AXIOMATIC DESIGN

Axiomatic Design (AD) is a systems engineering methodology that enables the creation of good system designs by improving the causality of system functionality, the system's physical realisation, and manufacturing processes. This is done by using mathematical principles that describe independence between these elements. AD contributes to a better understanding of main and alternative design options. Its applications extend across various industries, from mechanical engineering to (bio)medical systems and even social systems. AD is particularly suitable for addressing problems in high-tech systems development. This article outlines what AD is and how it can be applied.

ERIK PUIK AND RIK LAFEBER

Introduction

Axiomatic Design (AD) is a concept developed to address the challenges that arise during the design phase in the development of complex systems. The method was developed by Nam P. Suh of the Massachusetts Institute of Technology (MIT) in the second half of the 1970s [1]. AD declares 'Axioms' that cannot be proved or deduced from physical phenomena, which gives the method its name. Initially, a number of six design Axioms were defined. Two of the Axioms have stood the test of time, the others appeared to be corollaries of these two. Since then, the methodology has centred around two primary axioms:

- The first axiom, the 'Independence Axiom', directs the designer to ensure that the functional requirements are independent. This means that each functional requirement should be determined, commonly referred to as 'satisfied', by a specific design parameter and not influenced by others. The advantage of such independence is that changes to one design parameter should not interfere with multiple other functions. Consequently, the design is more robust, easier to control and improve, and less prone to unexpected outcomes.
- The second axiom, the 'Information axiom', encourages minimising the information content of the design. Essentially, the design should be as simple and clear as possible. As few steps or parameters as possible should be required to move from function to physical solution.

AUTHORS' NOTE

Erik Puik is professor of Smart Manufacturing at Fontys University of Applied Sciences in Eindhoven (NL). Rik Lafeber is researcher Microsystem Technology and lecturer Mechanical Engineering at HU University of Applied Sciences in Utrecht (NL).

erik.puik@fontys.nl
www.fontys.nl
www.hu.nl

The application of the Axioms contributes to a structured design process, reduces complexity and promotes robustness. It provides a rational basis for design choices and improves communication between team members using the formalised methodology. This introduction to AD focuses on how system requirements are decomposed in AD. To do this, the Independence Axion is applied. The Information Axiom will be described in a future article.

Organizing Domains

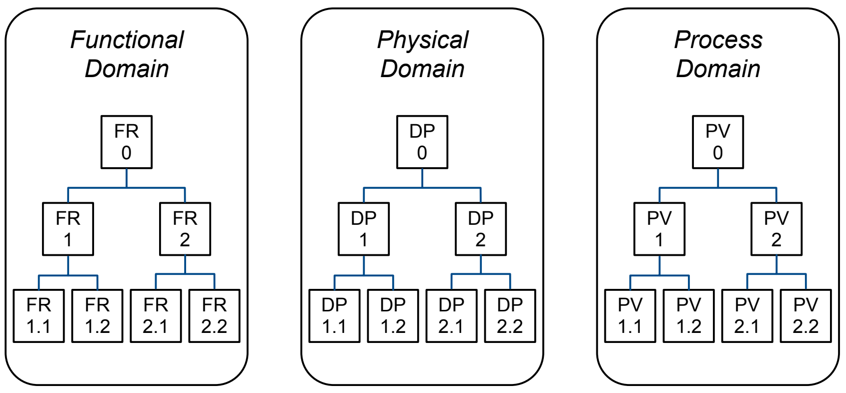
AD provides a systematic method of translating functional needs into functional design, reducing reliance on intuition or guesswork in the design process. This systematic approach creates a clear roadmap, starting with identifying functional requirements, fulfilling these requirements through design parameters, and finally representing these parameters in process variables.

