ELSEVIER

Chemical Engineering Science 60 (2005) 2069–2083

Chemical Engineering Science

www.elsevier.com/locate/ces

Modular dynamic simulation for integrated particulate processes by means of tool integration

Viatcheslav Kulikov[a], Heiko Briesen[a,*], Robert Grosch[a], Aidong Yang[a], Lars von Wedel[b], Wolfgang Marquardt[a]

[a]*Lehrstuhl für Prozesstechnik, RWTH Aachen University, Turmstr. 46, D-52064 Aachen, Germany*
[b]*AixCAPE, Intzestr. 1, D-52072 Aachen, Germany*

Received 3 August 2004; received in revised form 30 November 2004; accepted 30 November 2004

Abstract

In this contribution a sequential modular strategy for the dynamic simulation of particulate process flowsheets is presented and the efficiency of the approach is demonstrated by means of an example process for the crystallization of pentaerythritol. The flowsheet of the process consists of a number of different unit operations, e.g. evaporator, crystallizer, hydrocyclone, and mixer, which are described by mathematical models of largely varying complexity and structure. A key advantage of the presented sequential modular strategy is that specialized tools can be selected for the modelling and solution of each unit operation in the flowsheet. The tools are then coupled together by means of the tool integration framework CHEOPS in order to capture the overall structure of the flowsheet. In a case study, a startup of a crystallization flowsheet is carried out. As a result, detailed information about the dynamic plantwide process behavior is obtained. The practical relevance of the approach is demonstrated by means of a scenario where potential blocking of the filters following the crystallizer has been analyzed.

Keywords: Crystallization; Sequential modular approach; Dynamic simulation; Tool integration; Modular algorithm; Population balance

1. Introduction

Over the past decades process simulation has become an established tool in industrial applications and academic research. However, the tools available for process simulation still show certain limitations. Especially the dynamic simulation of complete process flowsheets of particulate processes is far from being a resolved issue.

Dynamic modelling of any fluid phase unit operation usually leads to the same type of equations: the material and energy balances are augmented by constitutive equations and closing conditions which results in a system of differential and algebraic equations (DAE). When these units are integrated into a process flowsheet the system size obviously increases but the type of equations remains unchanged. For the solution of the arising possibly large DAE system several powerful algorithms have been developed (e.g. Brenan et al., 1989) which have been implemented in stand-alone solvers like e.g. DASSL (Petzold, 1983) and LIMEX (Deufhardt et al., 1987) as well as integrated into commercial tools (e.g. Pantelides, 1996). This availability of easy-to-use software strongly promoted the widespread use of dynamic process simulation in industrial practice in the last decade.

Considering the simulation of processes involving particulate matter, the situation is quite different. The current lack of a versatile flowsheeting tool dealing with particulate matter has several reasons.

On the one hand one has to admit that, due to the higher complexity of the processes involved, the knowledge of particulate processes is not as advanced as for fluid phase processes where standard model libraries are already available. For particulate processes these libraries are not yet at hand,

* Corresponding author. Tel.: +49 241 8094861.
E-mail address: briesen@lpt.rwth-aachen.de (H. Briesen).

0009-2509/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ces.2004.11.037

although research efforts are undertaken to set up such a library at least for steady-state models.[1]

On the other hand the mathematical structure of particulate process models is often not of DAE type. Particulate products are often characterized by distributed properties (e.g. particle size). Therefore, the population balance concept (Hulburt and Katz, 1964; Randolph and Larson, 1988) is often employed to model particulate processes, which results in partial integro-differential-algebraic equation (PIDAE) systems.

The numerical solution of PIDAE systems is by far more complicated than that of DAE systems. Appropriate numerical schemes often depend on the particular problems to be solved. Therefore, current simulation tools mainly deal with specific problem classes and unit operations. Typically, a single unit or a single flowsheet of a fixed structure is considered. Such tools, however, exploit the available process knowledge and often use tailored numerics for an efficient solution.

In this contribution we suggest to realize the dynamic simulation of particulate process flowsheets by means of existing specialized software tools, instead of trying to build a new simulation tool from scratch. The models as well as the solution algorithms encoded in the simulation software for individual process units are reused in an a posteriori fashion by means of a modular dynamic simulation strategy (Marquardt, 1991) and a component-based software platform. The concept of dynamic modular simulation has again been in focus of a number of works done in recent years (Abdel-Jabbar et al., 1999; Grund et al., 2003; Garcia-Osorio and Ydstie, 2004). However, these efforts were predominantly concentrated on the development of the modular dynamic simulation algorithms from scratch and their usage for the simulation of general models.

The focus of this contribution is on the application of dynamic simulation algorithms for the software integration of different, existing process simulation tools. A clear benefit of such an approach is that if a certain process unit model is used within a mixed fluid-particulate process flowsheet, the unit model already implemented in some tool does not have to be reimplemented but can be reused directly. Also, the development of complex process models can be shared between different process specialists who use their preferred modelling and simulation tool, avoiding the usage of some general purpose tool which can hardly be found for all problem classes. Finally, the integration approach can take advantage of well-developed and highly specialized numerics in the particular software packages to increase solution robustness and efficiency.

In Section 2 an example crystallization flowsheet is presented. Section 3 describes the software solution for the tool integration, and the simulation algorithm used. The simulation case study and the results are discussed in Section 4, and general conclusions are presented in Section 5. The models used for the simulation can be found in the appendices.

[1] http://www.vt1.tu-harburg.de/vt1/toebermann/solidsim.htm (Accessed 05.03.2004).

2. Case study: crystallization of pentaerythritol

Crystallization from solution is one of the most important operations used in industry. The driving force for particle formation is the supersaturation in the liquid phase, which is depleted by nucleation and crystal growth. The supersaturation can be generated by cooling of the solution, by evaporation of the solvent, or by reactive generation of the solute. The crystal product is usually characterized by the *crystal size distribution* (CSD) (Randolph and Larson, 1988) represented by the population density function $n(l,t)$ of particle size l and time t.

In our case study, a process flowsheet for the evaporative crystallization of pentaerythritol from an aqueous solution is investigated. Pentaerythritol is commonly used in the production of high-quality alkyd resins, lacquers and lubricant additives, and its physico-chemical properties are well-studied and available in many databases. Also studies on the crystallization kinetics of pentaerythritol in have been carried out (O'Meadhra et al., 1996; Plácido et al., 2002).

The investigated process flowsheet shown in Fig. 1 does not claim to represent a real industrial process. Nevertheless, it comprises the most important steps of a typical crystallization process: (i) preparation of the concentrated solution; (ii) crystallization; (iii) separation of the product. Here, like in many other industrial processes the partial recycle of the product stream is used to minimize the loss of valuable materials. The flowsheet consists of four units—a mixer (4), an evaporator (1), a crystallizer (2) and a hydrocyclone (3), and shows one recycle stream.

In the evaporator, the solution is heated to 102 °C and partially evaporated in order to increase the concentration of pentaerythritol in the stream entering the crystallizer and to assist the generation of the necessary supersaturation. The concentrated solution is fed to the crystallizer where the supersaturation is generated by vapor withdrawal from the top of the unit. Thus the required level of supersaturation is generated and crystallization occurs. The crystal slurry leaving the crystallizer is passed to the hydrocyclone, where the larger crystals are separated from the smaller ones. The larger crystals in the bottom stream are the desired product whereas the overflow stream is recycled and mixed with the fresh feed. The fine particles contained in the recycle are dissolved during mixing, thus increasing the concentration of pentaerythritol in the solution.

The flow rate in the recycle can be controlled by the recycle ratio $\alpha = V_t^{hc}/V_f^{hc}$, and it is assumed that there is no purge in the recycle. The other process inputs are the feed flow rate V_{feed}, the feed composition w_{feed}, the heat supplied to the evaporator Q_{evap}, and vapor withdrawn from the crystallizer V_v^{cr}.

