POLITECNICO DI TORINO
Collegio di Ingegneria Gestionale - Classe LM-31
Corso di Laurea Magistrale in Engineering and Management

Tesi di Laurea Magistrale

Digitalization and the migration towards modular product architectures

Empirical analysis comparing non-digital and digital product pairs through architectural metrics to assess the impact of digitalization on component coupling and modularity

Relatore:
Prof.ssa Francesca Montagna
Co-relatore:
Dott.ssa Teresa Monti

Candidato:
Tommaso Guasti
Matricola: 331450

Anno Accademico 2025-2026

Sessione di Laurea Marzo-Aprile 2026

Bibliography
Webography

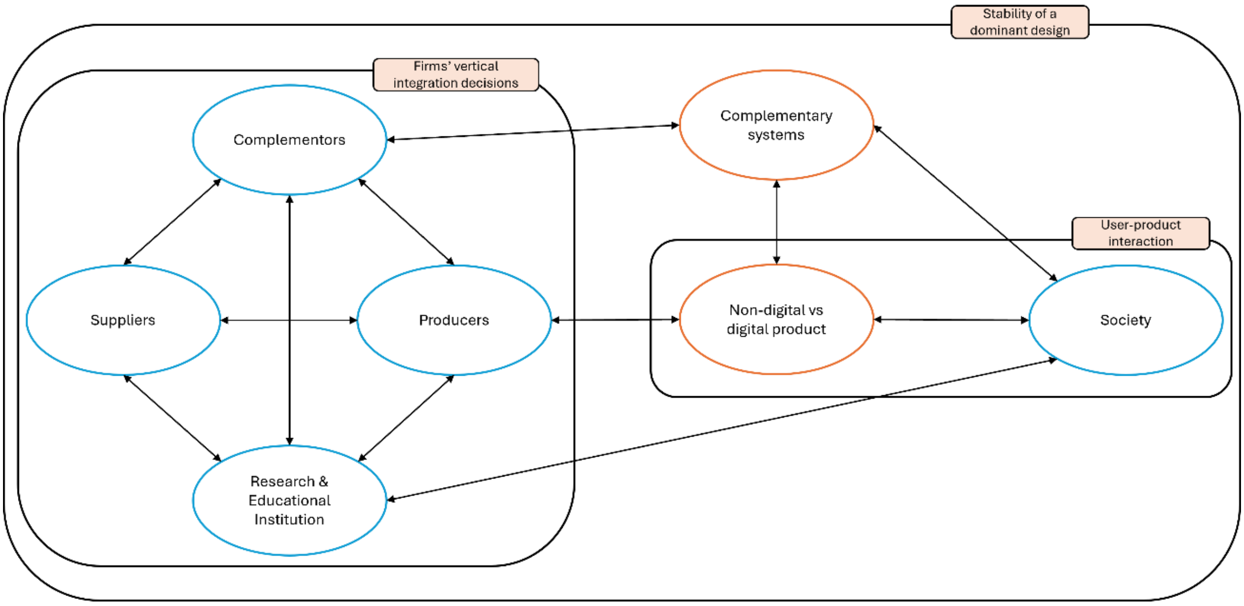
Chapter 1: Introduction

Technological innovation involves multiple actors and their interactions across the entire lifecycle of a product. These actors and their interactions constitute what Dosi termed a technological paradigm, which is a set of procedures, relevant problems and the specific knowledge related to their solution (Dosi, 1982). The technological paradigm defines the direction of technical change along a technological trajectory, being the pattern of normal problem-solving activity on the ground of a technological paradigm.

Producers design and manufacture products, but they rely on suppliers for components and materials, coordinate with complementors who develop related products or services, draw knowledge from research and educational institutions. On the demand side, products are used in combination with complementary systems, interact with users in specific ways, operate within regulatory and social frameworks established by society. Such user-product interaction influences how demand experiences the product and provides feedback that affects future design choices.

Overall, innovation takes place in this connected system, where firms make vertical integration decisions, complementary systems emerge around products and user-product interactions shape preferences and adoption patterns. Digital transformation introduces new paths of innovation that challenge established dominant designs and change the relations between products, firms and society.

Figure 1.1: Constituent elements of a technological paradigm



Source: Cantamessa and Montagna, 2023

Within this context, product architecture plays a central role in structuring innovation dynamics. Product architecture can be defined as the allocation of functions to physical components and the specification of interfaces among them (Ulrich, 1995). This allocation determines how easily a product can be modified, extended or combined with other systems and therefore influences which types of innovation are possible over time. Different architectural configurations in fact enable different forms of innovation, from incremental improvements to deeper architectural reconfigurations (Henderson & Clark, 1990).

The digitalization of physical products has emerged as one of the most significant forces reshaping product architectures within established technological paradigms, favoring flexible architectures, with clear and standardised interfaces that allow firms and complementors to innovation locally while keeping overall system integrity. While mechanical products often display a high degree of integrality, with many physical interdependencies (Fixson and Park, 2008), digitalization alters such constraints by introducing programmability and data homogenization (Yoo et al 2010).

Building on this perspective, the thesis places product architecture at the centre of the analysis to form the basis for the understanding of how innovation dynamics evolve when physical products undergo digital transformation,

The thesis uses a combination of qualitative and quantitative architectural analysis tools to empirically validate the increasing level of modularity brought by digitalization. A systematic comparison between six pairs of products allows to deeply investigate how functions are allocated to components and how extensively components are shared across functions in non-digital and digital products.

The structure of the thesis is as follows. Chapter 2 defines product architecture and presents the analytical tools used throughout the study. Chapter 3 introduces product architecture into the dynamics of innovation, focusing on dominant design and architectural change over product lifecycles. Chapter 4 examines digital transformation and discusses its influence on product architecture. Chapter 5 outlines the research methodology applied and the selected product pairs. Chapter 6 presents the results of the architectural analysis while the last chapter synthesises the findings and discusses their implications.

Chapter 2: Definition of product architecture

2.1: Architecture characteristics

Defining what product architecture is has been a topic discussed in several different articles and publications. At its core, product architecture refers to “the arrangement of functional elements and their interconnections”, which Karl Ulrich named a *function structure*. This leads to a study of a product's functions and selecting the level of abstraction that we need to adopt at the most general level a function for a product might be just the main one, but deeper analysis may identify many functional elements.

The transition from functional elements to physical components requires careful specification. A discrete physical product consists of one or more physical components. Components are the tangible embodiments of functional requirements (Gershenson et al., 2003).

Physical components are designed to implement functional elements of a product, and the mapping between them may be one to-one, many-to-one, or one-to-many (Ulrich, 1995). This mapping is not arbitrary but reflects deliberate product design decisions on how to decompose the overall product into components.

Product architecture is formally defined as “the scheme by which product architecture is allocated to physical components” (Ulrich, 1995). This allocation reflects product architecture’s design decisions made during the early stages of product development. Understanding product architecture extends beyond comprehending the way products are designed; it also covers the way they are manufactured and used during their lifecycle (Fixson 2005). The decisions made during the early design phase have cascading effects on the following phases of the product lifecycle, which covers the entire process of a product from its design phase, making architectural choices critical for the product success.

A product can be decomposed to multiple hierarchical levels. Starting with the major subsystems, which can be further decomposed into components and eventually subcomponents, at each level, the architecture specifies the functionality assigned to that level, the physical elements responsible for delivering that functionality and also the interaction point between these elements (Fixson, 2005). These interaction points, called

interfaces, may be non-contact (infrared communication link between remote control and a tv) or involving geometric connections between elements (as with a gear on a shaft).
The specification of interfaces is particularly important for managing complexity. As emphasised by Yoo Youngjin in the paper "The new organizing logic of digital innovation", the specification of interfaces enables designers to understand how components must interact without needing details about each component. Maintaining transparent interfaces while containing internal complexity is fundamental to effectively manage the product structure.

Returning to the relationship between components, when a single component contributes to different functions it is being called function sharing.
The degree of function sharing depends on the considered level of detail, which affects not only the complexity and number of components but also the degree to which a product exhibits modularity or integrality (Ulrich, 1995).

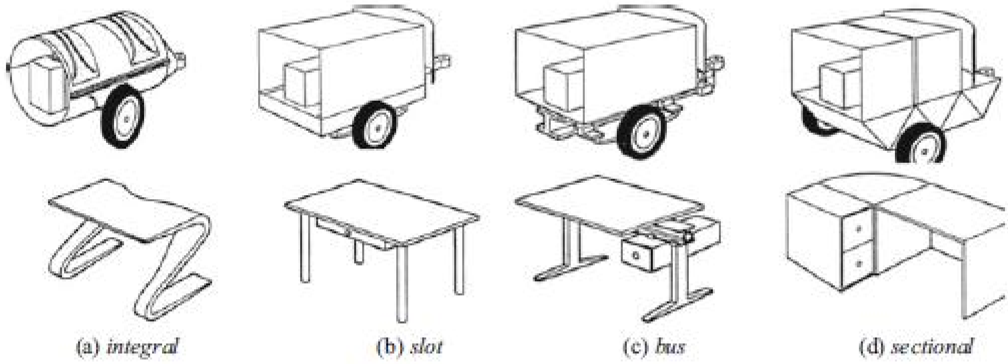
A closely related topic is the notion of coupling and decoupling. When a change to a component requires a corresponding change to another distinct component in order for the product to work correctly, those components are said to be coupled (Ulrich, 1995). Conversely, decoupled components are either unaffected by changes to other components or affected with a minimal impact.
The decoupling concept is critical to understand product modularity because of the ability to change single components without affecting other parts of the product.

Moreover, when describing product architecture, it is possible to distinguish between two categories of architecture, according to Ulrich:

- Integral architecture is characterised by a many-to-many relationship between functions and components. Each function is being fulfilled by multiple components and by components embodying multiple functions. Changes in a component often necessitate adjustments to other components, resulting in tightly coupled systems: as a result, there is not a clear one-to-one mapping between functions and components.
- Modular architecture is characterised by a near one-to-one relationship between functions and components. Each function is typically fulfilled by a distinct component, and each component is related to a specific function. Many

advantages come with this architecture, such as enabling component reuse across product variants. Also, it facilitates time reduction because of the possibility to develop in parallel by different teams (Baldwin & Clark, 1997). However, modular architecture also has some disadvantages: there is the risk of creating organisational barriers to architectural innovation, since teams become specialised in their respective modules and may resist changes that require cross-module coordination (Ulrich, 1995). Moreover, modular architectures can be implemented though different interface structures. In slot-based architectures, each module is connected to the system through a dedicated interface, which constrains where components can be integrated and limits interchangeability. Bus-based architectures rely instead on a shared interface, allowing multiple modules to connect to the same system bus and enabling higher flexibility. A further configuration is the sectional architecture, in which modules share standardized interfaces and can be freely combined without predefined positions, further increasing adaptability. These different interfaces have increasing degrees of modularity, with bus-based and sectional architectures offering greater flexibility than slot-based solutions.

Figure 2.1 Alternative architectures for two products



Source: Ulrich, 1995

The principles of modular architecture can also be studied beyond tangible goods. Furthermore, services can benefit from modularity too: according to Saara Brax et al. (2017), there are huge advantages that modularity offers in services. Examples of these advantages are personalisation, standardisation, and operational efficiency. That means that the same principles of decoupling and functional decomposition are applicable to service systems not only to physical products.

2.2 How to distinguish between integral and modular architecture

2.2.1 Qualitative methods to assess architecture

In order to understand how products are structured and the interaction between their parts can be analysed, it is essential to represent product architecture in a visual way.

Drawing the structure and the relation of a product allows to get a qualitative description of the architecture and most important to measure the degree of modularity. There are several graphical instruments useful for this purpose and this section introduces some key tools employed in architecture analysis.

A block diagram essentially serves a spatial representation of how product functions decompose into physical components (Ulrich, 1995). In this representation, each block corresponds to a physical component and the connection between blocks show how they interact to deliver the product's functionality. In this way, analysts are allowed to comprehend quickly the structure of a product and to individuate the major subsystems.

Another tool with which product architecture can be modelled are Design Structure Matrices, which offers a quantitative framework for analysing how components interact. The DSM is a square matrix that maps the dependencies of a component with respect to other components, allowing designers to assess which components are coupled and which are independent (Steward, 1981).

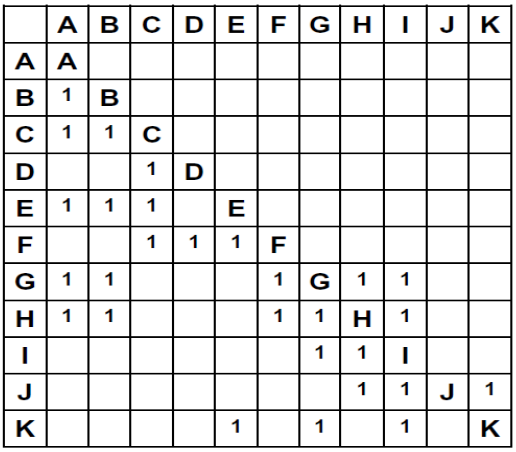
In the DSM, information about dependencies between variables is represented as a matrix following the rule "a mark in row *i,* column *j* means *i* has the predecessor *j*" (Steward, 1981).

As stated by steward, the way to read matrices is the following: "Going across the rows shows what precedes; going down columns shows what follows." This criteria allows to identify which components are coupled and which ones are independent, hence identifying modules of components.

A DSM complements block diagrams by providing quantitative feedback for analysing component interactions rather than an exclusively graphical one. Moreover, in a graphical setting, clusters are less easily identified rather than in a tabular format. By examining the matrix, one isolates blocks of *ones (*1 indicates dependency, 0 otherwise) and groups them together. These clusters represent integrated components that function as a sort of subsystem.

In conclusion, this matrix is useful to point out groups of components and potential modules, highlighting which parts of the product are highly integrated.

Figure 2.2. A DSM representation example



	A	B	C	D	E	F	G	H	I	J	K
A	A										
B	1	B									
C	1	1	C								
D			1	D							
E	1	1	1		E						
F			1	1	1	F					
G	1	1				1	G	1	1		
H	1	1				1	1	H	1		
I							1	1	I		
J								1	1	J	1
K					1		1		1		K

Source: Steward, 1981

Another essential tool for product architecture is the functional tree. It represents a hierarchical decomposition of product functions and helps map the product's components into functions. A graphical way of representing it is provided by functional trees. Unlike block diagrams, which focus just on physical components. The root of the functional tree corresponds to the core function of the product.

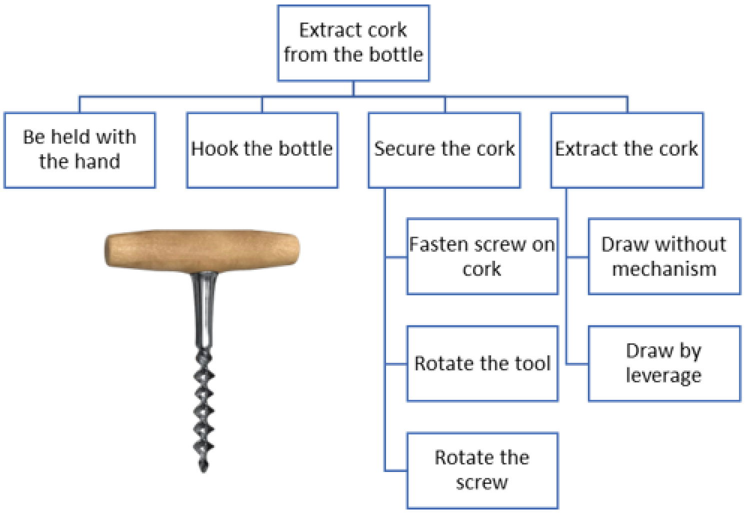
The root of the functional tree represents the core function of the product. The core function can be split into multiple functions, which are successively broken down into further detailed functions. This hierarchical decomposition continues until the chosen level of detail has been reached.

To illustrate this concept for instance, consider a corkscrew and the breakdown of its functional elements (Cantamessa and Montagna, 2023). The core function is to "extract cork from the bottle". The function can be broken down into "secure the cork", meaning enabling the screw to grip the cork, "rotate the screw" and "be held with the hand", hence enabling the user to grip the device. Each of these functions can be decomposed successively depending on the desired level of detail. Once the appropriate level has been reached, functions are connected to the physical components of the product. This mapping reveals the product architecture (Ulrich, 1995):

- When each function is implemented by exactly one component, the product exhibits modular architecture.

- when having multiple functions associated to a single component, the description of the product architecture tends toward an integral one

Figure 2.3. An example of function tree analysis



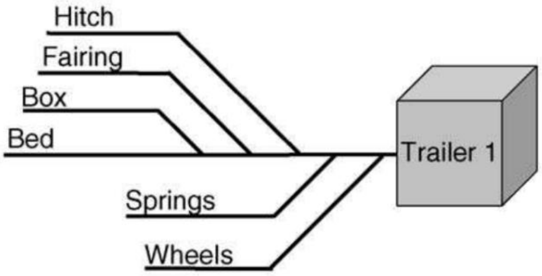
Source: Pahl and Beitz (1996)

Another tool that helps in studying architecture analysis is the assembly fishbone diagram. Assembly fishbone diagrams represent a structured approach to analyse the physical architecture of products through tear-down decomposition (Ulrich and Pearson, 1998). The method starts from the complete assembled product and progressively identifies the hierarchical relationships between components according to their physical assembly sequence. Components are organised along a main structural backbone that represents the primary assembly structreu, while secondary elements are positioned according to their attachment relationships. This representation makes explicit the physical organisation of the product and the dependency relationships embedded in this architecture.

By visualizing how components are connected in a sequence, connected to the main structure, fishbone diagrams help identifying modules. In this context of architectural analysis, the tool provides a systematic way to document the physical decomposition and to establish a consistent basis for further analysis on product architecture's couplings (Ulrich and Pearson, 1998).

It starts from a tear-down analysis in which a product is physically disassembled to understand its structure and parts (Ulrich and Pearson, 1998). It works backwards, beginning with the full product, all parts that are assembled to the main component are considered on the relevant level. In the image below, an example of a fishbone assembly diagram showing how the main components of a trailer, sequentially attached to a single assembly backbone.

Figure 2.4. An example of an assembly fishbone diagram



Source: Ulrich and Pearson (1998)

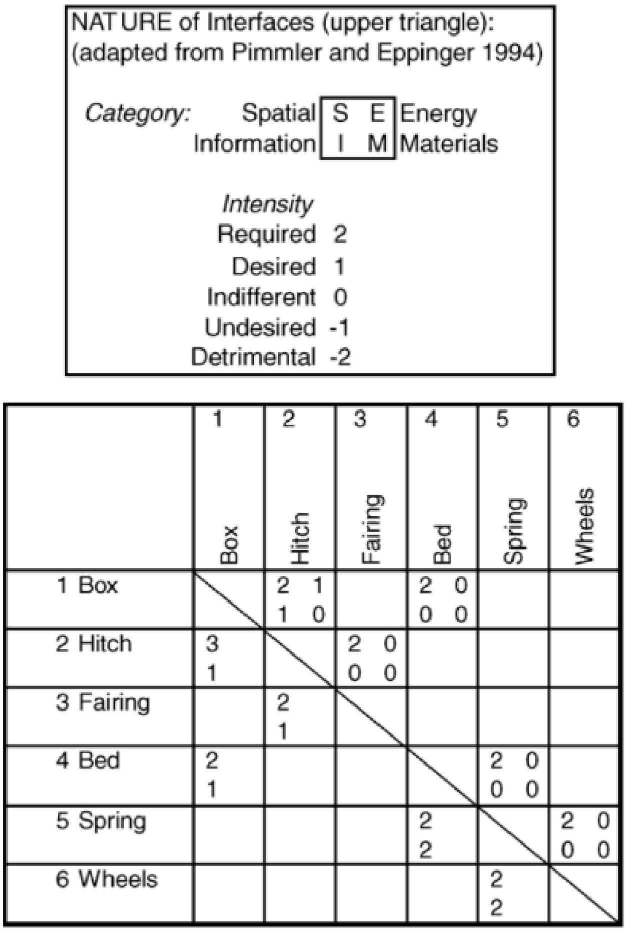
Another method used to analyse product architecture is the interface matrix, which helps identify and descried the interactions between product components and understand how strongly they are connected (Pimmler and Eppinger, 1994). While structural decomposition shows how a product is divided into parts, the interface matrix focuses on how these parts interact with each other.

The method is based on building a square matrix where both rows and columns represent the components of the product. Each cell of the matrix indicates whether two components interact and what type of interaction exists between them. According to Pimmler and Eppinger, interfaces can be classified in different categories, such as

- spatial interactions (components that must fit together physically),
- energy interactions (transfer of forces or power),
- information interactions (exchange of signals or data)
- material interactions (flow of physical substances)

By mapping these relationships, the interface matrix provides a clear overview of the level of coupling withing a product architecture, Products with many dense and interdependent interactions tend to have more integral architecture, while products with fewer and more standardised interfaces are usually more modular. Fort this aspect, the interface matrix is particularly useful when comparing analogue and digital interactions toward information-based connections. In this thesis, interface matrices are used to systematically document the connection between components in each product analysed. The results support the qualitative architectural assessment and help identify differences in interaction patterns between analogue and digital versions of similar products. Below, an interface matrix example is displayed:

Figure 2.4b. An example of an interface matrix



NATURE of Interfaces (upper triangle):
(adapted from Pimmler and Eppinger 1994)

Category:

Spatial	S	E	Energy
Information	I	M	Materials

Intensity

Required	2
Desired	1
Indifferent	0
Undesired	-1
Detrimental	-2

	1 Box	2 Hitch	3 Fairing	4 Bed	5 Spring	6 Wheels
1 Box		2 1 1 0		2 0 0 0		
2 Hitch	3 1		2 0 0 0			
3 Fairing		2 1				
4 Bed	2 1				2 0 0 0	
5 Spring				2 2		2 0 0 0
6 Wheels					2 2	

Source: Pimmler and Eppinger (1994)

2.2.2 Quantitative methods to assess architecture

Qualitative and more visual representation of architecture can be useful for identifying structural patterns and understanding how functions are allocated to components. These tools allow designers to detect clusters of coupled components and visualize interface structures.

However, quantitative analysis can sometimes be necessary to establish a more precise measure of the level of modularity of an architecture. Among the most widely used approaches are commonality indices, which are key in quantifying how many components are shared between functions within a specific product. When calculating it, the higher the degree of commonality, the more components are shared across different functions.

In product design, understanding this trade-off between standardising and differentiating is essential (Fixson, 2005).

The Degree of Commonality Index (DCI) introduced by Collier in 1981, measures the average number of common functions that each distinct component provides. Formally, DCI is calculated as follows:

$$DCI = \frac{\sum_{j=i+1}^{i+d} \Phi_j}{d}$$

Where:

- *Φ(j)* represents the number of functions that component *j* is used in, (also known as the number of parents of component *j*)
- *d* is the total number of distinct components in the product

The DCI provides insight into the amount of function sharing across components. A DCI value of 1 indicates no commonality as each component serves only one function, while higher values indicate greater function sharing.

While DCI provides useful information, it has a significant limitation: its value is unbounded, which makes it difficult to compare it with another product.

An index which addresses this issue is the Total Constant Commonality Index which was developed by Wacker & Trelevan in 1986. This indicator normalises the commonality measure to a bounded interval between 0 and 1, facilitating the comparison between different products.

The TCCI is based on the number of instances in which each component appears in the product bill of materials, which is the complete list of parts and materials needed to construct the product. The aim of the indicator, as stated by Altavilla S. and Montagna F., is “not only to consider the similarities between the two products, but also to understand the number of standard parts shared between the analysed cases and to investigate the architectural complexity of the entire product structure”.

The TCCI is calculated as follows:

$$TCCI = 1 - \frac{d - 1}{\sum_{j=1}^{d} \Phi_j - 1}$$

It ranges from 0 to 1, where 0 implies no component sharing between functions, corresponding to the maximum integral architecture. Conversely, the maximum modular architecture is reflected by a TCCI = 1.

These absolute boundaries, ranging from 0 to 1, facilitate the comparison among different products, as all values are normalised to the same scale.
By considering how extensively components are reused across compared products, the TCCI provides a comprehensive measure of architectural complexity (Altavilla and Montagna, 2019).

The Commonality Index proposed by Martin and Ishii in 1996 is a modified version of the DCI that measures commonality across an entire product family rather than withing a single product (Martin and Ishii, 1996), The index considers how many unique components exist across all variants and normalises this information on a bounded scale between 0 and 1. Formally, CI is defined as:

$$CI = 1 - \frac{u - \max p_j}{\sum_{j=1}^{v_n} p_j - \max p_j}$$

Where:

- *u* is the number of unique parts
- *p(j)* is the number of parts in model *j*
- *v(n)* is the final number of varieties offered

The CI ranges from 0 to 1. Higher CI indicates fewer unique components and greater standardisation across the product family. Values close to 0 indicate that variants rely on many distinct components, reflecting low commonality and a higher level of complexity in managing the product family.
In conclusion, it can be interpreted as the ratio between the number of unique components in a product family and the total number of parts in the family.

Another notable index is the Product Line Commonality Index (PCI). Introduced by Kota, Sethuraman and Miller in 2000, it extends this reasoning by focusing on non-product differentiating components and by explicitly capturing how similar these components are in size, materials and assembly methods across the family (Kota, 2000). It is designed to penalise only those differences that should ideally be standardised, given the chosen range of variants.
The PCI is calculated as reported below, where:

- *n(j)* represents the number of functions that component *j* is used in

- *f1(i)* is the size and shape factor for component i
- *f2(i)* is the materials and manufacturing processes factor for component i.
- *f3(i)* is the assembly and fastening schemes factor for component i

$$PCI = \frac{\sum_{i=1}^{P} n_i {}^* f_{1i} {}^* f_{2i} {}^* f_{3i} - \sum_{i=1}^{P} 1/n_i^2}{P^*N - \sum_{i=1}^{P} 1/n_i^2} * 100$$

The three factors f1(i), f2(i), f3(i) take values between the minimum 1/n(i) and a maximum of 1, depending on how similar corresponding characteristics are across variants. The PCI ranges from 0 to 100. When having value zero indicates no non-differentiating parts are shared, while when having maximum value equal to 100 means non-differentiating parts are identical across products in size, shape, material and assembly method.

Beyond indices alone, an insightful tool that is used for architecture analyses are Functional Component Allocation (FCA) maps, which provide a graphical representation of the associations between functions and components.

The function component allocation analysis works as follows:

1) construct a matrix with rows representing functions (from the functional tree) and columns representing physical components. Each cell in this matrix indicates whether and how much a particular component contributes to each function.
2) Identify and plot relationships. For each function, two indices are calculated
   - the first index identifying the number of components that jointly provide a function
   - another index, assessing the extent to which this set of components contributes to other functions
3) Map functions onto the map. These two indices position each function on a two-axis graph, revealing patterns of product architecture.

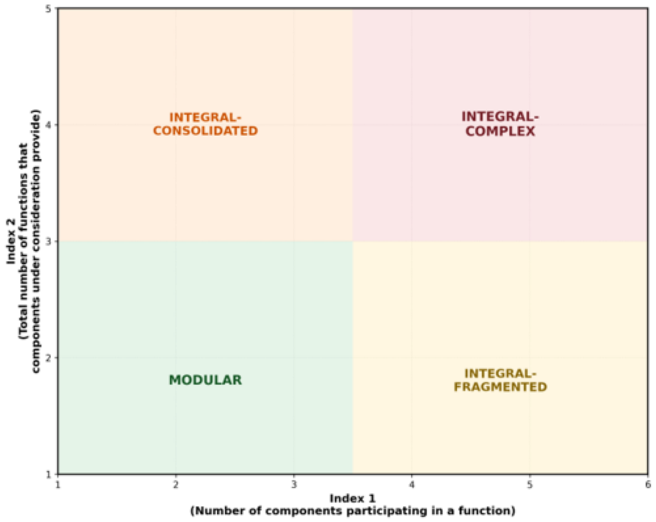
The maps can be better interpreted by dividing them into four regions, each representing a different architectural style. Each function can be located in one of these four regions (Fixson, 2004):

- lower left region (modular architecture): functions located in this region are closest to the ‘ideal’ modular one-to-one relationship between function and component, where a single component delivers a single function and that component is not involved in providing other functions.

