DISCRETE-MODELLING OF PROCESS COMPONENTS INTERACTIONS USING THE DESIGN STRUCTURE MATRIX

Klaus J. Diepold, Franz J. Winkler and Boris Lohmann
TU München, Germany

Corresponding author: K. J. Diepold, TU München, Institute of Automatic Control
85748 Garching, Boltzmannstraße 15, Germany, kj.diepold@mytum.de

Abstract. The product development community faces new challenges due to a drastic increase in the scale and complexity of engineered systems. Modelling these systems is becoming increasingly important as it allows a better process/product understanding for analysing changes while avoiding several iteration loops. In this article a recent decomposition principle, called DSM (*Design Structure Matrix*), is extended to a discrete-time dynamical system based on process components. The system is modelled as detailed as necessary by aligning the used interaction knowledge to an introduced weighting coefficient. Proportional, functional and qualitative knowledge of the coherences is considered and linguistic process knowledge is thereby included via *Fuzzy-Logic*. Hence, the developed strategy allows to handle nonlinear relations as well as uncertainties. First applications are shown on two simplified examples, together with analyses of process fixed points and attracting regions using well-known mathematical methods.

1 Introduction

Process models are very important for the analysis and implementation of process improvement projects. Therefore, structured methods helping to handle the complexity during modelling and analysing are required. One of the most famous process modelling approaches was established by Jay W. Forrester and is known as *System Dynamics* [8], [9] (and overview in [21]). The main relationships between the elements are expressed by a time-dependent integral interdependency. This results in an examination of levels' inflow and outflow, so that from system theoretical point of view the created dynamical system is described by coupled differential equations. Most of the applications of this approach are in urban commuter systems, social systems and industrial dynamics [19], [8]. Various theoretical and practical developments made *System Dynamics* to a powerful modelling and simulation tool. However, it still comes along with some disadvantages. *System Dynamics* models are based on *Causal Loop Diagramms* (CLD) or *Level-Rate-Diagramms* (LRD). Getting both in an appropriate way, requires knowledge of all coherences [15]. There exist several algorithms helping to transform an insufficient CLD into an appropriate LRD. The obtained representation of a system is therefore not unique, because of the different assumptions required by the algorithms. For more informations see [6], [5] and [4]. Additionally, the required integral process behavior is often not clearly defined and modelling purely discrete-time dynamical systems can cause problems [16]. Due to the required system knowledge, there exist a couple of simulation tools specifically designed for *System Dynamics* applications, helping to analyse such systems. However, exclusively trusting simulations for a better process understanding, can result in drawing false cause-and-effect conclusions, due to the summation of quantification mistakes [7].

Large, multidisciplinary and networked systems are affected by reams of interactions within and between several domains like components, tasks and persons [1]. A recent decomposition principle to handle such complexity and helping to get a better process/product understanding uses a simple matrix structure format called DSM (*Design Structure Matrix*), which was developed by D. Steward [17]. A basic DSM is a square matrix with identical column and row labels. Each of them represents an item of the specified system and the whole DSM documents which items are interacting. Depending on the considered domain, there exist several DSM variations, providing a more comprehensive view of the interactions taking place.

The effects of identified feedback loops inside the DSM can not most often be analysed using currently available methods. Currently these feedbacks are considered as potential sources of instability and therefore removed (if possible) by a system redesign [11]. Such redesigns are often still done by trial and error without really knowing their benefits.

The new approach in this paper is extending the component-based DSM to a discrete-time dynamical system in order to allow analyses of the feedbacks therein. The approach offers different options of filling the DSM and thereby of describing the dynamics:

- linear dynamics (i.e. proportional relations)
- nonlinear relations
- uncertainties

This allows the DSM to tackle a wide range of problems and analysis tools while simultaneously bypassing some disadvantages of *System Dynamics*.

The remainder of this paper is organized as follows: First, the handling of the *Design Structure Matrix* (especially the used component-based DSM) is presented in Section 2. The expansion of the DSM will be introduced in Section 3. In Section 4 two application examples of the illustrated approach are presented together with analytical analyses of the processes fixed points and attracting regions using well-known mathematical methods [20].

2 The Design Structure Matrix

A DSM displays the relationships between items of a system or process in a compact matrix structure and is therefore a visual and analytically advantageous format [3]. It is a square matrix with identical column and row labels. Each of them represents an item of the specified system. An off-diagonal marking signifies the dependency of one element on another. As there exist different notations of reading, we restrict ourselves in this article to the following notation: An "X" marking along a column reveals, which other variables that column‘s item depends on. Accordingly, it follows that marks above (or under) the main-diagonal clarifies feedforward dependencies. If interacting elements are linked across the main-diagonal, a feedback is revealed. Additionally, reading down a column of the DSM reveals system inputs, whenever no marking is found. Repeating this procedure along a row reveals system outputs. As mentioned before, there exist several DSM variations providing a more comprehensive view of the interactions taking place depending on the considered domain. A distinguished overview is given in [2]. In this article, the component-based DSM is used, which obviously documents the relationships between system components. An organized taxonomy can help to identify these dependencies and more than one interaction type can be included by making the DSM three-dimensional [13]. By restructuring the columns and rows in a proper way, subareas of the system interactions are detectable such as strictly feedforward dependency parts, feedbacks and decoupled clusters. Several algorithms exist, which restructure and subclassify the DSM accordingly. Therefore, this matrix is a simple, but powerful graphical representation of a complex dynamical system or process, based on a component-by-component view. Hence, the DSM is widely used in product development. Figure 1 shows an enlarged component-based DSM of a climate control system from T.R Browning [2] and the detected element-clusters therein. The DSM structure is adjusted to the introduced notation.

