI-driven SOC algorithm in MATLAB/Simulink.
- Run edge-case simulations using Python-based battery modelling.
- Deploy firmware on test vehicle BMS to validate real-world accuracy.

STAGE 5: TEST – VALIDATING BMS SOFTWARE PERFORMANCE

Key Testing Metrics for BMS Software

- SOC & SOH Accuracy – Comparison with real-world charge cycles.
- Thermal Performance – BMS response under extreme temperatures.
- Fault Handling & Safety – Ensuring robustness against short circuits and voltage surges.

Example - Real-World BMS Testing Approach:

- A/B Testing: Comparing traditional SOC algorithms vs. AI-enhanced models.
- Field Testing: Running BMS software on 10 different EV models to ensure cross-platform compatibility.
- User Feedback: Collecting data from drivers to refine SOC accuracy and user interface.

CONCLUSION & INDUSTRY BEST PRACTICES

- User-Centric Design: Prioritise driver and manufacturer needs.
- Rapid Iteration: Test BMS software in real-world conditions early.
- AI & Cloud Integration: Leverage predictive analytics for enhanced performance.
- Compliance & Safety: Align software with ISO 26262 and ASPICE guidelines.

Industry Best Practices for BMS Software Development:

- Digital Twin Testing: Validate BMS models before real-world implementation.
- Modular Architecture: Design software that supports different battery chemistries.
- Continuous Learning Algorithms: Adapt BMS parameters dynamically over time.
- Over-the-Air (OTA) Updates: Enable real-time software enhancements.
- Cross-Disciplinary Collaboration: Work with hardware, software, and AI teams for holistic solutions.