- If a function is provided by a larger set of components, which individually are not involved in other functions, then the function is located in the lower right region of the FCA map. This exhibits an integral-fragmented FCA style, where a function requires multiple dedicated components.
- If one component delivers several functions, these functions provided by that component will be located in the upper left region of the map. They show integral-consolidated behaviour.
- The functions would be located in the upper-right quadrant only if multiple components provide multiple functions in such a way that most components participate in most functions. This represents the most integral configuration where a product exhibits a deeply interconnected architecture.

In conclusion, the FCA map visually represents the architectural balance of a product. By plotting all the functions on it, designers can see clearly whether the product has predominantly modular characteristics, meaning functions clustered on the left, or a mixed architecture with functions located all over the map. Moreover, in this representation it is easy to identify which functions might benefit from redesigning to increase modularity.

Figure 2.5. FCA maps split into regions



Source: Fixson, 2004

Chapter 3: Architecture and dynamics of innovation

3.1. Architecture Innovation and Types of Technological Change

Product architecture plays a central role in shaping innovation dynamics over time. It defines how product functions are allocated to components and how these components interact within a system (Fixson, 2004). Because architectural choices determine interdependencies between elements, they strongly influence the types of innovation that firms can pursue, and the organisational capabilities required to implement change. Innovation can be understood as the economic exploitation of an invention, which is the act of devising a new solution to a problem. An invention becomes an innovation only when it is successfully introduced to the market and recognised as valuable by users. When discussing innovation, it does not necessarily involve products, as it could be concerning processes or services as well (Roberts, 1987). In particular, innovation can be defined as the “act or process of introducing ideas, devices or methods” (Mirriam-Webster Dictionary). Innovation may concern products, processes or services and their evolution typically follows technological trajectories characterisde by progressive performance over time.

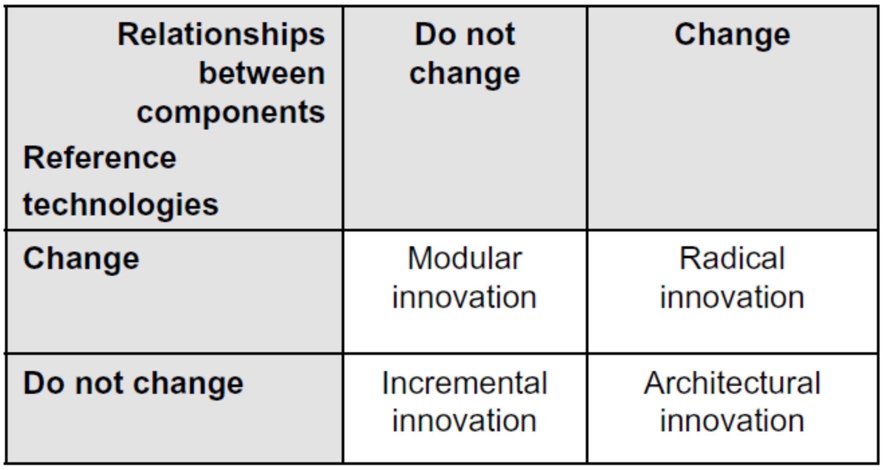
Innovation can be classified according to different taxonomies, depending on the perspective adopted. One of these taxonomies focuses on the product architecture. Henderso and Clark (1990) proposed a classification in which the type of innovation depends on whether the change affects individual components, the interfaces between them, or the overall system architecture.

Within these technological trajectories the nature of innovation varies depending on whether the change affects components, their interfaces or the entire system architecture. A widely adopted framework is the taxonomy proposed by Henderson and Clark (1990), which distinguishes four main types of innovation:

- Incremental Innovation: refinements within existing components without altering system structure. Neither the underlying technology nor product architecture will change. knowledge and capabilities are reinforced
- Modular Innovation: changes in individual components while maintaining stable interfaces. Underlying technology does change in one or more functional elements, but product architecture remains stable

- Architectural Innovation: reconfiguration of linkages between components and functions while core ones remaining largely unchanged. Through new structural relationships, it alters the system performance
- Radical innovation: simultaneous changes to both core concepts and architectural linkages, resulting in an entirely new system. Both underlying technology and product architecture change

Figure 3.2: a recap of the taxonomy proposed by Henderson and Clark



Relationships between components **Reference technologies**	**Do not change**	**Change**
Change	Modular innovation	Radical innovation
Do not change	Incremental innovation	Architectural innovation

Source: Henderson and Clark, 1990

The architectural configuration strongly shapes innovation feasibility. Integral architectures characterised by tightly coupled components, require coordinated changes across system and may increase the cost and complexity of innovation (Fixson and Park 2008). Modular architectures by contrast rely on stable interfaces that reduce interdependencies and enable parallel innovation activities (Schilling 2000).

Therefore, product architecture not only reflects technological choices but also influences how innovation capabilities develop within firms and industries.

3.2. The Abernathy-Utterback model and dominant design

When studying innovation, taking into consideration the role of time is key and the performance of the innovation as time passes. Technologies do not evolve randomly but rather following recognisable trajectories: these trajectories are represented through s-curves.

These curves describe the relationship between technological performance or cumulative adoption and time. S-curves which study the evolution of innovation from the market perspective, are called diffusion curves, which describe how innovations spread among users over time. This pattern articulates into distinct phases: an initial incubation phase, characterized by uncertainty and limited adoption; a diffusion phase, in which the technology spread rapidly, and economies of scale emerge; and a maturity phase, where growth slows significantly and technology becomes established (Rogers, 2003). These phases provide a useful analytical framework for positioning technologies within their lifecycle and also comprehend the competitive dynamics of the market.

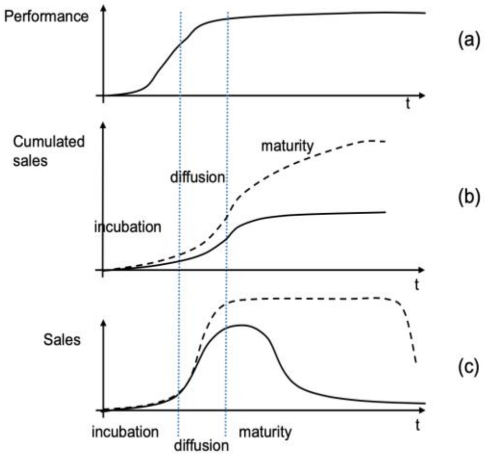
The innovation process is cyclical: it repeats over time and contributes to generating and placing new technologies on the market. The analysis of these evolutions in a dynamic context is what is called dynamics of innovation.

In analysing these dynamics, it is essential to introduce a second s-curve that roughly follows the performance s-curve, and which represents the diffusion of the innovation in the market. Diffusion (or penetration) can be defined as the fraction of potential users that, at a given time, have decided to adopt the technology (Rogers, 2003). To represent this diffusion cumulative sales of the technology over time are used. By developing graphically this curve, it can be split into three phases:

- incubation phase: technology is at its beginning and is not yet established. This phase is characterised by uncertainty and limited adoption
- diffusion phase: technology spread in the market and generates economies of scale
- maturity. phase in which growth slows significantly, as both sales and technology are stable

These phases provide a useful analytical framework for positioning technologies within their lifecycle and understanding competition dynamics of the market.

Figure 3.1: Performance and diffusion s-curves.



Source: Abernathy and Utterback, 1975

To understand how product architecture emerges and evolves during these phases, one of the most influential frameworks is the Abernathy-Utterback model. This model explains how the competitive landscape changes with the changing of the nature of the innovation, with technology moving through its lifecycle.

The evolution of innovation within industries is closely linked to the emergence of dominant designs. The Abernathy-Utterback model provides a dynamic framework explaining how technological change and competitive dynamics co-evolve over time (Abernathy and Utterback, 1975). The model divides in three stages:

- The earliest stage, when a new technology emerges, is called the fluid phase. This is a phase in which multiple alternatives coexist simultaneously. Firms experiment extensively with different product designs, proposing different technical solutions to address the same requirements. High rate of product innovation but the progress is limited as the efforts are not concentrated in the same direction.
- As the technology and market develop, the industry enters the transition phase. Once the product architecture emerges as the dominant design, product performance takes off as well as its diffusion on the market.
- Lastly, there is the specific phase, in which firms compete on cost and quality while demand increases. There is a shift towards process innovation specific to the dominant design and the competition is reduced to few big players

A dominant design can therefore be defined as the set of components and features that become widely accepted by the industry and the market, overcoming competing technological solutions (Abernathy and Utterback, 1975). Its emergence represents a turning point in the industry lifecycle, signaling the resolution of competitive struggles over design approaches and marking the point at which the market converges to a standard. The dominant design is therefore an architectural outcome: the configuration of components and their relationships have prevailed over alternatives. Once established, product architecture determines which innovations are feasible and which are not, constraining the direction of technological change for firms and industries (Henderson and Clark, 1990) .

The persistence of a dominant design over time is reinforced by several mechanisms.

- Learning effects allow firms to improve manufacturing efficiency and product quality as experience accumulates with the same architecture.

- Economies of scale: large investments in specialised production assets create switching barriers
- Complementary assets: ecosystems of suppliers and complementary products develop around the dominant architecture
- Network externalities. adoption increase the value of compatibility with existing standards

These mechanisms generate technological lock-in, often stabilising dominant designs for an extended period. Although originally developed for manufacturing industries, similar patterns emerge in service and digital context. However, digital technologies introduce greater flexibility because software components can be modified independently from physical architecture, potentially weakening traditional lock-in mechanisms (Yoo et al 2010).

3.3 Vertical Integration and architectural paradigms

Product architecture has significant implications for organisational structure, particularly regarding vertical integration and inter-firm relationships. The degree of modularity influences makes or buy decisions, supplier involvement and collaboration with complementors. In fact, product architecture does not only determine how functions are allocated to components but also influences which activity to keep in house and which one to externalise.

In integral architectures, components are tightly coupled and their performance depends on complex interactions.

In such systems, component design decisions are highly interdependent; interfaces are difficult to standardise; performance optimisation requires coordinated system-level control. Consequently, firms developing integral products are typical of high level of vertical integration systems. In fact, independent suppliers could struggle to design components without the knowledge on the full system.; coordination costs increase when development is distributed, performance optimisation requires unified control over design and production. Integral architectures are frequently observed during early industry stages when performance improvements are priorities and technological uncertainty remains high (Fixson and Park 2008).

Conversely, modular architectures rely on stable interfaces that allow components to function independently (Schilling, 2000) this structural separation supports:

- specialisation of external suppliers
- competitive sourcing of modules
- parallel innovation across different subsystems

In modular industries, firms often focus on system integration and architectural design, coordination of supply network, management of complementary innovation ecosystems. The modular paradigm encourages outsourcing and distributed innovation because suppliers can develop improvements without redesigning the entire system. The relationship between architecture and vertical integration is bidirectional

- architectural choices influence whether activities are either internalised or outsourced
- supplier capabilities and industry maturity shape the feasible degree of modularity

As dominant designs emerge and interfaces become standardised, modular architectures become more viable, allowing firms to externalise production while focusing on high-volume system integration activities.

Furthermore, architectural paradigms influence external relationships between suppliers. Complementors and ecosystem partners become central actors in modular industries where innovation occurs through coordinated contributions from multiple specialised firms (Yoo et al , 2010). Understanding the interaction between product architecture, vertical integration and innovation dynamics is therefore essential for explaining both firm strategy and industry evolution.

4. Digital transformation of physical products

4.1. Digitalization of products

Digitalization refers to the encoding of non-digital information into digital format. This process has progressively affected products in multiple industries in the last twenty years, and it represents a shift in how products are designed, manufactured, used during their lifecycle (Yoo et al, 2010). For a product to be digitalised it means that the product acquires properties which differentiate it structurally and functionally from their non-digital equivalents (Henfridsson, 2014).

When comparing non-digital and digital products, the functionality in the non-digital product is implemented through mechanical components and their fixed physical configuration; digital products instead are programmable (Yoo et al, 2010). Digital products in fact can execute new functions after production through software updates or modifications to algorithms. A thermostat temperature control logic for instance can be redesigned through firmware updates without replacing any physical component; similarly, a digital food processor can execute new mixing sequences through software updates.

The practical implication of programmability is an extended design window (Henfridsson, 2014) because in non-digital products, once the design is finalised, manufacturing processes require certain specifications. Instead, digital ones permit functionality design and redesign during its operational life. This flexibility impacts how companies manage innovation because the boundary between the design and operational phases of product development become less clear.

A second important property is data homogenization, which consists in standardizing information in the digital format, making data accessible across different devices (Yoo 2010). In non-digital systems, data is present in different formats. For instance, in music cassette tape, as magnetic variations. These format differences create coupling between information and physical substrate, limiting the compatibility of data with different systems.

Digitalization removes these issues of compatibility; a digital representation maps any non-digital signal into a set of binary numbers, i.e., bits (a contraction of binary digits). This leads to a homogenization of all data accessible by digital devices. Any digital contents (audio, video, text, and image) can be stored, transmitted, processed, and displayed using the same digital devices and networks (Yoo et al, 2010).

A food processor recipe sequence can be copied, shared and migrated across devices. Thai allows seamless information flow across systems that in non digital architectures would be impermeable functionally.

Architecturally, decoupling between physical components and information services. Components can be updated or reconfigured without disrupting data flows, provided standardised interfaces are maintained. This property becomes important when analysing how digital products maintain modularity even though software complexity increases (Yoo et al, 2010).

Another aspect coming with digitalised products are network effects. Network effects are positive externalities increasing in value as more devices adopt compatible standards. Typically, non-digital products operate as close systems while digital ones are involved in networked ecosystems with their utility strongly depending on the possibility to connect with external systems. For instance, taking into consideration a food processor, it gains functionality when it is possible to download recipes through cloud connection.

A repercussion on competition is represented by product platform strategies. In fact, digital products function both as autonomous product services, a core function and as a component in a larger ecosystem. This directly impacts competitive dynamics, with firms competing on ability to attract complementary services. The fixed edges of nondigital products are way less permeable than the analysed digital structure.

A key characteristic of digital innovation is that it allows the carrying out of new combinations of digital and physical components to produce new products.

In the following chapters, the use of the term digital innovation will be focused on product innovation, distinguishing it from IT innovation research that has occupied primarily process innovation (Yoo et al, 2010).

4.2. Architecture of digital products

The digitalisation of physical products has significant implications for product architecture. Traditional products typically are characterised by integral architectures and functions achieved through tightly coupled components.

In such architectures, changes to one component frequently require changes to others, limiting flexibility (Henderson and Clark, 1990).

Digitalisation modifies this structure, enabling a redistribution of function across physical components. The mechanical functions are replaced by digital processing units and sensors; as a result, several functions which were previously provided by multiple components can be consolidated into fewer ones (Fixson and Park 2008).

A key concept when discussing digital products is layered modular architecture, which is conceived as a "hybrid of the modular architecture of a physical product and the layered architecture of digital technology" (Yoo et al, 2010).

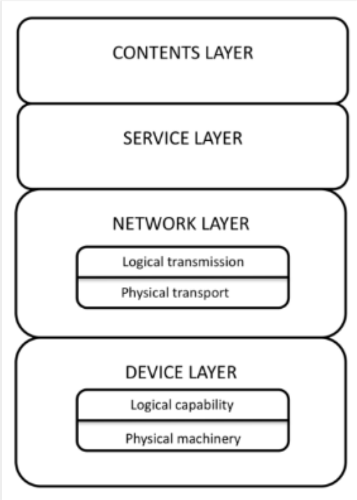
Layered modular structure extends the modular architecture of physical products by incorporating four loosely coupled layers of devices, networks, services, and contents created by digital technology. In particular, the layered modular architecture is composed of:

- device layer: this layer includes physical machinery (sensors actuators, processors, mechanical structures) and the embedded logical capabilities, such as operating systems and firmware, that control and coordinate the physical components (Yoo et al, 2010). It is where the core physical functions of the product are implemented.
- network layer: comprises both the physical transport infrastructure and the logical transmission, that enable communication between devices (Yoo et al, 2010). It connects the device to external systems and other products.
- service layer: this layer contains the application functionality directly used by end users: control algorithms, scheduling and optimisation routines user interfaces and other software services that process data (Yoo et al, 2010). It defines what the product does" from the user's perspective.
- content layer: consists of data handled by the product: text, images and data such as tags, location stamps. These contents can be stored, transmitted and combined across different devices and services.

Layered modular architecture is particularly effective in explaining innovation dynamics in software intensive systems, where functionality emerges from interactions across layers and where many-to-many relationships are common at the logical level (Yoo et al, 2010). Nevertheless, this perspective is limited when applied to analysis of physical products. It mainly focuses on logical structures rather than on physical components and also it often includes services that fall outside the scope of physical product architectures (Salvador, 2007).

For this reason, the discussion focuses on the physical architecture of products with tangible components, examining how digitalisation affects the allocation of functions to physical components. Moreover, it is expected that at this level, digitalisation reduces architectural coupling and increases modularity (Ulrich, 1995).

Figure 4.1 the layered architecture of digital technology



Source: Yoo et al, 2010

The four layers represent different design hierarchies (Clark 1985), meaning that design decisions for a component in one layer without depending on decisions made in others. Thanks to this separation, designers can pursue *combinatorial innovation* by recombining components drawing from different layers, relying on shared standards to integrate them into alternative digital products (Gao and Iyer, 2006).

The layered modular architecture describes therefore a product structure where devices, networks are organised into distinct layers. Each layer follows its own design logic, so design decisions can be made largely independently. Shared standards and protocols enable interaction across layers, allowing elements developed separately to be recombined into new solutions. This structure increases design flexibility and supports combinatorial innovation through the integration of components belonging to different layers (Yoo et al, 2010).

Another aspect which is important to be analysed in the analysis of digital product architecture concerns architectural frames. Architectural frames can be defined as shared cognitive structures that shape how designers and engineers interpret product architectures, identify relevant components and reason on relationships between functions and components. Their influence with design problems is perceived as crucial and which solution strategies are considered legitimate within a given technological context (Henderson and Clark, 1990).

Traditionally, products have been designed with a hierarchy of parts frame, where products are decomposed into hierarchically organised components each responsible for a specific function. This frame tends to produce a stable architecture but limits flexibility (Simon 1962).

Later on, with the upcoming digitalisation of products, another frame has been adopted, the *network of patterns* frames. In this perspective, functions are patterns being reused and combined across different components. Digital technologies enable functions to be realised independently of specific physical parts (Henfridsson, 2014). Rather than increasing complexity of the architecture, digitalisation often leads to modular and simpler architectures, characterised by clearer interfaces and more one-to-one function-component relationships, because it abstracts functions away from mechanical structures. Within the network of patterns, design does not primarily emphasize decomposition into physical parts and subsequent aggregation. Instead, it focuses on generalising design solutions in a pattern that can be specialised for different applications. A pattern describes a recurring problem and provides a reusable solution that can be adapted to multiple contexts. For instance, temperature control pattern can be implemented across different products such as thermostats, refrigerators. While the physical components differ, the underlying logic for sensing and regulating temperature can be reus. ed and adapted across devices.

The pattern approach is particularly valuable when facing new product challenges as it allows firms to recombine and adapt existing patterns rather than designing solutions entirely from scratch (Henfridsson, 2014).

In conclusion, while classical mechanical products typically employ hierarchy of parts, digitalised products increasingly rely on both frames simultaneously.

At the physical architecture level, this often results in a clearer allocation of functions to components. Digitalised products tend to exhibit a higher number of one-to-one relationships between functions and components, which is a defining characteristic of modular architectures. Interfaces become more stable and components can be modified or replaced with limited impact on the rest of the system (Gershenson, 2003).

It is important to note that digitalisation does not eliminate physical constraints. Mechanical spatial and structure requirements still play a critical role in product design. However, being that the functional complexity is managed by the digitalised parts of the product, physical architectures can become simpler and more modular compared to their non-digital counterparts (Ulrich, 1995).
Digital transformation can therefore be interpreted as a transition from integrated physical solutions towards architectures in which physical components act as carriers of programmable functional patterns. This interpretation is particularly useful for understanding changes in modularity observed through empirical architectural analysis.

Despite the growing body of research on digitalisation and product innovation, existing studies still provide limited empirical evidence on how digitalisation concretely reshapes the internal physical architecture of product. Much of the literature focuses either on digital technologies themselves or on innovation outcomes such as services, platform and ecosystem dynamics (Yoo et al, 2010; Henfridsson et al 2014), while architectural implications are often discussed at an industry level rather than through a systematic structural analysis. As discussed in the previous chapters, digitalisatin has the potential to modify the collection of functions withing products, shifting complexity from mechanical structures toward programmable components. This potentially leads to a increase in modularity architectural-wise (Ulrich, 1995; Gershenson 2003). however, empirical research rarely investigates how these transformations affect the decomposition of physical products: by analysing digital and non-digital versions pf the same artefact of products performing equivalent core functions, it can be analysed whether digital transformation leads to increased modularity in physical product design.

This thesis addresses this gap by conducting a comparative architectural analysis of non-digital and digital product pairs. The objective is to assess how the degree of architectural modularity changes when products undergo digital transformation, focusing specifically on the internal structure of physical components rather than on software features or service systems. To achieve this, the analysis applies a consistent set of architectural tools introduced in chapter 2. The following chapter presents the methodological framework adopted to select product pairs and to operationalise the measurement of architectural modularity along the non-digital digital comparisons.

Chapter 5: Methodology of analysis

This chapter describes how the analysis of product architecture modularity with digital transformation was conducted. The study uses a systematic approach, based on the theoretical frameworks presented in the previous chapters.

When products become digital, the way functions are organised in the product changes; functions are no longer embedded directly in mechanical structures, instead they are assigned to programmable components. This makes the relationship between functions and physical components clear and easier to measure.

Therefore, the main goal here was to measure the change in the degree of product architecture modularity resulting from digital transformation.

Five main tools were used to analyse product architecture, all presented in section 2.2. These tools together provide a completer picture of product architecture:

- block diagrams: show how the product is divided into components and how they are connected.
- design structure matrices: show which components depend on each other.
- functional trees: show all functions a product performs, organised from general to specific
- total constant commonality index (TCCI); measures how many components are shared among different functions
- Functional Component Allocation map: shows visually which components perform which functions

The analysis follows a logical process. First, functions are broken down into more detailed functions by asking "How is this function performed?". This continues until we reach the desired level of detail that is performed by single components. Then, we look at how many components work together for each function and how much those components are used for other functions. This allows us to measure how modular or integrated the product architecture is.

The analysis focuses on pairs of products having non-digital and digital versions. Nine paired comparisons of products have been analysed with a specific process. The products perform the same core functions, while relying on different architectures. The topic to be examined here is the increase in modularity associated with the digital version through architecture comparison. Only product pairs characterised by an established dominant design are considered, in order for the architecture to be clearly identifiable.

Digital products that enable service functions were excluded in order to avoid introducing additional layers of architectural complexity. Also, purely digital artefacts are excluded because of their intangible nature, as well as non-assembled products. That lead to the following list of products:

Non-digital product version	Digital product version
Cyclone vacuum cleaner	Autonomous floor cleaning robot
Music cassette deck	CD player
Electronic treadmill	Digital treadmill
Food processor	Thermomix
Electronic thermostat	Digital thermostat
Mechanical weighing scale	Digital weighing scale

Each pair was chosen because it shows a real change in technology. The non-digital version was eventually replaced by the digital one, even though both perform the same core function for the user.

Combining block diagrams and DSM therefore provides a clearer picture of the internal structure of the product pairs. These two tools along with the functional tree prepare the ground for the quantitative analyses of commonality and FCA map. Specifically, the FCA map has been developed across two indexes. The first one identifies the number of components that jointly provide an elementary function, the second one counts the number of other functions that the set of components for a specific elementary function supports. Consequently, using the two indexes four regions are plotted: lower left-modular region, lower right integral-fragmented, upper left integral-consolidated, upper right integral-complex, according to the terminology proposed by Fixson.

The theory presented in Chapter 4 suggests that digital products should be more modular than non-digital products. This is because digital products can update functions through software instead of changing mechanical parts, data in digital products is stored in standard formats, not locked to specific physical parts; interfaces are more standardized when dealing with digital products.

To perform the comparisons the first topic that was studied were patents about the product in a standard form; not the latest version but rather the first one to be established and commercialised.

In the following tables are reported all the analysed patents for each product pair.

Figure 5.1. Patents used in the analysis

Product	**Patents**
Music Cassette Deck	Tomita S., January 26, 1979, *Tape Stop Detecting Mechanism*, USA, US4238808, United States Patent Office Ida M., December 2, 1981, *Tape Transport Mechanism for Cassette Tape Player*, USA, US4453189, United States Patent Office Kato I., Kamimura T., Takemasa K., Onishi J., September 30, 1985, *Cassette Tape Deck*, USA, US4672477, Pioneer Electronic Corporation Nagaya H., Saito Y., January 9, 1985, *Cassette Tape Deck Operating Mechanism*, USA, US4704651, Pioneer Electronic Corporation Mizukami T., Kono K., Okumoto K., August 14, 1991, *Highly Compact Cassette Player*, USA, US5153791, Matsushita Electric Industrial Co
CD Player	Ueno M., Naito T., January 8, 1985, *Disc Player*, USA, US4653039, Sony Corporation Kimura T., Endo F., September 20, 1989, *Compact Disk Player*, USA, US5122999, Pioneer Electronic Corporation Nonaka Y., January 25, 1993, *CD Player*, Europe, EP0552987B1, Pioneer Electronic Corp Scibora M., May 4, 1999, *Universal Compressed Audio Player*, USA, US6122230, Advanced Communication Design, Inc.
Food Processor	Verdun P., January 24, 1974, *Apparatus for preparing food*, USA, US3892365, United States Patent Office Necas D. E., Khan A. U., November 3, 1980, *Food Processor*, USA, US4387860, United States Patent Office Shibata K., February 5, 1981, *Julienne cutter for food processors*, USA, US4393737A, United States Patent Office Podell A. F., February 25, 1981, *Magnetic safety interlock for a food processor utilizing vertically oriented, quadrant coded magnets*, USA, US4629131A, United States Patent Office Le Rouzic C., Gateaud A., Fleche M., August 31, 1989, *Device for the extraction of juice and pulp from fruit and vegetables*, USA, US5289763, United States Patent Office

Thermomix	Seidler M., Leaman K., August 27, 2009, *Food Mixers and Processors*, USA, US8348186, United States Patent Office Goderieux A., Bizard J.C., April 14, 2011, *Cooking Appliance with Stirring Means and Associated Method*, USA, US9498083, United States Patent Office Yan W., Lang T., Stach C., Kraut-Reinkober S., November 13, 2018, *Method for adjusting the heating power of at least one heating element of a domestic appliance*, USA, US11071404, Vorwerk & Co. Interholding GmbH Haase C., Resende M. J., Wehlig R., GANSS J., Hilgers S., November 10, 2014, *Systems and methods involving a computer system and a kitchen appliance*, USA, US10159382, Vorwerk & Co. Interholding GmbH
Electronic Thermostat	Behr J. L., January 10, 1978, *Control system for maximizing the efficiency of an evaporator coil*, USA, US4067203A, Emerson Electric Co Kabat J. L., February 7, 1978, *Proportional Temperature Control System with Linear Delay*, USA, US4072866A, Honeywell Inc. Mott R. C., June 20, 1978, *Proportional electrical control system*, USA, US4095741A, Honeywell Inc Johnson V. C., McCormick V. K., Whitney J. M., December 11, 1979, *Heating and cooling thermostat with changeover switching operated upon control point adjustment*, USA, US4177860, Honeywell Inc.
Digital Thermostat	Jones J. J., McIntosh J. O., June 14, 1983, *Electronically controlled programmable digital thermostat having variable threshold hysteresis with time*, USA, US4388692A, Honeywell Inc Sahay B. B., Jones J. J., April 17, 1984, *Electrically controlled programmable digital thermostat and method for regulating the operation of multistage heating and cooling systems*, USA, US4442972A, Honeywell Inc Brown B. T., Butler W. P., July 6, 1993, *Digital thermostat with single rotary encoder switch for establishing set point temperature*, USA, US5224649A, Emerson Electric Co Rosen H. B., November 30, 2004, *Programmable thermostat system employing a touch screen unit for intuitive interactive interface with a user*, USA, US6824069, United States Patent Office Lunacek M. G., Amundson J. B., Juntunen R. D., July 14, 2015, *Thermostat with electronic image display*, USA, US9081393, Ademco Inc
Electronic Treadmill	Cherry, January 16, 1973, *Drive and Control System for Diagnostic and Therapeutic Exercise Treadmill*, USA, US3711812A, Del Mar Engineering Laboratories Otte G., December 21, 1982, *Emergency shut-off switch and frame assemblies for exercise apparatus*, USA, US4364556A, Nissen Corporation Smith D. B., Moore J. M., May 27, 1986, *System for Elevating an Exercise Treadmill*, USA, US4591147, Precor Incorporated Ogden R., Johnson C. J., January 13, 1987, *Adjustable speed control arrangement for motorized exercise treadmills*, USA, US4635928A, Ajax Enterprises Corporation

Digital Treadmill	Shafe T. C., White M. E., December 10, 1997, *Motorized exercise treadmill*, USA, US6045490, United States Patent Office Chen J., October 23, 2002, *Replaceable control mainboard for treadmill*, USA, US6960154B2, United States Patent Office Oglesby G. E., Golen S. Jr., Fox J. B., Daniel J., Kohn R. D., Clawson C. E., Lantz K. F., Wille D. R., September 4, 2001, *Treadmill control system*, USA, US6783482B2, Brunswick Corporation Wei W. F., Huang C. C., Huang C. Y., Lin J. T., January 27, 2017, *Treadmill and control method for controlling the treadmill belt thereof*, USA, US10293210B2, United States Patent Office
Vacuum Cleaner	Dyson J., February 15, 1983, *Vacuum cleaning appliances*, USA, US4373228A, United States Patent Office Dyson J., November 5, 1991, *Shut-off device for cyclonic vacuum cleaner*, USA, US5062870A, United States Patent Office Dyson J., February 25, 1992, *Dual cyclonic vacuum cleaner with disposable liner*,USA, US5090976A, United States Patent Office Dyson J., September 24, 1996, *Dual cyclonic vacuum cleaner*, USA, US5558697A, United States Patent Office
Autonomous Floor Cleaning Robot	Jones J. L., Mack N. E., Nugent D. M., Sandin P. E., December 16, 2002, *Autonomous floor-cleaning robot*, USA, US6883201B2, United States Patent Office Ozick D. N., Okerholm A. M., Mammen J. W., Halloran M. J., Sandin P. E., Won C., December 23, 2008, *Autonomous coverage robot navigation system*, USA, US8380350B2, United States Patent Office Pack R. T., Lenser S. R., Kearns J. H., Taka O., May 23, 2013, *Simultaneous Localization and Mapping for a Mobile Robot*, USA, US9037396, United States Patent Office Sun C., Kim S., December 26, 2017, *Robot cleaner*, USA, US10548444B2, United States Patent Office

To enhance the readability of the results presentation, only the first comparison, between a music cassette deck and a cd player, is described in a complete way, including both the functional architecture and the component-function analysis. For the other pairs, instead, only the architectural description and results are presented, without reporting the same level of details as for the case.

