MODEL-BASED SAFETY MONITORING OF PRODUCT FLOW PATHS*

G. Quirós, R. Jorewitz, U. Epple
RWTH Aachen University, Germany

Corresponding author: G. Quirós, RWTH Aachen University, Chair of Process Control Engineering
52064 Aachen, Turmstraße 46, Germany, g.quiros@plt.rwth-aachen.de

Abstract. *Product flow paths* are the routes that products take while flowing through a plant. The development of systems that ensure and monitor the correct and safe transport of material is usually plant-specific and based on informal knowledge, and is time-consuming and error-prone. In this work we seek a synthesis solution for the task of monitoring the safety of product flow paths in processing plants. A formal model of the plant is used, which defines a simplified plant representation that considers the possibility of flow through its elements. Based on this model, we present a formalisation of the safety of a product flow path at a given plant state. This formulation may be used as a guideline for automating the construction of systems which perform safety monitoring of product flow paths. An outline for the design of such a system following a decentralisation scheme is also presented.

1 Introduction

1.1 Safety of product flow paths

A basic and essential operation performed by processing plants is the movement of material, i.e. products, between plant elements. This movement or flow is physically constrained by the structure of the plant itself and is caused either by gravity or by the operation of active devices such as pumps. We denote the routes that products take while flowing through a plant as *product flow paths*. Understanding this concept is of great importance when developing process automation systems. For instance, the operation of plants with flexible structures consisting of multiple and alternate product flow paths requires adequate working states of plant elements like valves in order to restrict the flow of material to a desired path, as well as to ensure the safety of the flow operation by avoiding undesired and potentially hazardous situations such as leaks (when the product flow diverges from the intended path and reaches unexpected plant locations) and unintended mixtures (when another product enters an active flow path unexpectedly). For example, Figure 1 shows a diagram of a simple filling station consisting of four tanks, two pumps, and two product input nozzles. Each of the four tanks may be filled by from any of the two product inputs by the corresponding pump. The product flow path shown in blue corresponds to one such filling operation. For this product flow path, the diverging path shown in red represents a potential leak to another tank. Likewise, the joining path shown in green represents a potential unintended mixture. These situations are avoided by closing valves V4 and V7 respectively, whenever this product flow path is in operation.

Process control systems usually fulfil the important task of ensuring and monitoring the correct and safe transport of material in processing plants, as they are designed and implemented with these requirements in mind. However, the development of these plant-specific solutions is based on informal knowledge and is time-consuming and error-prone, especially in the case of large and complex plants which follow flexible designs, e.g. multi-purpose plants in the pharmaceutical industry. Furthermore, this engineering phase must be repeated as soon as the plant structure itself changes. This serves as a motivation for studying automated approaches that may partly or completely replace this engineering work, ensuring its correctness at the same time. The automation of automation [10, 11] offers a technique for reducing the engineering effort of the development of process control systems, as well as improving their quality, by automating the construction of such systems (in part or in whole) through the application of engineering rules. Analogously, in this work we seek a synthesis solution for the task of monitoring the safety of product flow paths.

1.2 A formal and abstract plant model

In order to apply an automatic synthesis approach to systems which perform tasks regarding product flow paths, an adequate representation of the plant is required which describes its structure and the behaviour of its elements unambiguously. With the goals of enabling and supporting automation-of-automation approaches for such systems in general, we have developed a formal model, based on the RIVA model presented in [3, 4], which satisfies these requirements by defining a simplified plant representation that considers the possibility of flow through its components. This model, which we denote as *flow allowance model*, is generic enough to represent practically any type of plant and plant device, as it only requires knowledge about the plant's structure and the general flow-related behaviour of its elements. With the help of this model, it becomes possible to formally define the structure, type, state and safety of product flow paths, and use these formalisms as the base of automated solutions. Additionally, in many cases this model may be automatically created from machine-readable plant representations such as CAEX

*This research has been partially funded by the DFG Research Training Group 1298 "Algorithmic synthesis of reactive and discrete-continuous systems" (AlgoSyn).

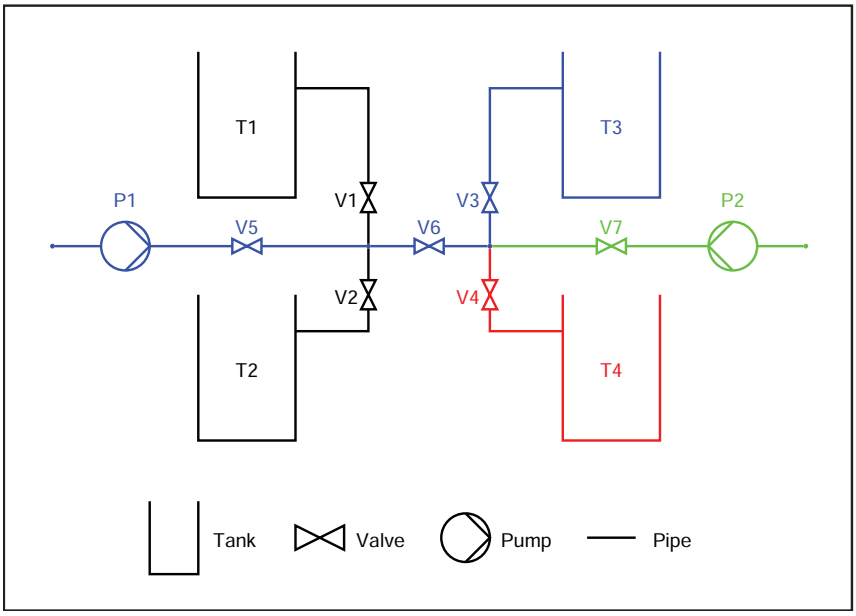

Figure 1: A tank filling station with flexible structure. Each of the four tanks may be filled by from any of the two product inputs by the corresponding pump. The product flow path shown in blue corresponds to one such filling operation. For this product flow path, the diverging path shown in red represents a potential leak to another tank. Likewise, the joining path shown in green represents a potential unintended mixture.