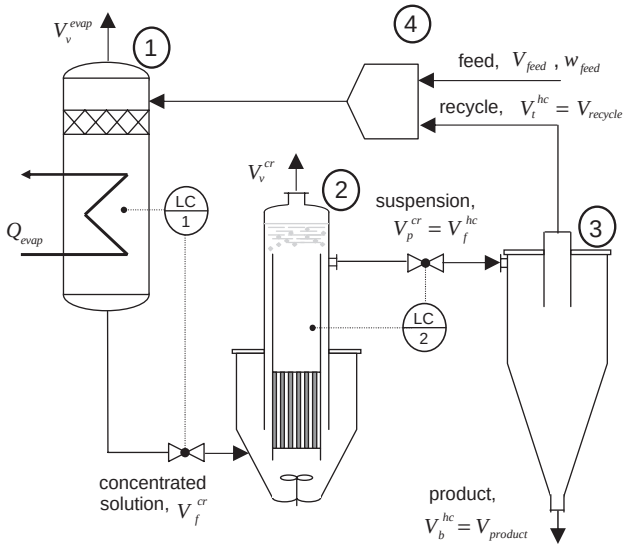

Fig. 1. Process system for the crystallization of pentaerythritol: (1)—evaporator; (2)—crystallizer; (3)—hydrocyclone; (4)—mixer.

In order to gain a detailed description of the processes in the individual units, a rigorous mathematical model is associated with each unit. Hence, every model is based on first principles describing the physical process phenomena to the extend possible, and not solely on empirical relations. In the framework of this concept, a population balance approach is used to model the crystallizer unit. The kinetic relations describing growth and nucleation of pentaerythritol in an aqueous solution were investigated by Chianese et al. (1995a, b) and O'Meadhra et al. (1996). In the present contribution, we adopt the dynamic model proposed by O'Meadhra et al. (1996), which has been validated with an industrial process by Plácido et al. (2002). The primary kinetic phenomena accounted for in the model are crystal growth and secondary nucleation in terms of attrition.

To simplify the modelling of the DTB type crystallizer, which is depicted in Fig. 1, the classical MSMPR assumptions (mixed suspension, mixed product removal) are made. A fines dissolution loop as it is sometimes employed in this type of crystallizers is not considered.

A summary of the kinetic expressions used in this model is presented in Appendix A.

The modelling of the hydrocyclone also requires closer attention. The separation process in this unit is characterized by the grade efficiency curve which relates the mass density of particles of certain size in the bottom stream (mass recovery) to the mass density of the particles of the same size in the feed. With the known grade efficiency curve the particle distributions in the overflow (the recycle stream) and underflow (the product stream) can be computed. Although the curve can be obtained by means of purely empirical models (Bradley, 1965), a more rigorous approach is to model the particle transport phenomena in the hydrocyclone. Bohnet (1969) presented an approach to compute the particle velocity in a hydrocyclone eddy, and Neesse and Schubert (1975) described a separation model based on the superposition of the sedimentation flux in the centrifugal force field and the turbulent diffusion flux. Further development of these models lead to a quasi-stationary multi-compartment model (Braun, 1989) which accounts for both geometry and particle fractions. This model, which is formulated in terms of differential equations representing the particle transport along the axial coordinate of the hydrocyclone for each particle size, is used in the present study. A more detailed explanation of the model is given in Appendix B.

The dynamic models of the remaining units, the evaporator and the mixer, are based on macroscopic mass and energy balance equations. For the evaporator, a vapor–liquid equilibrium model is employed where the vapor phase is assumed to act as an ideal gas. The vapor pressure is computed according to the Antoine correlation, and the activity coefficients in the liquid phase are computed according to the NRTL model. The parameters required for this equilibrium model for the pentaerythritol–water system are taken from the property data bank available in the software employed (see Section 3.1).

3. Dynamic modular simulation by tool integration

Before solving the flowsheet simulation problem using the suggested tool integration approach, several questions have to be addressed. First, it has to be decided which software tools will be used for the simulation of the individual units in the flowsheet. Second, a software solution for the integration of these tools should be found. And finally, a coordination algorithm has to be developed to guarantee the convergence of the flowsheet to the real solution of a plantwide simulation problem.

3.1. Simulation tools

The simulation tools for the representation of the specific units have to be selected with care. An essential requirement is that any tool must have a software interface for the communication with an external client. It should further provide a powerful and expressive modelling language to represent the model in its full detail. The model should be efficiently solved by means of built-in algorithms with an user-specified accuracy. Other factors (user-friendly interface, presence of property databank, etc.) should be considered as well. It should also be accounted for whether the model is already available within the tool as a part of its model library or as a result of previous investigation.

Of the four units involved, the models of the evaporator and the mixer consist of only standard balance and phase equilibrium equations. They can be represented by predefined models in available commercial flowsheet simulators.

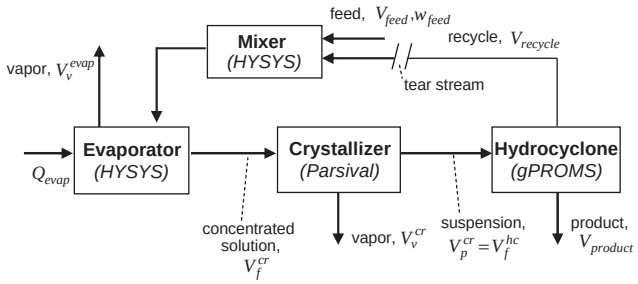

Fig. 2. Tool integration and modular simulation of the example flowsheet.

Hyprotech Hysys[2] has been chosen as a modelling tool for these units. Hysys contains a comprehensive model library for the standard process equipment as well as an extensive property databank with physico-chemical properties and correlations available for both pentaerythritol and water. The models of the crystallizer and the hydrocyclone, however, are neither available from the Hysys model library, nor can they be easily implemented there. Hence, other tools need to be considered for their implementation.

For the implementation of the hydrocyclone no dedicated simulator is known to the authors. Therefore, the process modelling tool gPROMS[3] was selected in which the model can be written in its mathematical representation.

For crystallization processes, a specialized simulation tool Parsival[4] has been developed in the recent years. The tool employs advanced adaptive numerical techniques for the discretization of population balance equations. The use of adaptation allows an efficient and error controlled simulation even for crystallization problems with very complex kinetics.

Thus, the "physical" process system is replaced by a corresponding "virtual" model flowsheet where the unit operations represented by the unit models and implemented in appropriate tools are connected by the couplings representing material and energy streams (Fig. 2).

3.2. Integration platform CHEOPS

The next step towards flowsheet simulation is the software solution for the integration of the selected tools. This is done by implementing the flowsheet model by means of an *integration platform*—a software environment which supports the setup of the flowsheet simulation problem and the communication between simulation tools during runtime in a generic way.

The development of such a platform has been carried out by Marquardt and coworkers (von Wedel and Marquardt, 1999; Scharwaechter et al., 2002; Schopfer et al., 2004). The integration platform CHEOPS introduced in these publications provides the generic class prototypes and interfaces for the integration of various tools, models and solver codes. The platform is component-based. Abstract classes are used to define and implement the integration framework in a generic way. This framework is instantiated during runtime by concrete software components for solvers, unit operation models, etc. The communication between the components is based on the middleware CORBA (Henning and Vinoski, 1999) which has been developed for cross-platform integration of components written in different programming languages.

To explain the functionality of CHEOPS, the major components of its architecture are outlined in the following. These components are the *model representation*, the *model source*, the *flowsheet representation* and *flowsheet solution algorithms*. Here we will only shortly outline the functionality of these components. A more detailed description of the architecture is provided by Schopfer et al. (2004).

The model representation is a core component of CHEOPS. It is a high-level abstraction of a model with its general interface functionality within CHEOPS. The component contains information about inputs and outputs of the model as well as state and parameter variables relevant to CHEOPS. It can exchange information with the other components via its interface.

Each model representation has an associated model source. A model source component can be a legacy model library, a CAPE-OPEN equation set object[5] (ESO), or the model in a dedicated simulator for a certain unit operation type. Depending on the type of model source, the instance of the model representation can in turn belong to either of two supported subclasses:

- an *open-form model representation* encoding a model source in open-form formulation which provides full access to the residuals and Jacobian matrix of the equation system (an example of such a model source is an ESO);
- a *closed-form model representation* encoding a model source in closed-form formulation, for which only inputs and outputs can be accessed, and which is therefore supposed to employ its own numerical algorithms as well as mapping procedures for unit inputs and outputs.

The communication between any model source and CHEOPS components occurs via model representations. Legacy code can directly be incorporated into CHEOPS as an instance of a model representation component. However, in case the models are available from an external tool, a dedicated *tool wrapper* has to be developed. The function of such a wrapper is to "translate" the commands of CHEOPS to the commands supported by the tool interface

[2] Aspen Technology Inc., http://www.aspentech.com (Accessed 12.07.2004).

