Structural Design of Systems with Safe Behavior under Single and Multiple Faults

Blanke, Mogens; Staroswiecki, Marcel

Published in:
Fault Detection, Supervision and Safety of Technical Processes

Link to article, DOI:
10.3182/20060829-4-CN-2909.00078

Publication date:
2006

Document Version
Publisher's PDF, also known as Version of record

STRUCTURAL DESIGN OF SYSTEMS WITH SAFE BEHAVIOR UNDER SINGLE AND MULTIPLE FAULTS

Mogens Blanke * **and Marcel Staroswiecki** **

* *Automation at Ørsted•DTU, Build. 326, Technical University of Denmark, Kgs. Lyngby, Denmark and CESOS at NTNU, Trondheim, Norway e-mail: mb@oersted.dtu.dk*
** *SATIE UMR 8029, Ecole Normale Superieure de Cachan, 61 avenue du President Wilson, 94235 Cachan cedex, France, e-mail: marcel.staroswiecki@univ-lille1.fr*

Abstract: Handling of multiple simultaneous faults is a complex issue in fault-tolerant control. The design task is particularly made difficult by the numerous different cases that need be analyzed. Aiming at safe fault-handling, this paper shows how structural analysis can be applied to find the analytical redundancy relations for all relevant combinations of faults, and can cope with the complexity and size of a real system. Being essential for fault-tolerant control schemes that shall handle particular cases of faults/failures, fault isolation is addressed. The paper introduces an extension to structural analysis to disclose which faults could be isolated from a structural point of view using active fault isolation. An example and results from a marine application illustrate the concepts.

Keywords: Fault-tolerant control, Structural analysis, Fault diagnosis.

INTRODUCTION

Fault-tolerant control uses control or sensor reconfiguration to accommodate failures in instruments, plant components or actuators. Aiming at utilizing existing redundancy in instrumentation and control devices as far as possible, fault-tolerant control can be applied to minimize the hazards associated with malfunction, even when several sensors or actuators fail, but several modifications need be made to the usual single fault FTC schemes in order to achieve the necessary level of safety.

Structural analysis is a theoretic method that aims at offering such possibility. Structural concepts were studied early in the applied mathematics community (Dulmage and Mendelsohn, 1959) various theoretical algorithms were developed in (Dulmage and Mendelsohn, 1963) and (Hopcroft and Karp, 1973). Structural analysis has been used intensively in Chemical Engineering for solving large sets of equations and issues on solvability have been pursued in a number of publications, see (Unger *et al.*, 1995) and (Leitold and Hangos, 2001) and the references herein. The structural approach and the features it offers for analyzing monitoring and diagnosis problems was first introduced in (Staroswiecki and Declerck, 1989) and further developed in (Staroswiecki *et al.*, 1993). Extensions to analysis of reconfigurability and fault-tolerance emerged in (Staroswiecki *et al.*, 1999) and (Staroswiecki and

Gehin, 2000). The structural analysis approach was brought into a digested form in (Blanke *et al.*, 2006). Structural analysis has hence evolved during several decades. However, the salient features of the theory and the possibilities it offers have only become apparent to a larger community in the field of automation and automatic control over the last few years (Åström *et al.*, 1986), (Izadi-Zamanabadi and Staroswiecki, 2000) with applications reported in e.g. (Izadi-Zamanabadi *et al.*, 2003) and (Lorentzen *et al.*, 2003). Reasons for the slow penetration into applications origin mainly in the lack of widely available tools to support the structural analysis method for automated industrial systems.

This paper considers the safety of fault-tolerant control schemes when multiple faults may be present. It is shown how structural analysis can be applied to analyze cases of multiple faults and to syntetisize residual generators. Fault isolation, which is instrumental for correct fault handling, is addressed and a new result shows how active isolation could be achieved from a structural point of view.

The paper first reviews the concept of behaviors and shows how the behavior of a system is equally well applied on the services offered by hardware and software components. It then interprets the impact on safety of a system that is supposed to work under conditions of multiple faults. Isolability conditions are highlighted, and a new concept *active structural isolability* is introduced. An example illustrates the aspects of this extension to the structural analysis approach. An application on a marine system briefly illustrates analysis for operation under multiple faults.

1. RECONFIGURABILITY AND SAFETY

A system consists of a set of components which each offer a service and performs this service through defined normal behaviours. A component can offer different versions of services and command to the component can define which version of a service is made available. Within a component, fault-tolerant techniques can use fault-diagnosis and fault-handling to switch between services or offer a service in a version with degraded performance if local malfunction should make this necessary.