[1], thereby simplifying the application of this approach further. Figure 2 shows a graphical representation of the structural model which corresponds to the plant of Figure 1, where the tanks, pumps, valves, pipes and pipe joins of the plant are all represented as elements uniformly, having an adequate set of connectors which are connected with those of other elements in a one-to-one fashion.

1.3 Related work

The Route Control programming package for SIMATIC PCS 7 [12] enables the automatic routing of products in flexible plants, as well as the administration, control and monitoring of these routes. However, it requires an engineering phase where partial routes are to be defined and configured manually using a traditional approach. In turn, we seek to develop a simple and fully automatic technique for flow path safety monitoring based on our abstract model of plant structure and plant working state.

The Multilevel Flow Modelling presented in [5] is a methodology for modelling goals and functions of complex processing plants, with the intention of aiding in the development of diagnostics and control systems. It considers not only the flow of mass but also that of energy, and considers for each case source, sink, storage, balance, transport and barrier elements. However, an explicit treatment of flow paths as entities which may identified in a plant is not given.

1.4 Goals of this paper

In [9], a technique for the automatic discovery of product flow paths is presented, and [8] outlines a mechanism for the automatic assurance of product flow paths which is inspired by a similar approach used in railway locking [6]. In this paper, we present a formalisation of the safety of a product flow path at a given plant state based on our model of flow allowance. This formulation may then be used as a guideline for automating the construction of systems which perform safety monitoring of product flow paths. An outline for the design of such a system following the decentralisation scheme used in [9, 8] is also presented.

2 Abstract plant model

In this section we give a formal definition of our abstract plant model, which encompasses the structure and flow allowance of the plant, as well as the product flow paths of the plant.

Definition 1 (Plant structure) *A plant is a tuple* $(T, E, C, \tau, \varepsilon, \circ)$ *where*

- T *is a set of element types,*
- E *is a set of plant elements,*
- C *is a set of connectors,*

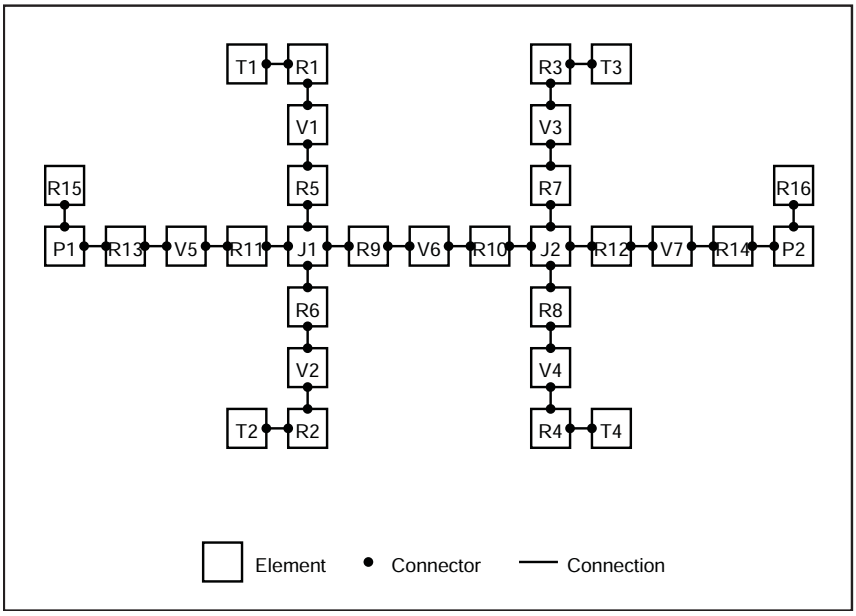

Figure 2: A graphical representation of the structural model of the filling station of Figure 1. The tanks (T#), pumps (P#), valves (V#), pipes (R#) and pipe joins (J#) of the plant are represented as elements uniformly, having an adequate set of connectors which are connected with those of other elements in a one-to-one fashion.

- $\tau : E \to T$ *is a function such that* $\tau(e)$ *is the type of element* e *for every* $e \in E$,
- $\varepsilon : C \to E$ *is a function such that* $\varepsilon(c)$ *is the element of connector* c *for every* $c \in C$,
- $\circ \subseteq C \times C$ *is the connection relation, which is*
  - *irreflexive:* $c_1 \circ c_2 \;\Rightarrow\; c_1 \neq c_2$
  - *symmetric:* $c_1 \circ c_2 \;\Leftrightarrow\; c_2 \circ c_1$
  - *functional:* $(c_1 \circ c_2 \wedge c_1 \circ c_3) \;\Rightarrow\; c_2 = c_3$

  for any $c_1, c_2, c_3 \in C$.

