講演番号	028

Building a Digital Thread Platform to Enable Bidirectional Design Convergence between Upstream Requirements and Downstream Specifications via MBSE

Seishi Shimamura　Haruhisa Tsuchikawa　Chisato Suzuki　Asami Hozumi　Kiyotaka Shoji
Ryuzo Noguchi　So Nishiwaki　Ryoji Katsuki

Nissan Motor Co., Ltd., Nissan Technical Center
560-2 Okatsukoku, Atsugi-shi, Kanagawa, 243-0192, Japan
(E-mail: seishi-shimamura@mail.nissan.co.jp)

KEY WORDS: **software and its underlying technologies, database, systems engineering/MBSE**

As vehicle development becomes increasingly complex due to electrification and software-intensive architectures, maintaining consistency among requirements, design specifications, and verification results has become a critical challenge. While Model-Based Systems Engineering (MBSE) provides a framework to manage these relationships, its effectiveness in practical development activities is often limited. A key reason is that requirement-fulfillment judgments remain largely tacit knowledge, relying on individual engineers' experience and manual interpretation. Consequently, even when analyses and simulations are automated, achievement-status updates frequently remain manual, creating a bottleneck in the overall development process.

To address this issue, this study proposes a digital thread platform that formalizes requirement-fulfillment judgment specifications and integrates them into a Simulation Process and Data Management (SPDM) environment. The proposed digital thread consistently links upstream requirements with downstream specifications, analysis results, and decision-making artifacts. By representing judgment specifications as data rather than implicit rules, the platform enables reproducible and consistent evaluations across the development process.

The platform consists of four core elements: a requirement list defining performance targets and constraints; a specification list managing design parameters and their alternative candidates; an achievement-status list recording OK/NG judgment results for each requirement–specification combination; and an evaluation board for cross-comparison and visualization of multiple candidates. These elements are connected through common identifiers, allowing relevant evaluations to be executed upon an engineer's execution command based on updated specifications, with results reflected along the digital thread. Fig. 1 shows the SPDM-based system configuration with data-defined judgment specifications and the relationships among these four core elements.

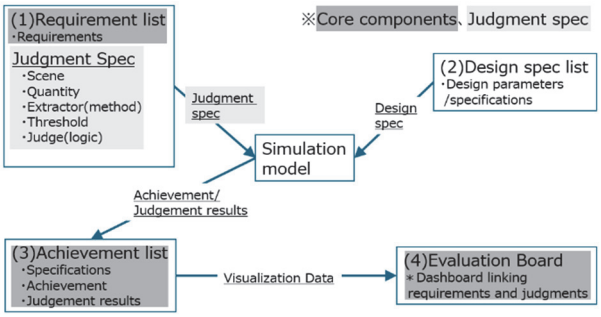

Fig. 1 SPDM-based system configuration with data-defined judgment specifications.

A key feature of the proposed approach is the formalization of judgment specifications. Each requirement is associated with structured data describing the evaluation condition (Scene), referenced physical quantity (Quantity), extraction logic applied to simulation outputs (Extractor), acceptance criteria (Threshold), and decision logic (Judge). Unlike conventional approaches that embed judgment logic directly within programs, the proposed method treats judgment specifications as reusable and manageable data assets.

Based on this concept, we implemented "Auto-Gate" as a mechanism for updating achievement levels. Auto-Gate executes managed evaluation specifications and simulation models based on instructions from the person in charge. It then applies extraction and evaluation logic to update the achievement level and reflects the results in the achievement level list and evaluation dashboard. This enables efficient evaluation of multiple design proposals while reducing the risk of missed re-evaluations that often occur when specifications change. Fig. 2 shows the operational data items and workflow of Auto-Gate.

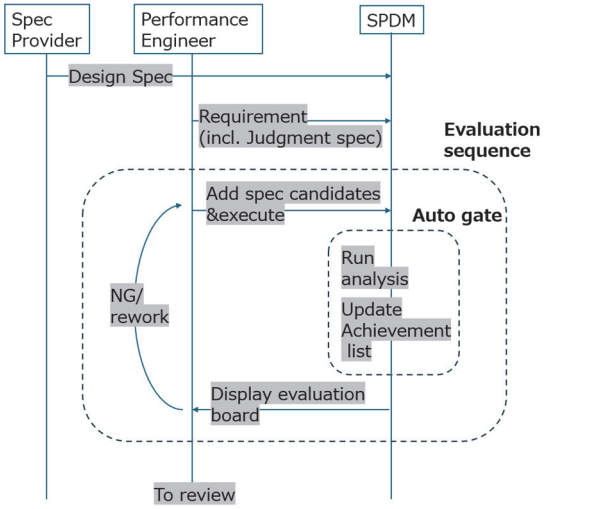

Fig. 2 Operational data items and workflow of Auto-Gate