[3] Process System Enterprise, www.psenterprise.com (Accessed 12.07.2004).

[4] Computing in Technology GmbH (CiT), http://www.cit-wulkow.de (Accessed 12.07.2004).

[5] COLAN CAPE-OPEN Specification, www.colan.org (Accessed 22.04.2004).

thus providing information exchange between the tool and CHEOPS.

The tool wrappers are also developed as instances of model representation components and are accessed via common CHEOPS interfaces. The actual tool interface is "hidden" behind the interface of the tool wrapper. This enables CHEOPS to communicate with the external tools as if they were its own components. A prerequisite to the development of such a wrapper is the availability of a documented tool interface to the external clients. Besides, the tool should be capable to act as a server to respond to the commands of CHEOPS controlling the execution of the integrated model.

A wrapper component for an ESO (as e.g. provided by gPROMS) is already integrated in CHEOPS and enables the mapping of a given ESO into both open-form and closed-form model representation components. However, most of the other commercial tools cannot create an ESO and do not provide information about the equation system being solved. Thus, tool wrappers derived from the closed-form model representation subclass have to be employed. However, the tool interfaces of the different tools differ very much from each other so that a dedicated wrapper instance has to be created for each simulator. Considering the tools used for the present study, both Hysys and Parsival provide COM interfaces to access the inputs, the outputs and, partially, the parameters of the model. In order to enable the communication, a software bridge connecting the COM interface of the simulator and the CORBA interface of CHEOPS had to be developed within the tool wrappers for Hysys and Parsival. For technical details we refer to Rosen and Curtis (1998).

The flowsheet representation component is implemented by means of several related components. The first component, the *unit operation*, represents a single unit model in the flowsheet and has one or more associated model representations. The second component is the *coupling* component representing a material and/or energy stream. A set of unit operations and couplings defines the whole flowsheet. Both the flowsheet and the tools assigned to simulate the individual units have to be specified by the user.

To solve the flowsheet, different flowsheet solution algorithm components are supported by CHEOPS. The integration platform CHEOPS defines separate components for equation-oriented and modular simulation strategies. The simulation strategy used for tool integration will be discussed in the next section.

3.3. Simulation strategy

Basically, there are two conceptually different strategies to simulate the complete flowsheet. The *equation-oriented simulation strategy* uses the actual equation system and its Jacobian from the individual model representations.This information is aggregated and solved by means of an appropriate solver (Barton and Pantelides, 1993). The model representations have to be of the open-form type in this case. In practice, however, existing simulation tools usually do not provide this open form representation. A closed form model representation, on the other side, is almost always accessible. This allows tool integration by means of a *modular simulation strategy*.

A modular strategy is based on the concept that the solution of the individual units of the flowsheet is delegated into the unit level. Which means that each unit operation is solved by means of an appropriate solver. Thus, the units provide only input–output information. Convergence is achieved by means of a certain coordination algorithm on the flowsheet level which is usually iterative.

To formally describe the simulation strategy for the flowsheeting problem, we present the mathematical models of individual units and the convergence procedure separately.

3.3.1. Abstract mathematical model

An abstract mathematical DAE model for the ith unit of the flowsheet can be written as follows (Gilles et al., 1988):

$$\dot{\mathbf{x}}_i = \mathbf{f}_i(t, \mathbf{x}_i(t), \mathbf{z}_i(t), \mathbf{u}_i(t), \mathbf{p}_i(t)), \tag{1}$$

$$\mathbf{0} = \mathbf{g}_i(t, \mathbf{x}_i(t), \mathbf{z}_i(t), \mathbf{u}_i(t), \mathbf{p}_i(t)), \tag{2}$$

$$\mathbf{y}_i = \mathbf{h}_i(t, \mathbf{x}_i(t), \mathbf{z}_i(t), \mathbf{p}_i(t)). \tag{3}$$

Here $\mathbf{x}_i$ and $\mathbf{z}_i$ are the vectors of differential and algebraic state variables in the ith unit, $\mathbf{u}_i$ is the vector of unit inputs, and $\mathbf{y}_i$ is the vector of unit outputs. The function vectors $\mathbf{f}_i$, $\mathbf{g}_i$ and $\mathbf{h}_i$ refer to the sets of state equations, algebraic equations and output equations, respectively. The resulting system of differential equations has a differential index of one. Suitable initial conditions $\mathbf{x}_i(t_0)$ have to be provided to fully specify the model.

The streams between the units can be represented by means of a *permutation matrix* for internal coupling of the flowsheet $\mathbf{P} = (\mathbf{P}_{ji})$. Each constitutive submatrix $\mathbf{P}_{ji}$ represents the coupling between the ith and the jth unit. Its elements are equal to 1, if an output variable contained in the vector of unit outputs $\mathbf{y}_i$ for the ith unit is mapped to the variable in the vector of unit inputs $\mathbf{u}_j$ of the jth unit, and 0, if no coupling exists between these variables. Thus a representation of the couplings in the flowsheet can be written by means of the *identity equations*

$$\mathbf{u}_j(t) = \sum_i \mathbf{P}_{ji}\mathbf{y}_i(t), \quad \forall j. \tag{4}$$

In case of more complex models comprising partial differential equations, state, input and output variables can additionally be distributed over a certain domain. However, within a certain simulation tool, discretization is always necessary to obtain a numerical solution. To formally apply Eq. (4) some discretized representation of the distributed variables, which is a discrete vector of algebraic variables, needs to be passed. Note, that this passed discretized representation can be different from the internal representation of the particular tools

in order to match the requirements of the other connected tools. While Parsival uses a polynomial representation on finite elements, the data passed to the hydrocyclone is represented by function values at selected collocation points.

3.3.2. Simultaneous and sequential modular simulation

A modular flowsheet simulation can be realized in simultaneous or sequential form, differing by the control logic used for solving the dynamic unit models and iterating the flowsheet couplings. In case of the *simultaneous modular strategy*, flowsheet tearing (Westerberg et al., 1979) is applied simultaneously to all the couplings. Since no input information is available from the previous unit operations, all input variables have to be guessed at the first step. With these input guesses the models of the individual units are integrated for each integration interval $[t_1, t_2]$ without noticing each other. When each simulation tool has computed a solution of the corresponding unit operation, the couplings are updated. The update of all couplings connecting the units in the flowsheet occurs simultaneously once per iteration on the flowsheet level. Thus, one can write the update procedure after the nth iteration as

$$\mathbf{u}_j^{n+1}(t) = \sum_i \mathbf{P}_{ji}\mathbf{y}_i^n(t), \quad \forall j,\ t \in [t_1, t_2]. \tag{5}$$

Therefore, independently of the flowsheet topology, the computation of new inputs to all units (unless they are fixed as the flowsheet inputs) is always based on the outputs computed at the previous and not on the current iteration. This concept is very useful for decreasing computational cost by parallelization of the simulation (Borchardt, 2001; Grund et al., 2003). The integration of the individual units can then be distributed on different computers and only the iterative update procedures require synchronization. However, to achieve good convergence properties, proper initial guesses for the values in *all* flowsheet couplings have to be made.

However, if parallelization is not a concern, a *sequential modular strategy* seems to be a more desirable choice. In this case, tearing is applied to only one stream per recycle denoted as *tear stream*, and the remaining streams are *untorn couplings*. The units connected by untorn couplings have to be integrated sequentially according to the topology of the flowsheet. The inputs of a downstream unit are then simply set equal to the outputs of the upstream unit at the current iteration. Thus, the simulation of the units always uses the latest information available from the upstream unit. Initial guesses only have to be made for the tear stream variables. Possible approaches for the selection of the tear streams can be found in the literature (Biegler et al., 1997).

3.3.3. General algorithm structure

The dynamic sequential modular simulation algorithm implemented in CHEOPS and employed in this study is schematically presented in the Fig. 3.

The algorithm operates on bounded integration intervals denoted here as Δt_k, where k is an index of the interval. In each of these intervals a fixed number of intermediate time points is selected. During the integration the variable values are evaluated at these points. Thus, these are not the single variable values, but the *variable trajectories* along the interval which are obtained during the integration and should be converged in order to get the solution of the flowsheet. Such an iterative process is called *iteration in the function space*.