AD demands clear formulation of design objectives through the establishment of 'Domains':

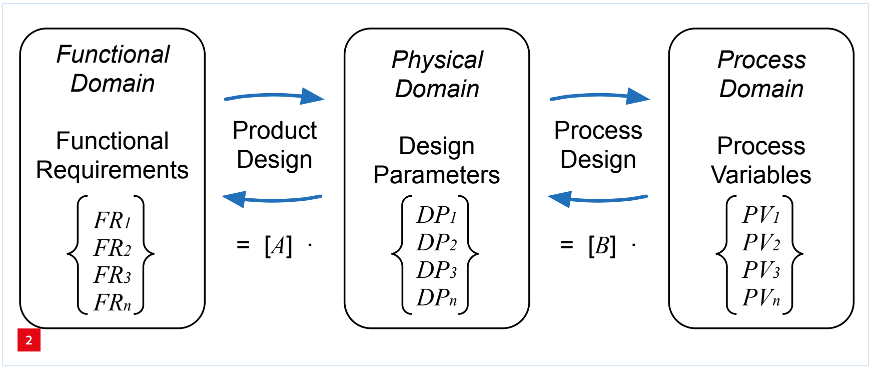
- functional domains, containing the Functional Requirements, from now on to be called FRs;
- physical domains, containing the Design Parameters, or DPs;
- process domains, containing the Process Variables, or PVs.

The domains are hierarchically decomposed as shown in Figure 1.

According to the definition in AD, the Independence Axiom advises to "Maintain the independence of the functional requirements". AD also explains how this can be done from a mathematical perspective, as shown in Figure 2.



1
Axiomatic Domains and their hierarchical organisation.



2

Axiomatic Domains and their relations.

The domains, in which functional requirements (FRs), design parameters (DPs), and process variables (PVs) are represented as vectors, are interrelated with design matrices. The design equations according to good AD practice are defined as:

$$\{FR\} = [A] \cdot \{DP\}$$
$$\{DP\} = [B] \cdot \{PV\}$$

Here, $[A]$ and $[B]$ are the product and process design matrices, respectively. If a product design has three FRs, DPs, and PVs, the product design matrix $[A]$ and the process design matrix $[B]$ have the following form:

$$[A] = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

$$[B] = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix}$$

The design equation for the FRs is then defined as follows:

$$\begin{bmatrix} FR_1 \\ FR_2 \\ FR_3 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \cdot \begin{bmatrix} DP_1 \\ DP_2 \\ DP_3 \end{bmatrix}$$

Here, FR_1 to FR_3 are three functional requirements and DP_1 to DP_3 are three relevant design parameters. In this representation, a '*good design*' would be an 'uncoupled' or a 'decoupled' one if the matrix is diagonal or triangular, respectively, as shown below:

$$[A] = \begin{bmatrix} X & 0 & 0 \\ 0 & X & 0 \\ 0 & 0 & X \end{bmatrix} \qquad \textit{(uncoupled)}$$

$$[A] = \begin{bmatrix} X & 0 & 0 \\ X & X & 0 \\ X & X & X \end{bmatrix} \qquad \textit{(decoupled)}$$

Here, the X-es indicate non-zero elements of the matrix and as such indicate a relation between the associated DPs and the FRs. In an uncoupled design, every FR is related to a single DP, while in a decoupled design, it may be related to more than a single DP, but if the right order is applied to adjust the FRs with the DPs, all FRs can be tuned sequentially. In AD, this design matrix takes a central place because it defines the structure and as such the behaviour of the design.

The design equation for the FRs does not yet include the relation to the manufacturability of the physical system. This is where the process design matrix $[B]$ is applied:

$$\begin{bmatrix} DP_1 \\ DP_2 \\ DP_3 \end{bmatrix} = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} \cdot \begin{bmatrix} PV_1 \\ PV_2 \\ PV_3 \end{bmatrix}$$

The full design equation, which correlates the process variables with the functional requirements, then has this form:

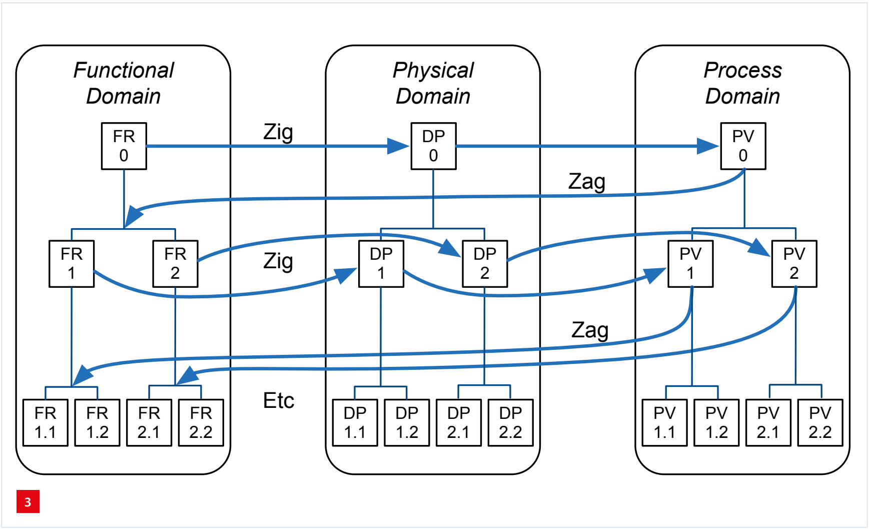
$$\begin{bmatrix} FR_1 \\ FR_2 \\ FR_3 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} \cdot \begin{bmatrix} PV_1 \\ PV_2 \\ PV_3 \end{bmatrix}$$