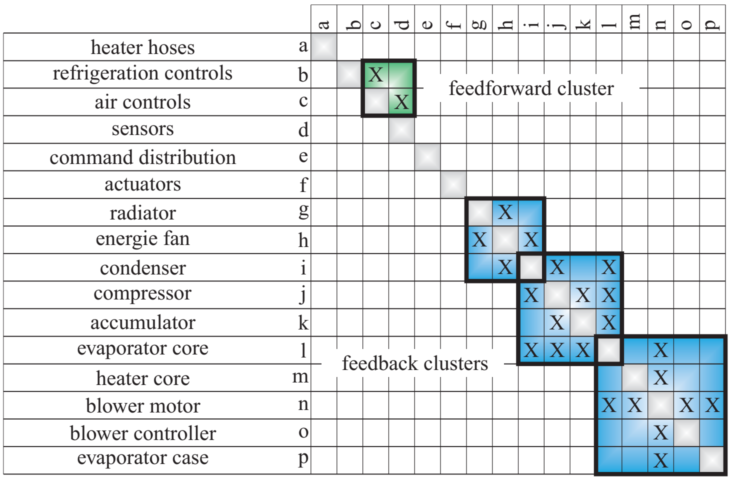

Figure 1: Component-based DSM of a climate control system - detected clusters are marked [2].

Further examples and a description of the algorithms can be found in [12] and [10].

The discovered feedback loops are essential for the dynamical behavior of the overall process. In order to reduce their potential source of instability, these feedbacks need to be further analysed, by adding more information to the DSM in a structured way.

3 Expanding the DSM: The DynS-DSM

The extended component-based DSM, denoted as DynS-DSM (*Dynamic System Design Structure Matrix*), pinpoints relevant interactions between component attributes, which are classified according to their priority. Therefore, first the participating attributes a_q of the concerned clusters elements j have to be detected. This is archived by merging all of the attributes in a matrix format, having the style of the DSM structure. The rows are in accordance with the rows of the component-based DSM. Reading across a row reveals the involved attributes of the component. Figure 2 shows an appropriate matrix format of the attribute detection. At the end an (n x m) attribute

	attribute 1	attribute 2	attribute 3	…	attribute m-1	attribute m
component 1	$a_{1,1}$	$a_{1,2}$			$a_{1,m-1}$	
component 2		$a_{2,2}$	$a_{2,3}$			$a_{2,m}$
⋮	…	…	…	…	…	⋮
component n					$a_{n,m-1}$	$a_{n,m}$

Figure 2: Matrix format of the attribute detection.

matrix

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & 0 & \dots & a_{1,m-1} & 0 \\ 0 & a_{2,2} & a_{2,3} & \dots & 0 & a_{2,m} \\ \vdots & \dots & \dots & \dots & \dots & \vdots \\ 0 & 0 & 0 & \dots & a_{n,m-1} & a_{n,m} \end{bmatrix} \tag{1}$$

is obtained, where n is the number of components and m is the number of detected attributes. The further studies are based on the attribute matrix **A**.

Now, the developed attribute matrix will be divided in several analysis matrices. The fragmentation of the attributes is fundamental for the finally obtained system description. This results in at least three analysis matrices,

- **X** (system state declaration)
- **I** (system input declaration)
- **O** (system output declaration),

with

$$\mathbf{X} = \begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,m} \\ x_{2,1} & x_{2,2} & \dots & x_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \dots & x_{n,m} \end{bmatrix}, \quad \mathbf{I} = \begin{bmatrix} i_{1,1} & i_{1,2} & \dots & i_{1,m} \\ i_{2,1} & i_{2,2} & \dots & i_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ i_{n,1} & i_{n,2} & \dots & i_{n,m} \end{bmatrix}, \quad \mathbf{O} = \begin{bmatrix} o_{1,1} & o_{1,2} & \dots & o_{1,m} \\ o_{2,1} & o_{2,2} & \dots & o_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ o_{n,1} & o_{n,2} & \dots & o_{n,m} \end{bmatrix}. \tag{2}$$

The entries of these matrices are equal to the corresponding attribute $a_{j,q}$ or zero. Adding further analysis matrices to ease the actual modelling is always possible, but should be done with respect to the particular kind of process and its complexity. For example a influence matrix **R** could be considered in a first step. The correct fragmentation can easily be ensured, by the condition

$$\mathbf{A} \stackrel{!}{=} \mathbf{X} + \mathbf{I} + \mathbf{O} + \ldots + \mathbf{R}. \tag{3}$$

Hence, it is guaranteed that every attribute according to **A** occurs only in one of the analysis matrices.

As mentioned before, the original component-based DSM will be extended to an attribute-based DSM, which can now be created from the analysis matrices. In order to keep it structured, all attributes belonging to the same component are listed next to each other. This can be done using a simple algorithm, by reading across the same row of all introduced analysis matrices sequentially and filtering all non zeros entries out, before going on with the next row. The resulting matrix structure is denoted with respect to the DSM as DynS-DSM (*Dynamic System Design Structure Matrix*). An example of a resulting form of the DynS-DSM is depicted on the left of figure 3, together with a detected cluster. In order to keep the figure clear, every considered component was assumed to have one system state, input and output attribute. The first attribute represents the state for all components. Thus, only the first column of **X** is unequal zero (see figure 2 and equation (2)). The second attribute represents all of the inputs and the third one all of the outputs. However, these are no restrictions, because the DynS-DSM will be a square matrix for any arbitrary case and so all properties of the DSM are always transferable.

The interactions between the attributes are clarified by using weighting coefficients in the range [0, 1]. Each coefficient expresses the priority of the associated interaction due to the dynamical behavior of the process and is thus equivalent to the requested approximation level. P.S. Raghuvanshi and S. Kumar used a similar method to describe the interactions between system elements in [14] and called it a *real valued connection*.