1.1 Subsystem services

A system breakdown in Fig.1 shows three different architectures, i.e. arrangement of the system components and their interaction. Component k has input u_k , output y_k, parameters θ_k

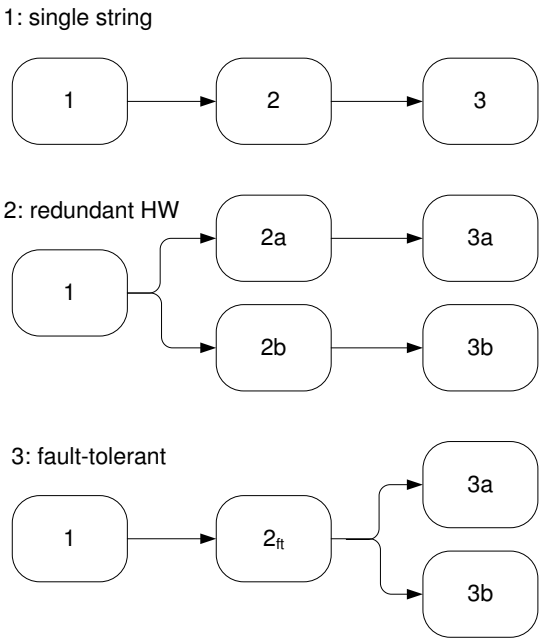

Figure 1. Three architectures, single line with no redundancy (1), hardware redundancy (2) and combined fault-tolerance and redundancy (3)

and a behavior $c_k(y_k, u_k, \theta_k) = 0$. The behavior may be constructed from a set of constraints $\{c_{k1}, c_{k2}, ..., c_{kn}\}$ associated with the subsystem and the exterior behavior of the component is the union of internal behaviors $c_k = c_{k1} \cup c_{k2} \cup .... \cup c_{kn}$ or for brevity, $c_k = \{c_{k1}, c_{k1}, ..., c_{kn}\}$. Following the generic component definition in (Blanke *et al.*, 2006), the service $S^{(k)}$ offered by component k is to deliver *produced variables* (output), based on *consumed variables* (input) and available *resources*, according to the specified behavior $S^{(k)}_{(v)}$ where $v \in \{1, 2, 3, ..\}$ is the version of the service. Clearly, the exterior behavior is associated with the service offered by the component, we denote this behavior by $c^{(v)}_k$.

In this context we particularly wish to consider versions of the same service that follow from the condition of the component, from normal over degraded to none. If a component has an internal failure, fault-tolerant techniques may still provide a version of the service with degraded performance $\left(S^{(k)}_{(d)}\right)$ or the service may not be available at all $\left(S^{(k)}_{(o)}\right)$. Hence, we consider the set of versions $v \in \{n, d1, d2, .., o\}$ where n : $normal; d1 : degraded1; d2 : degraded2; o : none$.

1.2 Service at system level

The service obtained by the system as an entirety is a function of the component architec-

ture $\mathcal{A}$ and the versions for the present condition k_i of components. With m components in a system, each component in one out of p conditions, $k_i \in \mathbb{N}^{p_i}$, we have a versions vector $\mathbf{v} = [v_1(k_1), v_2(k_2), ..., v_m(k_m)]$, and the set of available behaviors $C_v = \left\{c_1^{v(k_1)}, c_2^{v(k_2)}, ..., c_m^{v(k_m)}\right\}$.

Definition 1. The overall service available from a system is $S^{(s)}(c_i^{v(k_i)}) = \mathcal{A}(C_v|\mathbf{v}(\mathbf{k}))$, $i = 1, ..., m$.

With a single string architecture from Fig. 1, we obtain

$$S^{(s)} = S^{(1)} \cap S^{(2)} \cap S^{(3)} \tag{1}$$

With redundancy in the system, the hardware configuration with two parallel, totally redundant lines with only one component in common,

$$S^{(s)} = \tag{2}$$
$$S^{(1)} \cap \left(\left(S^{(2a)} \cap S^{(3a)}\right) \cup \left(S^{(2b)} \cap S^{(3b)}\right)\right)$$

This solution is expensive as it requires two completely redundant subsystems. A cost effective solution would be to have some components intrinsically safe $S^{(1)}$, have others equipped with fault-tolerant properties so their service $S^{(2)}_{v(2)}$ will be available but in degraded version when local faults occur, and just have hardware redundancy for few essential components $(3a, 3b)$. The fault tolerant architecture shown in part C of Fig. 1 is based on this idea. The service at system level is

$$S^{(s)} = S^{(1)} \cap S^{(2)}_{v(2)} \cap \left(S^{(3a)} \cup S^{(3b)}\right) \tag{3}$$

The paradigm in this architecture is that component failures should be detectable and control be switched to obtain a fault-tolerant service or reconfigure the system bypassing faulty components. This should be achieved by controlling the signal flow in the software of the system.