A caveat must be made here that only a limited number of system designs are sufficiently well understood in practice to successfully apply this equation. The examples in this article are therefore limited to the translation from DPs to FRs.

The process of zigzagging

To check that all FRs are satisfied by their DPs and subsequently the DPs are satisfied by their PVs, AD uses a procedure called 'zigzagging'. Zigzagging is a top-down descent through the design hierarchy, covering all domains sequentially. At each level, it is checked whether the FRs and DPs are satisfied before going down to the next level, as shown in Figure 3.

The process of zigzagging is always performed from the left- to the right-hand side. Zigzagging covers all domains. Successful completion of the zigzagging process will lead to an uncoupled or a decoupled design matrix and satisfies the Independence Axiom, which completes the conceptual



3

The process of hierarchically zigzagging through the domains.

phase of the project. Characteristic of the process is that even at the highest level the manufacturability of a system is already considered. All requirements are addressed in a structured and sound manner.

AD application examples

Example 1: Independence Axiom applied for the design of a water faucet
A traditional example for learning about the possible coupling between the Functional Requirements of a system is the 'Water Faucet Example'. We consider two FRs of a water faucet that need to be satisfied:

FR_1 = Control temperature
FR_2 = Control flow rate

These two FRs need to be controlled independently.

In a traditional faucet, as shown in Figure 4, there are two levers that control hot and cold water:

DP_1 = Control flow of hot water
DP_2 = Control flow of cold water

Together with the FRs, these lead to the following design equation:

$$\begin{bmatrix} FR_1 \\ FR_2 \end{bmatrix} = \begin{bmatrix} X & X \\ X & X \end{bmatrix} \cdot \begin{bmatrix} DP_1 \\ DP_2 \end{bmatrix}$$

The product design matrix, consisting of the four X-es, is a coupled matrix, which means that if temperature FR_1 needs to be adjusted this can be done by adjusting both the hot and cold water, DP_1 and DP_2, respectively. Unfortunately, the change of either DP_1 or DP_2 also affects the waterflow FR_2. If a user would want to achieve a particular temperature and flow, a situation that needs satisfaction of both FR_1 and FR_2 at the same time, an iterative attempt to adjust both DP_1 and DP_2 could lead to an approximation of the desired situation. However, the intended temperature and flow will never be reached exactly.

4
Faucet with two levers, for adjusting flow and temperature to preferred values. (Licence: Flickr creative commons)

5
This faucet with a single lever is easier to control. (Licence: Flickr creative commons)

A more convenient, modern faucet is shown in Figure 5. Its design has different DPs:

DP_1 = Rotate handle
DP_2 = Lift handle

The design matrix then has the following form:

$$\begin{bmatrix} FR_1 \\ FR_2 \end{bmatrix} = \begin{bmatrix} X & 0 \\ 0 & X \end{bmatrix} \cdot \begin{bmatrix} DP_1 \\ DP_2 \end{bmatrix}$$

This leads to a construction that is an uncoupled design, because FR_1, the temperature of the water, can be solely controlled by DP_1, rotating the handle, and control of the flow, FR_2, can be realised by lifting the handle, DP_2. This means that with this approved design, independently controlling temperature and flow is significantly easier for the user. The product design matrix in this case is a decoupled matrix, which means that the independence of the FRs is guaranteed. In AD, this is called a 'Good Design'.