Depending on the available information about the process, the introduced range has to be subdivided. Therefore, in this article three classical description forms, which are expressed by the weighting factor according to the level of knowledge, are considered (figure 3):

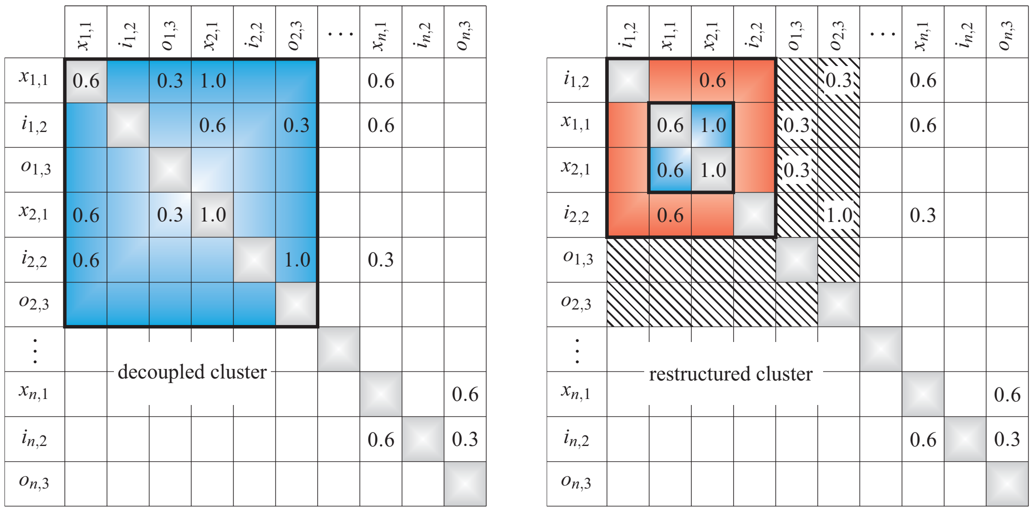

Figure 3: Weighted DynS-DSM with a marked decoupled cluster (at left) and for the system analysis restructured cluster (at right): feedback (inner), input-enlarged feedback (exterior) and shaded negligible output part.

- 0.3 (linguistical description required)
- 0.6 (proportional description required)
- 1.0 (functional description required)

The notation of reading is basically the same as for the DSM. Hence, it can be easily proven whether the DynS-DSM is designed properly. All $i_{j,q}$ and $o_{j,q}$ are not allowed to be linked in their corresponding column or row. If this not fulfilled, the analysis matrices have to be restructured or an additional one has to be included. For a system analysis, the outputs are negligible first (can be determined if the phase portrait is known) and the detected clusters have to be enlarged with their involved inputs (see figure 3 at right). Accordingly, high-weighted (significant) clusters are detectable and the involved interactions therein can be formulated geared to the individual weighting coefficient. The cluster enlargement with the inputs can also be done in the component-based DSM. Thus, beside input attributes also input components of a feedback cluster can be considered in the DynS-DSM (see example 4.2).

Next, the formulated interactions are inserted in the final DynS-DSM. Proportional relationships are inserted directly, the functional coherences of the attributes are entered in a separate section. Further description forms (like linguistical) are included in additional layers, by making the DynS-DSM three-dimensional. The structure of the final DynS-DSM is shown in figure 4, where the two corresponding layers of the detected feedback cluster from figure 3 are depicted. As mentioned before, interpreting the resulting DynS-DSM, reveals the system behavior over time by a discrete-time dynamical system. The notation of reading is introduced on three interactions from the marked cluster out of figure 3, one for each considered description.

$$
\begin{aligned}
x_{1,1}^{(k+1)} &= k_1 x_{1,1}^{(k)} + k_2 x_{2,1}^{(k)} + k_3 i_{2,2}^{(k)}, \\
x_{2,1}^{(k+1)} &= \underbrace{\sqrt{k_4 (x_{1,1}^2)^{(k)} + k_5 x_{1,1}^{(k)} + k_6 (x_{2,1}^3)^{(k)}}}_{\mathbf{f}_1} + k_7 i_{1,2}^{(k)}, \\
o_{1,3}^{(k+1)} &= \begin{cases} F_1 = \begin{cases} LOW & \text{if } x_{1,1}^{(k)} \text{ and } x_{2,1}^{(k)} \text{ are MEDIUM,} \\ MEDIUM & \text{if } x_{1,1}^{(k)} \text{ and } x_{2,1}^{(k)} \text{ are HIGH,} \end{cases} \\ F_2 = \begin{cases} HIGH & \text{if } x_{1,1}^{(k)} \text{ or } x_{2,1}^{(k)} \text{ are LOW,} \end{cases} \end{cases}
\end{aligned}
\tag{4}
$$

where (k) stands for the amount of iterations of discrete-time dynamical systems. The first interaction is purely proportional and its parts are expressed in the DynS-DSM by reading down the column of $x_{1,1}$. First of all, the prefix of the interaction part, than the proportional coefficient and last the connection sign between this coefficient and the attribute of the corresponding row. The functional relationship f_1 of the second interaction is represented in detail in an added sector. The nonlinear connection is summarised for each attribute on the self depending cell of f_1 and the corresponding proportional coefficients are included as already known. If an attribute appears several times in the nonlinear summarisation, the proportional coefficients are listed in the polynomial order of the attribute. If the nonlinearity is too extensive, a link to an external description can be placed instead. The linguistical operations "and" and "or" in (4) are merged together in the functions F_1 and F_2. These functions are inserted in the *Fuzzy*-layer (right of figure 4) and modeled using *Fuzzy-Logic* operations such as "MAX(MIN)". The notation of reading is almost the same as before, only the interaction between F_1, F_2 and $o_{1,3}$ is splitted up

	$x_{1,1}$	$i_{1,2}$	$o_{1,3}$	$x_{2,1}$	$i_{2,2}$	$o_{2,3}$	f_1
$x_{1,1}$	$+k_1\cdot$						$+k_4\cdot\ +k_5\cdot$
$i_{1,2}$				$+k_7\cdot$			
$o_{1,3}$							
$x_{2,1}$	$+k_2\cdot$						$+k_6\cdot$
$i_{2,2}$	$+k_3\cdot$						
$o_{2,3}$							
f_1				$+1\cdot$			$+\sqrt{x_{1,1}^2+x_{1,1}+x_{2,1}^3}$