The time interval Δt_k is usually larger than the time steps chosen by the different simulation tools. Therefore, it is possible to simulate each of the units according to its own transient behavior and to fully exploit the time step adaptation algorithms inherently available in effective simulation tools. If only one of the tools would be used for time step control, the corresponding physical process would dominate the choice of time steps. This would possibly lead to time steps too large to properly reflect the transient behavior of other units. Or the times steps would be chosen too small, resulting in unnecessary computational effort.

Before the integration starts, the total time horizon and the initial integration interval Δt_0 are set, the external inputs are initialized, and the guesses for the tear streams variables on the interval Δt_0 are decided on. With this information, the sequential integration of the units along the integration interval and the data propagation in all untorn couplings are carried out. After completion of the integration, the estimates on the error of the solution are computed for each of the tear streams in the current integration interval (Section 3.3.4), and the obtained values are compared with the prescribed tolerance.

If the convergence criterion is not met, another iteration on the current integration interval is needed. An update procedure has to be applied for the tear stream variables (Section 3.3.5). To facilitate convergence, an adjustment of the integration interval may be necessary. After the trajectories of all tear stream variables are updated, the sequential integration is repeated.

Otherwise, if the convergence criterion is met, the iteration on the current integration interval is stopped, and the output profiles are stored. Then, the next integration interval is initialized, with the length adjusted according to the strategy discussed in Section 3.3.6. To proceed with the integration, the guesses for the trajectories of the tear stream variables have to be estimated for the new integration interval by means of an extrapolation block as described in Section 3.3.7.

The integration continues until the end of the simulation time horizon is reached. If one of the selected tools will fail to converge, it will send an error message to CHEOPS, which in this case will terminate without further integration of the problem.

The algorithm is implemented in CHEOPS in a component-oriented way. The three stages of the algorithm—error check, tear stream update and trajectory extrapolation—are developed as separate components which together form the instance of the modular solver. The components realiz-

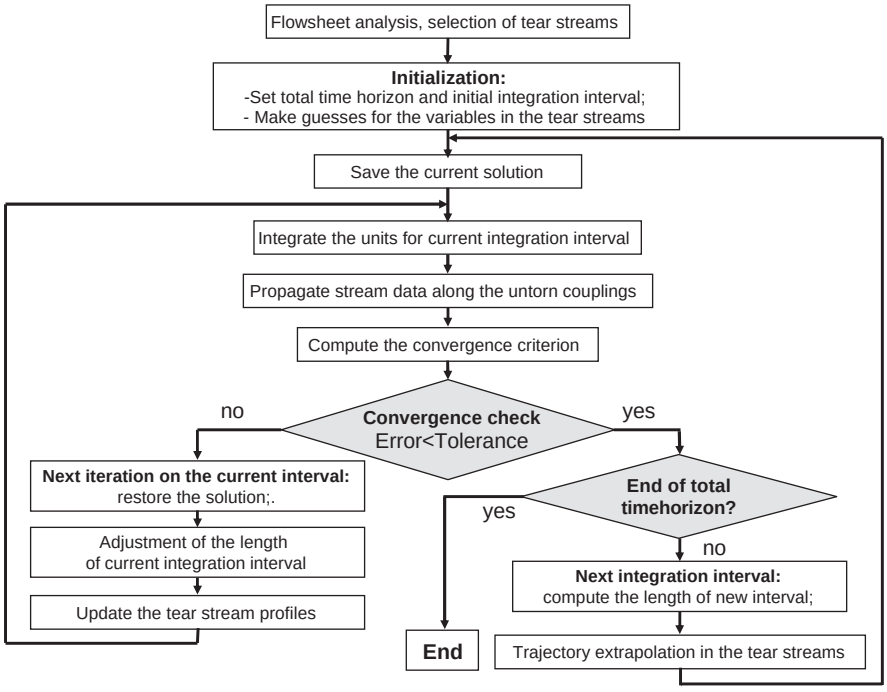

Fig. 3. Block scheme of the modular dynamic simulation algorithm.

ing one and the same part of the algorithm can be replaced easily enabling the user to select the most appropriate ones depending on the characteristics of the problem under investigation.

3.3.4. Convergence check

In order to have convergence, the intermediate solution at the nth iteration, $\mathbf{x}^n(t)$, on a given integration interval has to approach the true solution, $\mathbf{x}^*(t)$, for $n \to \infty$. As the true solution for the process states is unknown, an estimate on the error of the solution ERR^n has to be computed by comparing the outputs $\mathbf{y}(t)$ computed at the current iteration to the ones at the previous iterations:

$$\begin{aligned}\mathrm{ERR}^n &= \| \|\mathbf{y}^n(t) - \mathbf{y}^{n-1}(t)\|_{L_2}\|_\infty \\ &= \max_m \|y_m^n(t) - y_m^{n-1}(t)\|_{L_2} \\ &= \max_m \int_{t_{k-1}}^{t_k} (y_m^n(t) - y_m^{n-1}(t))^2 \,\mathrm{d}t, \\ &\quad t \in \Delta t_k = [t_{k-1}, t_k]. \end{aligned} \tag{6}$$

This error estimate is compared to the user-defined tolerance ε according to:

$$\mathrm{ERR}^n \leqslant \varepsilon \tag{7}$$

to assess convergence.

3.3.5. Update of the tear streams

The primary task of the tear stream update procedure is the computation of the guesses for the inputs of the downstream unit at the next iteration from the data at the current iteration. This procedure plays a key role in the convergence behavior.

The simplest and still widely used technique for tear stream updating in modular dynamic simulation is *direct substitution* (Liu and Brosilow, 1987) where the outputs of the upstream units in the tear stream at the nth iteration are directly assigned to the inputs of the downstream unit j at the $(n+1)$th iteration:

$$\mathbf{u}_j^{n+1} = \sum_i \mathbf{P}_{ji}\mathbf{y}_i^n(t). \tag{8}$$

As a modification of the method, the *relaxation*

$$\mathbf{u}_j^{n+1} = (1-\lambda)\mathbf{u}_j^n(t) + \lambda \sum_i \mathbf{P}_{ji}\mathbf{y}_i^n(t) \tag{9}$$

can be used introducing a relaxation parameter $\lambda \in (0, 1]$. For $\lambda = 1$ we recover direct substitution. A limitation of both methods is that they only exhibit linear convergence (Helget, 1997). The convergence properties are also restricted by a Lipschitz condition (Liu, 1983; Liu and Brosilow, 1987). Eq. (9) is actually implemented in CHEOPS.

An attractive alternative to accelerate convergence are Newton iterations

$$\mathbf{u}_j^{n+1}(t) = \mathbf{u}_j^n(t) - (\mathbf{J}^n)^{-1}\left(\mathbf{u}_j^n(t) - \sum_i \mathbf{P}_{ji}\mathbf{y}_i^n(t)\right) \tag{10}$$

which converge quadratically also in function space (Helget, 1997).

Typically, the Jacobian matrix $\mathbf{J}^n$ for the considered recycle is not available, and an approximation $\mathbf{B}^n$ has to be used

instead resulting in a *quasi-Newton approach*. To obtain this approximation, Grund et al. (2003) proposed a variation of Broyden update formulas for modular dynamic simulation.

3.3.6. Adjustment of the integration interval

The adaptation of the length of the integration interval, Δt_k, is another important measure to improve convergence. The fundamental idea is to select smaller integration intervals in case of steep variable trajectories, and to enlarge it for flatter ones in order to accelerate integration.

The integration interval at the first step is set by the user according to his knowledge of the system behavior. For further control of the integration interval, both heuristic and error controlled adjustment methods can be used (Helget, 1997). Methods based on *error controlled adjustment* compute the new integration interval, Δt_{k+1}, from the one at the previous integration interval, Δt_k, based on the ratio of the tolerance to the error estimate.

In the algorithm implemented in CHEOPS, an adaptive adjustment is used to decrease the interval length during iterations on the same integration interval Δt_k only. The following update formula

$$\Delta t_k^{n+1} = \Delta t_k^n \left(\frac{\varepsilon}{\mathrm{ERR}^n} \right)^{1/(p+1)} \tag{11}$$

is used with parameter p controlling the adaptation behavior (Liu and Brosilow, 1987). In CHEOPS we employ $p = 1$.

After convergence of one integration interval is reached, the length of the next interval is computed from the length of the current iteration Δt_k^* according to the heuristic

$$\Delta t_{k+1}^0 = 1.4 \Delta t_k^*. \tag{12}$$

Additionally, minimum and maximum bounds on the integration intervals can be specified by the user.