For every $t \in T$, *the set* E_t *of elements of type* t *is defined such that*

$$e \in E_t \;\Leftrightarrow\; \tau(e) = t.$$

For every $e \in E$, *the set* C_e *of connectors of element* e *is defined such that*

$$c \in C_e \;\Leftrightarrow\; \varepsilon(c) = e.$$

Similarly to the approach presented in [9, 8], a plant is formally represented by a tuple $(T, E, C, \tau, \varepsilon, \circ)$ with sets of element types T, elements E and product connectors C. The function τ maps every element to its type, and the function ε maps every connector to the element which owns it. Finally, the binary relation $\circ$ represents the interconnection of element connectors as is found in the physical plant. The plant is hereby modelled by a special kind of graph: the elements of the plant are represented by graph nodes (E), and rather than connecting the nodes directly, the edges of the graph ($\circ$) link so-called connectors (C), which are in turn embedded in the element nodes (ε) as shown in Figure 2. This corresponds on the one hand to typical plant-engineering representations like CAEX [1], and on the other hand allows for attributing the connectors with flow allowances in the following.

Having this formal representation of the structure of a plant, we wish to model the flow of products through this plant structure. As discussed in [9], we follow an approach for representing product flow which is *plant-oriented* (it considers characteristics of the plant itself such as connection structure and working state of the plant components, rather than the actual physical properties of the material) and *passive* (it considers the *possibility* of flow through the plant, rather than the *causes* of flow or the *actual* flow). We denote this possibility of flow through the plant as *flow allowance*, which represents a *necessary* condition for actual product flow. Therefore, the absence of flow allowance guarantees the absence of product flow in the plant, and we use this reasoning when defining the structure and safety of product flow paths.

The definition of flow allowance may be introduced by an analogy: an element is similar to a room with multiple and rather sophisticated doors, which correspond to its connectors. A door may be used for entering or exiting the room exclusively, or may be used simultaneously as an entrance and as an exit. Furthermore, some doors may be

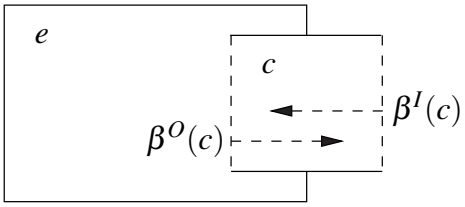

Figure 3: Flow allowance behaviour of a connector c of an element e: $\beta^I(c)$ refers to the flow which enters e through c, and $\beta^O(c)$ refers to the flow which leaves e through c.

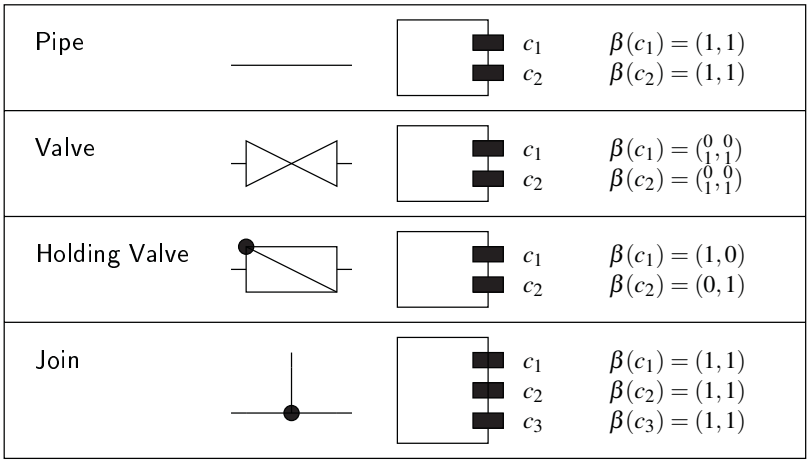

Figure 4: Modelling of various types of plant elements. A pipe is an element with two connectors which always allow flow in both directions. A valve is a 2-connector element whose connectors may allow or inhibit flow in both directions. A holding valve is a 2-connector element where one connector always allows incoming flow only while the other always allows outgoing flow only. Finally, a 3-way pipe join is a 3-connector element whose connectors always allow flow in both directions.

opened and closed, and this may be done for the entrance and exit aspects independently. In a similar way, product may flow into and out of an element through its connectors. The labelling of a connector as an entrance or an exit, as well as its ability to open and close, is described by the *flow allowance behaviour* of the connector, which is defined in the following.

Definition 2 (Flow allowance behaviour) *Given a plant $(T,E,C,\tau,\varepsilon,\circ)$, the flow allowance behaviour function of the plant is a mapping $\beta : C \to \{0,1,^0_1\} \times \{0,1,^0_1\}$. For a given $c \in C$ where $\beta(c) = (i,o)$, we denote i as the input flow allowance behaviour of c and o as the output flow allowance behaviour of c. The input flow allowance behaviour function $\beta^I : C \to \{0,1,^0_1\}$ is defined such that*

$$\beta^I(c) = i \;\Leftrightarrow\; \beta(c) = (i,o)$$

for any $o \in \{0,1,^0_1\}$. The output flow allowance behaviour function $\beta^O : C \to \{0,1,^0_1\}$ is defined such that

$$\beta^O(c) = o \;\Leftrightarrow\; \beta(c) = (i,o)$$

for any $i \in \{0,1,^0_1\}$.

The terms *input* and *output* in this definition respectively refer to the flow which *enters* and *leaves* the corresponding element $\varepsilon(c)$ through the connector c, as shown in Figure 3.