	$x_{1,1}$	$i_{1,2}$	$o_{1,3}$	$x_{2,1}$	$i_{2,2}$	$o_{2,3}$	F_1	F_2
$x_{1,1}$							$+1\cdot$	$+1\cdot$
$i_{1,2}$								
$o_{1,3}$								
$x_{2,1}$							$+1\cdot$	$+1\cdot$
$i_{2,2}$								
$o_{2,3}$								
F_1			L $\rightarrow$ M M $\rightarrow$ H				$\mathrm{MIN}(x_{1,1}, x_{2,1})$	
F_2			H $\rightarrow$ L					$\mathrm{MAX}(x_{1,1}, x_{2,1})$

Figure 4: DynS-DSM Layers: proportional and functional relations (at left), *Fuzzy*-relations (at right).

in the parts used in equation (4). Using this notation, a wide range of interactions can be displayed in a compact form and further interaction types can be added easily. Additionally, all proportional coefficients k_p can be seen as changeable inputs. Therefore, the analysis of the process can be balanced between qualitative and quantitative modelling [7].

4 Application examples

In this section the introduced approach is applied to two simplified industrial examples. The relationships between the elements are described as in Section 3 (linguistical, proportional and functional). The first example is a vehicle design iteration process, where initial conditions can be chosen by the customer. As second example an assembling process is considered, whose velocity depends on a process parameter λ.

4.1 Vehicle design iteration process

This example describes the interactions between components of a vehicle and their dynamical behavior when some modifications are required, because of initial conditions chosen by the customer. Both the associated component-based DSM and the reconstructed DSM are shown in figure 5. The detected feedback subarea is highlighted. The system is reduced to this subarea, modelled qualitatively and further analysed below. The detected attributes of the

	starter	control module	gearbox	engine	chassis	brake system	suspension	wheel	fender	toe/camber
starter										
control module										
gearbox					X					
engine	X	X	X		X			X		
chassis						X	X	X		
brake system				X				X		
suspension										X
wheel							X		X	X
fender										
toe/camber										

	starter	control module	toe/camber	suspension	fender	wheel	brake system	chassis	gearbox	engine
starter										
control module										
toe/camber										
suspension			X							
fender										
wheel			X	X	X					
brake system						X				X
chassis				X		X	X			
gearbox								X		
engine	X	X				X		X	X	

Figure 5: Basic component-based DSM (at left) and restructured DSM with marked feedback (at right).

component $a_{j,q}$ are assumed to be: the performance (alternatively the rating) $a_{j,1}$, the mass $a_{j,2}$, the costs $a_{j,3}$ and the lower limits of the ratings $a_{j,4}$. The order of the components are the same as in the marked DSM section. Thus, the attribute matrix based on (1) can be calculated as:

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & 0 & a_{1,3} & 0 \\ a_{2,1} & 0 & a_{2,3} & 0 \\ a_{3,1} & a_{3,2} & a_{3,3} & 0 \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{bmatrix}.$$

The subdivision of the attribute matrix results in four analysis matrices: **X** (system states), **I** (inputs), **O** (outputs) and **R** (influence). The division of the detected attributes is

$$\mathbf{X}=\begin{bmatrix}0&0&0&0\\a_{2,1}&0&0&0\\a_{3,1}&0&0&0\\a_{4,1}&0&0&0\end{bmatrix},\quad \mathbf{I}=\begin{bmatrix}0&\dots&0\\\vdots&\ddots&\vdots\\0&\dots&a_{4,4}\end{bmatrix},\quad \mathbf{O}=\begin{bmatrix}0&0&a_{1,3}&0\\\vdots&\vdots&\vdots&\vdots\\0&0&a_{4,3}&0\end{bmatrix},\quad \mathbf{R}=\begin{bmatrix}a_{1,1}&0&0&0\\0&0&0&0\\0&a_{3,2}&0&0\\0&a_{4,2}&0&0\end{bmatrix}. \tag{5}$$

Hence, there is only one fixed system input, the brake system has no state attribute and a proper fragmentation is guaranteed, because equation (3) is fulfilled. The used descriptions of the attribute interactions and thus the weighting coefficients are the same as in Section 3. The considered interactions weighting is shown in figure 6. Also the restructured weighted DynS-DSM is depicted and the detected clusters for the system analysis are marked. The linguistical connections are modelled via *Fuzzy IF-THEN-RULES*, which were introduced by Zadeh [22].

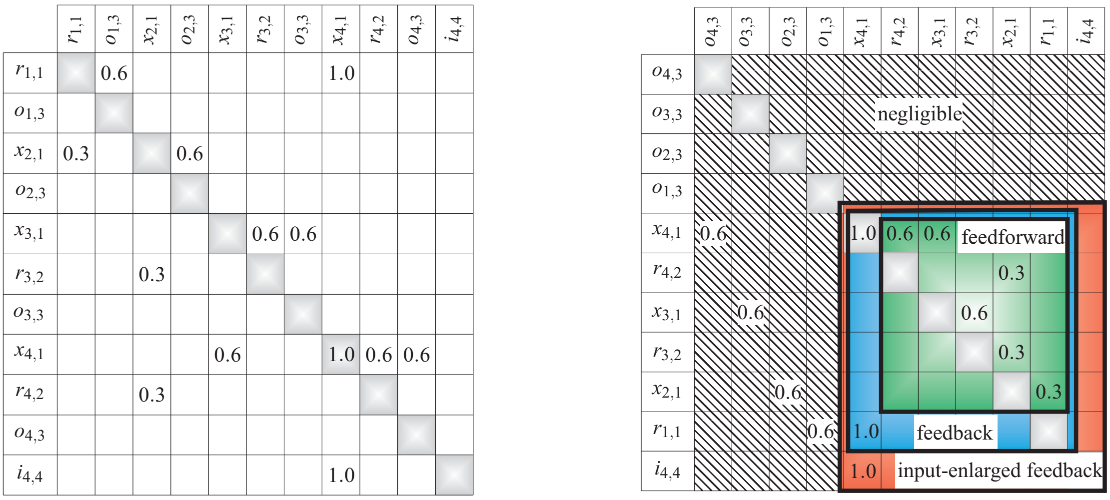

Figure 6: Weighted DynS-DSM (at left) and restructured with marked clusters for system analysis (at right).