3.3.7. Extrapolation of variable trajectories

After the solution has been converged on the current integration interval, and the length of the next interval has been computed, guesses for the tear stream variables have to be provided for the next integration interval. This procedure does not directly affect the stability of the algorithm but is still quite important, since good initial guesses accelerate convergence. Bad guesses may also cause computational problems during the integration of the unit models.

To predict the trajectories of the tear stream variables, we use an extrapolation technique. The known variable trajectory from the converged iteration on the integration interval Δt_k can be interpolated by means of the polynomial $\hat{\mathbf{u}}_k(t)$ of order q whose coefficients are determined on the basis of $q + 1$ known points on the trajectory. The extrapolation of this polynomial acts as an initial guess for the trajectory of the tear stream variables on the next integration interval Δt_{k+1}.

One of the methods to compute the coefficients of the polynomial is the usage of the Taylor expansion in the vicinity of the last converged point of the trajectory $\mathbf{u}_k(t_k)$. In practice, a second order truncation is sufficient to get a reasonable extrapolation of the trajectory. Hence,

$$\hat{\mathbf{u}}_k(t) = \mathbf{u}_k(t_k) + \frac{\mathrm{d}(\mathbf{u}_k)}{\mathrm{d}t}(t - t_k) + \frac{\mathrm{d}^2(\mathbf{u}_k)}{\mathrm{d}t^2}\frac{(t - t_k)^2}{2} + O(t^3), \quad t \in [t_k, t_{k+1}]. \tag{13}$$

The first and second order derivatives in the vicinity of t_k are approximated by finite differences. To compute them, three known points of the trajectory in the vicinity of t_k have been used. Currently, the approximation (13) is used in the algorithm implemented in CHEOPS.

4. Process simulation scenario and results

The dynamic simulation of complex processes finds its primary application in the analysis of transient process behavior. Although most continuous processes are designed to run in steady-state, different events may force the process into a transient. These may be unexpected external disturbances affecting process inputs or operating conditions as well as routine operating procedures such as load change, process startup or shutdown. To safely start or stop the process, multiple inputs have to be set in a proper sequence. To understand the response of the process to such changes, a dynamic simulation is necessary. Apart from that, the simulation results obtained can be used for process analysis as well as process design and optimization purposes.

In this section, we discuss scenario where a potential filter blocking caused by a large fines fraction interferes with a controlled process startup. For a desired startup without filter blocking certain constraints have to be met. The objective of this illustrative study is to determine the startup parameters in such a way that these constraints are met.

The initial conditions are chosen not to reflect the steady-state of the process for the given input specification. At $t = 0$ the crystallizer contains a suspension with $M_p^{\mathrm{cr}} = 1770\,\mathrm{kg}$ of particles of primarily large size. The initial temperature in the crystallizer is $T_{\mathrm{cr}} = 25\,^{\circ}\mathrm{C}$. The evaporator has a constant temperature $T_{\mathrm{evap}} = 102\,^{\circ}\mathrm{C}$.

The volume of the crystallizer, $V_{\mathrm{cr}} = 7\,\mathrm{m}^3$, is constant. Equipment design parameters for the hydrocyclone are given in Table 2 in Appendix B.

The input specifications are the feed flow rate, $V_{\mathrm{feed}} = 2700\,\mathrm{kg/h}$, its mass fraction of pentaerythritol, $w_{\mathrm{feed}} = 15\%$ mass, the evaporator heat duty, $Q_{\mathrm{evap}} = 3600\,\mathrm{kJ/h}$, and the vapor withdrawal rate in the crystallizer, $V_v^{\mathrm{cr}} = 0.27\,\mathrm{m}^3/\mathrm{h}$. The feed stream is crystal free. The feed is pre-heated to $T_{\mathrm{feed}} = 60\,^{\circ}\mathrm{C}$, and is undersaturated at that temperature.

As mentioned above, we consider the prevention of filter blocking as a constraint to the startup transient. In the considered example process, a filter unit is placed in the product stream downstream of the hydrocyclone, thus representing a second separation stage in addition to the hydrocyclone (not shown in Fig. 1). As the majority of the particles flowing into the filter belong to the coarse fraction, it is sufficient to

use a standard dead-end cake filter where the particles are accumulated on the surface of the filtration medium, and the clear liquor has to penetrate through the filter cake and the filtration medium layers. Hence, the cake also participates in the filtering process. If the suspension contains too many fine particles, the pores in the formed filter cake are too small. Then, the liquid cannot pass easily, leading to rapid filter blocking. To avoid this, we introduce a constraint on the mass flow rate of the fine crystals in the product stream of the hydrocyclone which is set at maximum $\dot{M}_{\text{fines}}^{\max} = 10\,\text{kg/h}$ of fine crystals. Here, the term "fines" refers to crystals of the size not exceeding $l_{\text{cryst}}^{\max} = 50\,\mu\text{m}$.

Preliminary simulation studies show that the mass flow of fines can be lowered either by lowering the feed to the process or by changing the recycle ratio α. The larger the recycle ratio α, the more crystals go to the recycle, and the larger flow rates are observed in the internal streams which leads also to a better separation of heavy particles in the hydrocyclone. In the following scenario the parameter α has been chosen as the manipulated variable to control the process behavior.

The process simulations have been carried out for the following values of the recycle ratio ($\alpha = 0.5$, $\alpha = 0.6$, $\alpha = 0.7$, $\alpha = 0.75$) for the same initial states and input specifications. The recycle stream leaving the hydrocyclone has been chosen as a tear stream. The tolerance level for flowsheet convergence (see Eq. (7)) and the maximum length of the integration interval have been set to $\varepsilon = 0.001$, and $\Delta t_{\max} = 10\,000\,\text{s}$, respectively. All simulations have run for approximately 60 h of process time which took 2.5–4 h of computer time (AMD Athlon, 2.2 GHz). None of the simulations had to be terminated due to convergence failure. In the initial phase of the simulation, when the system states changed rapidly, simulation required smaller time intervals and a larger number of iterations (6–7) per one time integration interval. Approaching the steady-state, the time intervals have increased and the number of iterations per time interval has dropped to 2–3.

The resulting transient particle size distributions are presented in Fig. 4 for $\alpha = 0.5$. Integral properties of the particle size distribution like the average particle size l_{50} or the third moment M_3 can be computed from the distribution. Fig. 5 shows that an increase of the recycle ratio results in lower values of the third moment in the product due to a larger number of crystals withdrawn into the recycle. The analogical behavior was observed for the other moments. The moments also show an oscillating behavior which often can be seen in real crystallization processes (Randolph and Larson, 1988). These oscillations are damped to steady-state after approximately 30–40 h of process time.

The information on the particle size distribution in the product stream facilitates to compute the mass flow rate of fines into the filter to check if the constraint is satisfied:

$$\dot{M}_{\text{fines}} = k_v \rho_{\text{cryst}} \int_0^{l_{\text{cryst}}^{\max}} l^3 n_{\text{prod}}(l)\,\mathrm{d}l < \dot{M}_{\text{fines}}^{\max}. \tag{14}$$

