

A Structured Approach for Function Analysis of Complex Automotive Systems

2011-01-1268

Published
04/12/2011

Ioan F. Campean, Edwin Henshall, David Brunson and Andrew Day
University of Bradford

Rod McLellan and Joseph Hartley
Tata Motors European Technical Centre

Copyright © 2011 SAE International

doi:[10.4271/2011-01-1268](https://doi.org/10.4271/2011-01-1268)

ABSTRACT

Function analysis provides the backbone of systems engineering design and underpins the use of Design for Six Sigma and Failure Mode Avoidance tools. Identification and management of interfaces is a key task in systems engineering design, in ensuring that the system achieves its functions in a robust and reliable way. The aim of the work presented in this paper was to develop and implement a structured approach for function analysis of a complex system, which focuses on the identification and characterization of interfaces. The proposed approach is based on the principle of separation of the functional and physical domains and development of function decomposition through iteration between functional and physical domains. This is achieved by integrating some existing / known engineering tools such as Boundary Diagram, State Flow Diagram, Function Tree and an enhanced interface analysis within a coherent flow of information. The approach is illustrated with a case study on system level design analysis of an electric powertrain for a full electric light commercial vehicle.

INTRODUCTION

Within a customer focused engineering approach, engineering design is directed towards *robust* and *reliable* delivery of customer required *functions*. Thus function analysis can be seen to provide the backbone of systems engineering design, through all stages of design and development. At the product definition stage customer needs and wants are converted into engineering functions, for which target specifications are established and documented in a

System Design Specification. Subsequent function decomposition and analysis is an essential part of concept generation and selection, and it provides the basis for the requirements cascade down to subsystem and components, where the design decisions are made and the design is generated. Within design verification the aim is to prove that the design can *robustly and reliably* achieve the required functions, i.e. that the design is free from function failure modes.

Function decomposition and analysis is also at the core of Design for Six Sigma (DfSS), where the key aim is to develop transfer functions relating critical functional requirements of the system (Ys), to key product design (x) and process (p) variables (1). These transfer functions are then used for robust optimization of the design, i.e. to select a design configuration with optimal parameter specification that would enable consistent achievement of the function under customer operating conditions.

The function analysis task for a complex system such as a vehicle is compounded by the simultaneous action of two factors:

- Functional complexity of the system itself - many functions being required at the system level, which have to be delivered within a compact and cost effective package;
- Highly complex “noise” space, as a result of a highly variable environmental and operational duty cycle, complex interactions between subsystems, and manufacturing variability inherent to high production rates.

Given the complexity of interaction between subsystems the focus of the systems engineering design task should be to

ensure that *all* system interfaces are identified, characterized and managed through function requirements. Such a detailed analysis is resource intensive, and hence very challenging within the normal time and budget constraints of a vehicle development program. However, the consequences of not carrying out this analysis thoroughly are also well known in industry: failure modes which are discovered late in the development program, requiring *countermeasures*, which are both technically difficult, given the design change constraints, and costly. The late development and implementation of countermeasures has a significant adverse effect on the product development factory efficiency.

The Failure Mode Avoidance framework (2,3,4) is one of the strategies emerging from the automotive industry as a way of enhancing the effectiveness of the product development process through the early identification of potential failure modes and the development of countermeasures. The Failure Mode Avoidance framework is underpinned by engineering tools such as Failure Mode and Effects Analysis (FMEA) (5), Fault Tree Analysis (FTA) (6) and P-Diagram (7), which provide a structure for engineering thinking for potential function failure mode identification. However, all of these tools require function analysis as a pre-requisite; for example the FMEA requires that system functions must be identified first (5), while the Function Fault Tree Analysis (6) requires the development of a function tree first.

With its roots in Value Engineering (8) the traditional practical engineering approach to function analysis is largely based on brainstorming. This is reflected widely in the literature providing guidance on FMEA development (9, 10) which generally explicitly includes the use of brainstorming to identify functions. Even where a more structured approach to function analysis is recommended such as the use of a Function Tree (11) this is typically generated through brainstorming and tends to focus on the main functions of the system, with no mechanism to ensure that *all* supporting functions are identified. The supporting functions are generally needed to manage exchanges at the interfaces both within and external to the system and so ensure that the system achieves its functional requirements in a robust and reliable way. If a function is not identified, then there will be no specification associated with it, and so the function will not be engineered in, resulting in a failure mode down the line. Field experience confirms that the overwhelming majority of reliability problems originate from the failure of supporting functions resulting in the function failure of the system. For example, Webb quoted a study of failures prompting vehicle recalls over a period of 10 years which showed that over 80% were directly associated with the management of interfaces (12), i.e. with failures of supporting functions.