Chapter 6: Results

6.1: The case of Music cassette deck vs CD player

6.1.1: Product architecture

The music cassette deck is the non-digital version of the first pair of comparisons. It represents the technological predecessor to the CD player.

The music cassette deck was introduced in the 1960s, representing a tool to store audio as continuous magnetic patterns on a tape (Wikipedia, 2025).

The architecture of the cassette deck relies on a magnetic head contacting the tape surface to detect magnetic variations. Those variations are then converted into electrical signals (US44531189, 1981).

Conversely the CD player was commercially introduced in the 1980s following the Red Book standard, developed by Philips and Sony.

Architecturally, the CD player uses a laser pickup system combined with a photodiode detector, which is able to noncontact read digital data directly from the disc surface. (US4653039, 1985).

Moreover, differences in the architecture of the two products arise in the transport/signal processing components, as the cassette deck requires a mechanism to maintain the tape speed constant. This mechanism is composed of a capstan and a pinch roller.

The CD player instead utilises a single motor to rotate the disc and a system to maintain laser alignment. This eliminates the need for direct contact with the disc.

Moreover, the way the signal is delivered is different, as the cassette deck processes non-digital signals through an amplification circuit, whereas the digital product has a digital signal processor incorporated and a digital to non-digital converter to transform encoded digital bits into audio.

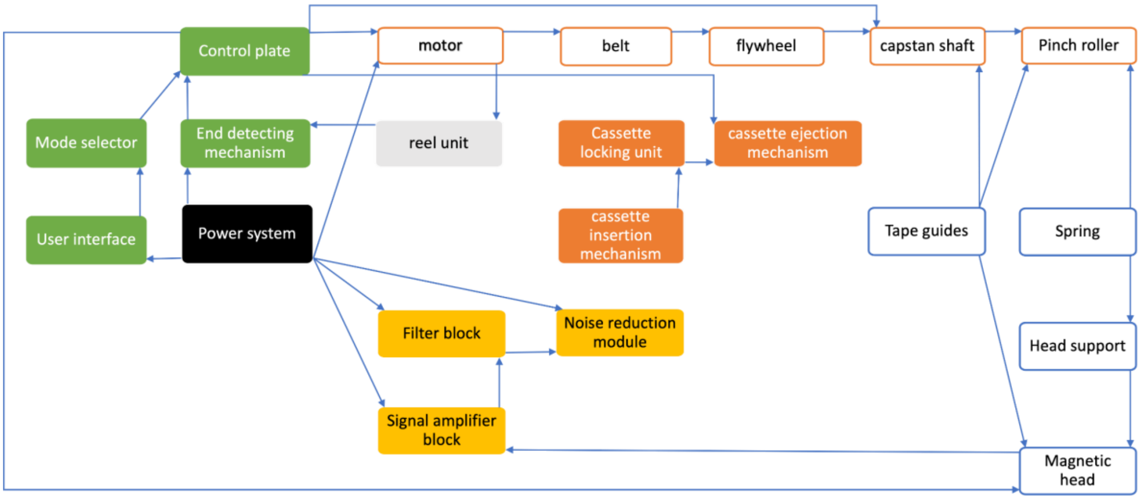
The transition from contact based to contactless digital data represents a shift quality-wise, with direct implications for component allocation.

6.1.2: Block Diagram, Design structure matrix and Functional tree

By building the block diagrams of the products we notice the aforementioned differences in structure. It is reported below the block diagram constructed for the cassette deck.

The non-digital product architecture is mechanical dominant: the transport chain, consisting in motor→ belt→ flywheel→ capstan, governs tape movement. The magnetic head and amplifier are following a sequential dependency.

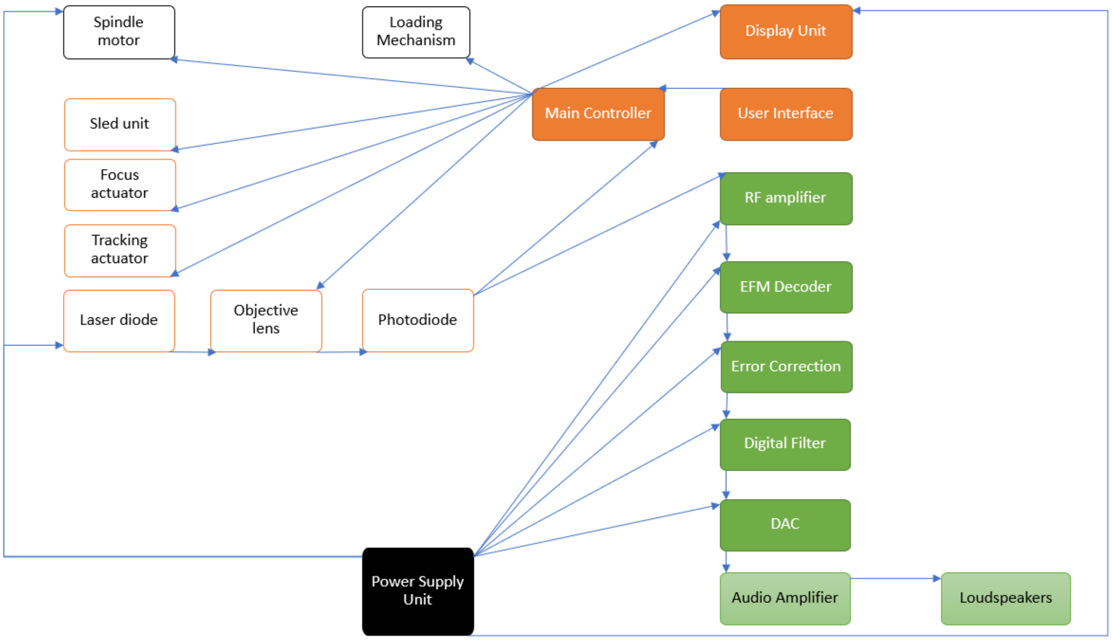
Figure 6.1a. block diagram music cassette deck



On the other side, the CD player exhibits distributed control logic. There are multiple subsystems which operate in autonomy, while being supervised by the main controller. The optical read path, formed by laser diode→ objective lens→ photodiode, functions independently from the signal processing (RF amplifier → EFM decoder→ error correction → DAC).

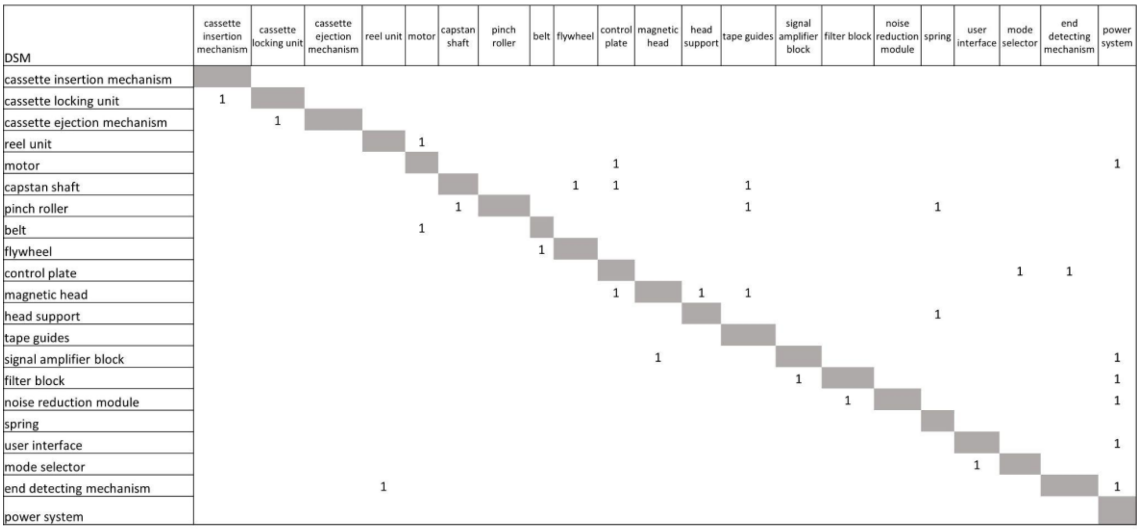
Architecturally, this separation of mechanical processes and digital ones is absent in the cassette deck where there are no intermediate layers.

Figure 6.1b. block diagram CD



Once having drawn the block diagrams, another useful tool is to represent graphically with a matrix the component relations. The tool used here is a Design Structure Matrix. From the image it is possible to see the block of components. It offers a compact tabular view and makes it easier to see patterns that are less evident in a graphical scheme.

Figure 6.1c. DSM cassette deck



DSM	cassette insertion mechanism	cassette locking unit	cassette ejection mechanism	reel unit	motor	capstan shaft	pinch roller	belt	flywheel	control plate	magnetic head	head support	tape guides	signal amplifier block	filter block	noise reduction module	spring	user interface	mode selector	end detecting mechanism	power system
cassette insertion mechanism	■																				
cassette locking unit	1	■																			
cassette ejection mechanism		1	■																		
reel unit				■	1																
motor					■					1											1
capstan shaft						■			1	1			1								
pinch roller						1	■						1				1				
belt					1			■													
flywheel								1	■												
control plate										■									1	1	
magnetic head										1	■	1	1								
head support												■					1				
tape guides													■								
signal amplifier block											1			■							1
filter block														1	■						1
noise reduction module															1	■					1
spring																	■				
user interface																		■			1
mode selector																		1	■		
end detecting mechanism				1																■	1
power system																					■

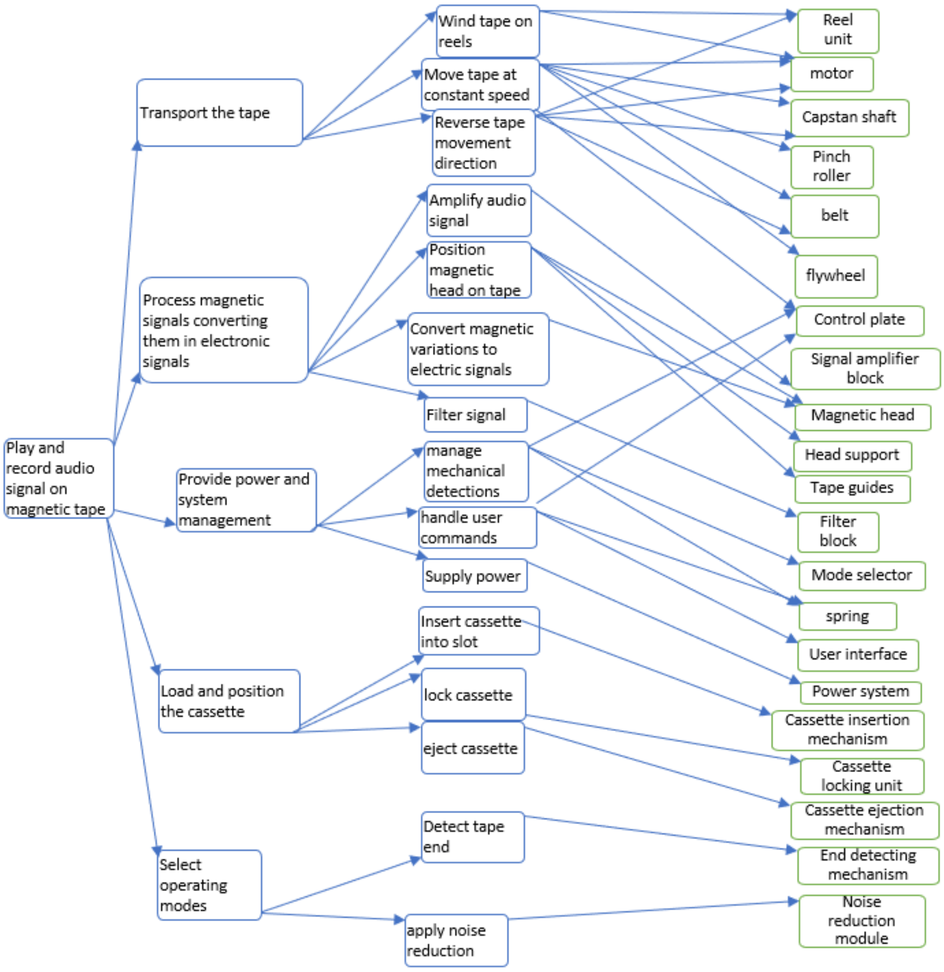
Figure 6.1d. DSM CD player

DSM	Laser diode	Objective lens	Photodiode	Focus actuator	tracking actuator	Sled unit	Spindle Motor	Loading Mechanism	RF amplifier	EFM Decoder	Error correction	Digital filter	DAC	Audio amplifier	Loudspeakers	User interface	Display unit	Main controller	Power supply unit
Laser diode																			1
Objective lens	1																	1	
Photodiode		1																1	
Focus actuator																		1	
tracking actuator																		1	
Sled unit																		1	
Spindle Motor																		1	1
Loading Mechanism																			
RF amplifier			1																
EFM Decoder									1										
Error correction										1									
Digital filter											1								
DAC												1							
Audio amplifier													1						
Loudspeakers														1					
User interface																			
Display unit																		1	1
Main controller			1													1			
Power supply unit																			

The analysis continues by developing the functional trees. Regarding the cassette deck we started from a generic function of the product called “Play and manage audio playback from optical disc”. Then, it gets divided into second level functions, which are “Manage CD operations”, “Read and decode digital data”, “Convert to audio output”, “Control system operations”, “Managing power and timing”. Beyond this the elementary functions are listed and connected to the elementary components. From this coupling we can notice how many components are serving different functions, i.e. the control plate of the cassette deck contributes to reversing the tape movement direction, handling user commands and managing mechanical detections.

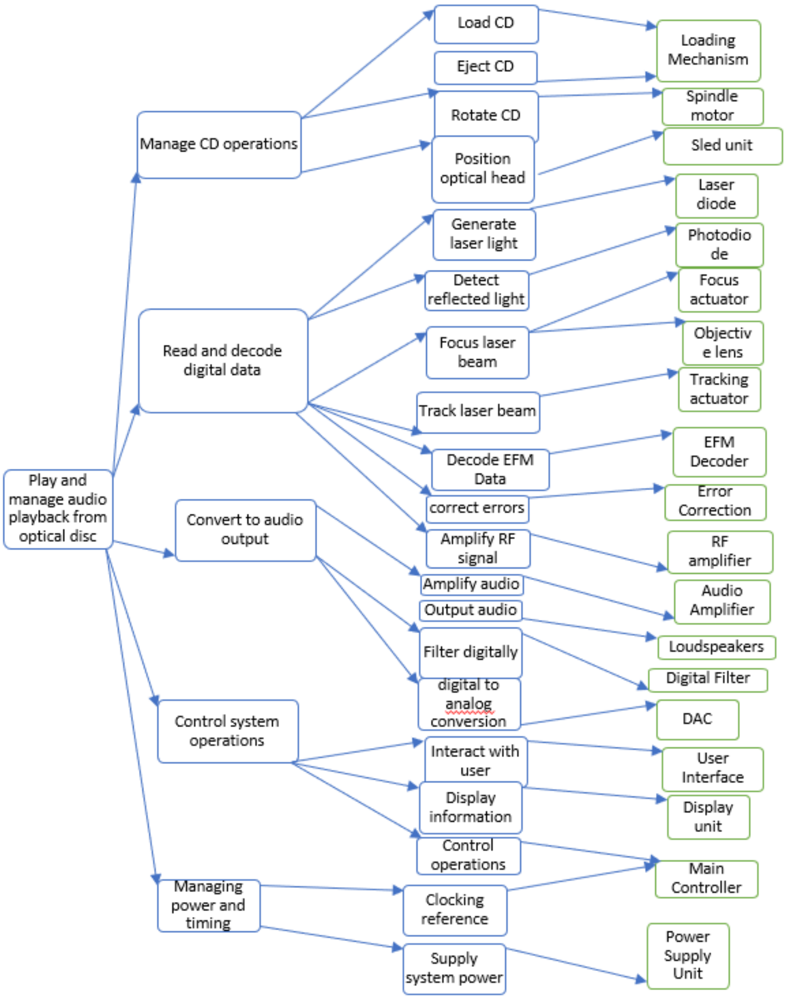
Overall, the connections are plenty and no isolated subsystems can be pointed out.

Figure 6.1e. FT CASSETTE



Instead in the cd player's functional decomposition, the operations remain isolated from data reading, Digital signal processing functions as "Decode EGM data", "Correct errors", "Filter digitally" for a processing chain operating independently from the optical mechanics. Each layer can be updated or substituted without redesigning adjacent layers.

Figure 6.1f. Functional tree CD player

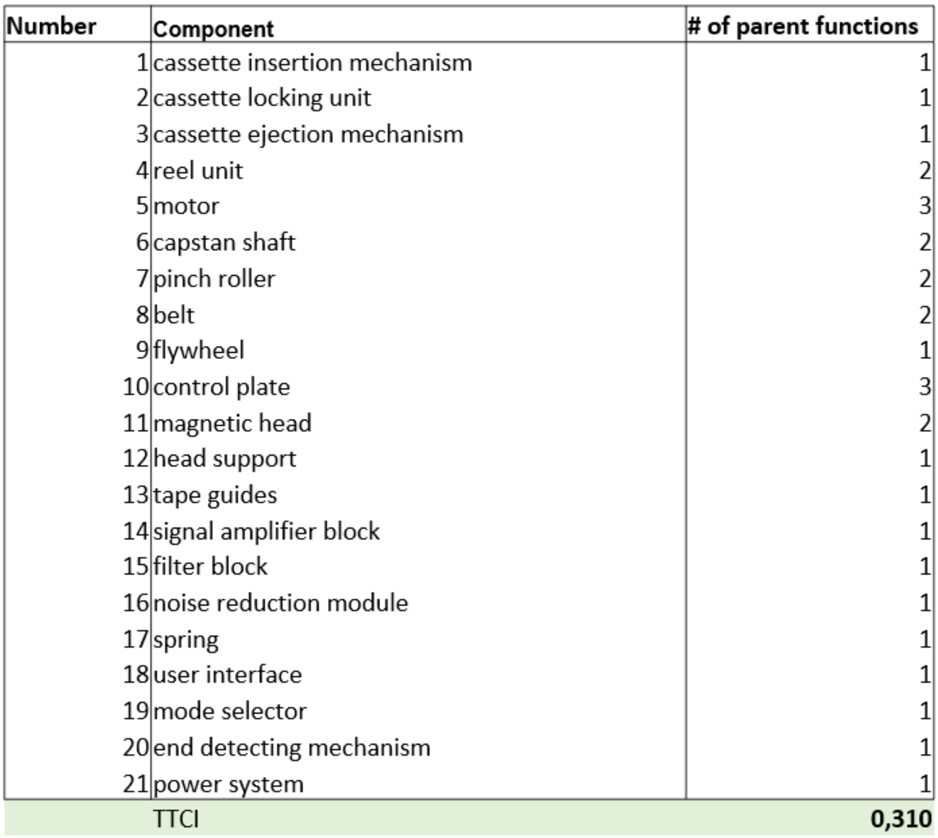


6.1.3 TCCI results and FCA map

Following the architectural decomposition of the pair of products, a commonality index calculation was performed to quantify the sharing of components among the functions of the product. The Total Commonality Index metric, bounded between 0 and 1, measures the degree to which components are reused across functional requirements.

Starting off with the non-digital product, the cassette deck has 21 components, numerous of which supporting more than one function. There is a substantial component reuse evident in elements such as the motor (3 parent functions), the control plate (3 parent functions), resulting in a TCCI of 0,310.

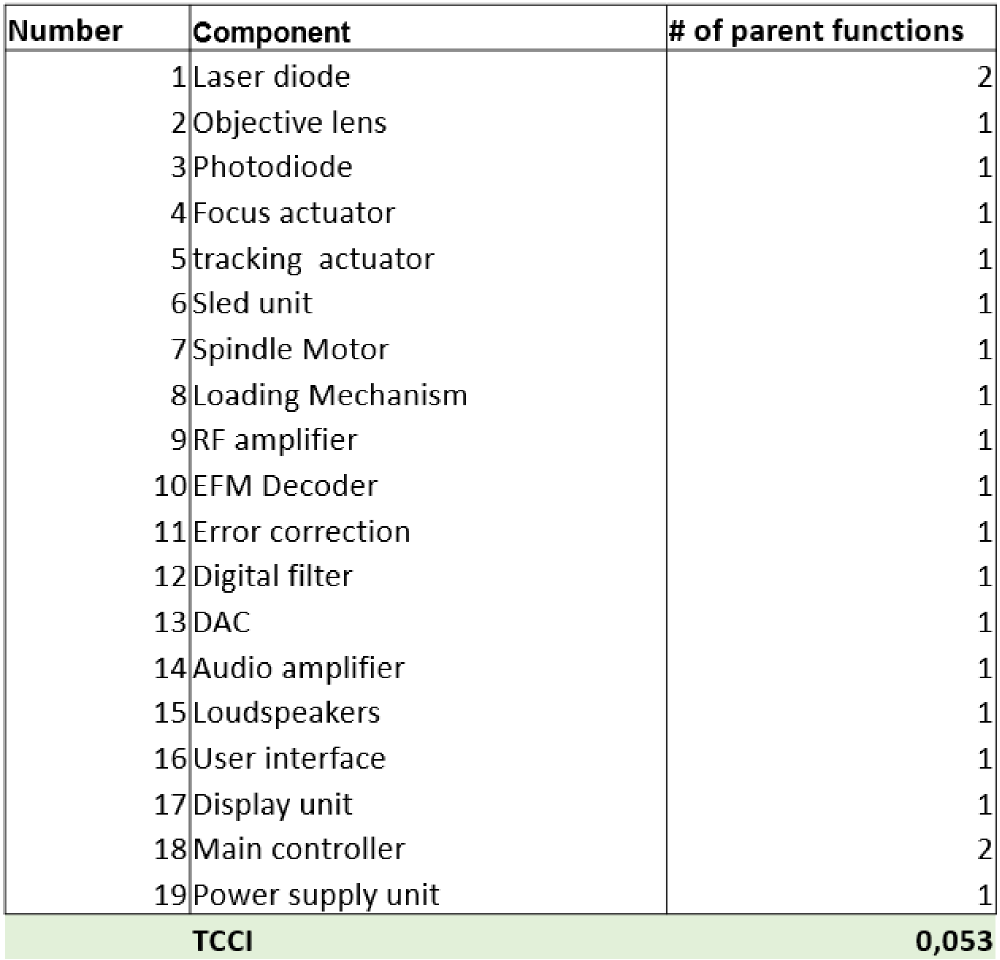
Figure 6.1g. TCCI CASSETTE



Number	Component	# of parent functions
1	cassette insertion mechanism	1
2	cassette locking unit	1
3	cassette ejection mechanism	1
4	reel unit	2
5	motor	3
6	capstan shaft	2
7	pinch roller	2
8	belt	2
9	flywheel	1
10	control plate	3
11	magnetic head	2
12	head support	1
13	tape guides	1
14	signal amplifier block	1
15	filter block	1
16	noise reduction module	1
17	spring	1
18	user interface	1
19	mode selector	1
20	end detecting mechanism	1
21	power system	1
	TTCI	**0,310**

By contrast, the cd player's components operate with high specialization. A TCCI of 0,053 corresponds to minimal functional reuse, and only the laser diode and main controller contribute to different functions out of the total 19 identified components.