1.3 Availability and safety

The plant at the system level is *available* as long as the predefined normal service is offered in some version, normal or degraded. A fault-tolerant version of the service is obtained when one or more of the component services are offered in a fault-tolerant version. A fail-operational version of the service is obtained when hardware reconfiguration has been made to bypass a failure in the redundant component.

When multiple local failures are present, the service at system level is

$$S^{(s)} = \mathcal{A}\left(C_v|\mathbf{v}\right)$$

Definition 2. Available. The system is available when $S^{(s)} \subseteq \mathcal{S}$ where $\mathcal{S} = \left\{S_1^{(s)}, S_2^{(s)}, ..., S_n^{(s)}\right\}$ is the set of admissible services that meet specified overall objectives for behaviour $\mathcal{O}$ of the system:$\forall S_i^{(s)}(c_i^{v(k_i)}) : C_v \subseteq \mathcal{O}$.

Definition 3. Fault. A fault is a deviation from normal behavior, $\exists i : c_i \neq 0$.

Definition 4. Critical fault. A fault in c_i is critical $c_i \in C_{crit}$ if it will cause the system's behavior to be outside the set of admissible behaviors, $c_i \in C_{crit} \;\; iff \;\; c_i \neq 0 \Rightarrow C_v \nsubseteq \mathcal{O}$.

Definition 5. Safe version of service. A version is safe if all critical faults are detectable, $\forall c_i \in C_{crit}$: $c_i \in C_{\det ectable}$

Assumption 6. It is a natural assumption that shut-down of the system is intrinsically safe and that the system can be shut down to the safe mode from any condition where $S^{(s)} \subseteq \mathcal{S}$.

Definition 7. Reconfiguration. A system is reconfigurable if $c_i \neq 0 \Rightarrow \exists j \neq i, \mathbf{v}(j) \neq \mathbf{v}(i) : C_{v(j)} \subseteq \mathcal{O}$

The task of fault-tolerant control is to find an appropriate $\mathbf{v}(j)$ when the fault c_i is detected and isolated and bring the system from version $v(i)$ to $v(j)$.

Having defined the system properties in terms of behaviors, it is natural to employ structural analysis where behaviors are defined in terms of constraints between variables and graph theory methods offer rapid and rigorous analysis.

2. STRUCTURE GRAPH

A structural model of a system can be represented as a bipartite graph that connects constraints and variables. The structure graph (Staroswiecki and Declerck, 1989) of a system (C, Z) is a bipartite graph $G = (C, Z, E)$ with two set of vertices whose set of edges $E \subseteq C \times Z$ is defined by $(c_i, z_i) \in E$ iff the variable z_i appears in constraint c_i.

The variables in Z are divided into known K and unknown variables X. Similarly, the constraints C are divided into constraints C_K that only apply to the known variables and C_X that involve at least one unknown variable. An incidence matrix S describes the structure graph where each row in the matrix represents a constraint and each column a variable. $S(i, j) = 1$ means that variable x_j appears in constraint c_i, $S(i, j) = x$ denotes a directed connection.

2.1 Constraints

Constraints represent the functional relations in the system, i.e. originating in a physical model using first principles. The constraints needed for structural analysis are far more simple. Instead of using the explicit system equations, structural analysis need to know whether a certain constraint makes use of a particular variable. Parameters that are known from the physics of the plant or from properties of the automation system, e.g. a control gain, are treated as part of the constraint in which the particular parameter is used. A constraint can be directed. This implies that a variable on the left hand side of the constraint can not be calculated from the right hand side of the constraint.

2.2 Variables

There are three different kinds of variables: *Input variables* are known, externally defined; *Measured variables* are entities measured in the system; *Unknown variables* are internal physical variables. Input and measured variables both belong to the set K but are separated for calculation of controllability.

2.3 Matching and results

The central idea in the structure graph approach is to match all unknown variables using available constraints and known variables, if possible. If successful, the matching will identify overdetermined subgraphs that can be used as analytical redundancy relations in the system.