Example 2: Decomposition of an armchair using the process of zigzagging
The second example describes the design of a chair. Its functionality will be decomposed and the design including the processes for manufacturing will emerge while doing. The process starts with the quest for the highest-level FR of a chair. A good FR is, according to the definition in AD, an answer to the question "What should the device do?"

The answer should be defined as activating as possible and positively formulated. It should describe what our chair should do, not what it should not do. Typically, the answer begins with an activating verb (with 'to be', 'to have' and 'to provide' being considered as non-activating verbs). In this case, FR_1 is defined as "Enable people to stay comfortably in a place for a longer period of time". This is shown in the upper left part of Figure 6.

The next step is to determine "how this is done". This is basically a transformation from the functional to the physical domain, describing what the solution will look like. In this case, we determine that there will be a mechanical construction that will support our body parts while we are staying somewhere.

The third and last decision to make at this hierarchical level is to choose what our goals are when the production of this product starts; will we develop a new manufacturing method, e.g. a particular kind of 3D printing or rather use standard processes? In this case, we decide that we will use standard, known process technology, as we rather focus on the functionality of the chair and what it looks like. This choice completes the first 'zig' of our hierarchical descent.

Next, we 'zag' back to the functional domain and continue the decomposition of the functionality of our chair. We decide that a good and comfortable chair needs to support both our back and bottom. In the physical domain, it can be seen that this will be realised with vertical and horizontal surfaces or planes, respectively. This decision is followed by the choice in the process domain that we intend to use some pressing process to manufacture the chair, which concludes the second hierarchical level and allows us to zag down one level further.

At the third level, something important happens while defining the embodiment of our chair. This can be investigated by looking at the FRs:

$FR_{1.2}$ = Ensure comfy back angle
$FR_{2.2}$ = Ensure comfy knee angle

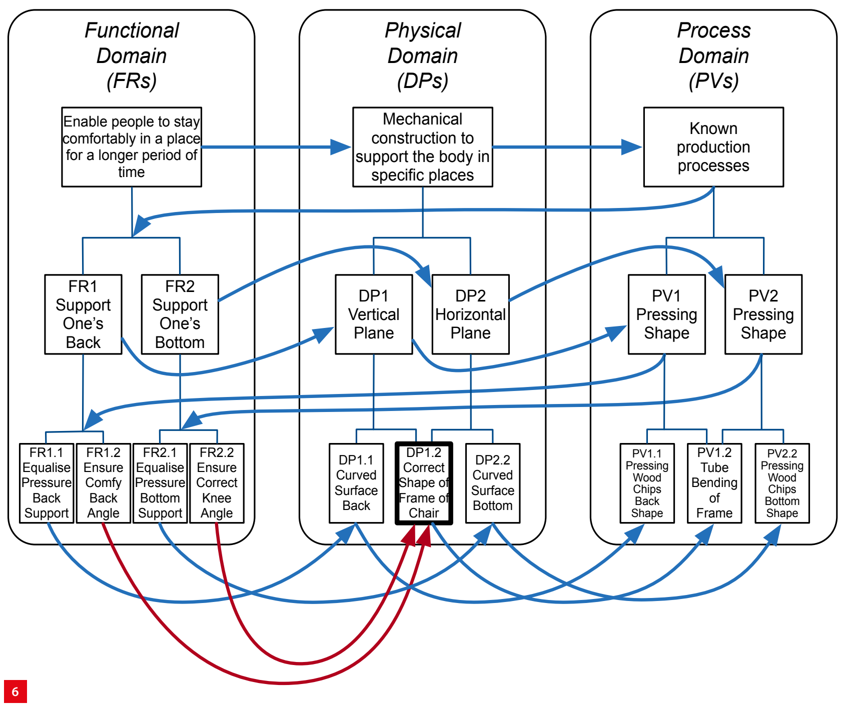
These FRs need to be satisfied with a single DP:

$DP_{1.2}$ = Correct shape of the frame of our chair

The design equation then has the shape of:

$$\begin{bmatrix} FR_{1.2} \\ FR_{2.2} \end{bmatrix} = \begin{bmatrix} X \\ X \end{bmatrix} \cdot [DP_{1.2}]$$

The two planes of the chair, vertical and horizontal, are attached to a single frame that needs to meet two functions, which warns us that we are dealing with a coupled system.