Next, the *Fuzzy IF-THEN* modelling will be explained in details but will not be inserted in the final DynS-DSM (figure 9), because of the analogy to the *Fuzzy*-layer explained in Section 3. First, the chassis rating dependency on the engine and the gearbox mass is considered. The *Fuzzy IF-THEN-RULES* are assumed to be:

$$x_{2,1}^{(k+1)}=\begin{cases}\mathit{HIGH} & \text{if } r_{4,2}^{(k)} \text{ is HIGH} \quad\text{or}\quad r_{3,2}^{(k)} \text{ is HIGH},\\ \mathit{HIGH} & \text{if } r_{4,2}^{(k)} \text{ is MEDIUM and } r_{3,2}^{(k)} \text{ is MEDIUM},\\ \mathit{MEDIUM} & \text{if } r_{4,2}^{(k)} \text{ is MEDIUM and } r_{3,2}^{(k)} \text{ is LOW},\\ \mathit{MEDIUM} & \text{if } r_{4,2}^{(k)} \text{ is LOW} \quad\text{and } r_{3,2}^{(k)} \text{ is MEDIUM},\\ \mathit{LOW} & \text{if } r_{4,2}^{(k)} \text{ is LOW} \quad\text{and } r_{3,2}^{(k)} \text{ is LOW}.\end{cases}$$

The possible intervals of the inputs $r_{4,2}$ and $r_{3,2}$ are standardised to the used *Fuzzy*-space $r_{4,2}^*$ and $r_{3,2}^* \in [1, 10]$. Suitable trapezoidal membership functions $\mu_{Fuzz,i}$ are applied for *Fuzzification* and the membership functions $\mu_{DeFuzz,i}$ for the *Defuzzification*, required to get $x_{2,1,i}^*$, are chosen as singletons. Both final used membership functions are shown in figure 7.

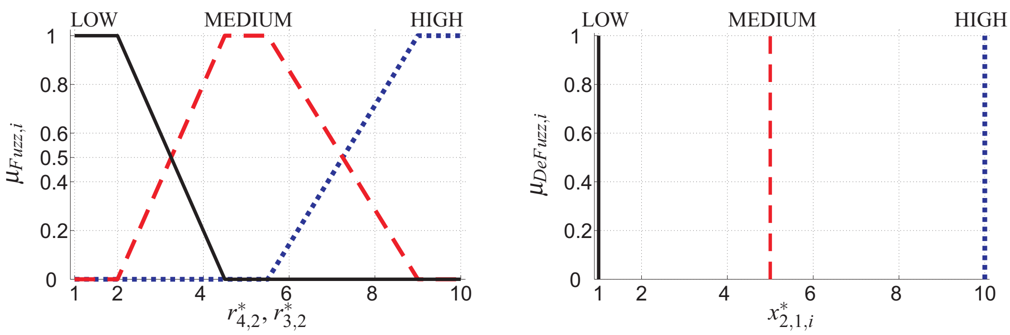

Figure 7: Membership functions: For *Fuzzification* $\mu_{Fuzz,i}$ (at left) and for *Defuzzification* $\mu_{DeFuzz,i}$ (at right).

The chosen M. Sugeno *Defuzzification* method [18] in combination with singletons is transferred to a simple *Center* *of Maximum* summation

$$x^*_{2,1,Sug} = \frac{\sum_{i=1}^{3} \mu_{DeFuzz,i} \cdot x^*_{2,1,i}}{\sum_{i=1}^{3} \mu_{DeFuzz,i}}, \tag{6}$$

where $x^*_{2,1,Sug}$ is the final value of the chassis rating in the *Fuzzy*-space. Additionally, due to the used singletons it is guaranteed that the whole *Fuzzy*-space can be utilized. The nonlinear characteristic function $x^*_{2,1,Sug} = g(r^*_{4,2}, r^*_{3,2})$ can be approximated, by solving (6) for all combinations of the inputs. The functional surface is shown on the left side of figure 8.

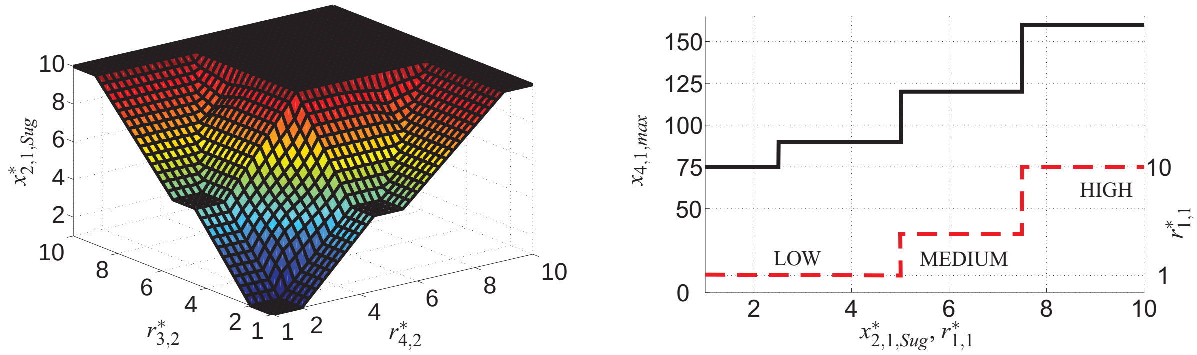

Figure 8: Functional *Fuzzy*-surface $g(r^*_{4,2}, r^*_{3,2})$ (at left) and transformation $h(x^*_{2,1,Sug})$ (dashed), $f_2 := r^*_{1,1} \rightarrow x_{4,1,max}$ equal $f_2 := x^*_{2,1,Sug} \rightarrow x_{4,1,max}$ (solid) (at right).