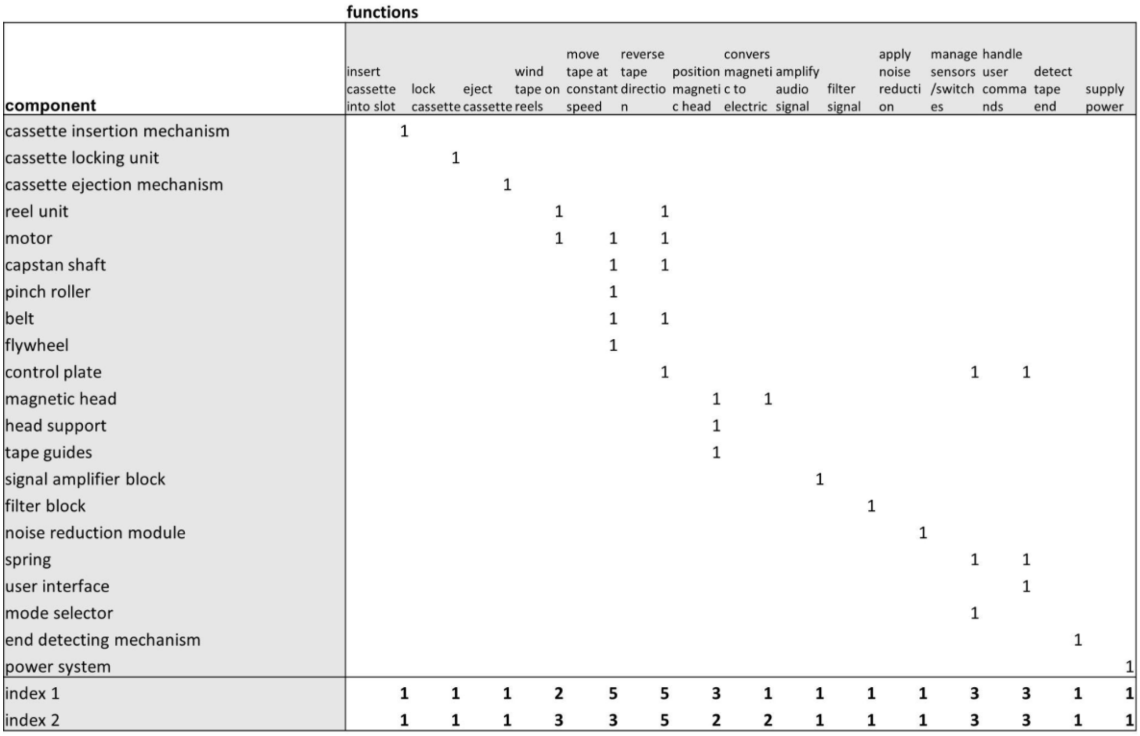
Figure 6.1h. TCCI CD PLAYER



Number	Component	# of parent functions
1	Laser diode	2
2	Objective lens	1
3	Photodiode	1
4	Focus actuator	1
5	tracking actuator	1
6	Sled unit	1
7	Spindle Motor	1
8	Loading Mechanism	1
9	RF amplifier	1
10	EFM Decoder	1
11	Error correction	1
12	Digital filter	1
13	DAC	1
14	Audio amplifier	1
15	Loudspeakers	1
16	User interface	1
17	Display unit	1
18	Main controller	2
19	Power supply unit	1
	TCCI	**0,053**

Furthermore, the Functional Component Allocation map has been developed.

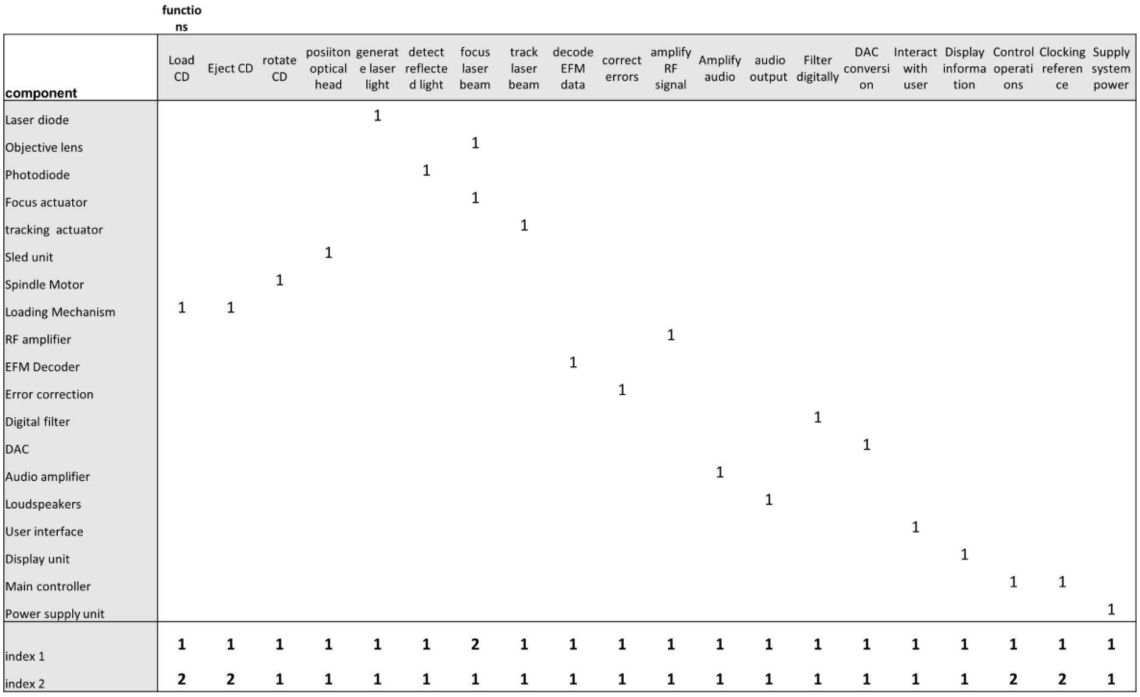
Figure 6.1i. FCA Matrix cassette deck



functions

component	insert cassette into slot	lock cassette	eject cassette	wind tape on reels	move tape at constant speed	reverse tape direction	position magnetic head	convers magnetic to electric	amplify audio signal	filter signal	apply noise reduction	manage sensors /switches	handle user commands	detect tape end	supply power
cassette insertion mechanism	1														
cassette locking unit		1													
cassette ejection mechanism			1												
reel unit				1		1									
motor				1	1	1									
capstan shaft					1	1									
pinch roller					1										
belt					1	1									
flywheel					1										
control plate						1						1	1		
magnetic head							1	1							
head support							1								
tape guides							1								
signal amplifier block									1						
filter block										1					
noise reduction module											1				
spring												1	1		
user interface													1		
mode selector												1			
end detecting mechanism														1	
power system															1
index 1	**1**	**1**	**1**	**2**	**5**	**5**	**3**	**1**	**1**	**1**	**1**	**3**	**3**	**1**	**1**
index 2	**1**	**1**	**1**	**3**	**3**	**5**	**2**	**2**	**1**	**1**	**1**	**3**	**3**	**1**	**1**

Figure 6.1l. FCA Matrix cd Player



functions

component	Load CD	Eject CD	rotate CD	posiiton optical head	generate laser light	detect reflected light	focus laser beam	track laser beam	decode EFM data	correct errors	amplify RF signal	Amplify audio	audio output	Filter digitally	DAC conversion	Interact with user	Display information	Control operations	Clocking reference	Supply system power
Laser diode					1															
Objective lens							1													
Photodiode						1														
Focus actuator							1													
tracking actuator								1												
Sled unit				1																
Spindle Motor			1																	
Loading Mechanism	1	1																		
RF amplifier											1									
EFM Decoder									1											
Error correction										1										
Digital filter														1						
DAC															1					
Audio amplifier												1								
Loudspeakers													1							
User interface																1				
Display unit																	1			
Main controller																		1	1	
Power supply unit																				1
index 1	**1**	**1**	**1**	**1**	**1**	**1**	**2**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
index 2	**2**	**2**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**2**	**2**	**1**

Through the FCA map it is possible to locate every function in one of the four regions to categorize the functions:

- Functions in the lower left region of the map exhibit a modular-like behaviour: in the case of the CD player most of the functions are located here, which means that the digital product's structure here is mainly modular.

- The upper left region, instead, is representing functions having one component delivering several functions. This is the case for the CD player, as functions such as “load CD”, “eject CD” are both provided by the “loading mechanism” set of components.
- The lower right region represents functions served by multiple components that each have limited additional responsibilities; in this comparison, looking at the Cassette Deck FCA chart, the function “position magnetic head” falls here.
- Finally, the upper right area contains integral-complex functions. For the cassette deck, «reverse tape direction» is here, meaning it is provided by multiple components, whereas for the CD player there are no integral-complex functions.

Figure 6.1m. FCA CASSETTE

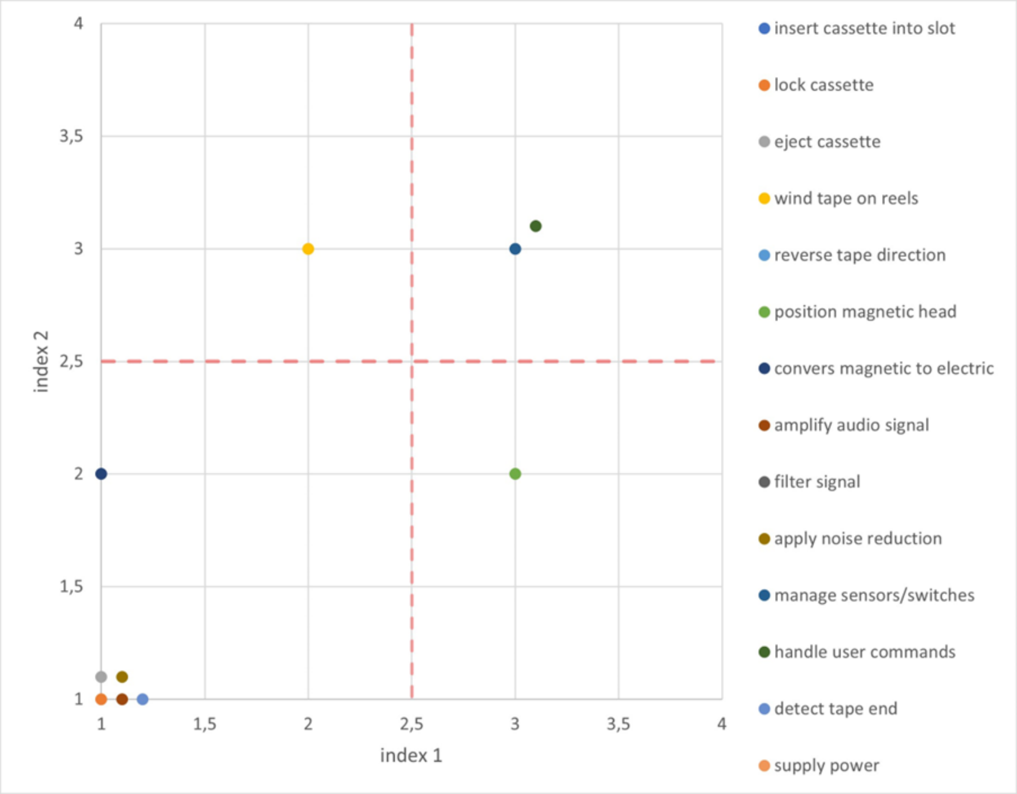
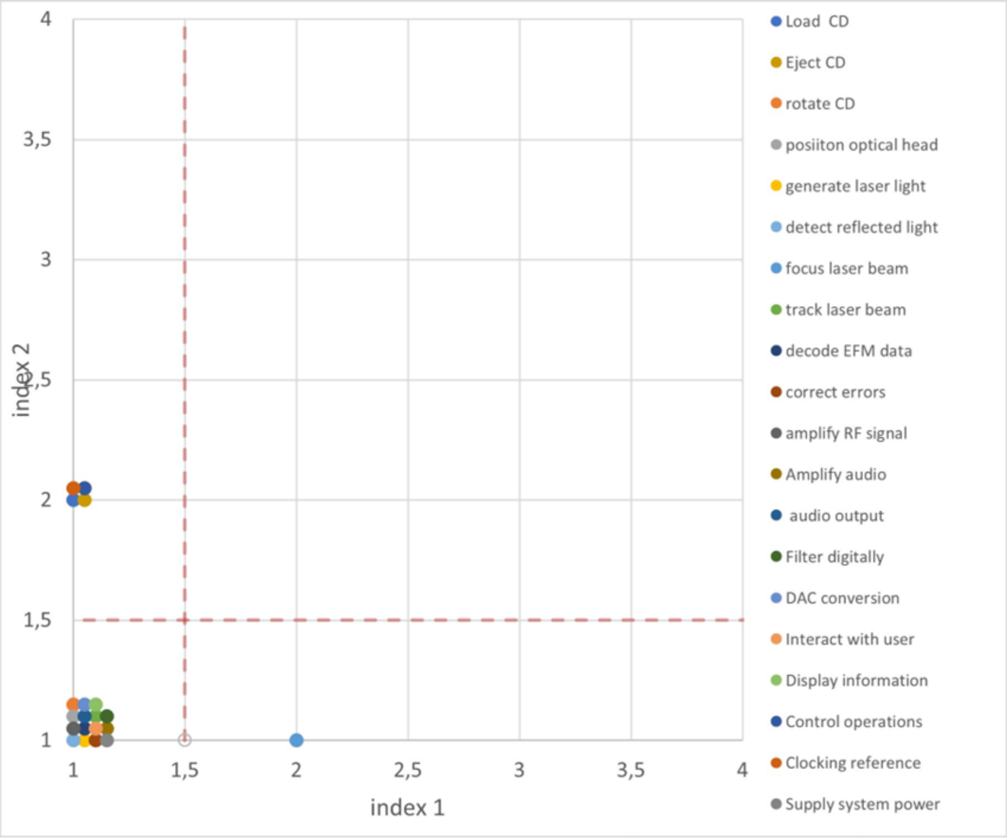


Figure 6.1n. FCA CD PLAYER



Overall, the FCA map reveals architectural differences: CD player's functions are mostly located in the lower left region, while the music cassette deck structure has three critical functions being integral complex, demonstrating clear differences in structure between non-digital and digital solutions.

These conclusions validate the Total Commonality Index findings: the music cassette deck, which has a TTCI of 0,310, is more integrated than the CD player. The CD player's TCCI of 0,055 confirms that mechanical constraints create more interdependencies rather than digital processing features.

6.2: Food processor vs Thermomix

6.2.1 Product architecture

Food processors emerged in the 1970s as versatile kitchen tools designed to perform multiple cutting and chopping. The main mechanism is made of interchangeable blades and discs (US3892365, 1974).

The Thermomix was developed later, as an integrated cooking appliance with programmable heating, and also automated stirring (US10159382, 2014) represents an evolution digitally controlled compared to the traditional mixing products.

Both the food processor and Thermomix share the fundamental requirement of containing food. Nonetheless they also share plenty of elementary functions, such as transmitting rotational motion to processing tools, ensuring user safety. Their approach diverges on how these shared functions are architecturally realized.

The food processor architecture is organised in five primary functional blocks:

- the food containment system with processing bowl, lid, bowl lock;
- mechanical drive transmission with electric motor, driveshaft and gear unit;
- interchangeable processing tools: blade, cutting and grating disc, juicing basket.
- control system: control knob and speed regulator
- support structure: base housing and rubber feet

The motor to tool coupling is achieved via a direct mechanical transmission through the shaft and driving coupling, where a single electric motor provides rotational energy being modulated by the speed regulator. Notably, both the motor and power supply unit are participating in multiple functions as well as the processing bowl.

Figure 6.2a. Block diagram food processor

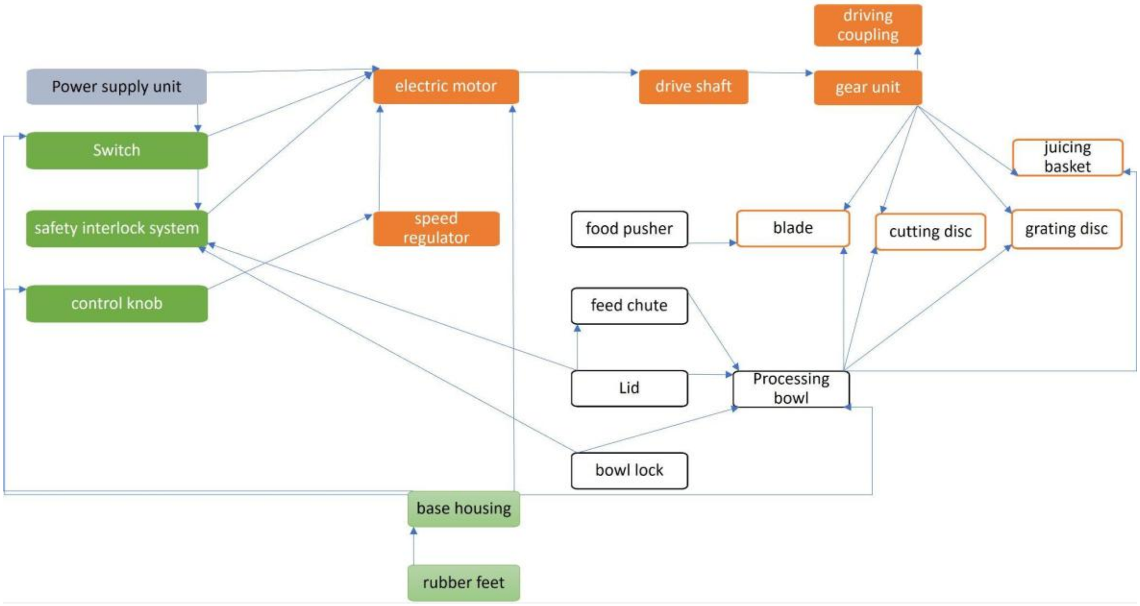
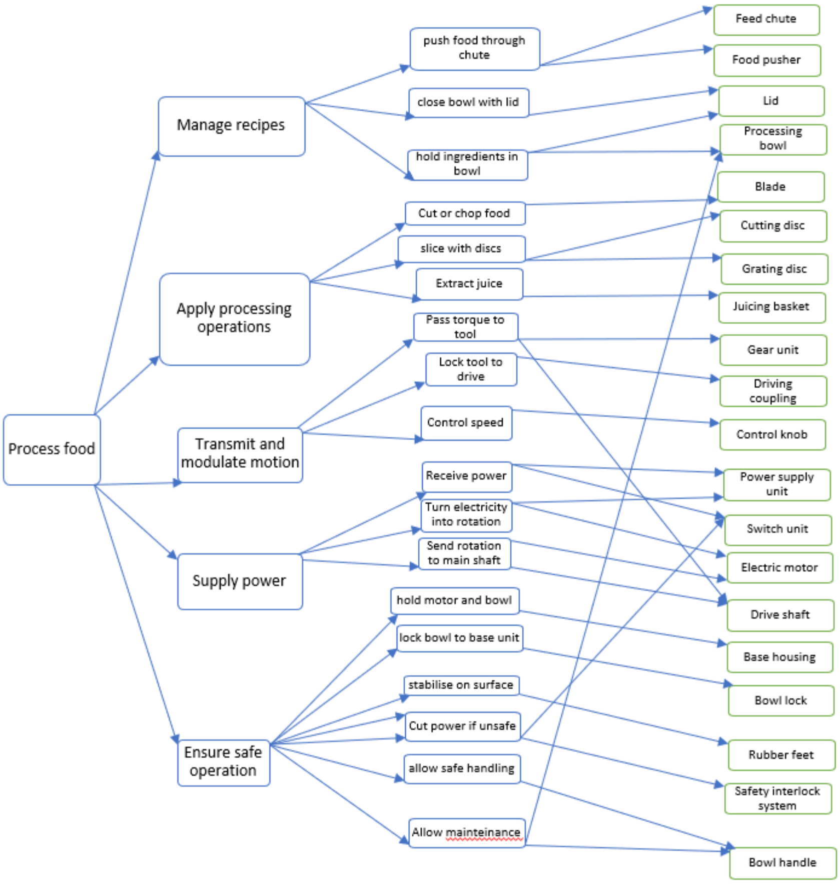


Figure 6.2b. Functional tree food processor



In contrast, the Thermomix structure is characterised by the following layers

- thermal generation system
- digital processing unit
- network connectivity layer
- user interface

Overall, the two products have similarities in the mechanical part: the motor, drive transmission, blade, bowl and lid. Nevertheless, the Thermomix delegates the control of these elements to a centralized microprocessor, separating them from the user input. Furthermore, the Thermomix introduces temperature regulation, recipe automation and remote connectivity, which were not present in the non-digital product.

Figure 6.2c. Block diagram Thermomix

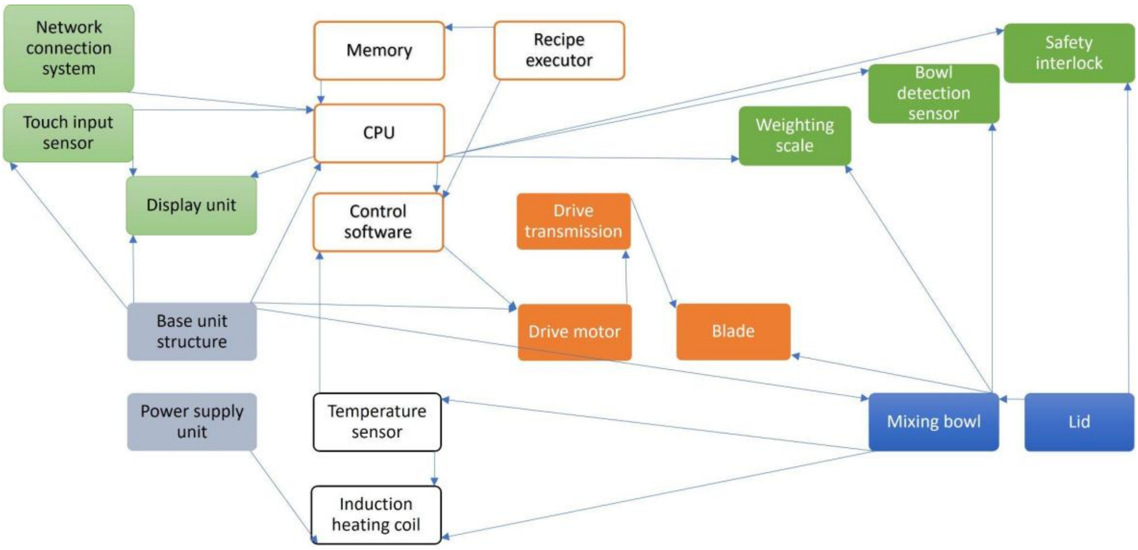
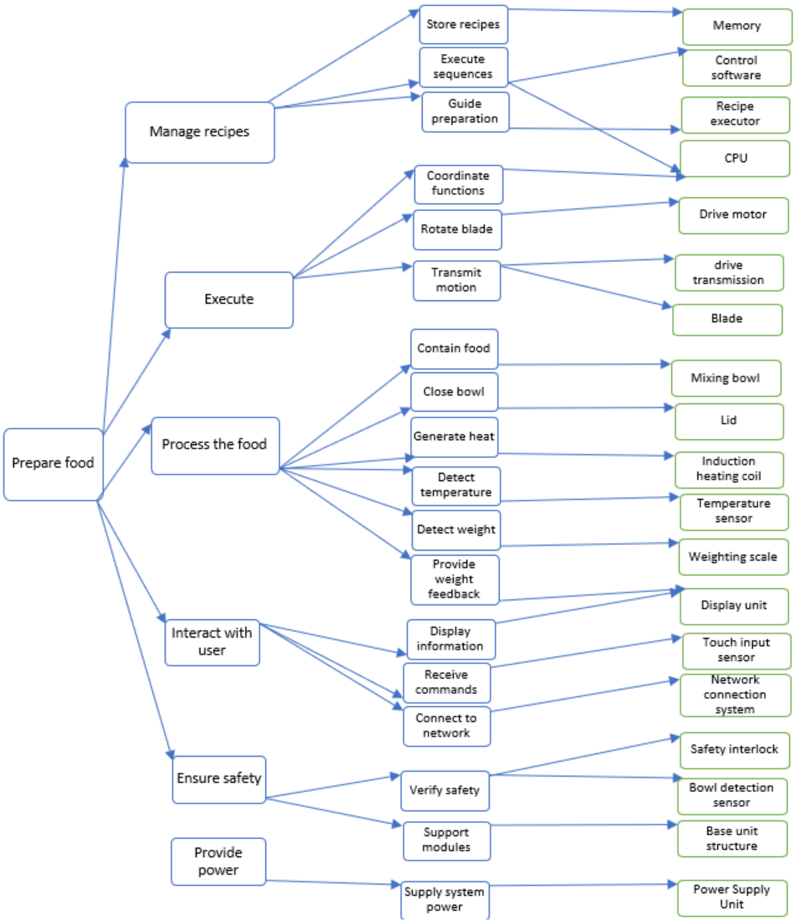


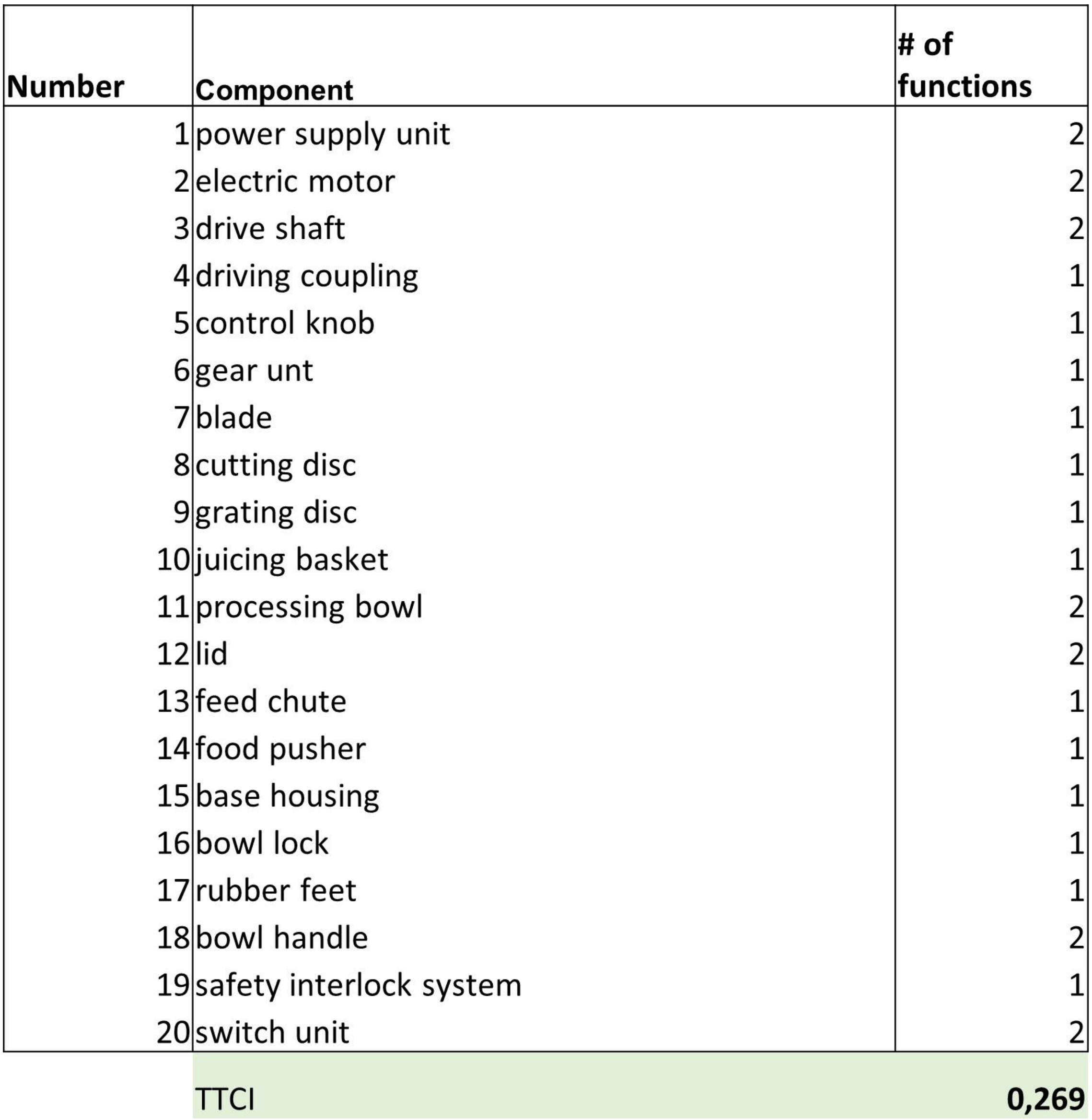
Figure 6.2d. Functional tree Thermomix



6.2.2 TCCI results and FCA map

The food processor is characterised by its mechanical design, so the commonality index results in 0,269. The electric motor generates rotational energy while being essential to motion generation; the drive shaft transmits and modulates this motion across tool changes; the processing bowl, lid and bowl handle each fulfil multiple purposes as well. These overlapping responsibilities reflect the integration requirements of purely mechanical systems.