Other structured tools such as the Boundary Diagram and Interface Matrix have been introduced to support the function

analysis of a complex system (12, 13). Boundary diagrams provide a concise graphical representation of the system, identifying the boundaries and the external interfaces of the system, as well as the relationships between the internal subsystems and components. The Interface Matrix is a powerful interface analysis tool, which aims to systematically identify interfaces and exchanges at interfaces both between the system being designed and neighboring systems and the environment (external interfaces), as well as exchanges between subsystems inside the boundary (internal interfaces). The Interface Matrix can also be used to evaluate the effect of external and internal interfaces on system function (12). While these tools support the development of a good understanding of the way in which the system achieves its function, their use is inconsistent in engineering practice, with the tools not being efficiently integrated in an information flow that supports the engineering design process; e.g. the information gained from interface analysis is not necessarily used to update the function tree. Given the low integration of the tools in a coherent information flow, the engineering interest for using these tools on a broad basis is also low.

The aim of this paper is to present a disciplined and rigorous approach for function analysis of a complex automotive system that coherently integrates known engineering tools in a way that supports the system engineering design process effectively. The approach builds upon the use of existing engineering tools by introducing an enhancement to these tools which facilitates the engineering thinking required to identify all function failure modes in a system and hence significantly increases the likelihood of producing a design which is largely failure free. The approach will be illustrated with a case study on the system level analysis of an electric powertrain for a full electric vehicle.

OUTLINE OF THE APPROACH

Based on the analysis presented above it can be stated that a coherent and efficient approach for function analysis of a complex automotive system must fulfill the following requirements:

1. Facilitate functional decomposition of the system, for the design of a completely new system this would provide support for concept design and selection. For an existing system (iterative / carryover design) it will provide a structured approach for the analysis of the system, delivering a good understanding of how the system achieves its functions, and the supporting functions requirements specification and cascade;
2. Support a systematic identification, characterization and management of interfaces through functional requirements. The aim is to identify *all* supporting functions that need to be provided in order to achieve a failure free integration of the system with robust delivery of system functions requirements;

3. Integrate with the system engineering design flow (i.e. support the requirements specification, requirements cascade, system integration and robust design verification), and with other structured processes such as Design for Six Sigma and Failure Mode Avoidance. For example, in order to integrate with other engineering tools within the Failure Mode Avoidance framework, the requirement is to provide a coherent core for the information flow to other tools such as FMEA, FT A and P-Diagram.

It is also desirable that the function analysis framework makes use of existing / known tools (such as Boundary Diagram, Interface Matrix, Function Tree) to facilitate adoption by the engineering community.

FUNCTION DECOMPOSITION

The theoretical basis for function decomposition analysis consists of the iterative mapping of functions and their solutions (sub-function structures) at increasing level of detail until a solution concept is reached (13). The axiomatic design paradigm (14) provides a useful conceptual framework in which engineering design is presented as an iterative (zigzagging) mapping between the functional domain and the physical domain. This is illustrated in Figure 1, where the customer needs and requirements (C_i) in the *Customer Domain* are mapped onto functions (F_i) in the *Functional Domain*. Within any system functions are achieved through hardware, hence the function requirements are mapped onto subsystems (S_i) within the *Physical (or Structural) Domain*. The separation between the Functional and Physical domains is essential, as it encourages engineers to focus on the functional requirements that need to be delivered by the system, rather than jumping straight to a particular design (hardware) solution. The systems engineering design cascade can be described as zigzagging iterations between the *Functional* and *Physical* domains, until a level of resolution is achieved where engineering design can be carried out (i.e. component level).

The zigzagging between the *Functional* and *Physical Domains* is also key in the iterative development of the transfer functions within DfSS, linking the *customer* perceived function (Y, or Key performance Output Variable, KPOV), to *function* requirements (y, Critical To Satisfaction, Quality, etc, i.e. CTX), and physical design parameters (x, Key Product Input Variables, KPIV).

The Pahl & Beitz (15) system black box model is commonly used to represent a system's function as illustrated in Figure 2. According to this model the system function can be defined as transformation of the input energy, material or information into the output energy, material or information. On this basis any engineering function can be defined in terms of transformation or transfer of energy, material and information. This can be a useful paradigm for the

decomposition analysis in the functional domain, by iteratively evaluating the chain or flow of transformations and / or transfers that can deliver the overall function of the system.

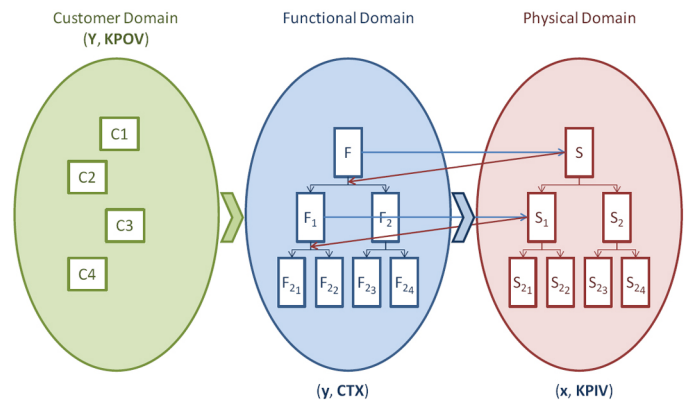


Figure 1. Functional decomposition by mapping and zigzagging between Customer / Functional and Physical Domains