Functional Domain (FRs)
Physical Domain (DPs)
Process Domain (PVs)
Enable people to stay comfortably in a place for a longer period of time
Mechanical construction to support the body in specific places
Known production processes
FR1 Support One's Back
FR2 Support One's Bottom
DP1 Vertical Plane
DP2 Horizontal Plane
PV1 Pressing Shape
PV2 Pressing Shape
FR1.1 Equalise Pressure Back Support
FR1.2 Ensure Comfy Back Angle
FR2.1 Equalise Pressure Bottom Support
FR2.2 Ensure Correct Knee Angle
DP1.1 Curved Surface Back
DP1.2 Correct Shape of Frame of Chair
DP2.2 Curved Surface Bottom
PV1.1 Pressing Wood Chips Back Shape
PV1.2 Tube Bending of Frame
PV2.2 Pressing Wood Chips Bottom Shape

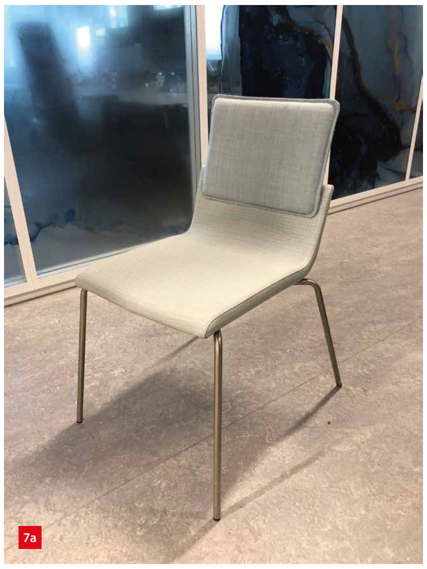
6

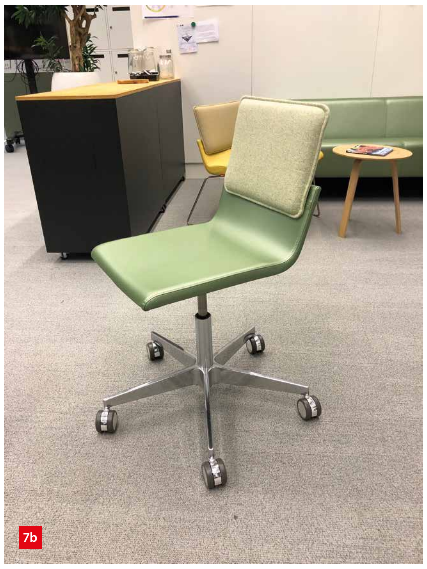
Decomposition of FRs, DPs and PVs while zigzagging downwards.

This is obviously caused by the fact that a single geometry of the frame determines the satisfaction of $FR_{3.2}$ and $FR_{3.4}$.

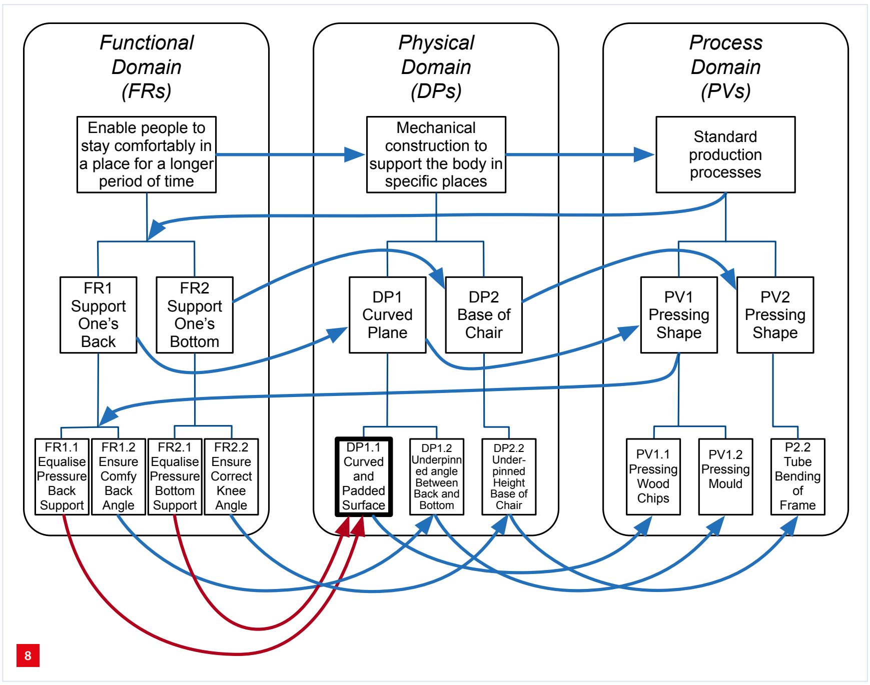
Note that this is not necessarily a bad outcome, as there may be reasons to combine functions, e.g. for aesthetic or manufacturability reasons. However, it warns us that the coupling is present, limiting our design freedom, and that it could bother us now or later. Figure 7 shows chairs where the coupling is eliminated to enable a modular design.