Figure 6.2e. TCCI food processor



Number	Component	# of functions
1	power supply unit	2
2	electric motor	2
3	drive shaft	2
4	driving coupling	1
5	control knob	1
6	gear unt	1
7	blade	1
8	cutting disc	1
9	grating disc	1
10	juicing basket	1
11	processing bowl	2
12	lid	2
13	feed chute	1
14	food pusher	1
15	base housing	1
16	bowl lock	1
17	rubber feet	1
18	bowl handle	2
19	safety interlock system	1
20	switch unit	2
	TTCI	**0,269**

Figure 6.2f. TCCI Thermomix

Number	Component	# of parent functions
1	CPU	2
2	memory	1
3	display	2
4	touch input sensor	1
5	Network connection system	1
6	recipe executor	1
7	control software	1
8	induction heating coil	1
9	temperature sensor	1
10	power supply unit	1
11	mixing bowl	1
12	blade	1
13	drive motor	1
14	drive transmission	1
15	lid	1
16	safety interlock	1
17	bowl detection sensor	1
18	base unit structure	1
19	weighing scale	1
	TCCI	**0,150**

The Thermomix FCA map displays a markedly different pattern-; most functions are distributed across the lower left and right quartans: the thermal mechanical and control functions have minimal interdependence. Weight detecting but also heat generating and recipes storage are all served by isolated components.
Furthermore, functions requiring coordination are delegated to the CPU, which acts as a centralised mediator of information flows.

These analyses reveal that digitalisation enables architecture simplification not by reducing complexity, but by reframing how complexity is managed, replacing mechanical coupling with information coupling.

Figure 6.2g. FCA food processor

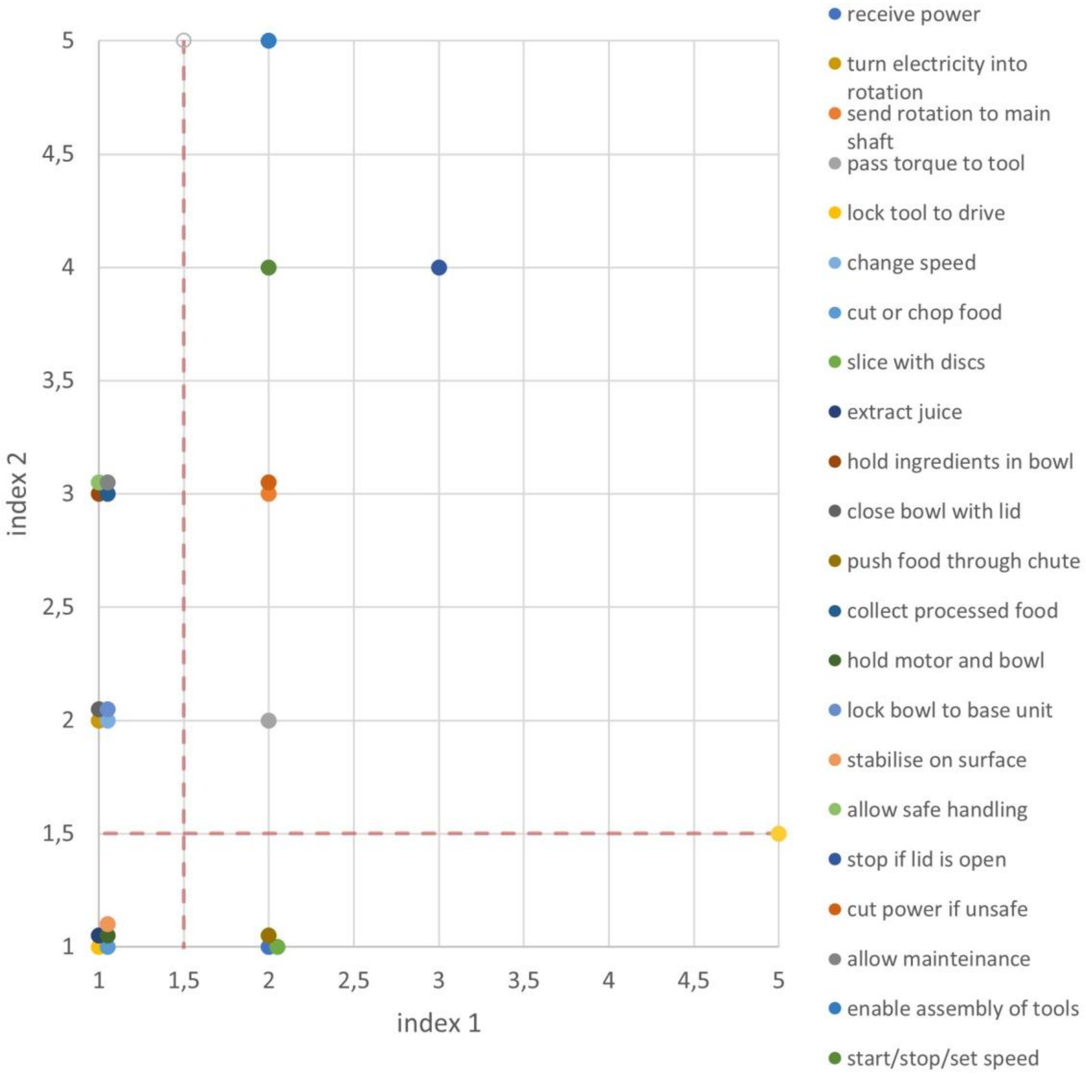
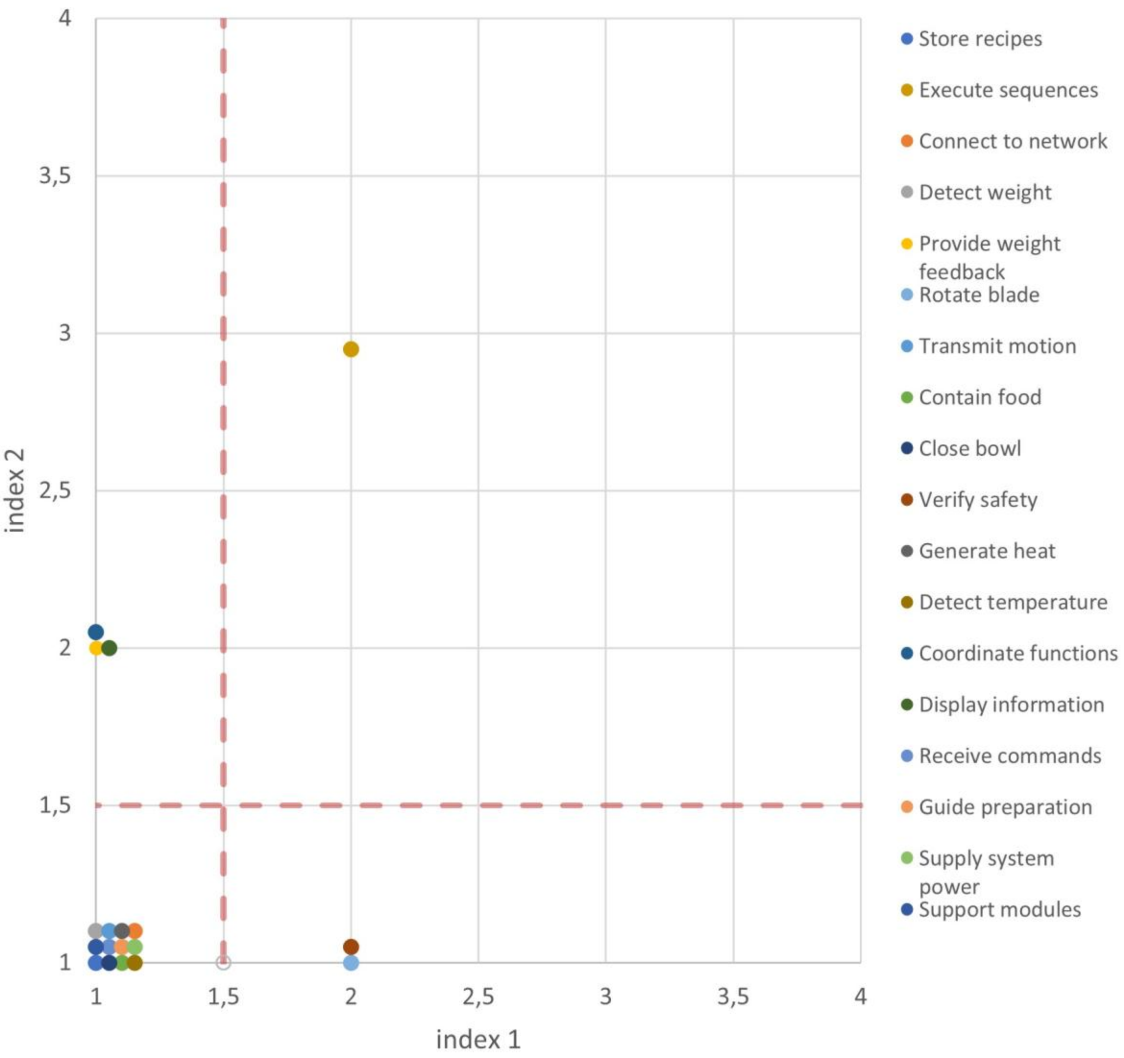


Figure 6.2g. FCA thermomix



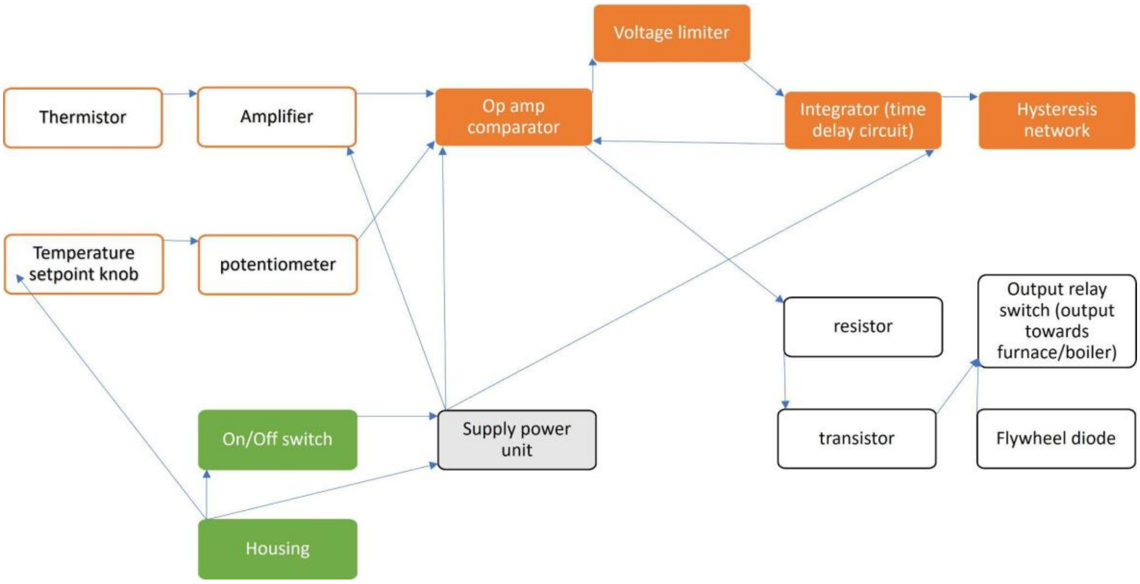
6.3: Electronic thermostat vs Digital thermostat

6.3.1 Product architecture

The third pair to be compared is the electronic versus the digital thermostat, which allow to regulate the ambient temperature by switching a furnace or a boiler on and off, doing so with different internal architectures.

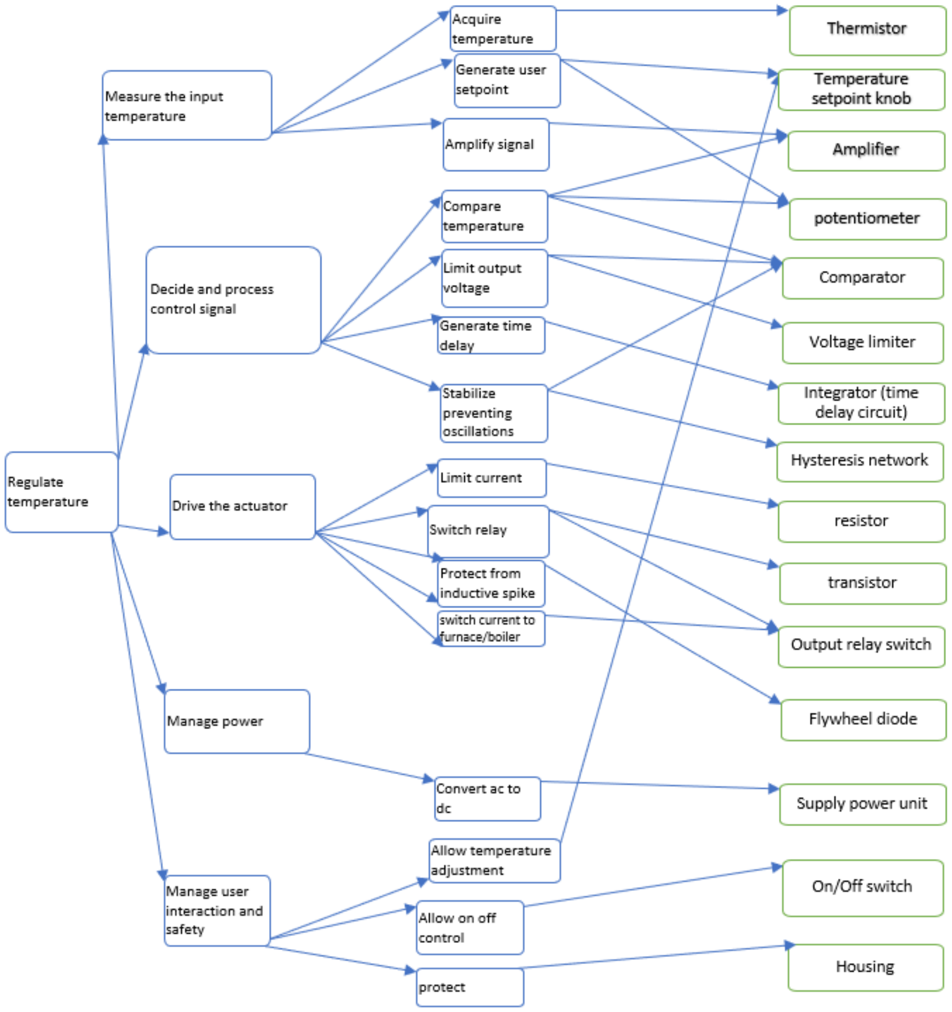
The electronic thermostat operates through non-digital signal processing, relying on components to monitor and respond to temperature variations (US4177860, 1979). Its architecture is organized around four big blocks: the thermistor and amplifier unit; the control unit, composed of comparator, integrator and hysteresis network and the user interface (temperature setpoint knob and on (off switch), and the power drive unit (relay switch and protection circuit).

Figure 6.3a. BD ELECTRONIC THERMOSTAT



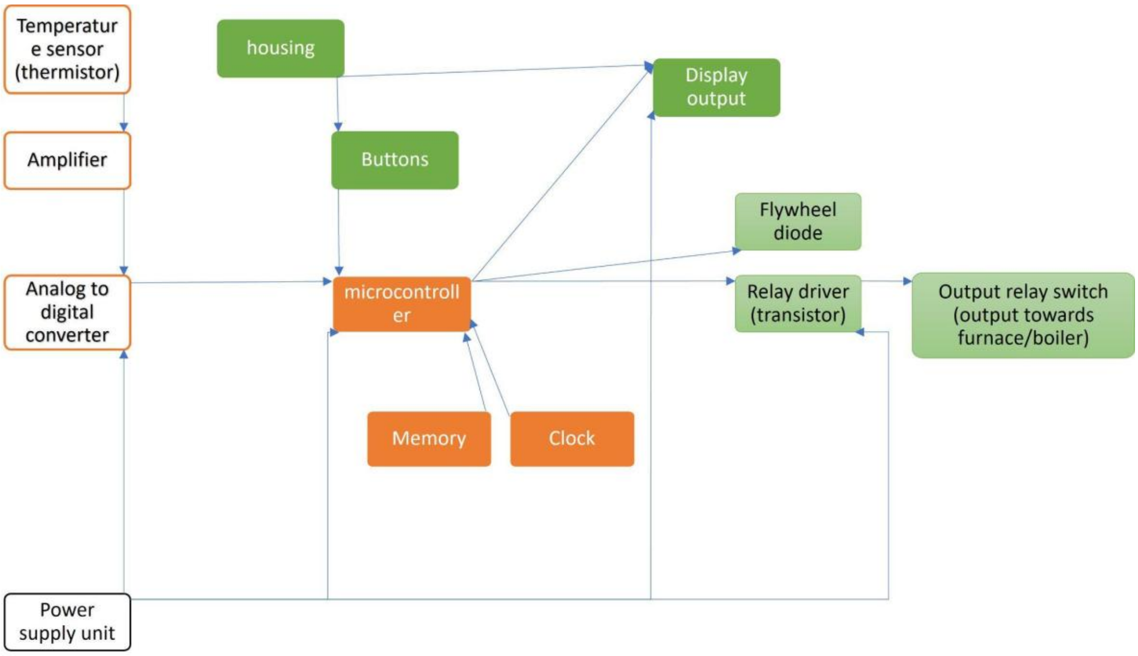
Its functional tree reveals that temperature regulation is achieved through several components operating simultaneously: the comparator manages three distinct functions, hysteresis stabilizations, current limitation, temperature comparison; creating an early component function binding where a single element serves multiple control purposes. Another key component here is the amplifier, which contributes to acquire the signal and amplify functions demonstrating component reuse at the non-digital processing level (US4072866A, 1978).

Figure 6.3b. FT ELECTRONIC THERMOSTAT



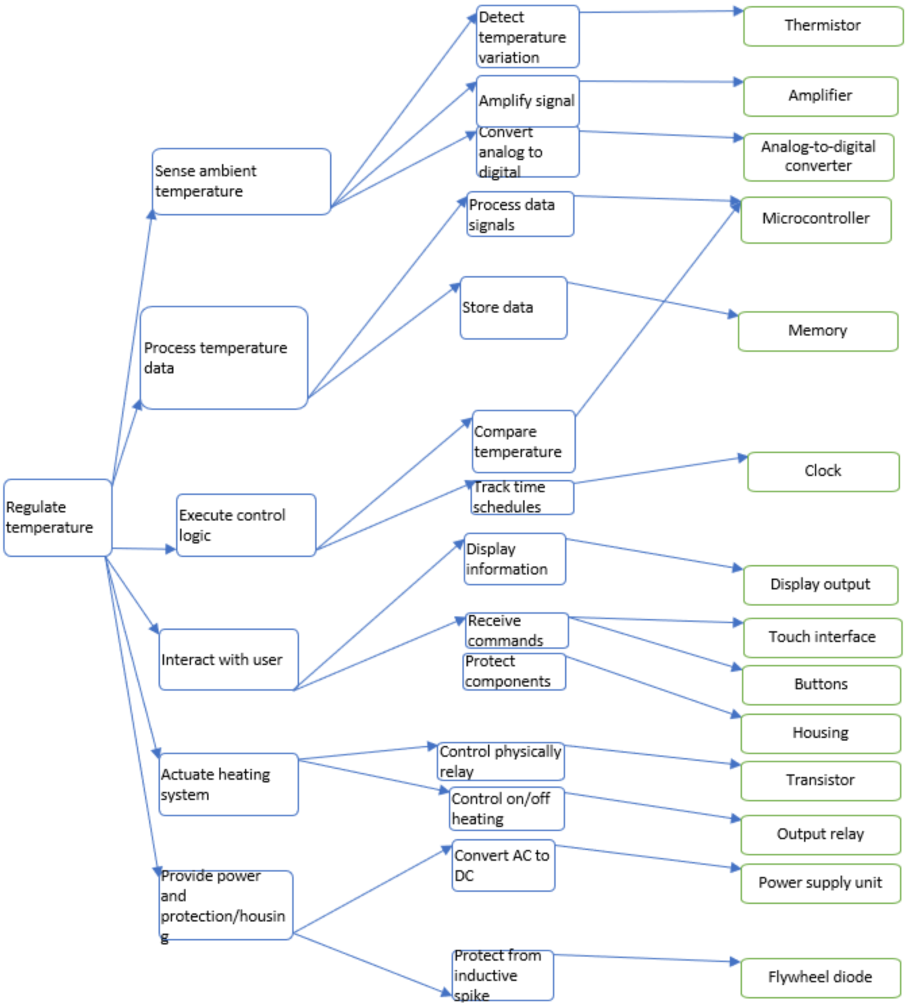
Describing the digital thermostat, it introduces an architecture based on a microcontroller. It fundamentally reorganises the decomposition of functions, with five blocks: the digital processing unit (microcontroller, clock, memory=, the sensing unit (thermistor and non-digital to digital converter) the user interface (buttons and display output= the power management unit (power supply), and the output control (relay driver and flywheel diode) reflect a distinct separation between data processing and physical actuation (US6824069, 2004).

Figure 6.3c. BD DIGITAL THERMOSTAT



Moreover, the functional tree shows how the microcontroller takes a central role in the digital thermostat, temperature comparison, data storage, schedule tracking control logic execution are all managed by it. Nevertheless, each component maintains a more specialized role compared to its non-digital counterpart, with the microcontroller being the only component participating in multiple functions.

Figure 6.3d. FT DIGITAL THERMOSTAT



6.3.2 TCCI results and FCA map

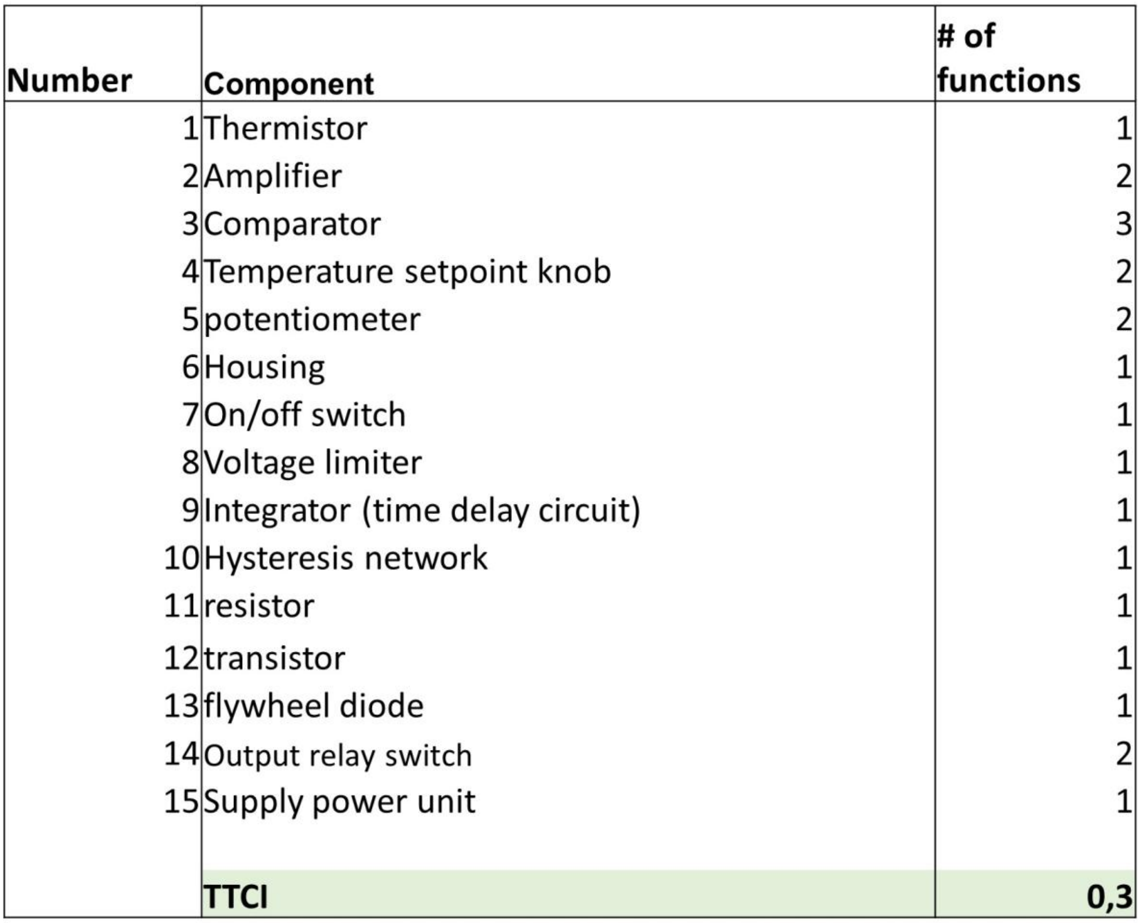
Following the architectural analysis, the relations between elementary functions and components have been used to compute the Total Commonality Index and find out the degree of component-sharing among functions.

Electronic thermostat has fifteen elementary components, achieving functional integration through component resume in its control circuit. The comparator contributes not only to temperature comparison but also to hysteresis stabilisation and preventing oscillations; other components are useful for different reasons such as the potentiometer. The output relay manages both heating activation and current switching. This pattern of multi-function component participation yields a higher commonality value, resulting in a value of 0,300.

Conversely, the digital thermostat with 13 components identified, exhibits different relationships: the microcontroller participates in data processing and logic execution and is the only component with multiple uses. The other 12 components maintain single functional assignments.

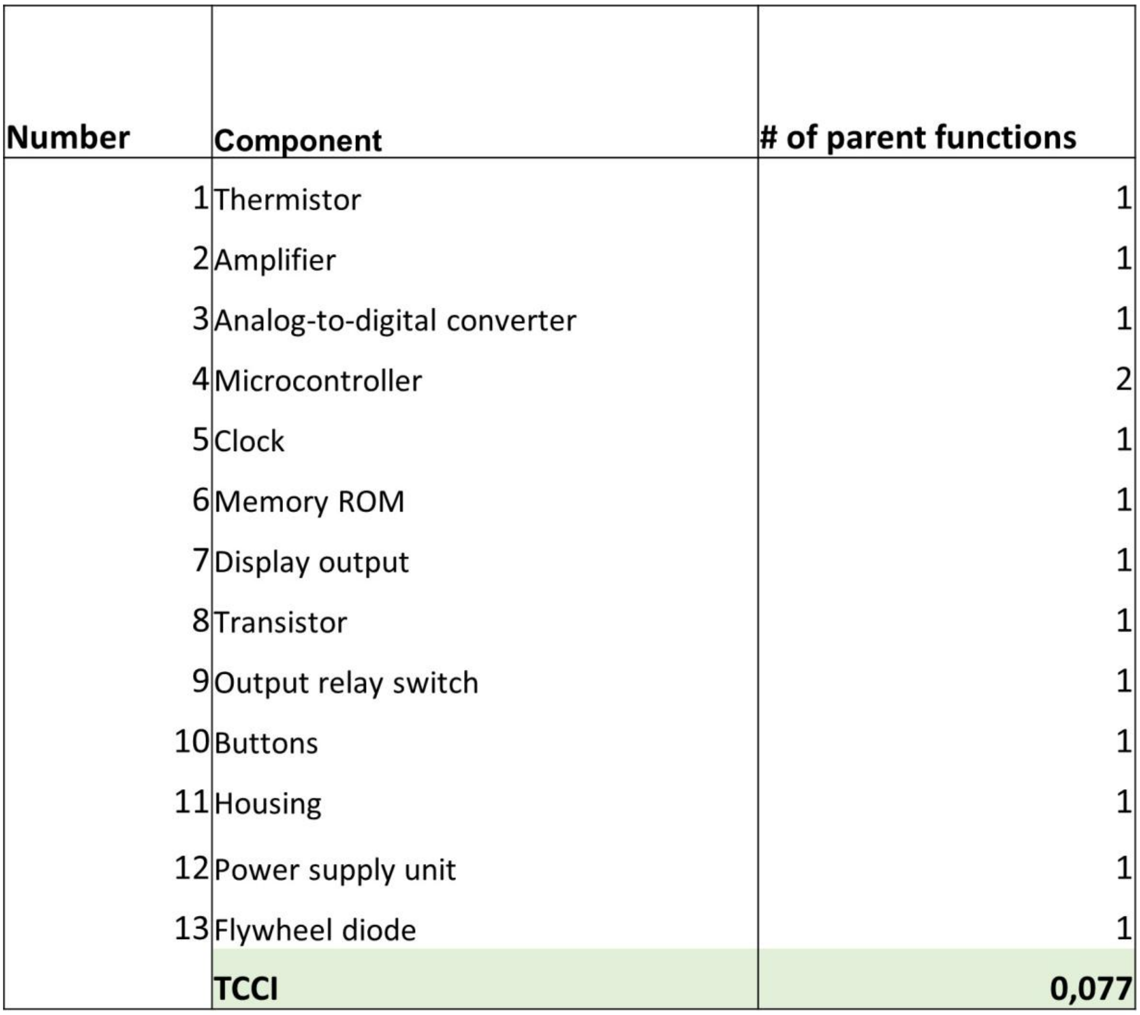
The TCCI of 0,070 reflects minimal component commonality and a high functional specialization, and this aspect is also deductible from the Function Component allocation map.