As mentioned before, the brake system has no state attributes in **X** (see (5)) and therefore no attribute to be analysed in the system representation. In addition, it is part of a feedforward cluster (see figure 6). Hence, the corresponding elements can be fully eliminated from the system analysis. This is done in the considered example as follows: As can be seen from figure 6, the chassis rating $x_{2,1}$ affects the brake system rating $r_{1,1}$, which in turn affects the engine rating $x_{4,1}$. The feedforward part of these interactions can be merged in combination with the transformation from the *Fuzzy*- into the origin-space. The *Fuzzy*-interaction $r^*_{1,1} = h(x^*_{2,1,Sug})$ is depicted by the dotted line on the right of figure 8. The solid line shows the chosen transformation f_2 from $r^*_{1,1}$ to the maximal possible value of the engine performance $x_{4,1,max}$. Hence, f_2 is also the transformation from $x^*_{2,1,Sug}$ to $x_{4,1,max}$, whereby $r^*_{1,1}$ is eliminated.

The used proportional and functional connections are just inserted in the final DynS-DSM as shown in figure 9. By

	$r_{1,1}$	$o_{1,3}$	$x_{2,1}$	$o_{2,3}$	$x_{3,1}$	$r_{3,2}$	$o_{3,3}$	$x_{4,1}$	$r_{4,2}$	$o_{4,3}$	$i_{4,4}$	f_1	f_2
$r_{1,1}$		$+k_1\cdot$											
$o_{1,3}$													
$x_{2,1}$				$+k_2\cdot$									$+1\cdot$
$o_{2,3}$													
$x_{3,1}$						$+k_4\cdot$	$+k_5\cdot$						
$r_{3,2}$													
$o_{3,3}$													
$x_{4,1}$					$+k_3\cdot$				$+k_6\cdot$	$+k_7\cdot$		$+1\cdot$	
$r_{4,2}$													
$o_{4,3}$													
$i_{4,4}$												$+1\cdot$	
$\mathbf{f_1}$								$+1\cdot$				$\min(\max(x_{4,1}, i_{4,4}), f_2)$	
$\mathbf{f_2}$													$x^*_{2,1,Sug} \rightarrow x_{4,1,max}$

Figure 9: DynS-DSM Layers: proportional and functional relations between the attributes.

reading the DynS-DSM as introduced in Section 3 the process behavior is mapped in the discrete-time dynamical system

$$\begin{bmatrix} x^*_{2,1,Sug} \\ x_{3,1} \\ x_{4,1} \end{bmatrix}^{(k+1)} = \begin{bmatrix} g(r^*_{4,2}, r^*_{3,2}) \\ k_3\, x_{4,1} \\ \min(\max(x_{4,1}, i_{4,4}), f_2(x^*_{2,1,Sug})) \end{bmatrix}^{(k)} = \begin{bmatrix} g((k_6\, x_{4,1})^* + (k_4\, x_{3,1})^*) \\ k_3\, x_{4,1} \\ \min(\max(x_{4,1}, i_{4,4}), f_2(x^*_{2,1,Sug})) \end{bmatrix}^{(k)}. \tag{7}$$

Analysing this system reveals a cascade arrangement starting with the last row of (7). Hence, by knowing the dynamics of $x_{4,1}$ the whole system behavior is describable. For it, $f_2(x^*_{2,1,Sug})$ from the right of figure 8 and the lower limit of the engine rate $i_{4,4}$ are needed. The meaningful intervals for the initial conditions (selectable by the customer), resulting from $f_2(x^*_{2,1,Sug})$ and the cascade arrangement of (7), are

$$x_{4,1}^{(0)} \in [i_{4,4},\, 160]\ \text{PS},\ x_{3,1}^{(0)} \in [k_3\, i_{4,4},\, k_3\, 160]\ \text{PS},\ {x^*_{2,1}}^{(0)} \in [1,\, 10]. \tag{8}$$

The *Fuzzy*-inputs, which are needed for the standardization described before, are now also allocable. However, even if the costumer chooses an inexpedient initial condition, the min-operator together with the cascade system format guarantee that the system gets back to the intervals of (8). Therefore, these span a positive invariant set in the phase space, defined in [20], and thus it is impossible for the system to become unstable.

The general condition for such systems' fixed points $\tilde{x}_{j,1}$ is

$$\begin{aligned} \begin{bmatrix} x^*_{2,1,Sug} \\ x_{3,1} \\ x_{4,1} \end{bmatrix}^{(k+1)} &\overset{!}{=} \begin{bmatrix} x^*_{2,1,Sug} \\ x_{3,1} \\ x_{4,1} \end{bmatrix}^{(k)} \\ \Rightarrow \begin{bmatrix} \tilde{x}^*_{2,1,Sug} \\ \tilde{x}_{3,1} \\ \tilde{x}_{4,1} \end{bmatrix} &= \begin{bmatrix} x^*_{2,1,Sug} \\ x_{3,1} \\ x_{4,1} \end{bmatrix}^{(k)} . \end{aligned} \tag{9}$$

This applied to the system (7) reveals

$$\tilde{x}_{4,1} = x_{4,1}^{(k+1)} \overset{!}{=} x_{4,1}^{(k)} \overset{!}{=} x_{4,1}^{(k-1)} \overset{!}{=} x_{4,1}^{(k-2)}, \tag{10}$$

where at least three iterations have to be done to be sure that the considered fixed point value of the engine rating is accounted for each state calculation. A warranty that (9) and (10) are fulfilled for arbitrary initial conditions is thereby also given through the systems properties (min-operator and cascade format). Thus, an asymptotic stable fixed point, which depends on the initial conditions, is always detectable. The other state values of the active fixed point are given by

$$\tilde{x}_{3,1} = k_3\, \tilde{x}_{4,1}, \quad \tilde{x}^*_{2,1} = g((k_6\, \tilde{x}_{4,1})^* + (k_4\, \tilde{x}_{3,1})^*). \tag{11}$$

Based on this fixed point, the nearest expedient combination of the car components is chosen and the final outputs $o_{j,q}$ can be determined using the DynS-DSM of figure 9.