The flow allowance behaviour of a connector is expressed in *ternary logic*. The ternary value 0 corresponds to the Boolean value 0 and represents a constant inhibition of flow. In turn, the ternary value 1 corresponds to the Boolean value 1 and represents constant allowance of flow. Finally, the ternary value $^0_1$ corresponds to both Boolean values, and represents a switchable behaviour which may either inhibit or permit flow at a given plant state. This allows us to model the flow allowance behaviour of almost any type of plant element; Figure 4 shows some examples of this.

As the flow allowance behaviour of a connector describes the possible flow allowance configurations of a connector at any given time, the composition of the flow allowance behaviour of the connectors of an element e describe the possible flow allowance configurations of e, and this compositional approach may be further used to describe the possible flow allowance configurations of the entire plant. These configurations correspond to the *flow allowance states* of the plant, which is defined using the function λ which maps each ternary logic value to the set of corresponding Boolean values ($\lambda(0) = \{0\}$, $\lambda(1) = \{1\}$, $\lambda(^0_1) = \{0,1\}$).

Definition 3 (Flow allowance state) *Given a plant* $(T,E,C,\tau,\varepsilon,\circ)$ *and its flow allowance behaviour function* β*, the set of flow allowance states of the plant is the set of mappings* $S \subseteq [C \to \{0,1\} \times \{0,1\}]$ *such that for every* $\sigma \in S$ *and every* $c \in C$ *it holds that*

$$\sigma(c) = (i,o) \;\Rightarrow\; [i \in \lambda(\beta^I(c)) \wedge o \in \lambda(\beta^O(c))].$$

For a given $\sigma \in S$ *and* $c \in C$ *where* $\sigma(c) = (i,o)$*, we denote* i *as the input flow allowance state of* c *at* σ*, and* o *as the output flow allowance state of* c *at* σ*. For a given* $\sigma \in S$*, the input flow allowance state function* $\sigma^I : C \to \{0,1\}$ *is defined such that*

$$\sigma^I(c) = i \;\Leftrightarrow\; \sigma(c) = (i,o)$$

for any $o \in \{0,1\}$*. For a given* $\sigma \in S$*, the output flow allowance state function* $\sigma^O : C \to \{0,1\}$ *is defined such that*

$$\sigma^O(c) = o \;\Leftrightarrow\; \sigma(c) = (i,o)$$

for any $i \in \{0,1\}$*.*

The flow allowance state of a connector is expressed in Boolean logic, where 0 represents the inhibition of flow and 1 represents the allowance of flow. Also, the flow allowance states of the plant are determined by the flow allowance behaviour function β as expected. For behaviours with values of 0 or 1, the corresponding values of the flow allowance states are fixed and fully determined by the model, as in the case of static plant elements like pipes or tanks. However, when a flow allowance behaviour has the value $\frac{0}{1}$, the actual value of a state may be either 0 or 1. At a given time during the operation of the plant, the physical state of the corresponding plant element determines the actual value of the flow allowance state. As these values are commonly available to a process control system, e.g. from the acknowledgement signals of controllable valves, we may assume that flow allowance states are known when developing algorithms which are based on this model.

Based on this formulation of the input and output flow allowance at every connector in the plant, we may now describe the allowance of flow of material *among* neighbouring plant elements by means of a binary relation over the set E.

Definition 4 (Flow allowance relation) *Given a plant* $(T,E,C,\tau,\varepsilon,\circ)$ *and its flow allowance behaviour function* β*, the flow allowance relation* $\to \subseteq E \times E$ *is defined such that* $e_1 \to e_2$ *if and only if there exist connectors* $c_1 \in C_{e_1}$ *and* $c_2 \in C_{e_2}$ *such that* $c_1 \circ c_2$*,* $\beta^O(c_1) \neq 0$ *and* $\beta^I(c_2) \neq 0$*.*

Definition 5 (Flow allowance relation at a state) *Given a plant* $(T,E,C,\tau,\varepsilon,\circ)$ *and a flow allowance state* $\sigma \in S$*, the flow allowance relation* $\xrightarrow{\sigma} \subseteq E \times E$ *at* σ *is defined such that* $e_1 \xrightarrow{\sigma} e_2$ *if and only if there exist connectors* $c_1 \in C_{e_1}$ *and* $c_2 \in C_{e_2}$ *such that* $c_1 \circ c_2$*,* $\sigma^O(c_1) = 1$ *and* $\sigma^I(c_2) = 1$*.*

The meaning of these flow allowance relations is an intuitive one: $e_1 \to e_2$ whenever it may be possible for a product to flow directly from e_1 to e_2 according to the flow allowance behaviour of the intermediate connectors, which in turn occurs whenever there exists a flow allowance state $\sigma \in S$ that permits such a flow; consequently, $e_1 \xrightarrow{\sigma} e_2$ whenever the flow allowance state σ permits a direct flow from e_1 to e_2. In both cases, we say that there exists a *flow step* from e_1 to e_2.

As the flow allowance relations describe individual flow steps in a plant, a natural extension of this concept is to chain several flow steps together in order to obtain a *flow path*. Indeed, this is the basic idea behind our definition of a product flow path: a flow path is a finite sequence of neighbouring plant elements which may be used by a product to flow from an initial element to a final element, and where each pair of consecutive elements conforms a flow step. This is formalised in the following definition, where the notation X^+ denotes the set of all non-empty sequences of elements of set X.