Figure 6.3e. TCCI ELECTRONIC THERMOSTAT



Number	Component	# of functions
1	Thermistor	1
2	Amplifier	2
3	Comparator	3
4	Temperature setpoint knob	2
5	potentiometer	2
6	Housing	1
7	On/off switch	1
8	Voltage limiter	1
9	Integrator (time delay circuit)	1
10	Hysteresis network	1
11	resistor	1
12	transistor	1
13	flywheel diode	1
14	Output relay switch	2
15	Supply power unit	1
	TTCI	**0,3**

Figure 6.3f. TCCI DIGITAL THERMOSTAT



Number	Component	# of parent functions
1	Thermistor	1
2	Amplifier	1
3	Analog-to-digital converter	1
4	Microcontroller	2
5	Clock	1
6	Memory ROM	1
7	Display output	1
8	Transistor	1
9	Output relay switch	1
10	Buttons	1
11	Housing	1
12	Power supply unit	1
13	Flywheel diode	1
	TCCI	**0,077**

The functional component allocation maps reveal complementary architectural patterns between the two thermostats.

The non-digital product FCA distributions show critical functions in the integral complex quadrant (upper right), including temperature comparison and oscillations prevention, which require coordinated operation across comparator, hysteresis network and related non-digital components. Conversely the digital thermostat FCA map displays most functions concentrated in the lower left and lower right regions. There are no integral complex functions appearing in the upper right quadrant, revealing the decoupled nature of the digital version.

Figure 6.3g. FCA ELECTRONIC THERMOSTAT

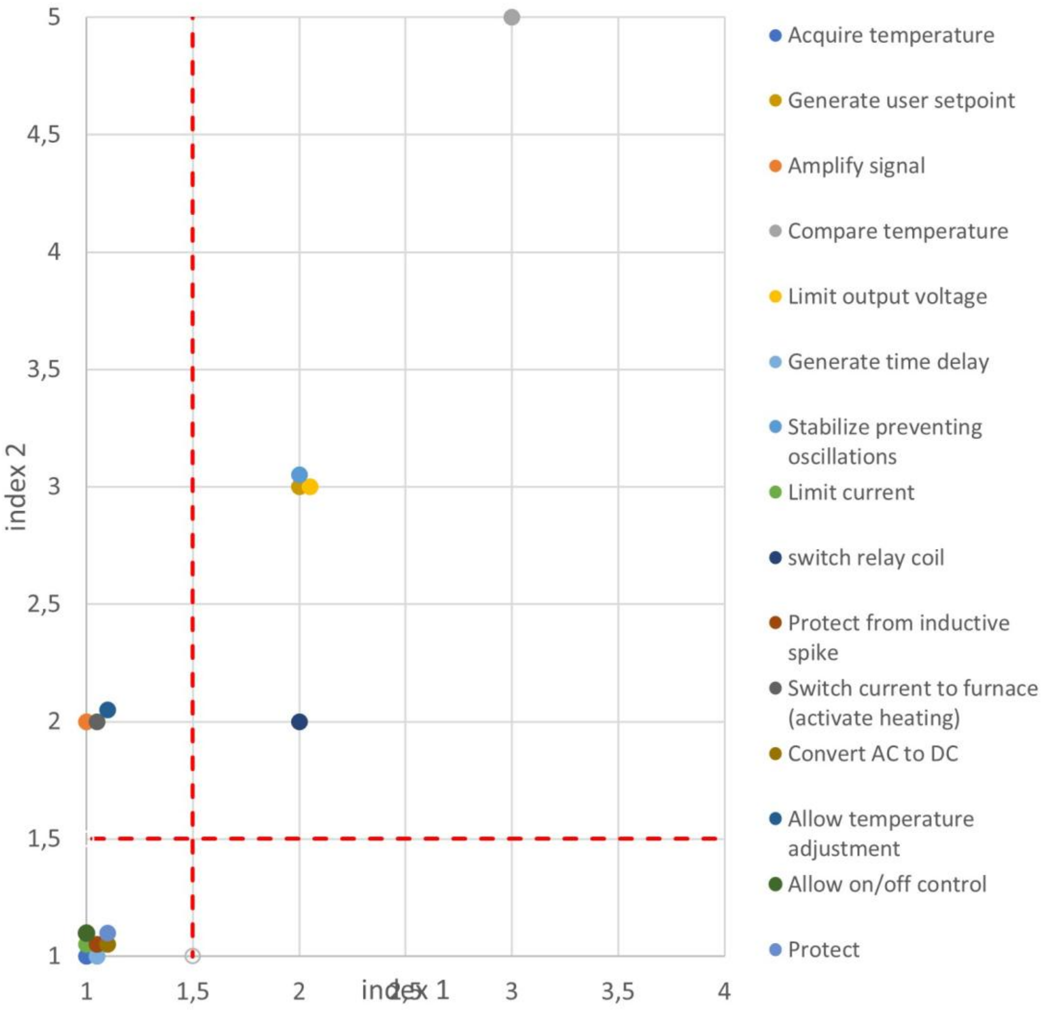
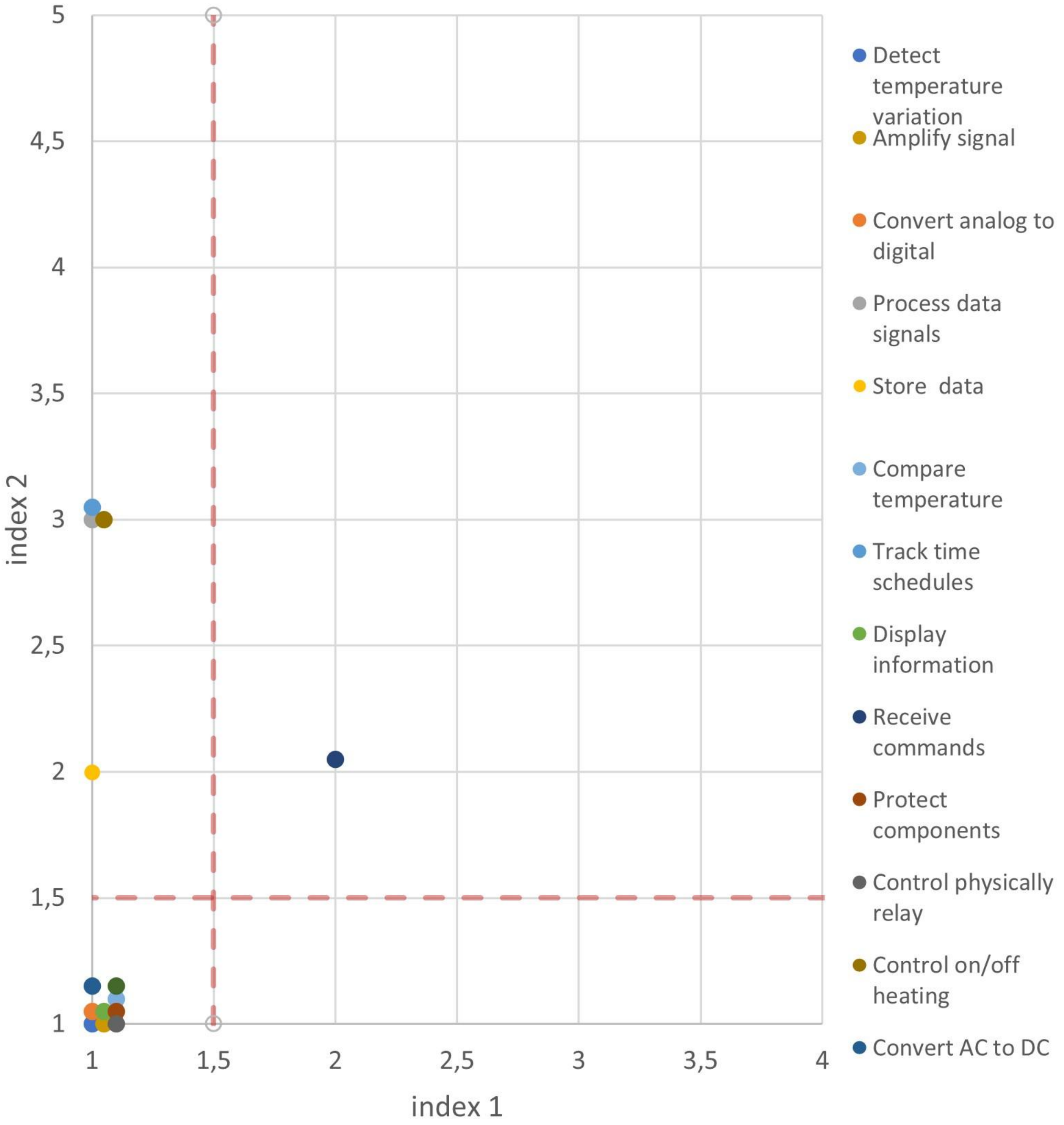


Figure 6.3h. FCA DIGITAL THERMOSTAT



6.4. Cyclone vacuum cleaner vs Autonomous cleaning robot

6.4.1. Product architecture

The cyclone vacuum cleaner emerged during the 1980s pioneered by James Dyson, who introduced the dual cyclone technology to improve dust separation and also maintaining suction power without clogging traditional filters (US4733228, 1983).

The cyclone vacuum cleaner architecture in five primary functional blocks:

- the intake and suction unit consisting of vacuum motor, impeller and cleaning head inlet
- the airflow management system: air channel, hose unit and cyclone chamber

- the user control interface: on/off button and handle
- the dust collection and filtration unit: cyclone chamber, filter block, bin
- the mobility support: wheels and main body

The motor driven impeller operates by generating a pressure differential which draws air and dirt through the system, while the cyclone chamber separates the particles and the filter block captures those particulates before exhaust.

Figure 6.4a. block diagram VACUUM CLEANER

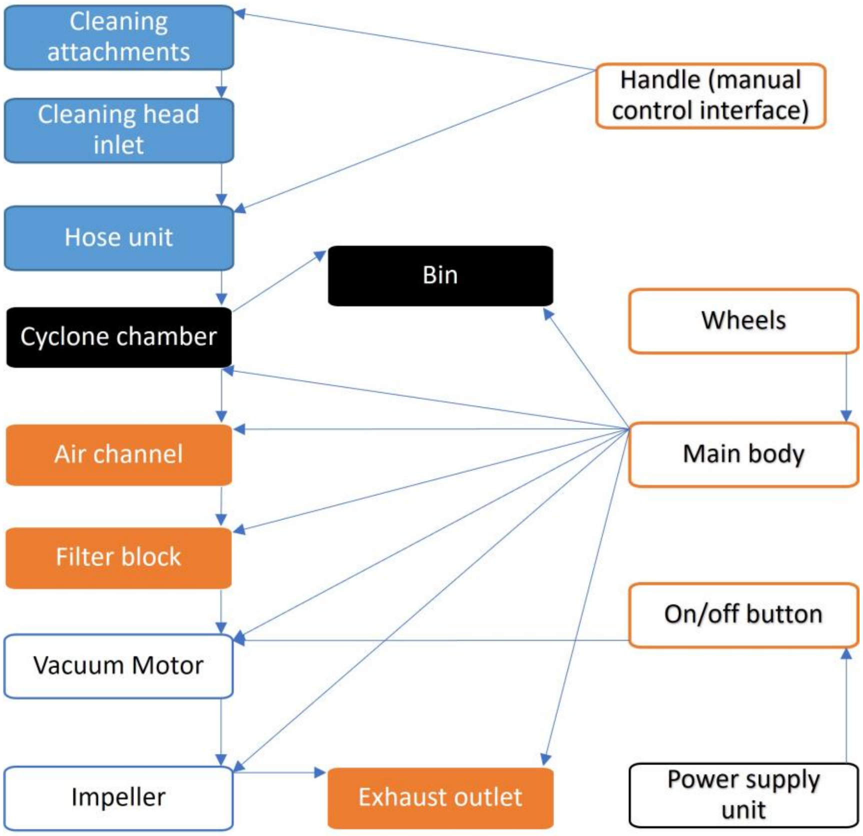
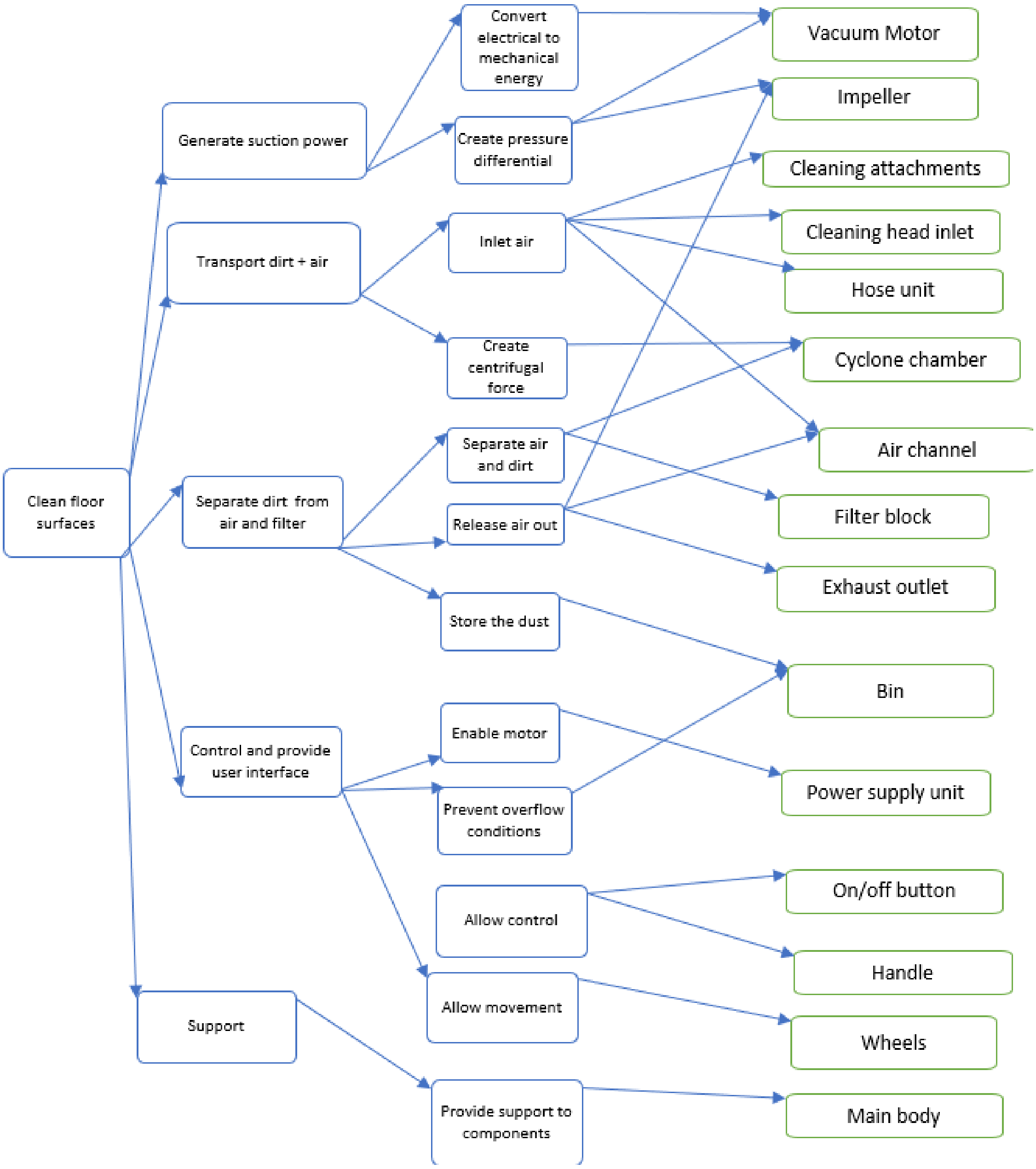


Figure 6.4b. functional tree VACUUM CLEANER



In contrast the autonomous floor cleaning robot integrates additional layers

- sensing unit: bumper sensor, wall sensor and cliff sensor
- main controller unit
- mobility unit: the drive motor, wheel modules and support to the wheel
- user interface unit

The cleaning mechanism is similar, as both contain rotating brushes, an impeller, and a vacuum motor. The robot awareness of the environment to the centralised digital controller. The robot sensor continuously monitors the conditions of the surface and presence of obstacles and feeds that information to the controller. Consequently, the controller adjusts brush rotation and motor speed accordingly. Here the manual control interface is absent, differently from the vacuum cleaner which is completely based on manual control.

Figure 6.4c. block diagram cleaning robot

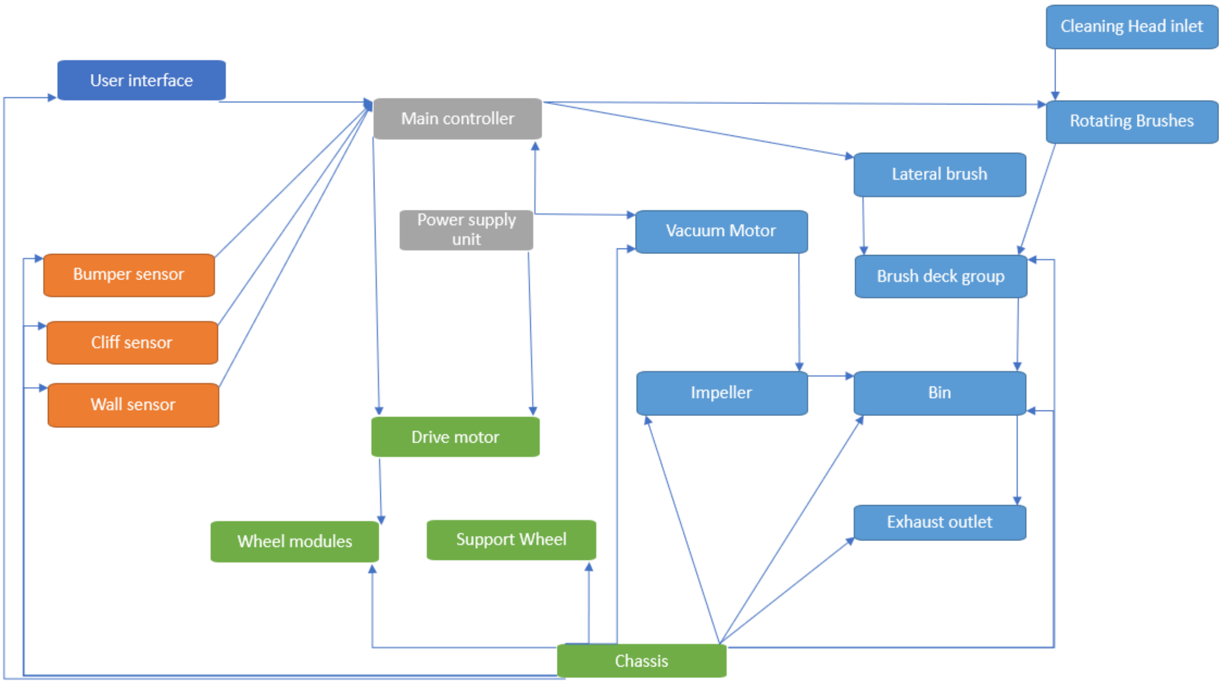
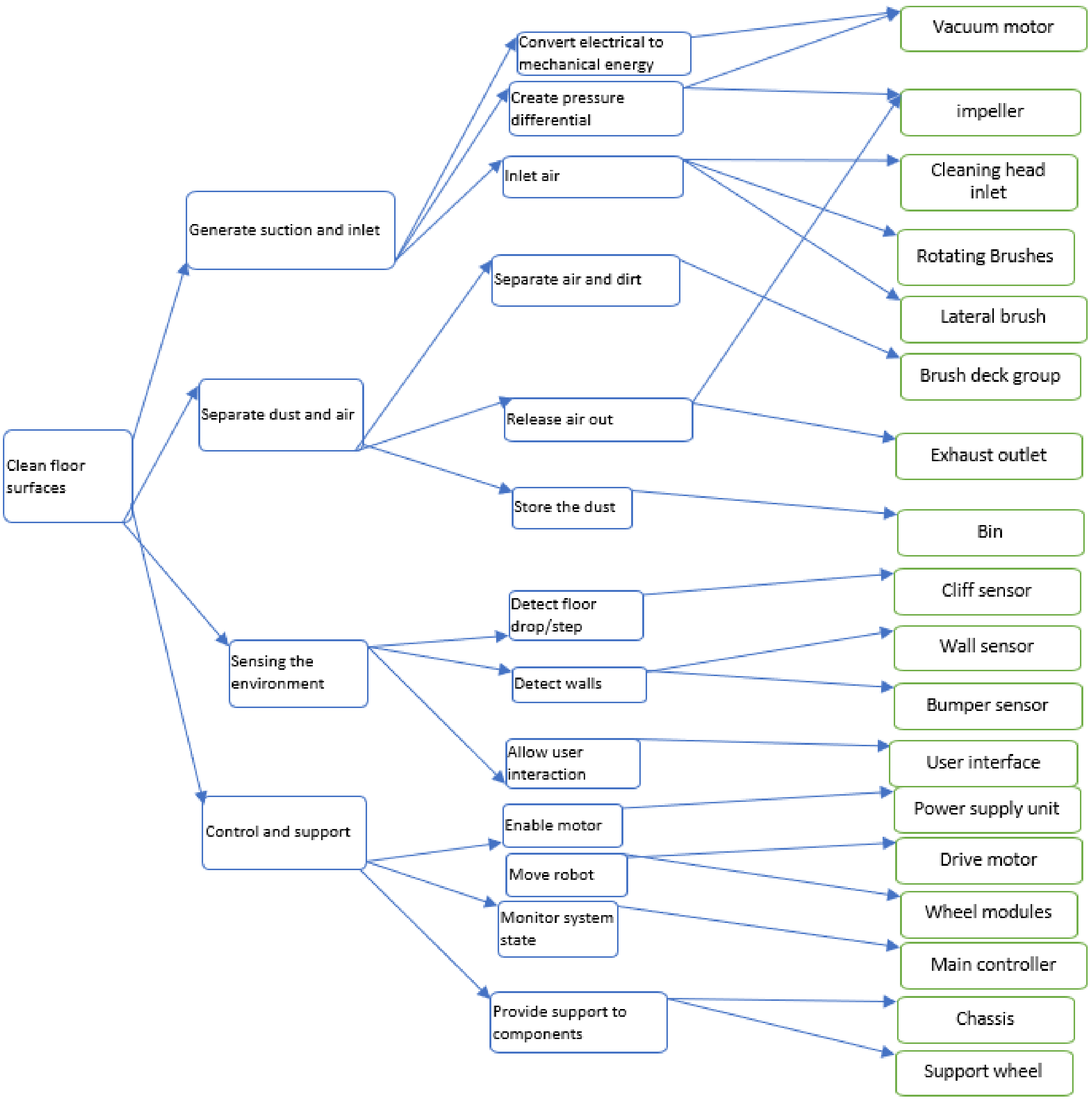


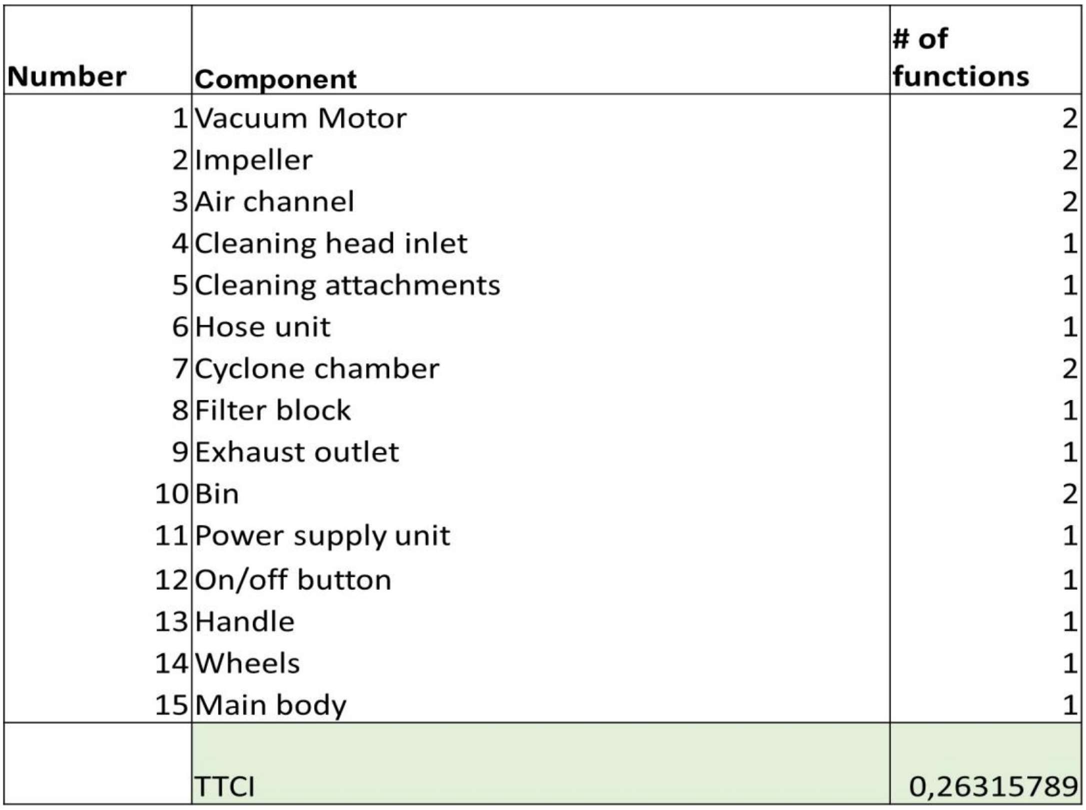
Figure 6.4d. functional tree cleaning robot



6.4.2. TCCI results and FCA map

The cyclone vacuum cleaner demonstrates a TCCI value of 0,263. The components which contribute to different parts for the non-digital product are the motor, which serves both energy generation and creation of suction power. Also the impeller contributes to generating pressure differential and creating an air inlet; lastly, air channel and cyclone chamber and the bin serve multiple functions. This pattern of component use is intrinsic to mechanical systems: physical elements simultaneously serve multiple purposes as they cannot be logically separated.