At the end of this example it should be mentioned that, if *Fuzzy*-conditions are needed to prove stability, a two dimensional linearisation around the center of the considered region or even a surface fitting can be done. Also a derivation fitting at the borders of the region helps to find a function that approximates the linguistical knowledge beyond the considered area.

4.2 Assembly operation

The application described next is a simplified automated wheel assembling process, while the car is moved. Because of the detailed explanation of the first example, this one is kept short and only proportional and functional relationships between the elements are considered. The focus of this example is not on modelling, but on showing another important stability behavior, which is also detectable using the DynS-DSM approach. The component based DSM in figure 10 (at left) is reduced to the major components and already restructured. The basic feedback cluster and, as mentioned in Section 3, its enlargement by the involved inputs are marked. The enlarged feedback will be further analysed. The detected attributes, already distributed to the considered analysis matrices, are

$$\mathbf{X} = \begin{bmatrix} x_{1,1} & 0 & 0 \\ x_{2,1} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},\ \mathbf{I} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & i_{2,3} \\ 0 & i_{3,2} & 0 \end{bmatrix},\ \mathbf{O} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{12}$$

where the first columns represent the component velocity, the second the component mass and last the amount of to-affix screws. To stay consistent with the rest of this contribution, the same subdivision of the weighting coefficients is used. Figure 10 (at right) shows according to (12) the considered weighted DynS-DSM with marked feedback and the input-enlarged feedback cluster.

The required functional interaction describing $x_{2,1}$ is assumed as a second-order polynomial. This could for example be achieved by an approximation based on several measurements of the states while changing the inputs. Additionally, all proportional coefficients are assumed to be $k_p \geq 0$, with $p \in [1,\, 5]$ to guarantee that both velocities $x_{1,1}$ and $x_{2,1}$ are ≥ 0 during the assembly. The resulting DynS-DSM follows analog to the general explained functional layer in Section 3 (figure 4) and will be omitted. Thus, the dynamic of the feedback cluster is assumed to be described by

$$\begin{bmatrix} x_{1,1} \\ x_{2,1} \end{bmatrix}^{(k+1)} = \begin{bmatrix} k_1\, i_{3,2}\, k_2\, x_{2,1} \\ k_3\, (k_4\, i_{2,3}\, x_{1,1} - k_5\, x_{1,1}^2) \end{bmatrix}^{(k)} . \tag{13}$$

	mover car	conveyer	grabber	car
mover car				
conveyer			X	
grabber	X	X		
car	X	X		

	$i_{3,2}$	$x_{2,1}$	$x_{1,1}$	$i_{2,3}$
$i_{3,2}$			0.6	
$x_{2,1}$			0.6	
$x_{1,1}$		1.0		
$i_{2,3}$		1.0		

Figure 10: Component based DSM with marked feedback cluster (at left): feedback (inner), input-enlarged feedback (exterior); and weighted DynS-DSM with marked clusters (at right): feedback (inner), input-enlarged feedback (exterior).

The first row of (13) reveals all possible equilibriums ($x_{j,1}^{(k+1)} \overset{!}{=} x_{j,1}^{(k)}$, like in (9)) of the phase space. These are all straight lines through zero with the gradient $k_1\, i_{3,2}\, k_2$. To analyse the stable equilibriums on this possible lines, $x_{1,1}$ in the second row of (13) is replaced by the first line of the system equation. This leads to

$$x_{2,1} = k_3\,[k_4\, i_{2,3}\,(k_1\, i_{3,2}\, k_2\, x_{2,1}) - k_5\,(k_1\, i_{3,2}\, k_2\, x_{2,1})^2]. \tag{14}$$

By neglecting the trivial stable equilibrium $x_{1,1} = x_{2,1} = 0$, equation (14) can be transformed to

$$\begin{aligned} 1 &= k_3\,[k_4\, i_{2,3}\, k_1\, i_{3,2}\, k_2 - k_5\,(k_1\, i_{3,2}\, k_2)^2\, x_{2,1}] \\ \Leftrightarrow \underbrace{i_{2,3}}_{\lambda} &= \underbrace{\frac{1}{k_3\, k_1\, i_{3,2}\, k_2\, k_4}}_{\lambda_c} + \underbrace{\frac{k_5\, k_1\, i_{3,2}\, k_2}{k_4}}_{\lambda_d}\, x_{2,1}, \end{aligned} \tag{15}$$

so that a local transcritical bifurcation is detected. The amount of to-affix screws $i_{2,3}$ is the bifurcation parameter and is therefore renamed λ. The critical value λ_c, where the local bifurcation takes place and the bifurcation gradient λ_d are merged in (15). Because the possible velocities $x_{1,1}$ and $x_{2,1}$ are bounded, the lowest of these bounds is a saturation of the process. Figure 11 clarifies the possible equilibrium straight line and the local bifurcation together with the resulting stable fixed points depending on λ. The saturation, which leads to the limit of screws λ_{lim} is also depicted. The only realisable region of the process is within the interval $\lambda \in\,]\lambda_c, \lambda_{lim}]$, because therein both states are $0 < x_{j,1} \leq$ Saturation. Hence, the desired interval of the to-affix screws can be achieved by varying the car mass $i_{3,2}$ and adjusting the proportional coefficients k_p of (13).