Results of the structural analysis are

- List of parity relations that exist
- Auto-generated suggestion of residual generators
- List of detectable faults
- List of isolable faults

When a matching has been found, backtracking to known variables will give a suggestion for parity relations that could be used as residual generators. A system with m constraints and n parity relations will give a relation showing which residuals depend on which constraints.

One view on these relations is the boolean mapping,

$$\mathcal{F} : r \leftarrow M \otimes (c_i \neq 0) \tag{4}$$

from which structural detectability and isolability can be found.

Lemma 8. A fault is structurally detectable iff it has a nonzero boolean signature in the residual, $c_i \in C_{\det ectable}$ iff $\exists j : c_i \neq 0 \Rightarrow r_j \neq 0$

Lemma 9. A fault is structurally isolable iff it has a unique signature in the residual vector, i.e. column m_i of M is independent of all other columns in M, $c_i \in C_{isolable}$ iff $\forall j \neq i : m_i \neq m_j$

Example 10. With $m = 6$ and $n = 4$, the result could have the form

$$\begin{matrix} & & & & & & c_1 & c_2 & c_3 & c_4 & c_5 & c_6 \end{matrix}$$
$$\begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{bmatrix} \leftarrow \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \tag{5}$$

Since columns $(2, 3, 4, 5)$ are independent, but $(1, 6)$ are dependent, $\{c_2, c_3, c_4, c_5\}$ are isolable and $\{c_1, c_5\}$ are detectable. The pair (c_1, c_5) is isolable.

3. ACTIVE ISOLATION

In some cases faults are group-wise isolable, i.e within the group individual faults are detectable but not isolable. This implies that with the given architecture of the system, these faults are group-wise not isolable. This does not necessarily imply that isolation can not be achieved in other ways. Indeed, although the same set of residuals will be "fired" when either one or the other of non-structurally isolable constraints is faulty, the time response of the residuals may be different under the different fault cases. Exciting the system with an input signal perturbation may therefore make it possible to discriminate different responses of the same residual set when different constraints within the group are faulty. The proof of the following result is obvious :

Proposition 11. Active isolation is possible if and only if both a structural condition and a quantitative condition are true.
Structural condition : the known variables in the set of residuals associated with a group of non-structurally isolable constraints include at least one control input.
Quantitative condition : the transfer from control inputs to residuals is affected differently by faults on different constraints.

Lemma 12. Input to output reachability. Let $p^{(i,j)} = \{c_f, c_g, ..., c_h\}$ be a path through the structure graph from input u_j output y_j and $\prod^{(i,j)}$ the union of valid paths from u_j output y_j. Let $C^{(i,j)}_{reach} = \left\{ c_g \mid c_g \in \prod^{(i,j)} \right\}$. A constraint c_h is input reachable from input u_j if a path exists from

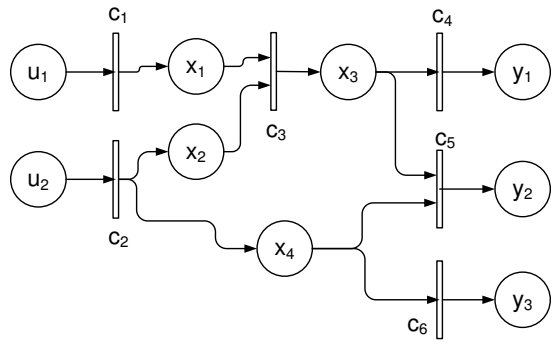

Figure 2. Structure graph for the example

u_j to any output y_k and the path includes the constraint, $c_h \in C_{reach}^{(i,k)}$

Lemma 13. Active isolability. Two constraints c_g and c_h are actively isolable if $\exists i,j,k,l : c_g \in C_p^{(i,j)}, c_h \in C_p^{(k,l)}$ and $\{c_g.c_h\} \notin C_{reach}^{(i,j)} \cap C_{reach}^{(k,l)}$

Algorithmic aspects. A path through a graph can be determined from an adjacency matrix,

$$\mathbf{A} : [C, K_i, K_m] \curvearrowright [C, K_i, K_m]$$

that shown which nodes in a graph are connected. As the graph is bipartite, the adjacency matrix is easily obtained from the incidence matrix (**?**) as

$$\mathbf{A} = \begin{bmatrix} \mathbf{O} & \mathbf{S} \\ \mathbf{S}^T & \mathbf{O} \end{bmatrix}$$