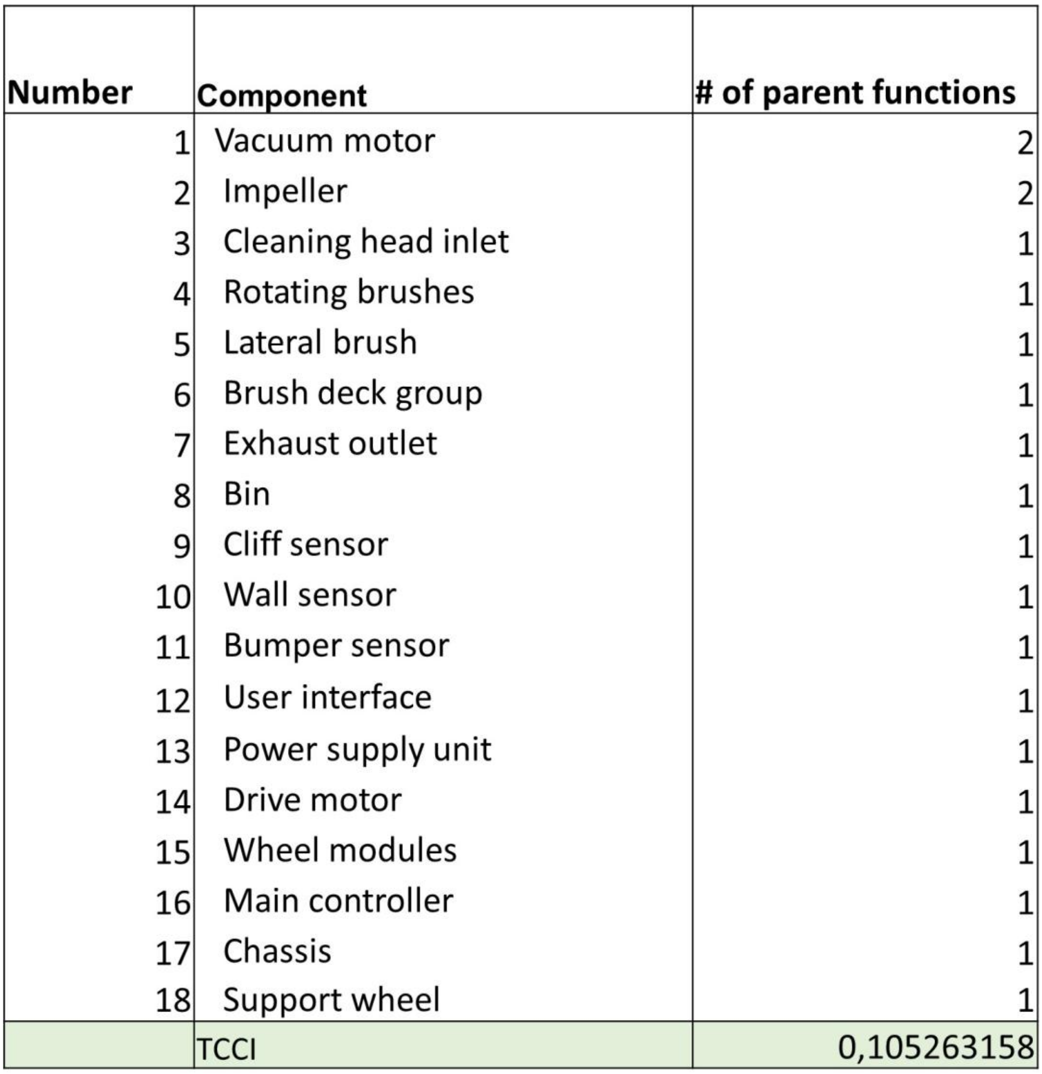
Figure 6.4e. TCCI VACUUM CLEANER



Number	Component	# of functions
1	Vacuum Motor	2
2	Impeller	2
3	Air channel	2
4	Cleaning head inlet	1
5	Cleaning attachments	1
6	Hose unit	1
7	Cyclone chamber	2
8	Filter block	1
9	Exhaust outlet	1
10	Bin	2
11	Power supply unit	1
12	On/off button	1
13	Handle	1
14	Wheels	1
15	Main body	1
	TTCI	0,26315789

The cleaning robot exhibits a lower TCCI of 0,105 indicating that most components maintain highly specialised roles. The only elements participating in multiple functions are the vacuum motor, impeller and power unit. This architecture is enabled by the digital controlling: the main controller mediates interaction between sensors, motors, brushes and this removes the need for mechanical connections between functions; the coordination is achieved through information processing in the robot.

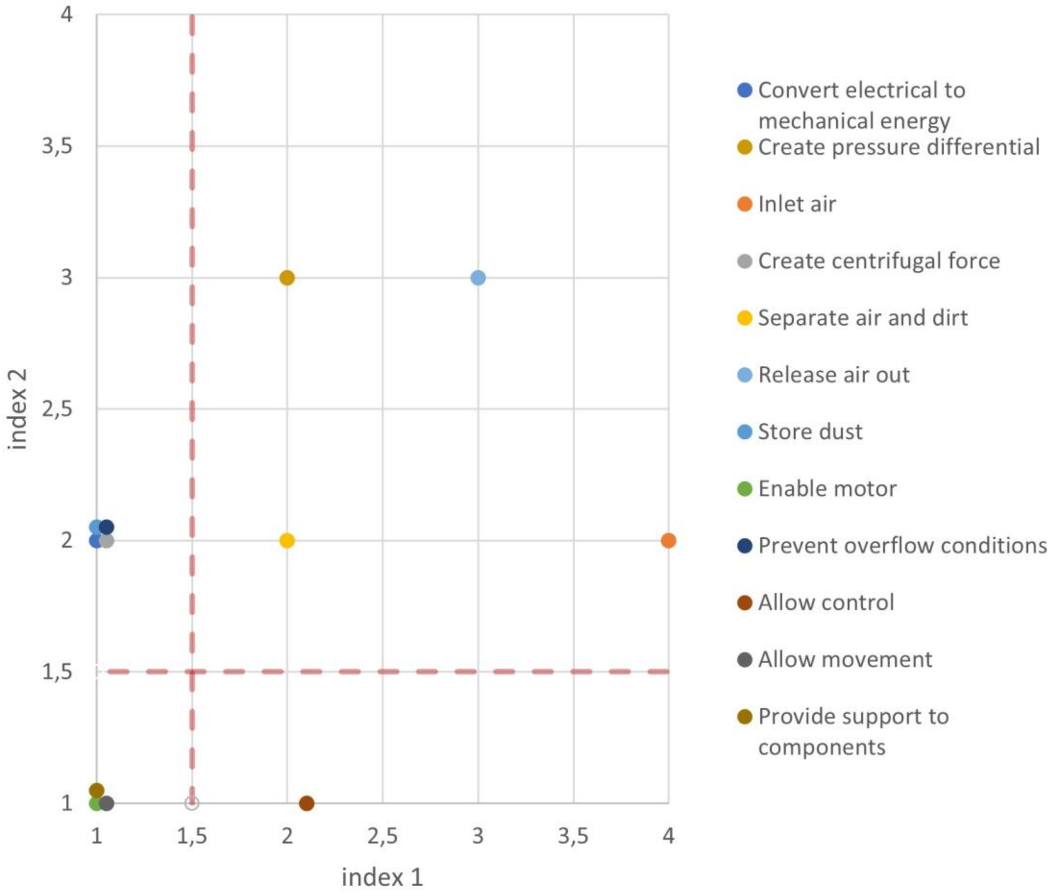
Figure 6.4f. TCCI AUTONOMOUS FLOOR CLEANING ROBOT



Number	Component	# of parent functions
1	Vacuum motor	2
2	Impeller	2
3	Cleaning head inlet	1
4	Rotating brushes	1
5	Lateral brush	1
6	Brush deck group	1
7	Exhaust outlet	1
8	Bin	1
9	Cliff sensor	1
10	Wall sensor	1
11	Bumper sensor	1
12	User interface	1
13	Power supply unit	1
14	Drive motor	1
15	Wheel modules	1
16	Main controller	1
17	Chassis	1
18	Support wheel	1
	TCCI	0,105263158

Discussing the FCA maps, the cyclone vacuum cleaner one shows clustering in the upper left region. Here “create centrifugal force”, “separate air and dirt” and “create pressure differential” are located and they are functions requiring coordinated mechanical operation among the cyclone chamber impeller and motor.

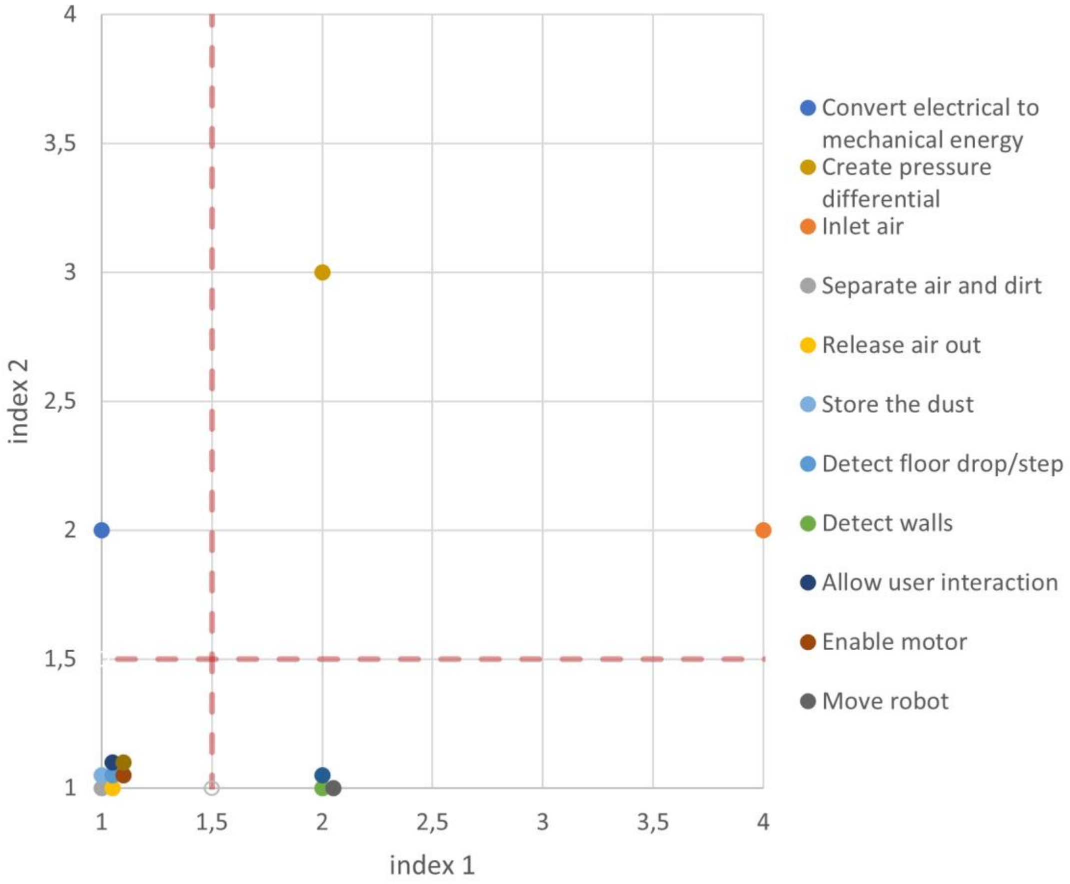
Figure 6.4g. FCA VACUUM CLEANER



Considering the autonomous robot FCA map displays a situation in which most functions scatter across the lower left region and lower righty, which is the integral fragmented one,. “Detect walls”, “Detect floor step” and “Allow user interaction” are each being performed by single components. The main controller contributes to monitoring the system and decision making activities but does not interact physically with the cleaning mechanism, instead it gives them commands.

In conclusion the spatial distribution of the FCA map validates the difference in TCCI, with digital mediation replacing mechanical coupling.

Figure 6.4h. FCA AUTONOMOUS FLOOR CLEANING ROBOT



6.5. Mechanical weighting scale vs Digital weighting scale

6.5.1. Product architecture

The mechanical weighting scale evolved through the 19th and 20th centuries into spring based designs that converted gravitational force into mechanical motion which could be read by a dial (US3759338, 1972). The digital weighting scale, introduced in the 1990s shifted from purely mechanical to electric signal processing and numerical display (US5004058A, 1990). This analysis compares a traditional mechanical kitchen scale with a contemporary one.

At first glance, both address an identical functional requirement, which is accepting a load, measuring its mass and communicating it to the user. In particular, for the mechanical conversion, when an object is placed on the scale, its weight pushed down and compresses a spring. This movement is passed through gears which make the pointer move. The pointer slides over a dial and shows the weight, while a tare knob is used to reset the scale to zero before weighting, Every component in the sequence here (spring hanger, transmission unit, pointer, dial, tare knob) contributes to turn a physical displacement into visual feedback.

Figure 6.5a. Block Diagram Mechanical weighting scale

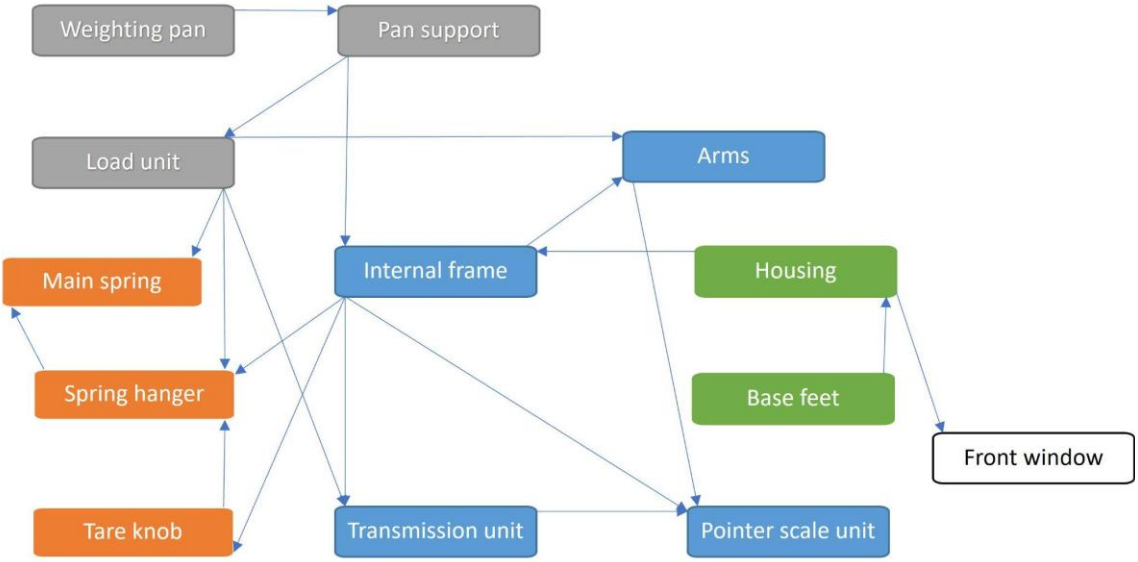
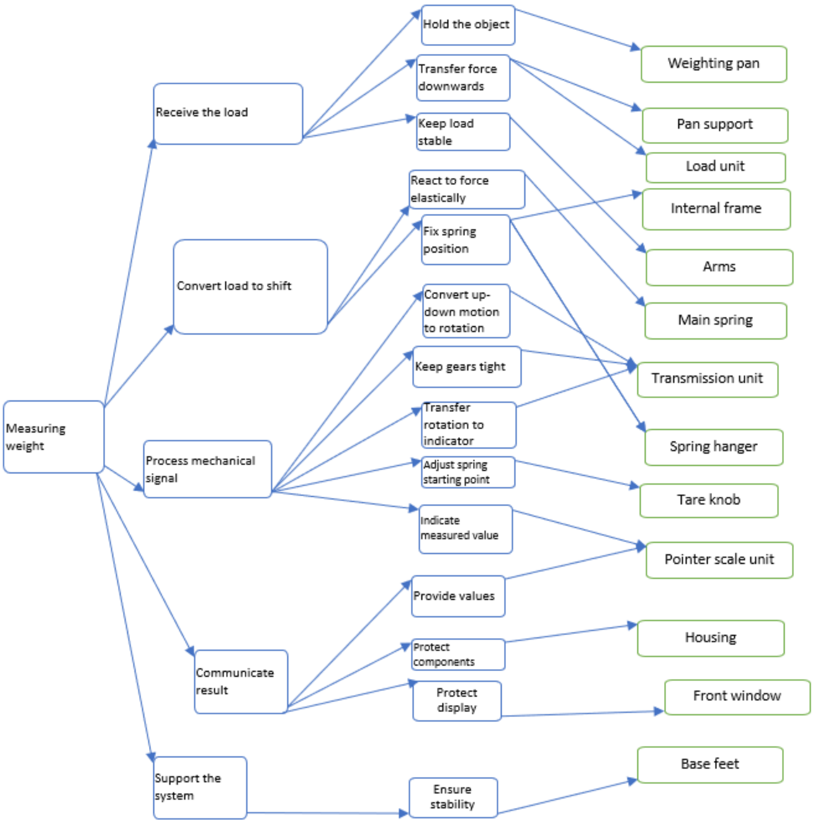


Figure 6.5b. Functional tree Mechanical weighting scale



Instead, the digital scale introduces a computational layer as an intermediate between sensing and indication. A digital scale works by turning weight into electrical information: when an object is placed on a scale, a sensor detects the pressure changing it to an electrical signal. The signal is processed by circuits and a microcontroller, which calculates the weight. If the rotate button is pressed, the scale subtracts the container's weight. The final result is shown as numbers on a screen. Unlike a mechanical scale, the parts do not move much as the system mainly processes information using electricity.

Figure 6.5c. Block diagram digital weighting scale

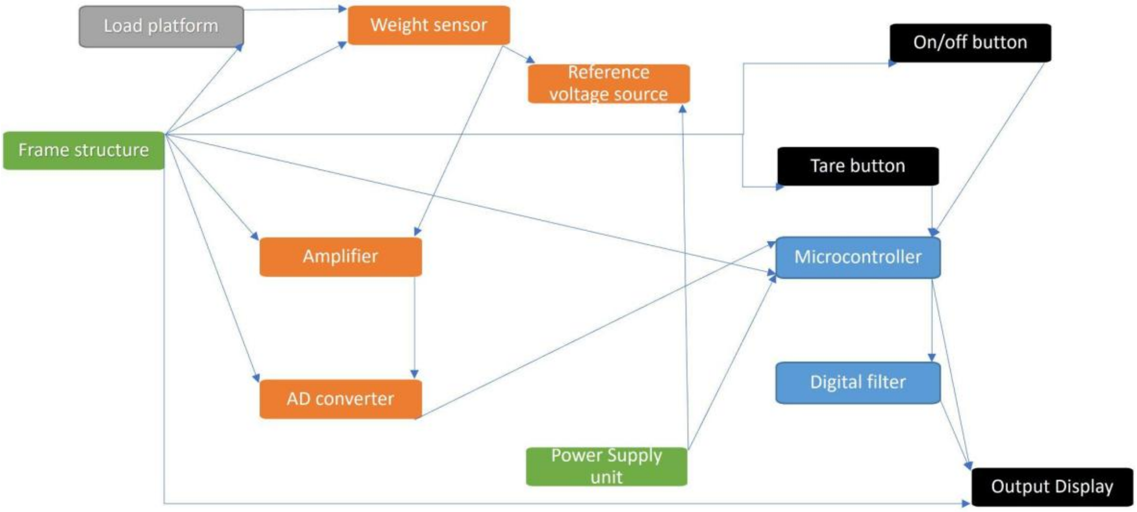
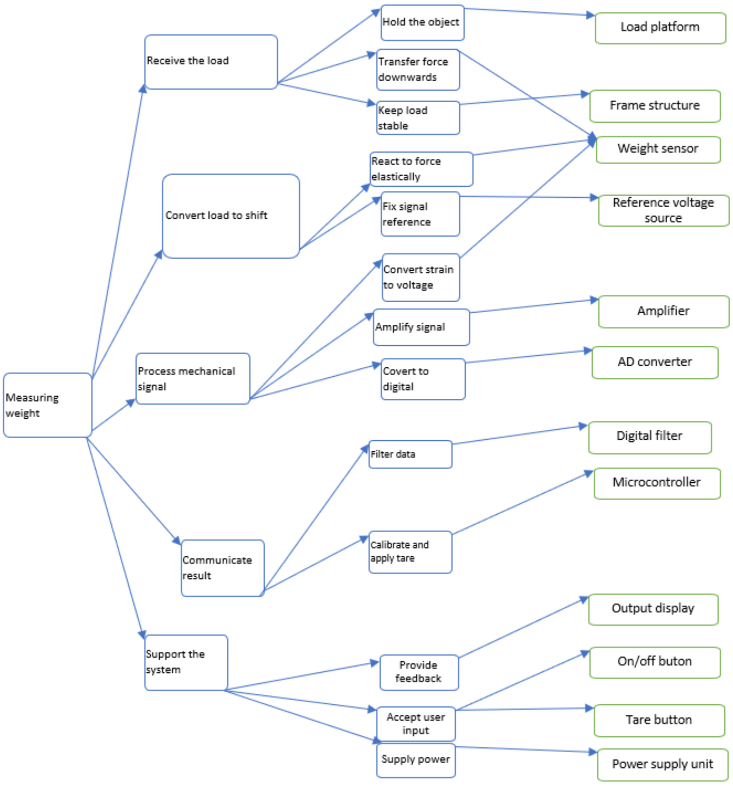
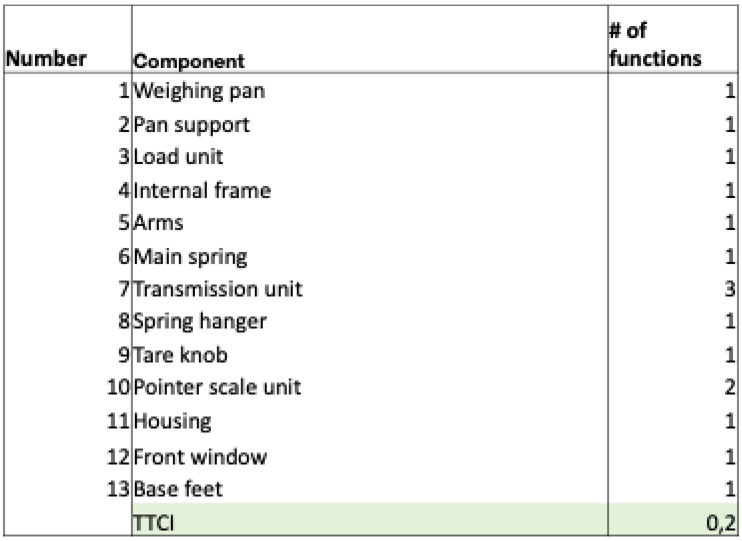


Figure 6.5d. Functional tree digital weighting scale



The mechanical scale exhibits a TCCI of 0,2. The transmission unit participates in three functions: keeping gears tights, transferring rotation to the indicator and processing the mechanical signal, while the pointer scale unit serves both indication and display roles. The main spring and spring hanger each fulfil dual functions. Those components contributing to multiple functions reflect the sequential nature of mechanical measurement. Each component, in fact, specialised in a particular stage of force transductions from elastic deformation, through mechanical amplification to visual presentation. The system is integrated even though it is not clear by looking at the index here, because most components, such as the weighing pan, pan support, load unit, arms, maintain single discrete roles aligned with their position in the measurement chain.

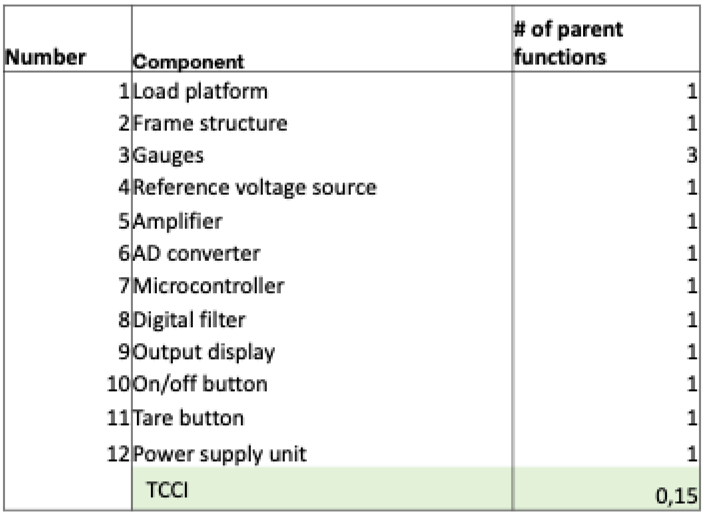
Figure 6.5e. TCCI Mechanical weighting scale



Number	Component	# of functions
1	Weighing pan	1
2	Pan support	1
3	Load unit	1
4	Internal frame	1
5	Arms	1
6	Main spring	1
7	Transmission unit	3
8	Spring hanger	1
9	Tare knob	1
10	Pointer scale unit	2
11	Housing	1
12	Front window	1
13	Base feet	1
	TTCI	0,2

The digital scale demonstrates an even lower TCCI of 0,15 with only the gauges participating in multiple functions. Every component addresses a single functional domain because of the digital intermediary, which mediates between sensing and display but does not physically interact with either, it transforms information rather than transmitting force.

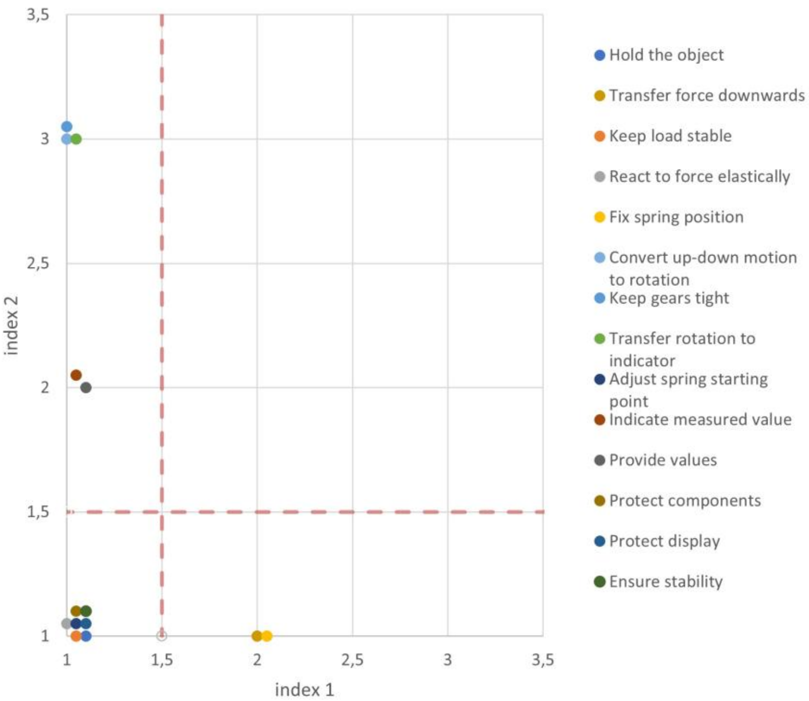
Figure 6.5f. TCCI digital weighting scale



Number	Component	# of parent functions
1	Load platform	1
2	Frame structure	1
3	Gauges	3
4	Reference voltage source	1
5	Amplifier	1
6	AD converter	1
7	Microcontroller	1
8	Digital filter	1
9	Output display	1
10	On/off button	1
11	Tare button	1
12	Power supply unit	1
	TCCI	0,15

Building then the FCA maps, the mechanical scale exhibits clustering in the integral consolidated quadrant, with functions such as "keep gear tight", "transfer rotation to indicator", and a couple functions in the lower right one. All things considered, it is not super integral, more so if looking at other analysed products, still functions scatter around three regions. Critical measurement functions remain integral consolidated.

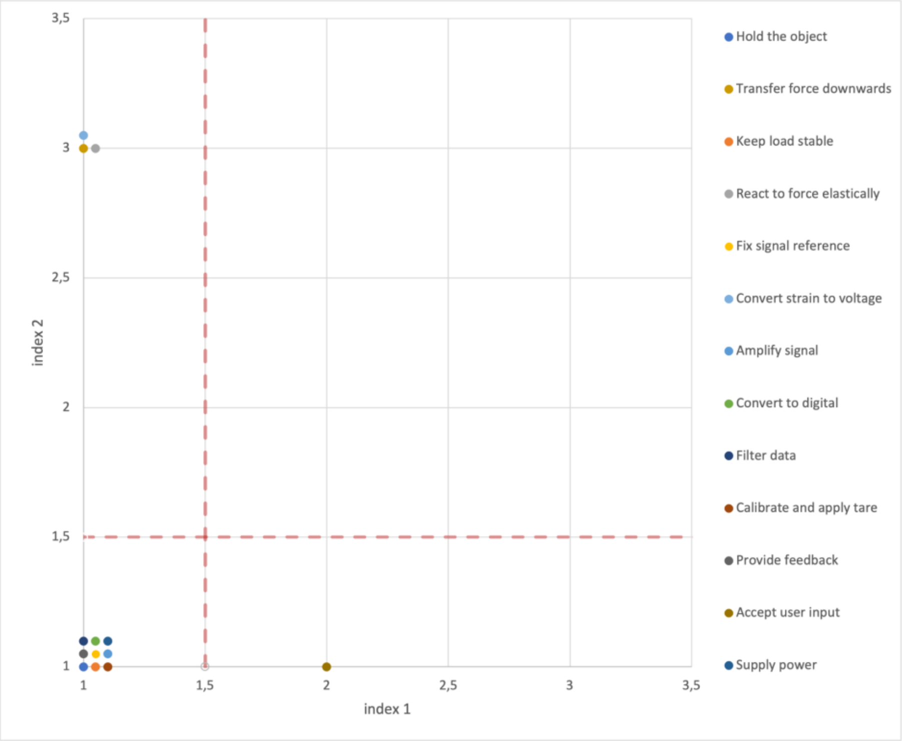
Figure 6.5g. FCA Mechanical weighting scale



Nevertheless, the digital scale map is centred in the modular quadrant. Only functions exhibiting a different tendency are the "transfer force downwards", "amplify signal", "react with force elastically". The spatial distribution validates the quantitative difference:

the mechanical scale TCCI of 0,2 corresponds to a sequential chain of mechanical operation where force flows through integrated components in series, whereas the digital scale TCCI of 0,15 reflects informational isolation.