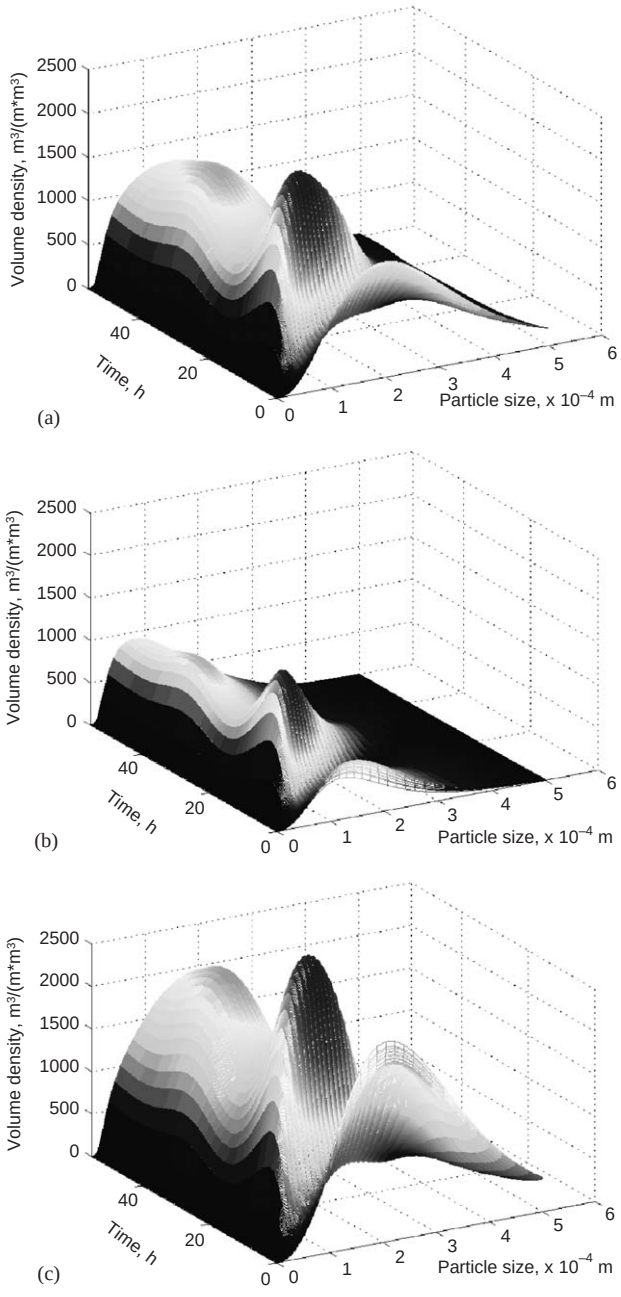

Fig. 4. Volume density distributions for $\alpha = 0.5$: (a)—in the crystallizer; (b)—in the recycle stream; (c)—in the product stream.

Fig. 6 shows the trajectories of the mass flow rates of fines for simulation runs for different recycle ratios. The higher the recycle ratio, the less is the mass flow rate of fines entering the filter. The results obtained can be used to get an approximate threshold value for the recycle ratio, above which the constraint is not violated. Note that the steady-state values of the mass flow rate are significantly smaller than the peak value reached during startup because of the oscillations in the crystallizer. Therefore, when estimating a lower bound on the recycle ratio parameter, one has to distinguish between the values for the steady-state and for the peak flow rates.

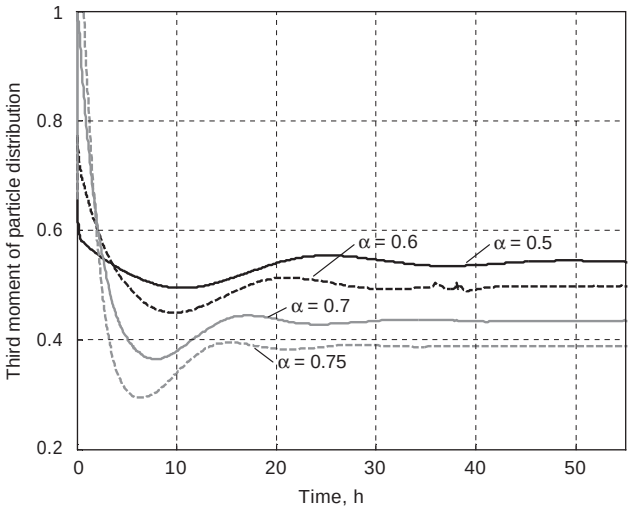

Fig. 5. Third distribution moment in product for different values of the recycle ratio α.

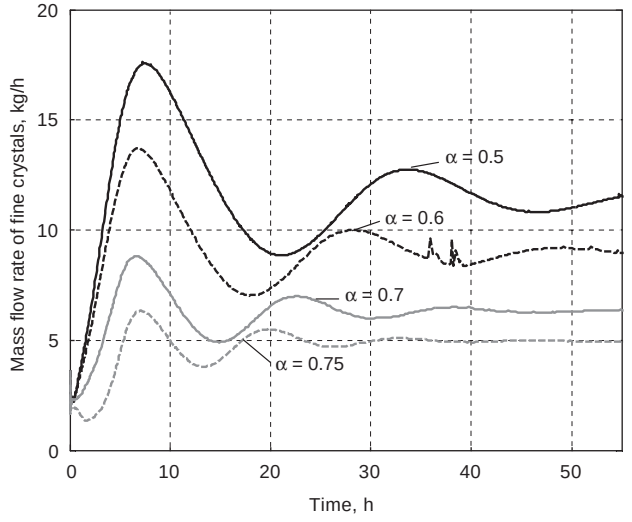

Fig. 6. Mass flow rate of fines in product stream for varying recycle ratios α.

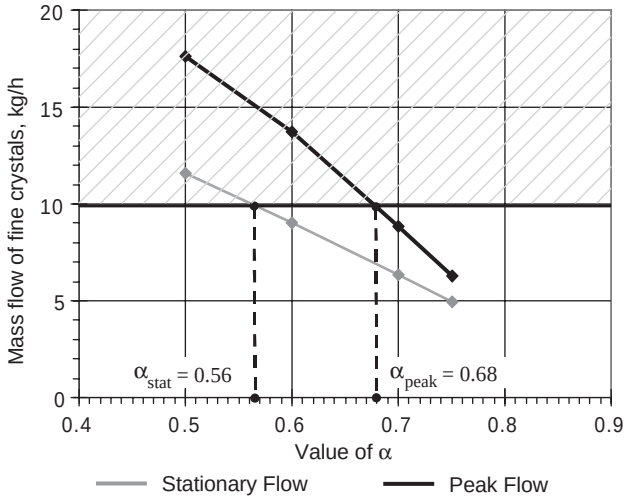

Fig. 7. Steady-state and peak values of recycle ratio α.

To meet the constraint for the steady-state of the process, a minimum value of $\alpha = 0.56$ should be chosen for the recycle ratio (see Fig. 7). This parameter value could also have been obtained from steady-state simulation. However, it can only ensure stable operation in steady-state, but not during the transient where strong oscillations are observed. In the latter case, dynamic simulation is required to find $\alpha = 0.68$ which guarantees that the constraint is met even for the peak flow rate to avoid the filter blocking also during startup.

5. Summary, conclusions, and outlook

In the present contribution, we focused on the integration of different simulation tools as an approach for the dynamic simulation of complex particulate process flowsheets with rigorous models of the unit operations. The approach reuses available commercial simulation tools where the models of the unit operations are represented and solved by means of specialized algorithms. For the integration of the various tools the integration platform CHEOPS has been used as a software solution. The potential of the approach has been demonstrated by means of a case study for a pentaerythritol crystallization problem with complex models of the crystallizer and the hydrocyclone. The simulation case study provides detailed information on the dynamic behavior inside the several units as well as on the plantwide level.

One important advantage of the suggested tool integration approach is its versatility. Each part of the flowsheet can be modelled in a specialized simulation tool. This way a maximum degree of modelling flexibility is provided to the user. This is even crucial if no single software tool is available to solve the overall flowsheet with an desired accuracy. With the modular sequential simulation strategy each tool cannot only help to set up the flowsheet simulation most user-friendly and flexible, but also allows to exploit the full power of the tailored numerical algorithms within each simulation tool. In the presented case this has been demonstrated by the use of Parsival which calculates the full CSD in high resolution instead of determining only moments of the distributions or rough particle size classes. Such a high resolution representation reduces the numerical error of the solution of the population balance and allows to address questions where the shape of the distribution function must be known, like in the case of filter blocking.

Another consequence of this versatility is that the approach can be applied time-efficiently, when parts of a flowsheet are already available in different tools. The existing models can directly be reused. Due to the continuing expansion of commercial simulators in science and industry, this aspect of model reuse and integration is of growing importance.

The approach can be further extended in two directions. On the one hand further simulation tools can be incorporated into the framework, thus broadening the bandwidth of unit operations and processes that can be treated (e.g. development of the wrapper to a computational fluid dynamics (CFD) package to account for fluid dynamic interaction). On the other hand solution algorithms for more complex problem formulations like optimization and parameter estimation can be implemented, to make the integration platform CHEOPS a more powerful tool.

An open issue is still the computation time which is relatively large in case of modular simulation algorithms due to the iterative convergence of the recycles on the flowsheet level. Thus, further algorithmic development is necessary in order to further improve the efficiency of the simulation, especially when larger flowsheets are addressed.