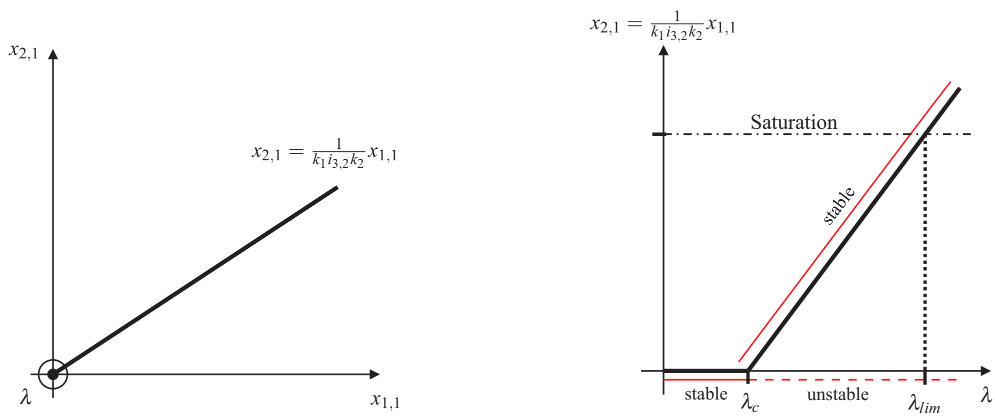

Figure 11: Possible equilibriums of the process (at left) and bifurcation diagram (at right).

By knowing this essential behavior of the process, the critical parameters are clear and an optimisation according to the possible ranges of the inputs can be done more effectively as only based on simulations.

5 Conclusion

A new approach for a structured modelling of complex processes has been introduced. The method is based on a recent decomposition principle called DSM (*Design Structure Matrix*). The DSM is thereby step by step extended to a discrete-time dynamical system based on component attributes. The new developed matrix is called DynS-DSM (*Dynamic System Design Structure Matrix*). It has been shown that all the advantages of the DSM are transferable to the DynS-DSM, so that already existing algorithms of the DSM can be used. This extension allows the DSM to tackle a wide range of problems and analysis tools. Different options of filling the DSM and thereby describing the dynamics in the DynS-DSM were introduced. Therefore, weighting coefficients to pinpoint different levels of required interaction knowledge have been used. The handling of nonlinear and linguistical relation-knowledge was shown. The approach was applied to two examples, where attracting regions of fixed points and a local bifurcation have been detected.

Accordingly, the dynamical behavior of a process can be reduced to the significant part and a dynamical model is obtained without the knowledge of a representative process time. The analytical analyses of the process model using well-known control theories and mathematical methods reduce the required amount of simulations for system understanding and system improvements.

6 Acknowledgment

The authors thank the Deutsche Forschungsgesellschaft (DFG) for funding this research as a part of the collaborative research "Managing cycles in innovation processes - Integrated development of product service systems based on technical products" (SFB 768).

7 References

[1] J. A. Bradley and A. A. Yassine. A multi-domain analysis framework for product development. In *Proc. of the International Design Engineering Technical Conferences*, New York, 2008.

[2] T. R. Browning. *Modeling and Analyzing Cost, Schedule, and Performance in Complex System Product Development*. PhD thesis, Massachusetts Institute of Technology, 1998.

[3] T. R. Browning. Applying the design structure matrix to system decomposition and integration problems: A review and new direction. *IEEE Transaction on Engineering Management*, 48(3):292–306, 2001.

[4] J. R. Burns and O. M. Ulgen. An integrated approach to the development of continuous simulations. *Socio-Econ. Plan. Sci.*, 12:313 – 327, 1978.

[5] J. R. Burns and O. M. Ulgen. A sector approach to the formulation of system dynamics. *Int. J. System Science*, 9(6):649 – 680, 1978.

[6] J. R. Burns and O. M. Ulgen. An algorithm for converting signed digraphs to forrester schematics. *IEEE Transactions on Systems, Man and Cybernetics*, 9(3):115 – 124, 1979.

[7] G. Coyle. Qualitative and quantitative modelling in system dynamics: some research questions. *System Dynamics Review*, 16(3):225–244, 2000.

[8] J. W. Forrester. *Industrial dynamics*. M.I.T. Press, Cambridge, 8th print, 1973.

[9] J. W. Forrester. *Principles of systems*. Wright-Allen, Cambridge, 8th print, 1976.

[10] M. Kreimeyer, M. Eichinger, and U. Lindemann. Assessment of process networks using graph and network theory based key figures. In *4th Annual Future Business Technology Conference (Fubutec)*, pages 24 – 26, Delft, 2007.

[11] U. Lindemann, M. Maurer, and M. Kreimeyer. Intelligent strategies for structuring products. In *Engineering Design - Theory and practice*, Cambridge (UK), 2005.

[12] M. Maurer, N. O. Boesch, G. Sheng, and B. Tzonev. A tool for modelling flexible product structures - mofleps. In *Proc. of the International Conferences on Engineering Design (ICED)*, Melbourne, 2005.

[13] T. U. Pimmler and S. D. Eppinger. Integration analysis of product decompositions. In *Proc. of the ASME 6th International Conference on Design Theory and Methodology*, Minneapolis, USA, 1994.

[14] P. S. Raghuvanshi and S. Kumar. On the structuring of systems with fuzzy relations. *IEEE Transactions on Systems, Man and Cybernetics*, 29(4):547 – 553, 1999.

[15] G. P. Richardson. Problems in causal loop diagrams revisited. *System Dynamics Review*, 13(3):247–252, 1997.

[16] K. E. Sahin. System dynamics models: Some obscurities. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(2):91–93, 1979.

[17] D. Steward. The design structure system: A method for managing the design of complex systems. *IEEE Transaction on Engineering Management*, 48(4):428–442, 1981.

[18] M. Sugeno. *Industrial applications of fuzzy control*. Elsevier Science, Amsterdam, 1985.

[19] F. Vester and A. v. Helser. *Sensitivitätsmodell*. Planungsgemeinschaft Untermain, Frankfurt am Main, 1980.

[20] S. Wiggins. *Introduction to applied nonlinear dynamical systems and chaos*. Springer, New York, 2003.

[21] J.-P. Ylèn and V. Hölttä. System dynamics - a tool for designing and analysing complex processes. In *Proc. of the 5th MATHMOD*, Vienna, 2006.

[22] L. A. Zadeh. Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Transactions on Systems, Man and Cybernetics*, 1(1):28 – 44, 1973.