In the first case (Figure 7a), the vertical and horizontal supports are integrated into a single body that determines


7a


7b

Two chairs with a modular construction that prevents the frame from being coupled.


Functional Domain (FRs)
Physical Domain (DPs)
Process Domain (PVs)
Enable people to stay comfortably in a place for a longer period of time
Mechanical construction to support the body in specific places
Standard production processes
FR1 Support One's Back
FR2 Support One's Bottom
DP1 Curved Plane
DP2 Base of Chair
PV1 Pressing Shape
PV2 Pressing Shape
FR1.1 Equalise Pressure Back Support
FR1.2 Ensure Comfy Back Angle
FR2.1 Equalise Pressure Bottom Support
FR2.2 Ensure Correct Knee Angle
DP1.1 Curved and Padded Surface
DP1.2 Underpinned angle Between Back and Bottom
DP2.2 Under-pinned Height Base of Chair
PV1.1 Pressing Wood Chips
PV1.2 Pressing Mould
P2.2 Tube Bending of Frame

8

Decomposition of FRs, DPs and PVs for the chairs of Figure 7.

10

Example of a decoupled chair as discussed in the text.

the right angles. The frame itself just supports the height of that body. Figure 7b shows an alternative version of the same chair where adjustment of the height is enabled.

Further analysis of these chairs learns that the designs also have theirs limitations. This can be seen when the decomposition is revisited, and the design parameters are updated as shown in Figure 8.

The design equation for the alternative chair design with the curved body to support both back and bottom, has the following form:

$FR_{1.1}$ = Equalise pressure for back
$FR_{2.1}$ = Equalise pressure for bottom

These FRs need to be satisfied with a single DP:

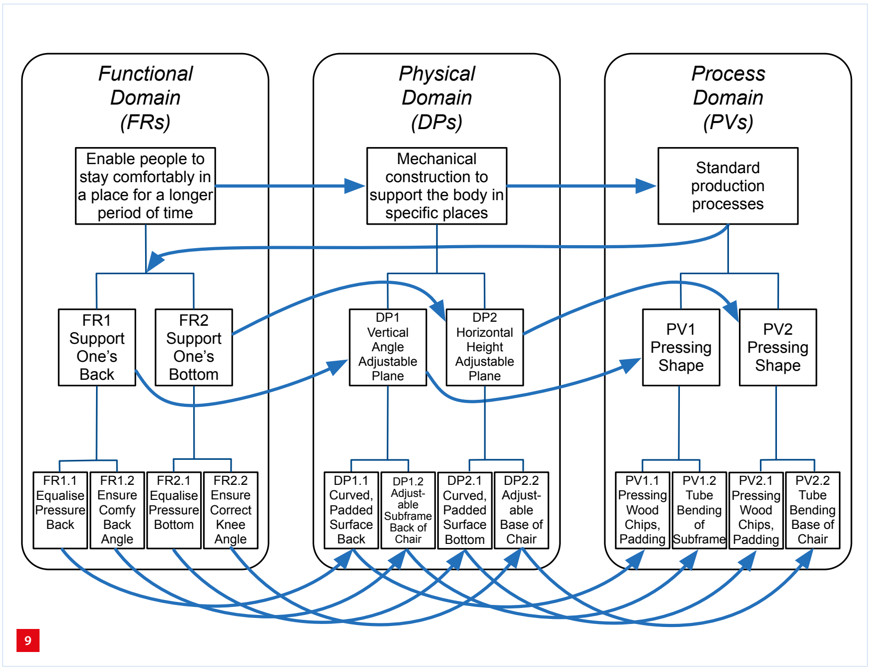
$DP_{1.1}$ = Curved and padded part

$$\begin{bmatrix} FR_{1.1} \\ FR_{2.1} \end{bmatrix} = \begin{bmatrix} X \\ X \end{bmatrix} \cdot [DP_{1.1}]$$