Notation

B	nucleation rate, m^{-1}
B^n	approximation of the Jacobian matrix at *n*th iteration
c	concentration of particles, kg/m^3
D_p	diffusion coefficient of particles, m^2/s
ERR	error estimate
G_{attr}	attrition rate, m/s
G_{eff}, G_{kin}	effective and kinetic crystal growth rate, m/s
h_{hc}	total height of the hydrocyclone, m
h_t	height of the overflow tube, m
J	Jacobian matrix
k_v	form factor
l	particle size, m
$\dot{M}_{fines}$	mass flow rate of the fine crystals in product, kg/s
M_p^{cr}	mass of the crystals in the crystallizer, kg
$M_{p,j}$	mass flow of particles in the *j*th compartment, kg/s
$n(l,t)$	particle size distribution, $(m * m^3)^{-1}$
$\mathbf{p}_i$	set of parameters for the *i*th unit
$\mathbf{P}$	matrix for the flowsheet couplings
Q_{evap}	amount of heat supplied to the evaporator unit, kJ/s
r	radial coordinate, m
t	time, s
Δt	time interval, s
$\mathbf{u}_i$	inputs for the *i*th unit
v_r	radial velocity of fluid due to convection, m/s
V	mass flow rate of the stream, kg/s
V_{attr}	volume of crystal fragments built by attrition, m^3
V_{cr}	volume of the crystallizer, m^3
V_{hc}	volume of the hydrocyclone, m^3
V_j	volumetric flow rate in the *j*th compartment, m^3/s
w	mass fraction of the crystallizing substance
w_p	sedimentation velocity of particles in radial direction, m/s
$\mathbf{x}_i$, $\mathbf{z}_i$	differential and algebraic states of the *i*th unit
$\mathbf{y}_i$	outputs of the *i*th unit
z	axial coordinate, m

Greek letters

α	recycle ratio
ε	tolerance
η	survival coefficient
λ	relaxation parameter
μ_3	third moment of particle size distribution
ρ_{cryst}	density of crystals, kg/m^3
σ	supersaturation

Subscripts/superscripts

b	refers to the bottom outlet (product)
cr	refers to the crystallizer
evap	refers to the evaporator
f	refers to the feed inlet (stream from the crystallizer)
feed	refers to the feed stream
hc	refers to the hydrocyclone
i, j	indices of the units in the flowsheet
j	index of the compartment
k	index of the time interval
m	index of the variable
n	iteration number
p, product	refers to the product stream
recycle	refers to the recycle stream
t	refers to the top outlet (recycle)
v	refers to the vapor stream

Acknowledgments

The authors gratefully acknowledge the financial support by the Ministry of Science and Research of the Federal State of North Rhine-Westfalia (Germany) granted in the framework of joint project VerMoS.

Appendix A. Model of the crystallizer

In the case study presented here, we have used the model of O'Meadhra et al. (1996) to represent the dynamics of an evaporative crystallizer producing pentaerythritol crystals.

Population balance: The evolution of the CSD is modelled by a population balance equation for a well-mixed

crystallizer with constant volume V_{cr}:

$$V_{cr}\frac{\partial n(l,t)}{\partial t}+V_{cr}\frac{\partial G_{eff}(\sigma,l)n(l,t)}{\partial l}+V_p^{cr}n(l,t)=V_{cr}B(\sigma,l). \quad (A.1)$$

The initial condition is $n(l,0)=n_0(l)$, the boundary condition is $n(0,t)=0$. Primary nucleation is neglected. In Eq. (A.1) the first left-hand term reflects the holdup of particles in the control volume, the second one corresponds to the effective growth, and the third one to the removal of particles by the outlet stream with volumetric flow rate V_p^{cr}. The right-hand side in Eq. (A.1) corresponds to the particle formation due to secondary nucleation.

In the underlying model the attrition phenomenon is captured by a negative growth rate G_{attr} and the nucleation rate $B(\sigma,l)$. Here, the negative growth rate accounts for the abrasion of large crystals, whereas the attrition fragments are introduced into the solid phase by the nucleation rate. Because of the crystals are also subject to the kinetic growth rate G_{kin} an effective growth rate of the crystals G_{eff} is observed:

$$G_{eff}(\sigma,l)=G_{kin}(\sigma,l)-G_{attr}(l). \quad (A.2)$$

The population balance is augmented together with the material and energy balances for the solute to build up a full description of the crystallizer.

Crystal growth: The crystal growth rate, $G_{kin}(\sigma,l)$, is a function of both the supersaturation of the solution and the crystal size:

$$G_{kin}(\sigma,l)=k_{kin}\sigma\left(1-\exp\left(-\frac{l}{p_1}\right)^{p_2}\right). \quad (A.3)$$

The supersaturation is defined as $\sigma=w/w_s-1$, where w is a mass fraction of the solute, and w_s its value in the saturated solution. p_1 and p_2 are empirical parameters which affect the shape of the growth rate curve.

Attrition: The attrition rate $G_{attr}(l)$ of a crystal is a complex function of crystal properties and process conditions (Gahn and Mersmann, 1999). Generally, the larger the crystal size, the stronger the crystals are affected by attrition. The attrition function in the pentaerythritol–water system can be represented by the "negative growth" model (O'Meadhra et al., 1996):

$$G_{attr}(l)=p_3\left(1-\frac{1}{1+(\frac{l}{p_4})^{p_5}}\right) \quad (A.4)$$

with empirical parameters p_3, p_4 and p_5.

Secondary nucleation: Here, the attrition fragments are introduced into the solid phase as new nuclei. In the underlying secondary nucleation model the nucleation rate $B(\sigma,l)$ is assumed to be proportional to the volume abraded from large crystals V_{attr}, to the normalized distribution of the nuclei $H(l)$, and to an empirical survival coefficient $\eta(\sigma)$, which depends on supersaturation and gives the number of the actually growing crystals:

$$B(\sigma,l)=\eta(\sigma)V_{attr}H(l), \quad (A.5)$$

$$V_{attr}=3k_v\int_0^\infty G_{attr}(l)nl^2\,dl, \quad (A.6)$$

$$\eta(\sigma)=p_6\sigma^{p_7}. \quad (A.7)$$

Model parameters: The values of the kinetic parameters are summarized in the Table 1.

Table 1
Kinetic parameters for the pentaerythritol crystallization

p_1	120 μm
p_2	1.5
p_3	2.4×10^{-9} m/s
p_4	305 μm
p_5	5
p_6	125
p_7	1
k_{kin}	1×10^{-6} m/s
k_v	0.471

Appendix B. Model of the hydrocyclone

The model of the hydrocyclone used in the study has been proposed by Braun (1989). The hydrocyclone is partitioned into three compartments as shown in Fig. 8. For simplification, the cylindric-conical geometry of the hydrocyclone is replaced by a purely cylindric geometry of equal volume, with an effective cylinder radius $r_a^*=\sqrt{V_{hc}/\pi h_{hc}}$, where V_{hc} is the actual volume of the hydrocyclone, and h_{hc} is its total height. The inlet compartment I ranges from the top of the hydrocyclone ($z=0$) to the lower edge of overflow pipe ($z=h_t$). Below compartment I the hydrocyclone is divided into two more compartments: the downward flow compartment II, and the upward flow compartment III. These compartments correspond to the areas of primary and secondary vortices in the hydrocyclone, they are delimited with a cylindrical surface of radius r_i.

The model is quasi-stationary and describes the evolution of the particle mass flow rate $M_{p,j}(z)$ in every compartment j along the vertical axis z of the hydrocyclone. The mass flow rate in turn can be related to the particle concentration by $M_{p,j}(z)=V_j(z)c_j(z)$ where $V_j(z)$ is the volumetric flow rate, and $c_j(z)$ the concentration of particles in compartment j (Table 2).