Definition 6 (Flow paths) *Given a plant* $(T,E,C,\tau,\varepsilon,\circ)$ *and its flow allowance relation* $\to \subseteq E \times E$*, the set of flow paths* $P \subset E^+$ *of the plant is defined such that*

$$e_1 e_2 \ldots e_n \in P$$

where $n \geq 1$*, if and only if the following hold:*

- $e_i \to e_{i+1}$ *for every* $i \in [1, n-1]$*,*
- $i \neq j \;\Rightarrow\; e_i \neq e_j$ *for every* $i, j \in [1,n]$*.*

For a flow allowance state $\sigma \in S$ *of the plant, the set* $P_\sigma \subseteq P$ *of open flow paths at* σ *is defined such that* $e_1 \ldots e_n \in P_\sigma$ *if and only if* $e_i \xrightarrow{\sigma} e_{i+1}$ *for every* $i \in [1, n-1]$*.*

The function $\kappa : P \to 2^E$ *maps every flow path* $p = e_1 \ldots e_n$ *to the set* $\{e_1, \ldots, e_n\}$ *of elements it contains.*

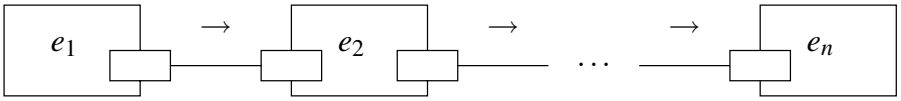

Figure 5: A flow path $e_1 e_2 \dots e_n$.

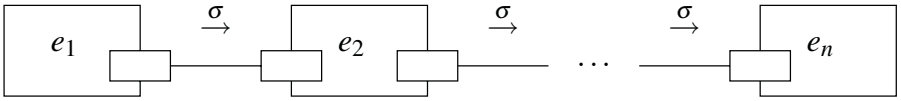

Figure 6: An open flow path $e_1 e_2 \dots e_n$ at a flow allowance state σ.

A flow path may be also seen as simple path in a *flow allowance graph* with a set of nodes E and a set of edges $\rightarrow$. Furthermore, an open flow path at a flow allowance state σ is a flow path in the plant which additionally follows the flow allowance relation $\xrightarrow{\sigma}$, which may be seen as a simple path in a *flow allowance graph at* σ with a set of nodes E and a set of edges $\xrightarrow{\sigma}$. Figure 5 depicts a flow path in graphical model representation, and Figure 6 shows a corresponding open flow path.

Apart from having to follow the flow allowance relation $\rightarrow$, flow paths must also be free from any repeated elements. We have included this restriction for several reasons. First, disallowing repeated elements in a flow path causes flow paths to be free of cycles, and therefore, guarantees that flow paths have a finite length (given that the plant is itself finite). This simplifies the representation of flow path data and the algorithms which work with flow paths. Second, allowing repeated elements would permit paths of the form $\dots e_1 e_2 \dots e_2 e_1 \dots$, which would model simultaneous bidirectional flow between the elements e_1 and e_2. As the intention of this work is to model physical product flow, we must rule out any form of simultaneous bidirectional flow from our model. A flow path where no element appears twice represents a form of "forward only" flow, which best describes the actual flow of products in processing plants. Finally, our intention is to define the concept of a product flow path in a way which is useful to describe and specify the spatial bounds of general product flow operations in a plant. We feel that the model of simple flow paths presented here which describes product flow between two plant locations captures the essence of these operations. More complex operations involving recirculation (cycles), forking and joining of flows may be also described using our model by using multiple flow paths. Therefore, we achieve a model which is accurate and comfortable to work with while at the same time expressive enough to handle both simple and complex plant designs.

3 Product flow path safety

The notion of safety in processing plants is a very broad and important part of the corresponding engineering field [7]. Based on our flow allowance model, we formulate a definition of safety of a product flow path at a given plant state with the intention of identifying general scenarios which correspond to undesired and potentially hazardous situations that may arise during the usage of a product flow path. Though this formulation is sufficient for most applications, refinements are possible and may be necessary in special cases.

We identify two subclasses of plant elements, namely *sources* and *sinks*. The former are elements which yield product that may flow to other points in the plant, such as tanks or input nozzles; the latter are elements which consume product which flows from other points in the plant, such as tanks or output nozzles. A similar classification which describes the general function of an element with respect to product flow may be found in [5].

Definition 7 (Sources and sinks) *Given a plant* $(T, E, C, \tau, \varepsilon, \circ)$*, the set* $E^{\uparrow} \subseteq E$ *is the set of sources of the plant, and the set* $E^{\downarrow} \subseteq E$ *is the set of sinks of the plant.*

Usually, product flow paths will begin at a source element and end at a sink element, although we do not require this when asserting their safety. The basic principle behind the definition of product flow path safety is the avoidance of leaks and unintended mixtures, in a similar way to how a train interlocking system avoids derailments (similar to leaks) and collisions (similar to mixtures) whenever a train travels through a given track segment [6]: derailments occur when a train (or a part of it) deviates from its intended course; collisions occur when an approaching train enters a track segment already in use by another train. With the help of this analogy, we may characterise these two situations for the case of processing plants as follows:

- *Leaks*. A leak occurs when the flow of a product diverges from the intended path p, and in our model, when there exists a diverging open flow path p' which begins at an element in p and which allows flow to a sink element.
- *Unintended mixtures*. An unintended mixture occurs when another material is able to flow into, and mix with, the product flowing through a flow path p, and this occurs when there exists a joining open flow path p' which begins at a source element and which allows flow to an element in p.