The adjacency matrix shows the result of a walk of length 1. A walk of length n will be described by A^n. Reachability of element i from element j in the graph can be determined by investigating the element (i,j) in the sequence of matrices

$$\mathbf{A}^1, \mathbf{A}^2, \mathbf{A}^3, ..., \mathbf{A}^{2cn}$$

where cn is the number of elements in $\{C, K_i, K_m\}$. With the i^{th} column of $\mathbf{A}$ being an input, and the j^{th} row a measurement, a path of length m exists from i to j iff $\mathbf{A}_{ij}^m \neq 0$. The nodes passed on the walk are determined by tracing the nonzero elements of $\mathbf{A}^m, \mathbf{A}^{m-1}, .., \mathbf{A}^1$. While this algebraic method is intuitive and is related to the structure graph $\mathbf{S}$, it is computationally inefficient, and algorithmic methods exist that can find all paths from a given input in a graph.

4. EXAMPLE

Let a system be given by the structure graph shown in Figure 2. Inputs are $K_i = \{u_1, u_2\}$, outputs are $K_m = \{y_1, y_2, y_3\}$, unknown variables are $X = \{x_1, x_2, x_3, x_4\}$. The associated incidence matrix is shown in Table 1

$\nearrow$	u_1	u_2	y_1	y_2	y_3	x_1	x_2	x_3	x_4
c_1	1	0	0	0	0	1	0	0	0
c_2	0	1	0	0	0	0	1	0	1
c_3	0	0	0	0	0	1	1	1	0
c_4	0	0	1	0	0	0	0	1	0
c_5	0	0	0	1	0	0	0	1	0
c_6	0	0	0	0	1	0	0	0	1

Table 1. Incidence matrix for example

$\nearrow$	c_1	c_2	c_3	c_4	c_5	c_6
r_1	1	1	1	1	1	0
r_2	0	0	0	0	1	1

Table 2. Dependency table for the example

A complete matching on the unknown variables can be achieved using the ranking algorithm (**?**), leaving c_6 and c_3 as unmatched constraints. The path found by the matching is:

$$\begin{aligned} &c_1(u_1) \rightarrow x_1; c_4(y_1) \rightarrow x_3; \\ &c_5(x_3) \rightarrow x_4; c_2(u_2, x_4) \rightarrow x_2 \\ &\quad \Rightarrow c_3(x_1, x_2, x_3) = 0 \\ &\quad \Leftrightarrow c_3(c_1(u_1), c_2(u_2, x_4), c_4(y_1)) = 0 \\ &\quad \Leftrightarrow c_3(c_1(u_1), c_2(u_2, c_5(c_4(y_1))), c_4(y_1)) = 0 \end{aligned}$$

and

$$\begin{aligned} c_6(y_3, x_4) = 0 &\Leftrightarrow c_6(y_3, c_5(x_3)) = 0 \\ &\Leftrightarrow c_6(y_3, c_5(c_4(y_1))) = 0 \end{aligned}$$

The analytical redundancy relations associated with c_3 and c_6 constitute two parity relations for the system considered in the example and two residual generators are

$$\begin{aligned} r_1 &= c_3(c_1(u_1), c_2(u_2, c_5(c_4(y_1))), c_4(y_1)) \\ r_2 &= c_6(y_3, c_5(c_4(y_1))) \end{aligned}$$

. The dependency matrix between residuals and constraints are shown in Table 2 shows detectability and isolability as achievable from the two residuals. Linearly independent columns show that the violations of constraints $\{c_5, c_6\}$ can be isolated. The set $\{c_1, c_2, c_3, c_4\}$ is block-wise isolable but violation of any of the individual constraints will only be detectable.

4.1 Active structural isolation.

In a fault-tolerant control setting, inputs u_1 and u_2 can be individually perturbed by the control system. The expected behavior of an output to perturbation at an input follows from Lemma 12 . The set of paths through constraints from $u_1$to the outputs are represented in the reachability table 3

The reachability from u_2 is shown in Table 4.

$u_1 \downarrow$	c_1	c_2	c_3	c_4	c_5	c_6
y_1	1	0	1	1	0	0
y_2	1	0	1	0	1	0
y_3	0	0	0	0	0	0

Table 3. Output reachability from u_1

$u_2 \downarrow$	c_1	c_2	c_3	c_4	c_5	c_6
y_1	0	1	1	1	0	0
y_2	0	1	0	0	1	0
y_3	0	1	0	0	0	1

Table 4. Output reachability from u_2

		a		b			m					
f_1	f_2	1	2	1	2	3	1	2	3	4	5	6
a_3	m_5	i	i	i	i	d	d	i	i	i	–	d
a_3	m_6	i	i	0	0	d	0	i	0	i	0	–
a_3	m_7	i	i	i	i	d	d	i	i	i	0	d

Table 5. Analyzing two simultaneous faults in marine application

Following Lemma 13 , it is easily seen that $\{c_1, c_2, c_3\}$ are structurally isolable when active isolation is employed, while c_4 remains detectable.