Here, the design equation indicates again coupling of the FRs, in a different way, however. In this case, it enables the designer to produce two versions of the chair, in Figures 7a and 7b, the latter being height adjustable. It may be difficult to change the angle of the back support of the chair, since it needs an adjustment of the mould, which requires a large investment. Again, it is the choice of the designer whether this is acceptable or not. If this coupling is perceived as a problem the chair could be further improved as shown in Figure 9.

The FRs and DPs of the improved system have the following form:

$FR_{1.1}$ = Equalise pressure for back
$FR_{1.2}$ = Ensure comfortable angle of back support
$FR_{2.1}$ = Equalise pressure for bottom
$FR_{2.2}$ = Ensure correct angle of knees


Functional Domain (FRs)
Physical Domain (DPs)
Process Domain (PVs)
Enable people to stay comfortably in a place for a longer period of time
Mechanical construction to support the body in specific places
Standard production processes
FR1 Support One's Back
FR2 Support One's Bottom
DP1 Vertical Angle Adjustable Plane
DP2 Horizontal Height Adjustable Plane
PV1 Pressing Shape
PV2 Pressing Shape
FR1.1 Equalise Pressure Back
FR1.2 Ensure Comfy Back Angle
FR2.1 Equalise Pressure Bottom
FR2.2 Ensure Correct Knee Angle
DP1.1 Curved, Padded Surface Back
DP1.2 Adjustable Subframe Back of Chair
DP2.1 Curved, Padded Surface Bottom
DP2.2 Adjustable Base of Chair
PV1.1 Pressing Wood Chips, Padding
PV1.2 Tube Bending of Subframe
PV2.1 Pressing Wood Chips, Padding
PV2.2 Tube Bending Base of Chair

9

Decomposition of FRs, DPs and PVs for a fully decoupled design of a chair.

These FRs need to be satisfied with the following DPs:

$DP_{1.1}$ = Curved, padded surface of back support
$DP_{1.2}$ = Adjustable subframe back of the chair
$DP_{2.1}$ = Curved, padded surface of bottom support
$DP_{2.2}$ = Adjustable base of the chair

The full design equation is then as follows:

$$\begin{bmatrix} FR_{1.1} \\ FR_{1.2} \\ FR_{2.1} \\ FR_{2.2} \end{bmatrix} = \begin{bmatrix} X & 0 & 0 & 0 \\ 0 & X & 0 & 0 \\ 0 & 0 & X & 0 \\ 0 & 0 & 0 & X \end{bmatrix} \cdot \begin{bmatrix} DP_{1.1} \\ DP_{1.2} \\ DP_{2.1} \\ DP_{2.2} \end{bmatrix}$$

This indicates that we are dealing with an uncoupled system of which all the FRs can be satisfied with their own DP. Such a fully decoupled chair could look like the one in Figure 10.

Conclusion

AD is a tool that can help designers analyse and evaluate design decisions. Designers may need some time to familiarise themselves with the method, but the functionality that comes with it is well worth the effort. It exposes where a design has dependencies and what can be done about them. In many cases, the coupling can even be eliminated with little effort, making the design more robust. In many cases it increases the possibilities to fine-tune a design at a later stage of the project.

To provide insight into how the method can be applied in a low-threshold manner, two examples of AD have been worked out. While these examples are not high-tech systems, it does not take much imagination to see how AD can be applied in complicated or even complex systems. In addition, it helps the user to keep an overview of the coupled structures that occur in system designs.

REFERENCE
[1] *www.axiomaticdesign.org*

BILZ PRODUCTS – MADE IN GERMANY

WE SOLVE ANY VIBRATION PROBLEM. YOUR PARTNER FOR BETTER RESULTS.

Bilz®
www.bilz.ag

OUDE REIMER
Trillingsisolatie

Meer informatie:

Mail: info@oudereimer.nl I Tel: 035 646 0820 I www.oudereimer.nl I www.oudereimer.shop