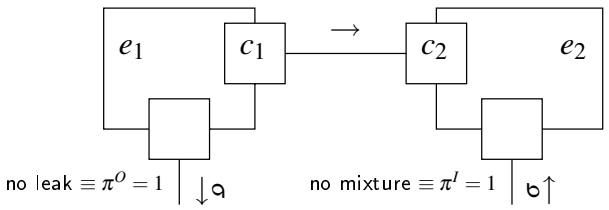

Figure 7: Determination of the safety of a flow step at a flow allowance state σ. A flow step from e_1 to e_2 is safe if there exists no diverging open flow path which begins at e_1 and which reaches a sink, and if there exists no joining open flow path which begins at a source and which reaches e_1. This is determined by the value of the function π^O at e_1 and π^I at e_2.

The conditions for the occurrence of these situations may be determined in a step-wise manner, as shown in Figure 7. In a flow path $p = \ldots e_1 e_2 \ldots$, the flow step represented by $e_1 \rightarrow e_2$ is safe if flow may occur from e_1 to e_2 exclusively. Additional outgoing flow from e_1 and additional incoming flow to e_2 are violations of this principle, as they represent leaks and mixtures respectively. Thus, we may determine the safety of a flow step by verifying the impossibility of these additional flows. The safety of a flow path may now be determined by the application of this rule to each flow step in the path, as shown in Figure 8. A given intermediate element e_i in a flow path p is both start and end of a flow step, and should be therefore free from both leaks and mixtures for p to be safe. On the other hand, the initial element of a p need only be free from leaks, and the final element of a p need only be free from mixtures for p to be safe. This follows from the determination of safety based on the point of view of a flow step as shown in Figure 7, and additionally has the nice property of allowing product to flow into and out of a flow path correctly, that is, via its end points, while considering this safe.

In order to formalise the absence of leaks and unintended mixtures at an element e with respect to a flow path p and a flow allowance state σ, we present the following definition of *flow protection functions*. As these and some other subsequent functions presented here denote Boolean values, we take the liberty of defining them in terms of first-order logic formulae.

Definition 8 (Flow protection) *Given a plant $(T,E,C,\tau,\varepsilon,\circ)$ and a flow allowance state $\sigma \in S$, the input flow protection function $\pi^I : E \times P \times S \rightarrow \{0,1\}$ is defined as*

$$\pi^I(e,p,\sigma) = \neg[\exists\, e_1 \ldots e_n e \in P_\sigma : (\{e_1,\ldots,e_n\} \cap \kappa(p) = \emptyset) \wedge e_1 \in E^{\uparrow}].$$

Given a plant $(T,E,C,\tau,\varepsilon,\circ)$ and a flow allowance state $\sigma \in S$, the output flow protection function $\pi^O : E \times P \times S \rightarrow \{0,1\}$ is defined as

$$\pi^O(e,p,\sigma) = \neg[\exists\, e e_1 \ldots e_n \in P_\sigma : (\{e_1,\ldots,e_n\} \cap \kappa(p) = \emptyset) \wedge e_n \in E^{\downarrow}].$$

Notice that the negations cause these functions to yield the value 0 when the potentially hazardous situations are present, and 1 when they are absent. Furthermore, for a diverging path to be considered a leak, and for a joining path to be considered a mixture, these paths must not have any element in common with the path p whose safety is being determined, excluding the element e. This causes paths which leave and rejoin p to be excluded from this condition, as well as paths which have a common section with p. This in turn limits the detection of leaks and mixtures to strictly diverging and strictly joining paths.

We may now present the *flow step safety function* which determines if the flow step represented by $e_1 \rightarrow e_2$ from flow path p is free of leaking deviations from e_1 to additional sinks (with the help of the output flow protection function π^O), as well as free of incoming flow from additional sources to e_2 (with the help of the input flow protection function π^I) at a state σ, following the idea from Figure 7.

Definition 9 (Flow step safety) *Given a plant $(T,E,C,\tau,\varepsilon,\circ)$ and a flow allowance state $\sigma \in S$, the flow step safety function $\delta : E \times E \times P \times S \rightarrow \{0,1\}$ is defined as*

$$\delta(e_1,e_2,p,\sigma) = [\pi^O(e_1,p,\sigma) \;\wedge\; \pi^I(e_2,p,\sigma)].$$

Finally, we may define the safety of a product flow path p at a given flow allowance state σ following the technique shown in Figure 8.

Definition 10 (Flow path safety) *Given a plant $(T,E,C,\tau,\varepsilon,\circ)$ and a flow allowance state $\sigma \in S$, the flow path safety function $\alpha : P \times S \rightarrow \{0,1\}$ is defined as*

$$\alpha(p,\sigma) = \alpha'(p,p,\sigma)$$

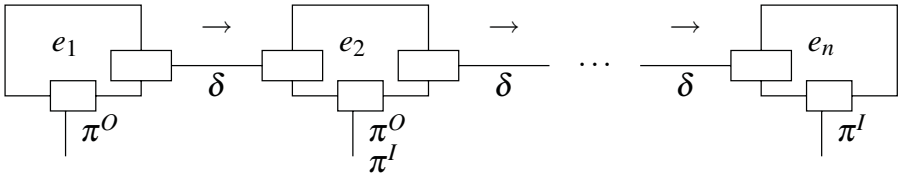

Figure 8: Determination of the safety of a flow path at a flow allowance state σ. A flow path $e_1 e_2 \ldots e_n$ is safe if every flow step it contains is also safe: every element except the final element e_n must be free of leaks, and every element except the initial element e_1 must be free of mixtures.

with the help of the auxiliary function $\alpha' : P \times P \times S \to \{0,1\}$ *defined as*

$$\begin{aligned} \alpha'(e,p,\sigma) &= 1 \\ \alpha'(e_1 e_2 \ldots e_n,p,\sigma) &= \delta(e_1,e_2,p,\sigma) \wedge \alpha'(e_2 \ldots e_n,p,\sigma). \end{aligned}$$

The successive application of the flow step safety function δ to each flow step in the flow path p is achieved by means of a recursive function which defines the safety of p inductively over the structure of p. In this manner, a simple and unambiguous way of determining the safety of a product flow path based on our abstract plant model is obtained.