Figure 6.5h. FCA digital weighting scale



6.6. Electronic treadmill vs Digital treadmill

6.6.1. Product architecture

The treadmill emerged as an exercise device in the 1970s. Early designs incorporated speed and incline control through mechanical linkage and pulley systems (US3711812A, 1973).

In the early 2000s the digital version was developed representing a shift toward computerized exercises management (US6783482, 2001). Here a basic electronic treadmill is compared with a contemporary digital treadmill.

In this particular comparison, the two cases have almost identical design requirements: rotating a running belt, allowing incline adjustment; however, the methods to contribute to this function are different.

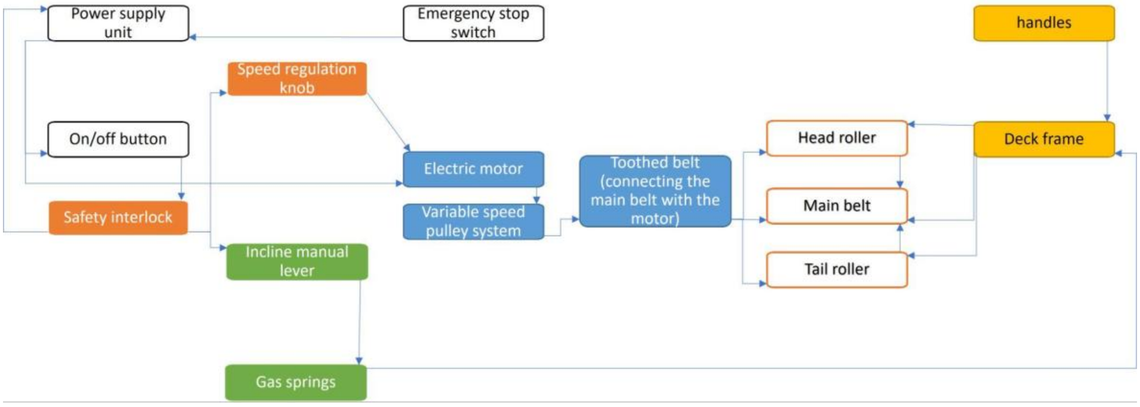
The electronic treadmill architecture is organised in the following blocks

- drive unit: electric motor, toothed belt, variable speed pulley system

- belt and roller system: head and tail rollers, main belt
- control unit: speed regulation knob, incline manual lever and on/off button
- safety unit: emergency stop switch and safety interlock
- structure unit: deck frame, handles and gas springs

Speed control is achieved simply with the user manually adjusting a knob, which directly modulates the pulley system, altering the gear ratio and thus motor speed. Incline adjustment similarly uses a manual lever mechanically connected to has springs to raise the tail of the deck. The central role is taken by the electric motor, driving the toothed belt which, by transmitting motion to the rollers, spins the main belt. The control signals are all mechanical: the user input is directly associated to a mechanical action, speed regulating wise and also emergency or on-off.

Figure 6.6a. Block diagram electronic treadmill



In contrast the digital treadmill introduces a centralized microcontroller. This microcontroller receives an electric signal coming from a button the user pushed in order to give a command. The tachometer sensor measures the belt speed. The microcontroller regulates power delivered to the DC motor through a circuit dynamically adjusting belt speed to match the user programmed target. The incline is adjusted electronically through an invisible system. The microcontroller acts as a computational intermediary decoupling user intent from mechanical actuation.

Figure 6.6b Block diagram digital treadmill

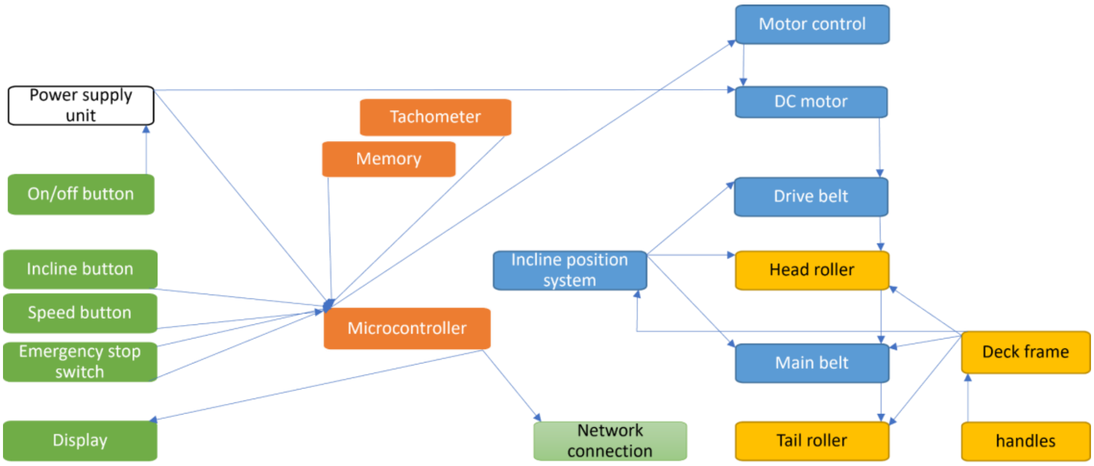


Figure 6.6c Functional tree electronic treadmill

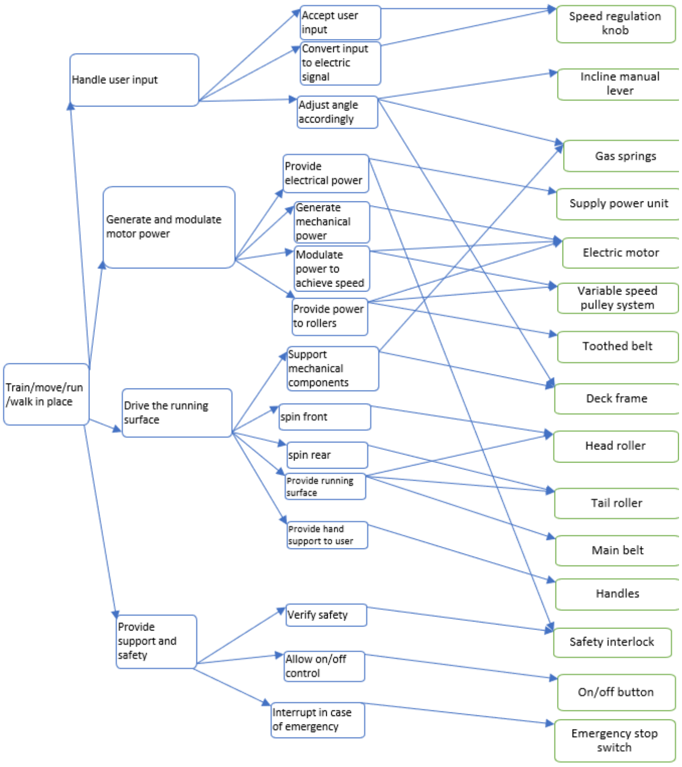
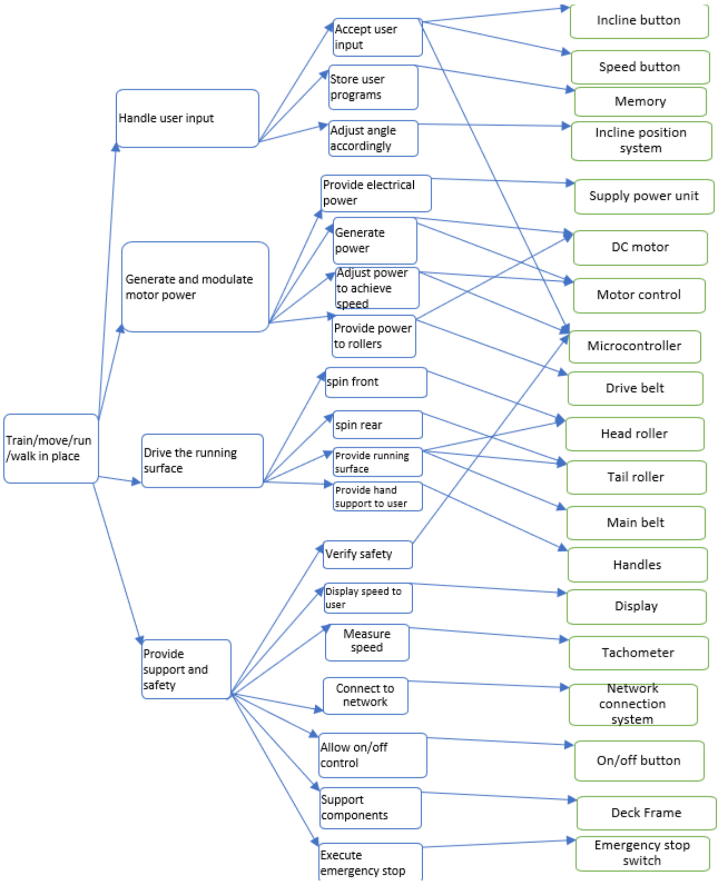


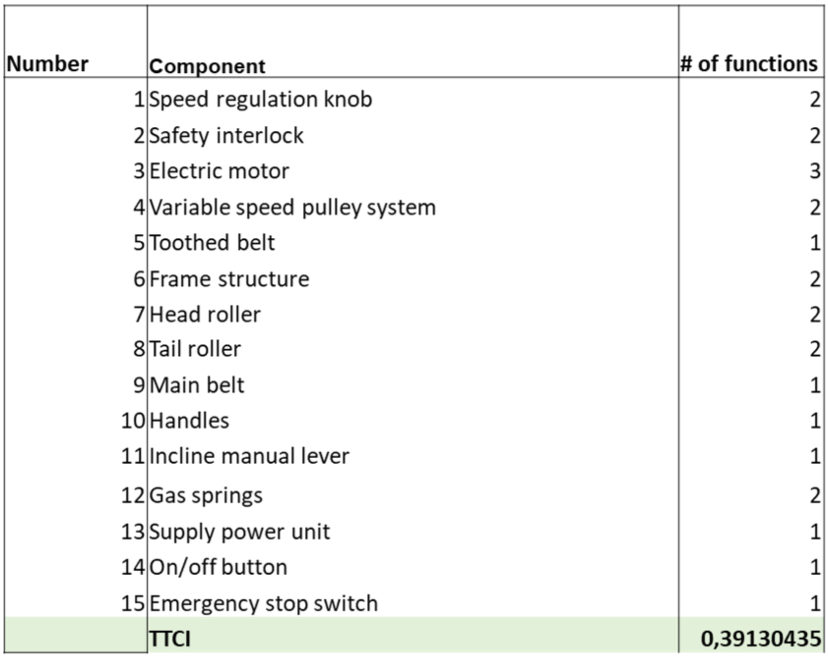
Figure 6.6d Functional tree digital treadmill



6.6.2. TCCI results and FCA map

From the functional tree of the electronic treadmill, it can be seen that several components contribute to multiple functions. For instance, the motor both generates and modulates power through the pulley system; the safety interlock interrupts the power and also verifies safety. The overlapping lead to a TCCI of 0,391 reflecting necessary mechanical components' coupling for the traditional exercise treadmill.

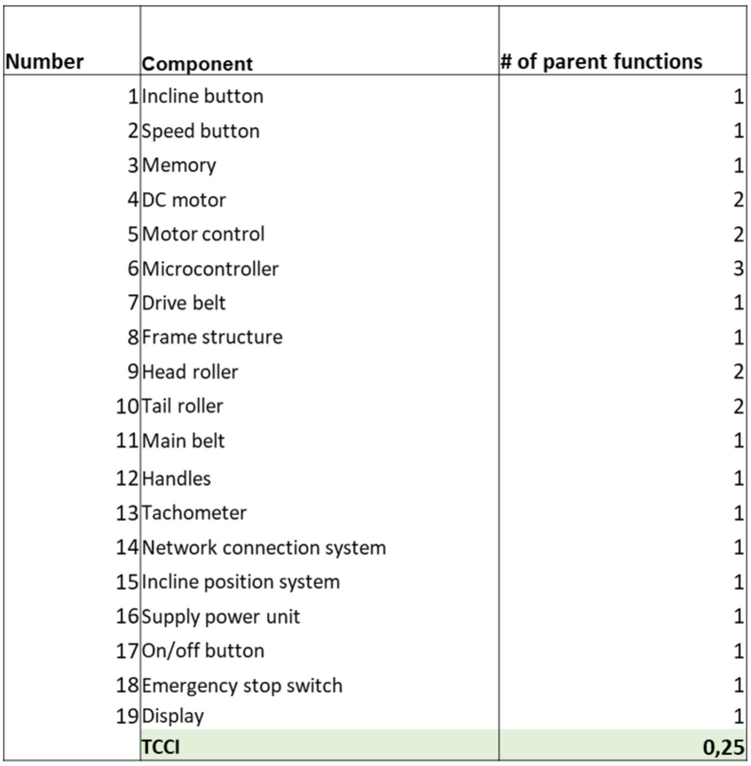
Figure 6.6e TCCI electronic treadmill



Number	Component	# of functions
1	Speed regulation knob	2
2	Safety interlock	2
3	Electric motor	3
4	Variable speed pulley system	2
5	Toothed belt	1
6	Frame structure	2
7	Head roller	2
8	Tail roller	2
9	Main belt	1
10	Handles	1
11	Incline manual lever	1
12	Gas springs	2
13	Supply power unit	1
14	On/off button	1
15	Emergency stop switch	1
	TTCI	**0,39130435**

The analysis of the digital treadmill shows architectural reorganization compared to the non-digital product. The microcontroller, motor control unit and DC motor contribute to multiple functions; motor control is intermediary, commanding from the microcontroller e translating to voltage adjustments to the DC motor. Nevertheless, the remaining components have a one-to-one relationship with functions: every component keeps a highly specialised role.

Figure 6.6f TCCI electronic treadmill



Number	Component	# of parent functions
1	Incline button	1
2	Speed button	1
3	Memory	1
4	DC motor	2
5	Motor control	2
6	Microcontroller	3
7	Drive belt	1
8	Frame structure	1
9	Head roller	2
10	Tail roller	2
11	Main belt	1
12	Handles	1
13	Tachometer	1
14	Network connection system	1
15	Incline position system	1
16	Supply power unit	1
17	On/off button	1
18	Emergency stop switch	1
19	Display	1
	TCCI	0,25

Constructing the Function Component allocation map of the electronic treadmill, it shows clustering in the integral consolidated region (upper left). “Generate mechanical power”, "Modulate power to achieve speed” and “spin front / rear” require high coordination between electric motor, variable speed pulley, rollers.

Nevertheless, many functions have modular behaviour, reflecting the mixed integration necessitated by the mechanical design.

The digital version instead has an FCA map fundamentally different from a spatial distribution perspective. Most functions scatter across the lower left and lower right regions. Functions such as “accept user input”, “display speed to user", "measure speed” and “store user programs” are all addressed by isolated components (speed button, display, tachometer and memory respectively) with minimal interdependency. The microcontroller occupies a unique position, participating in three distinct functional domains. This spatial separation validates the findings of the commonality indexes. The electronic treadmill of 0.391 corresponds to a mechanically integral architecture where components must physically couple, whereas the digital treadmill’s TCCI of 0.25 reflects modular product architecture.

Figure 6.6g FCA map electronic treadmill

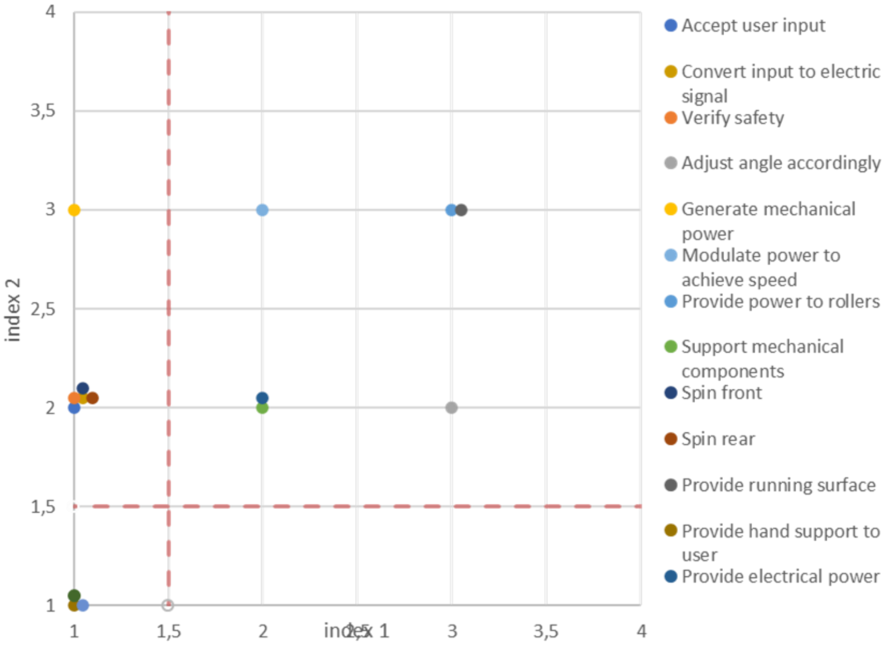
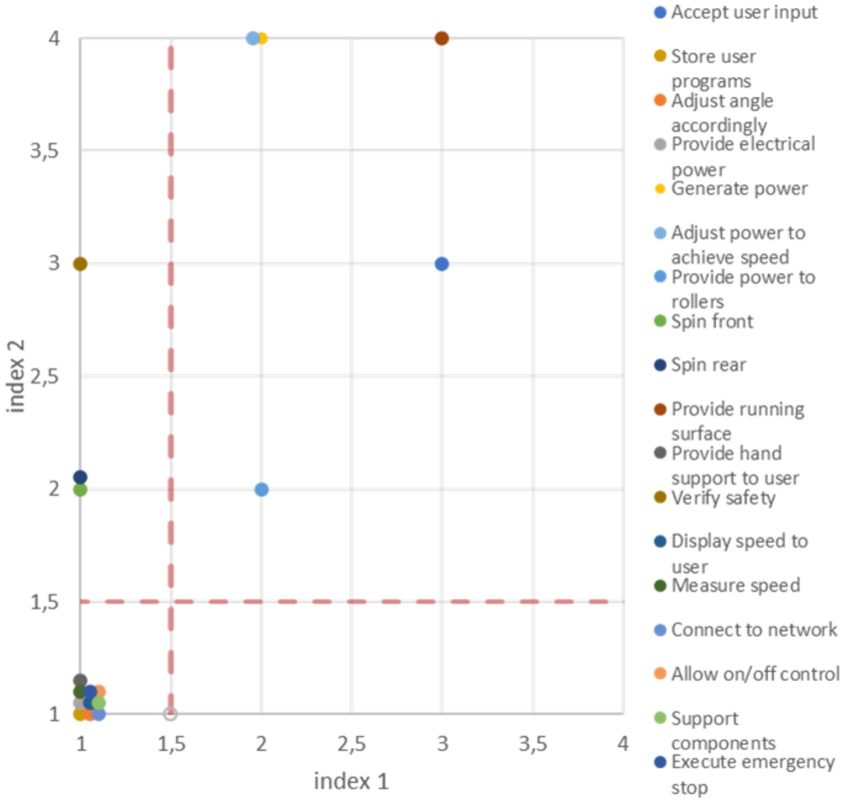


Figure 6.6h FCA map digital treadmill



Conclusion

The analysis presented here points to a clear outcome. When products transition from non-digital to digital, their physical architecture changes: across all examined cases, digital products exhibit lower levels of function sharing between components, even though the overall functional scope remains comparable. After concrete measurement of structural change, the analysis demonstrates systematic patterns across all six product pairs: lower commonality in digital products with TCCI reduction ranging from 25% to 83%, reduced physical coupling because of standardised digital interfaces, and clearer allocation of functions representing a shift from integral complex to modular or integral-consolidated architectures.

These findings provide empirical confirmation of what the literature on digital innovation has long suggested. The theoretical work mentioned and quoted in this thesis (Yoo et al on programmability and data homogenisation, Ulrich about modular architectures, Henderson and Clark's analysis of architectural innovation) all point to the expectation that digitalisatoin should reduce coupling and increase modularity. However, systematic empirical evidence comparing non-digital and digital versions of the same product has been lacking. This thesis addresses that gap by providing concrete, quantitative validation of these theoretical claims.

Moreover, an additional takeaway is that architectural change does not derive from a complete redesign of hardware. In many cases, core physical subsystems remain largely comparable across technological generations.

The key difference lies in how functions are coordinated. In digital products control and decision making are conferred in microcontrollers, processors or programmable components. They allow physical parts to assume a more specialised role. As a consequence, complexity is not removed from the system but relocated away from the physical structure. In addition, the number of tightly integrated components is reduced in digitalised products.

By comparing pairs of non-digital and digital products sharing the same core functions, and by analysing their architectures through commonality index and function-component allocation methods, the work provides evidence that the digital products have functions that are more clearly separated. This confirms that digitalization pushes systems towards flexible architectures that make it easier to upgrade parts and also add variants.

From an innovation management perspective, higher modularity has several implications. Firstly, a direct impact on how firms organize innovation activities over time. Modular architectures make it easier to introduce incremental innovations. Also, it supports radical innovation at a subsystem level as new technologies can be integrated into existing architectures without redesigning the whole product. This increases the firm ability to answer technological changes.

The empirical comparison between non-digital and digital products therefore provides concrete evidence that digitalization not only changes product technology but also changes the way innovation is managed.

More importantly, it validates existing theoretical frameworks and established a foundation for future research. By demonstrating that the transition to digital architectures systematically increases modularity across different product categories, this work lays a first building block for further investigations into how architectural change affects competitive dynamics, supply chain organisation, and the stability of dominant designs in digitalising industries. Moreover, future research could extend this analysis by considering software architecture and digital services, analysing a complete layered architecture.

Bibliography

Abernathy W.J., Utterback J.M. (1975). A Dynamic Model of Process and Product Innovation. International Journal of Management Science, 7(6), 639-656.

Altavilla and Montagna, A Product Architecture-Based Framework for a Data-Driven Estimation of Lifecycle Cost

Baldwin C.Y., Clark K.B. (1997). Managing in an age of modularity. Harvard Business Review, 75(5), 84-93.

Browning TR (2001) Applying the design structure matrix to system decomposition and integration problems: a review and new directions. IEEE Trans Eng Manag 48(3):292–306

Cantamessa M, Milanesio M, Operti E (2006) Value chain structure and correlation between design structure matrices Springer, New York

Cantamessa M., Montagna F. (2023), Management of innovation and production development. Springer, pp 461-482.

Clark K.B., Eppinger S.D. (2000). Product Design and Development (3rd ed.). McGraw-Hill, Boston, MA.

Collier D.A. (1982). Aggregate Safety Stock Levels and Component Part Commonality. Management Science, 28(11), 1297-1303.

Dosi G (1982) Technological paradigms and technological trajectories. A suggested interpretation of the determinants and directions of technical change. Res Pol 11(3):147–162

Fixson S.K. (2004), Product architecture assessment: a tool to link product, process, and supply chain design decisions. Journal of operations management.

Fixson and Park 2008; The power of integrality: Linkages between product architecture, innovation, and industry structure

Gershenson J.K., Prasad G.J. & Zhang Y. (2003), Product modularity: definitions and benefits. J. Eng. Design, Vol. 14, No. 3, pp. 295-313.

Henderson R., Clark K. (1990). Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms. Administrative Science Quarterly, pp. 9-30.

Henfridsson O., Mathiassen L. & Svahn F. (2014), Managing technological change in the digital age: the role of architectural frames. Journal of information technology, 29, 27-43.

Kota S., Sethuraman K., Miller R. (2000). A metric for evaluating design commonality in product families. J. Mech. Design, 122, 403–410.

Martin M., Ishii K. (1996). Design for variety: a methodology for understanding the costs of product proliferation. In Proceedings of the ASME Design Engineering Technical Conferences, Paper No. 96-DETC/DTM-1610. American Society of Mechanical Engineers, Irvine, CA.

Pahl, G., Beitz, W. (1996). Engineering Design: A Systematic Approach. London: Springer

Pimmler T.U., Eppinger S.D. (1994). Integration analysis of product decompositions. In Proceedings of the ASME Design Theory and Methodology Conference, Minneapolis, MN.

Roberts E.B. (1987). Generating technological innovation. Oxford, New York.

Rogers E.M. (2003). Diffusion of Innovations, 5th Edition. Free Press, New York.

Schilling M. (2000), Toward a general modular system theory and its application to interfirm product modularity. The academy of management review, Vol. 25, No. 2, pp. 312-334.

Steward, Donald V. (1981) "The Design Structure System: A Method for Managing the Design of Complex Systems." Administrative Science Quarterly,

Stone B.R., Wood K.L. & Crawford H.R. (1998) A heuristic method to identify modules from a functional description of a product. In: ASME Design Engineering Technical Conferences, Atlanta, USA, 13–16.

Simon, H. A. (1962). "The Architecture of Complexity." Proceedings of the American Philosophical Society, 106(6), 467–482.

Thevenot H.J., Simpson T.W. (2007), Commonality indices for product family design: a detailed comparison. Journal of engineering design.

Tidd, J., Bessant, J., & Pavitt, K. (2005). Managing Innovation: Integrating Technological, Market and Organizational Change.

Ulrich K. (1995) The role of product architecture in the manufacturing firm. Research Policy, 24.

Ulrich, K., Pearson, S. (1998). "Assessing the Importance of Design Through Product Archaeology." *Management Science*, 44(3), 352–369

Yoo Y., Hallfredsson O. & Lyytinen K. (2010), Research commentary – The new organizing logic of digital innovation: an agenda for information systems research. Information systems research, 21(4): 724-735.

Voss C.A. & Hsuan J. (2009) Service architecture and modularity. Decision Sciences, 40(3), 541–569

Wacker J.G., Treleven M. (1986). The Interplay of Design Complexity and Inventory Costs in Safeguarding Service Level Performance. Decision Sciences, 17(1), 25-43.

Webography

Capture. "How Do Cassette Tapes Work?" https://www.capture.com/blogs/insights/how-do-cassette-tapes-work?utm_source=chatgpt.com

Wikipedia, 2026. “Cassette deck.” *Wikipedia, The Free Encyclopedia*.

IRSAP. "The History of the Thermostat." https://www.irsap.com/en/blog/the-history-of-the-thermostat

In Sheep's Clothing. "A Primer on Vintage Cassette Decks: How to Find a Good One." https://insheepsclothingthrift.com/a-primer-on-vintage-cassette-decks-how-to-find-a-good-one/

Triton Store. "Scale vs. Balance." https://tritonstore.com.au/scale-vs-balance/

Vorwerk Support. "Are All the Thermomix Accessories Dishwasher-Safe?" https://support.vorwerk.com/hc/en-us/articles/360003821359

iFixit. "Smontaggio Thermomix+TM5+Teardown/117133." https://it.ifixit.com/Smontaggio/Thermomix+TM5+Teardown/117133

YouTube. "Companies presenting their autonomous cleaning robots when they were first launched." https://www.youtube.com/watch?v=Y0lIK2UzSQ

YouTube. "Brief look at the history of the robot vacuum." https://www.youtube.com/watch?v=rJwJhctz3M4

IRSAP. "The History of the Thermostat." https://www.irsap.com/en/blog/the-history-of-the-thermostat