4.2 Scenarios with multiple faults

Scenarios of multiple faults are dealt with, in the structural analysis context, by removing one or more constraints that represent the faulty parts of the system. Should c_6 be subject to a local failure, the remaining system $\mathbf{S}_f = \mathbf{S} \setminus \{c_6\}$ need be re-analyzed. The results can show which residual generators exist for the faulty system, and which further faults could be isolated or detected.

An application to a marine control system was treated in (Blanke, 2005) where analysis of multiple faults was a part of a fault-tolerant design. The tool SaTool (Lorentzen and Blanke, 2004)was used to generate the analytical relation.

5. CONCLUSIONS

Aiming at safe fault-handling, this paper showed how structural analysis was applied to find the analytical redundancy relations for all relevant combinations of faults. Being essential for fault-tolerant control schemes that shall handle particular cases of faults/failures, fault isolation was addressed. The paper suggested an extended analysis of the results from a structural analysis to disclose which faults could be isolated from a structural point of view using active isolation.

REFERENCES

Åström, K. J., J. J. Anton and K. E. Årz n (1986). *Expert Control.*. Vol. 22.

Blanke, M. (2005). Fault-tolerant sensor fusion with an application to ship navigation.. In: *Proc. IEEE MED.* pp. 1385–1390.

Blanke, M., M. Kinnaert, J. Lunze and M. Staroswiecki (2006). *Diagnosis and Fault-tolerant Control.* 2nd ed.. Springer.

Dulmage, A. L. and N. S. Mendelsohn (1959). A structure theory of bi-partite graphs.. *Trans. Royal Society of Canada. Sec. 3.* **53**, 1–13.

Dulmage, A. L. and N. S. Mendelsohn (1963). Two algorithms for bipartite graphs.. *Journal of the Society for Industrial and Applied Mathematics.* (1), 183–194.

Hopcroft, J. E. and R. M. Karp (1973). An n/sup 5/2 algorithm for maximal matchings in bipartite graphs.. *SIAM Journal on Computing* pp. 225–231.

Izadi-Zamanabadi, R. and M. Staroswiecki (2000). A structural analysis method formulation for fault-tolerant control system design. In: *39th IEEE CDC.* pp. 4901–4902.

Izadi-Zamanabadi, R., M. Blanke and S. Katebi (2003). Cheap diagnosis using structural modeling and fuzzy-logic based detection. *Control Engineering Practice* **11**(4), 415–422.

Leitold, A. and K. M. Hangos (2001). Structural solvability analysis of dynamic process models. *Computers and Chemical Engineering* **25**, 1633–1646.

Lorentzen, T. and M. Blanke (2004). Industrial use of structural analysis - a rapid prototyping tool in the public domain. In: *Proc. Workshop on Advanced Control and Diagnosis.* pp. 166–171.

Lorentzen, T., M. Blanke and H. H. Niemann (2003). Structural analysis - a case study of the rømer satellite.. In: *Proc. IFAC Symp. Safeprocess'2003.*. Elsevier - IFAC.

Staroswiecki, M. and A. L. Gehin (2000). Control, fault tolerant control and supervision problems. In: *IFAC Safeprocess' 2000.* Budapest, Hungary.

Staroswiecki, M. and P. Declerck (1989). Analytical redundancy in nonlinear interconnected systems by means of structural analysis. In: *Proc. IFAC AIPAC'89 Symposium.*. Vol. 2. Elsevier - IFAC. pp. 23–27.

Staroswiecki, M., J. P. Cassar and V. Cocquempot (1993). Generation of optimal structured residuals in the parity space. In: *12th IFAC World Congress.* Vol. 8. pp. 299–305.

Staroswiecki, M., S. Attouche and M. L. Assas (1999). A graphic approach for reconfigurability analysis. In: *Proc. DX'99.*

Unger, J., A. Kröner and W. Marquardt (1995). Structural analysis of differential-algebraic equation systems - theory and applications.. *Computers and Chemical Engineering* (8), 867–882.