4 Decentralised safety monitoring of product flow paths

As discussed in [9, 8], the decentralisation of process control systems is advantageous for many reasons (more flexibility, scalability and maintainability than centralised or monolithic systems, more robust handling of errors and service interruptions, easier modifications and upgrades of system components, computation independence thanks to locality, support for dynamic adaptability to new contexts, support for system synthesis for specific cases, etc.). A system for product flow path safety monitoring which operates in a decentralised fashion may be developed using the model-based definition of product flow path safety presented in this paper. This section gives a general description of the composition and operation of such a system.

We follow the decentralised component-based scheme presented in [9, 8]. Every plant element $e \in E$ is assigned a component of the system which controls and monitors the element, and which has connection ports for every connector $c \in C_e$. These ports are interconnected through bidirectional communication links in accordance with the relation $\circ$. Thus, the structure of the decentralised system is an analogy of the plant layout. Each component interacts with each of its neighbouring components by sending and receiving messages. A realisation of this scheme may be accomplished using IEC 61131-3 function blocks [2] for the components, which are common in process control systems. Furthermore, such a system may be automatically constructed from a flow allowance model of the plant by instantiating, parametrising and linking component blocks. This offers a simple and effective technique for synthesising systems such as the one outlined in this paper.

According to Definition 10, the function α is inductively defined over the structure of the flow path, meaning that the safety of a flow path p is determined incrementally in terms of each flow step of p, which in turn corresponds to the recursive invocations of α'. Therefore, we may use this same "calling" scheme for the definition of the messaging scheme of the components. Figure 9 shows the safety monitoring of a flow path $ab \ldots z$ by means of a decentralised system. Beginning with the component that corresponds to the first element in the path, a `MONITOR` message is issued along the components in the path, in correspondence to the evaluation of the function α. The path p is explicitly sent as part of the message, whereas the element e and the flow allowance state σ are given by the context of the execution of every component. The evaluation of the function δ is achieved as follows: if the element which corresponds to the component is not the final element of the path, the function π^O is evaluated by issuing `PROTO` messages in every diverging direction; likewise, if the element which corresponds to the component is not the initial element of the path, the function π^I is evaluated by issuing `PROTI` messages in every joining direction. These messages cause a flow path search to occur, according to the formulation of Definition 8 and similar to the analysis of flow paths presented in [9]. The responses to these messages are either `PROTO_1` (resp. `PROTI_1`), meaning that no offending flow path was found and therefore the value of π^O (resp. π^I) is 1, or they are `PROTO_0` (resp. `PROTI_0`) meaning that an offending flow path was found and therefore the value of π^O (resp. π^I) is 0. Additionally, the offending path is also sent in the message for reporting purposes. In order to reduce the number of messages sent, `PROTO` and `PROTI` messages are combined into a single `PROTIO` which performs the evaluation of both protection functions simultaneously.

After having received a `MONITOR` message and having sent the corresponding `MONITOR`, `PROTO` and `PROTI` messages, a component waits for a response to every message sent. When all responses have been obtained, the component may issue back a response to the original `MONITOR` message. In this case, two possible messages may be issued according to the responses received: `SAFE` corresponds to the case where α has the value 1, and `UNSAFE`

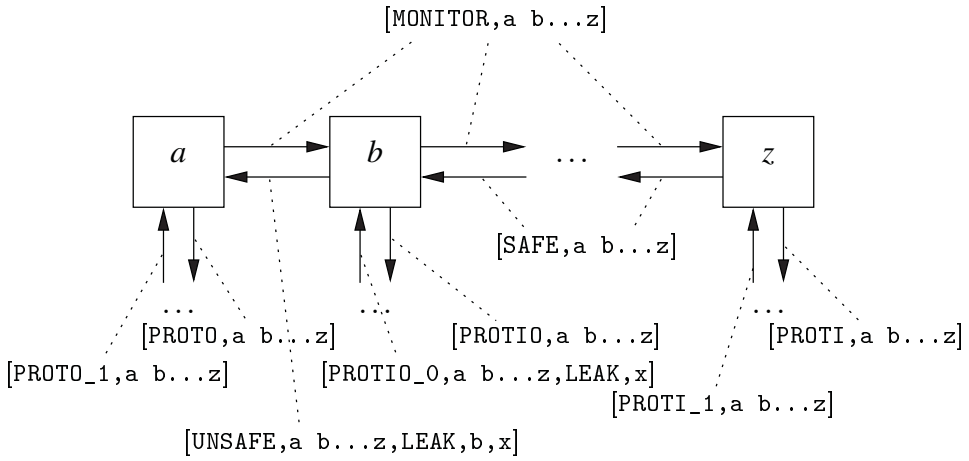

Figure 9: Decentralised, component-based safety monitoring of a flow path. Every element of the plant is represented by a component, and the components are interconnected with each other in the same way as their corresponding elements by means of bidirectional communication links. Messages are sent along the component connections, carrying out the evaluation of the safety functions.

corresponds to the case where α has the value 0. In the latter case, additional information regarding the reason for this determination may be included. This chain of messages eventually reaches the component which corresponds to the initial element of the path, which may then determine the safety value of α for the entire path p.