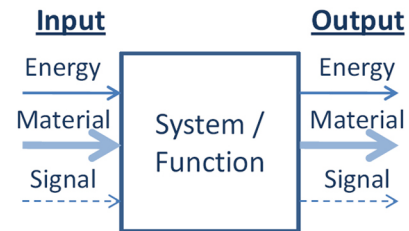


Figure 2. System function representation (15)

A particularly useful engineering tool that supports the mapping of the flow of energy / material / information between the input and the output is the System State Flow Diagram (4, 6). A State Flow Diagram is a block diagram of the system in which the blocks denote system states (referring to energy / material / information states), while arrows represent the flow or transformation of energy / information / material between system states, i.e. functions. Once the functions are identified they can be mapped onto the physical domain, by identifying subsystems / components that achieve the function. This physical mapping can then be represented as a Boundary Diagram, which is a graphical representation of the system in the physical domain, identifying subsystems together with the relationships between the subsystems in terms of flows of energy / material / information. This will be illustrated in the case study presented later in the paper.

INTERFACE ANALYSIS

Within the function decomposition analysis presented above the focus was on the achievement of the main function of the system, i.e. the flow (transformation and / or transfer) of energy / material / information between the input and output. However, within the physical domain there will be multiple

interactions between subsystems within the system, and between the system and the neighboring systems and the environment. These interactions could involve exchanges of energy / material / information, and could affect the functional performance of the system. Therefore, these interfaces need to be identified, characterized (i.e. define the type of exchange that takes place at the interface and evaluate the effect on the system function), and managed through function requirements. What this means is that a *function* must be defined to manage each interface exchange in order to ensure that a path for energy / material / information flow is provided and maximized if the exchange is beneficial to the main function of the system, or prevented / minimized if the exchange is detrimental to the main function.

From a practical point of view, it is useful for the engineering interface analysis also to identify, analyze and specify any *Physical Contact* requirements at an interface, referring to the spatial / geometry requirements and compatibility. Physical contact usually provides a path for energy / material / information exchange, and therefore it is commonly used as a first point in interface identification and analysis.

In practice interface analysis is usually managed through a matrix based tool, the *Interface Matrix* (12, 13). In its conventional form this tool is limited to the identification of interfaces between pairs of subsystems while in more advanced versions of the tool (12, 16), the interfaces are characterized by indicating the nature of the exchange (*Energy* (E), *Material* (M), *Information* (I) and *Physical Contact* (P)). An assessment of the effect of the interface exchange on the main function of the system is also indicated using a numerical scale, as shown in [Table 1](#) below (15).

Table 1. Numerical scale for evaluation of the effect of an interaction on the main function

Rating	Description
2	Necessary for function
1	Beneficial, but not essential to achieve function
0	No effect
-1	Negative effects but does not prevent function
-2	Must be prevented to achieve function

The approach developed and presented in this paper is based on an enhancement to the Interface Matrix, by introducing a more structured and formal analysis of the interfaces in which *functions* and *function requirements* are defined to manage the exchanges at an interface.

Within this framework the analysis should consider all pairs of subsystems in turn, both those within the system boundary and those external to the system. If an interface is identified between 2 subsystems, then this interface must be characterized in terms of the nature of the interface (P / E /

M / I) and its potential effect on the main function (as in [Table 1](#)), and *functions* have to be defined in relation to this exchange. The exchanges between complex systems are usually multiple and in both directions (i.e. from S_i to S_j and from S_j to S_i), therefore the analysis of each interface would typically yield several functions which may be associated with just one type of exchange or multiple types of exchange. It is useful that the function semantics reflect the interface being considered; thus, while keeping with the common guidelines (8, 11) of using a verb-noun structure for wording a function, each function should identify both systems that are related through the function; e.g. “*transfer energy from S_i to S_j* ”. It is important that these functions are documented at the time of the analysis; a tabular format is suggested as a simple tool to complement the Interface Matrix, which will be illustrated later in the paper.

From a system point of view, the functions identified to manage interfaces are *supporting* functions; their role is to ensure the system integration with the other systems, and also to ensure that the main system functions are robustly delivered.

CASE STUDY: FUNCTION ANALYSIS FOR AN ELECTRIC POWERTRAIN

The function analysis methodology proposed in this paper will be illustrated with a case study on the system level design analysis of an electric powertrain for a full electric light commercial vehicle developed for the European market.

At the time of this analysis the system was at the concept design stage; high level requirements specification were available and most technology decisions already made (e.g. that the vehicle will use conventional electrical energy storage / battery pack). However, the detailed system design and integration and subsystem design was not completed.

The purpose of the analysis was to carry out a rigorous top-down analysis to establish the function requirements cascade and to support robust subsystem design and validation within a Failure Mode Avoidance framework. While the scope for the analysis presented in this paper was limited to the high level system design, as carried out by a multi-disciplinary engineering team, it should be noted that methodology used is equally applicable to lower levels (e.g. component) of the systems hierarchy.

SYSTEM FUNCTION ANALYSIS

The system analysis started with the evaluation of the *engineering functions* that the system has to achieve. The main engineering function requirements of the system have been defined as follows: (i) to *provide controlled torque at the rear axle* and (ii) to *provide power for low voltage (LV)*