5 Summary

A basic and essential operation performed by processing plants is the movement of material, i.e. products, between plant elements. *Product flow paths* are the routes through the plant that products may use in order to flow from an initial element to a final element in the plant. Based on an abstract plant model which represents the structure of the plant and the possibility of product flow through its elements, a formalisation of the structure of a product flow path has been presented. Additionally, based on this same model, a definition of the safety of a product flow path at a given plant state has been given. These results may be used for specifying and automating the tasks of monitoring and assuring the safety of product flow paths in processing plants.

The definition of a product flow path is based on a plant structure model, which defines the elements of the plant and their interconnections by means of embedded connectors, and on a flow allowance model, which defines the ways in which product may flow into and out of every element through each corresponding connector. A product flow path is then defined as a sequence of neighbouring plant elements whose connectors may allow flow in the direction of the flow path at some plant state. Additionally, a flow path is said to be open at a given plant state if this flow can occur at this state.

The safety of a given product flow path is determined in a step-wise manner: a product flow path is safe if every flow step it contains is also safe. A flow step in turn is safe if product may flow exclusively from its first element to its second element, that is, if its first element is free from leaking deviations to product sinks, and if its second element is free from incoming mixtures from product sources. When applying this safety criterion to every flow step in a flow path, a simple formulation for the safety of a flow path is obtained which guarantees that a product flow path is free from product leaks and unintended product mixtures. Furthermore, it is applicable to any plant for which an abstract plant model is provided.

An outline of a system for product flow path safety monitoring has been presented, which operates in a decentralised manner and according to our model-based definition of product flow path safety. It follows a decentralised, component-based functional abstraction, where every plant element is assigned a component of the system which controls and monitors the element, and which has a connection port for each connector of the element. These ports are interconnected by means of bidirectional communication links, and in analogy to the structure of the plant. Each component interacts with each of its neighbouring components by sending and receiving messages. These messages correspond to the evaluation of the safety functions as defined in this paper, thus achieving the automatic determination of the safety of a product flow path in a decentralised manner. Furthermore, the automatic synthesis of such a system from an abstract plant model may be implemented in a straightforward way by instantiating, parametrising and linking component blocks.

6 References

[1] DKE, Deutsche Kommission Elektrotechnik Elektronik Informationstechnik im DIN und VDE. *DIN V 44366:2004-12: Specification for Representation of process control engineering requests in P&I Diagrams and for data exchange between P&ID tools and PCE–CAE tools*. DIN Deutsches Institut für Normung e. V., Berlin, 2004.

[2] IEC, International Electrotechnical Commission. *IEC 61131-3 Ed. 1.0 en:1993: Programmable controllers — Part 3: Programming languages*. International Electrotechnical Commission, Geneva, Switzerland, 1993.

[3] R Jorewitz, U Epple, A Münnemann, R Böckler, W Wille, and R Schmitz. Automation of performance monitoring in an industrial environment. In *PCIC Europe 2005, 2nd European Conference on Electrical and Instrumentation Applications in the Petroleum and Chemical Industry*, pages 116–120, Basel, Switzerland, Oct. 26-28 2005.

[4] Reiner Jorewitz, Gustavo Quirós, and Ulrich Epple. Modelling of multiple, semantically-coupled flow-graphs. In *SOLI 2008, IEEE International Conference on Service Operations and Logistics, and Informatics*, pages 965–971, Beijing, China, October 12-15, 2008.

[5] Morten Lind. Modeling goals and functions of complex industrial plants. *Applied Artificial Intelligence*, 8(2):259–283, 1994.

[6] Jörn Pachl. *Railway Operation and Control*. VTD Rail Publishing, Mountlake Terrace, USA, 2004.

[7] Martin Polke, editor. *Process Control Engineering*. VCH Verlagsgesellschaft, Weinheim, Germany, 1994.

[8] Gustavo Quirós and Ulrich Epple. From railroads to processing plants: Decentralised automatic assurance of product flow paths. In *EKA 2008, Entwurf komplexer Automatisierungssysteme: Beschreibungsmittel, Methoden, Werkzeuge und Anwendungen*, Magdeburg, Germany, 15-17 April, 2008.

[9] Gustavo Quirós, Martin Mertens, and Ulrich Epple. Function blocks for decentralised analysis of product flow paths. In *ETFA 2008, 13th IEEE International Conference on Emerging Technologies and Factory Automation*, Hamburg, Germany, 15-18 September, 2008.

[10] T. Schmidberger, A. Fay, and R. Drath. Automatisiertes Engineering von Prozessleitsystem-Funkionen. *atp - Automatisierungstechnische Praxis*, 02/2005, 2005.

[11] Stefan Schmitz and Ulrich Epple. On rule based automation of automation. In *Breitenecker F., Troch I. (Hrsg): 5th MATHMOD 2006: 5th Vienna Symposium on Mathematical Modelling*. ARGESIM - Verlag, Vienna, February 2006.

[12] Siemens AG. *Route Control V7.0 SP1 Programming and Operating Manual*. Siemens AG, Automation and Drives, 2007.