The evolution of the particle concentration is determined by the following phenomena acting in the radial direction:

- particle sedimentation to the hydrocyclone wall due to centrifugal forces and sliding along the wall in compartment II,
- particle sink from compartment III to compartment II due to centrifugal forces,

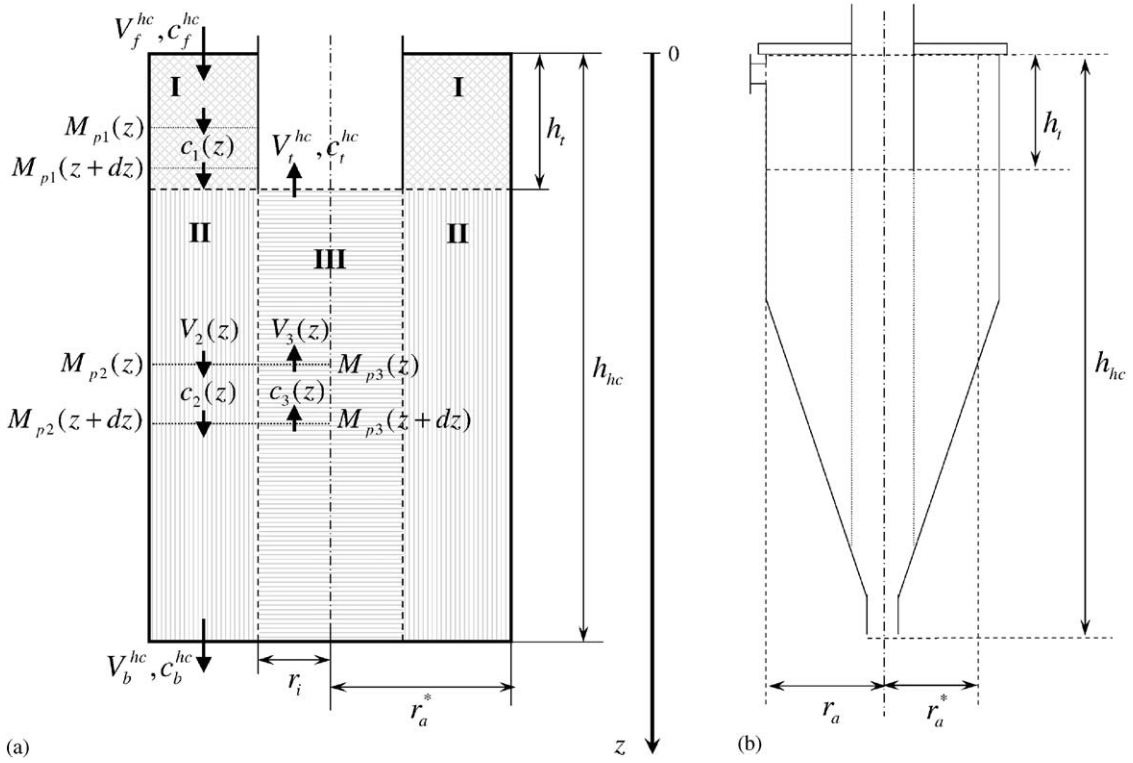

Fig. 8. Partitioning of the hydrocyclone: (a)—compartment model; (b)—original hydrocyclone.

Table 2
Equipment design parameters for the hydrocyclone

Hull diameter of hydrocyclone	d_p	0.45 m
Apex diameter of hydrocyclone	d_a	0.225 m
Height of hydrocyclone	h_{hc}	1.5 m
Height of apex tube	h_t	0.4 m
Radius of delimiting surface	r_i	0.05 m

- turbulent diffusion causing the particles to move from compartment II to compartment III,
- convective transport from compartment II to compartment III, counteracting the particle sink.

It is assumed that neither crystallization nor any kind of interaction between particles occurs in the hydrocyclone.

Total mass balance: Under the assumptions that the density of the suspension is constant, the total mass balance can be written as

$$V_f^{hc} - V_t^{hc} - V_b^{hc} = 0, \tag{B.1}$$

where V_f^{hc}, V_t^{hc} and V_b^{hc} are the volumetric flow rates of the feed, and the top and bottom outlets, respectively.

Compartment I : In the inlet compartment, the change of the particle mass flow M_{p1} occurs purely due to the sedimentation of the particles to the wall surface

$$\frac{\partial M_{p1}(z,l)}{\partial z} = 2\pi r_a^* w_p(r_a^*, l) c_1(z,l), \tag{B.2}$$

where $w_p(r_a^*, l)$ is the particle sedimentation velocity near the wall for particles of size l. For constant volumetric flow rate, we obtain

$$\frac{\partial c_1(z,l)}{\partial z} = -2\pi r_a^* \frac{w_p(r_a^*, l)}{V_f^{hc}} c_1(z,l). \tag{B.3}$$

Compartment II : The flow in this compartment is directed downwards. The particle mass flow M_{p2} is determined by

$$\frac{\partial M_{p2}(z,l)}{\partial z} = -\dot{m}_{p2} - \dot{m}_{p23} + \dot{m}_{p32}, \tag{B.4}$$

where the individual physical processes affecting the concentration of the particles are represented as follows:

- particle sedimentation and counter-current convective transport in the radial direction,

$$\dot{m}_{p32}(z,l) = 2\pi r_i (w_p(r_i, l) - v_r(r_i)) c_3(z,l), \tag{B.5}$$

- turbulent diffusion counteracting particle sedimentation,

$$\dot{m}_{p23}(z,l) = 2\pi r_i \frac{D_p(l)}{(r_a^* - r_i)} (c_2(z,l) - c_3(z,l)), \tag{B.6}$$

- particle removal from the compartment at the outer wall of the hydrocyclone due to sedimentation and sliding,

$$\dot{m}_{p2}(z,l) = 2\pi r_a^* w_p(r_a^*, l) c_2(z,l). \tag{B.7}$$

The volumetric flow rate $V_2(z)$ in the compartment II depends on the axial coordinate z according to

$$V_2(z) = V_f^{hc} + \frac{V_b^{hc} - V_f^{hc}}{h_{hc} - h_t}(z - h_t). \tag{B.8}$$

The resulting differential equation for compartment II is

$$\frac{\partial c_2(z,l)}{\partial z} = \frac{1}{V_2(z)}\left[-\frac{V_b^{hc} - V_f^{hc}}{h_{hc} - h_t}c_2(z,l) - 2\pi r_a^* w_p(r_a^*, l)c_2(z,l) - 2\pi r_i \frac{D_p(l)}{r_a^* - r_i}(c_2(z,l) - c_3(z,l)) + 2\pi r_i c_3(z,l)(w_p(r_i,l) - v_r(r_i))\right]. \tag{B.9}$$

The quantities $w_p(r_i, l)$ and $v_r(r_i)$ represent particle sedimentation velocity and the fluid radial velocity, respectively, evaluated at the surface delimiting compartments II and III. $D_p(l)$ is a turbulent diffusion coefficient for the particles. Braun (1989) provides a more detailed explanation and computational guidelines for these quantities.

Compartment III : The flow in compartment III is directed upwards. Therefore, the local particle balance for the compartment has to be computed in "backward" direction from

$$-\frac{\partial M_{p3}(z,l)}{\partial z} = \dot{m}_{p23} - \dot{m}_{p32}. \tag{B.10}$$

The volumetric flow rate in the compartment can be computed from

$$V_3(z) = V_t^{hc}\frac{h_{hc} - z}{h_{hc} - h_t}. \tag{B.11}$$

The following differential equation for compartment III is obtained

$$\frac{\partial c_3(z,l)}{\partial z} = \frac{1}{V_3(z)}\left[2\pi r_i(w_p(r_i,l) - v_r(r_i)c_3(z,l) - 2\pi r_i \frac{D_p(l)}{r_a^* - r_i}(c_2(z,l) - c_3(z,l)) + c_3(z,l)\frac{V_t^{hc}}{h_{hc} - h_t}\right]. \tag{B.12}$$

Boundary conditions: Boundary conditions are given for the particle concentrations at the boundaries of all hydrocyclone compartments:

$$c_1(0,l) = c_f^{hc}(l), \tag{B.13}$$

$$c_2(h_t,l) = c_1(h_t,l), \tag{B.14}$$

$$c_3(h_{hc},l) = c_2(h_{hc},l). \tag{B.15}$$

The concentrations in the outlet streams can then be calculated from the solution of the differential equations (B.3), (B.9), (B.12) with boundary conditions (B.13)–(B.15) as

$$c_2(h_{hc},l) = c_b^{hc}(l), \tag{B.16}$$

$$c_3(h_t,l) = c_t^{hc}(l). \tag{B.17}$$

References

Abdel-Jabbar, N., Carnahan, B., Kravaris, C., 1999. A multirate parallel-modular algorithm for dynamic process simulation using distributed memory computers. Computers and Chemical Engineering 23, 733–761.

Barton, P.I., Pantelides, C.C., 1993. Equation-oriented dynamic simulation—current status and future perspectives. Computers and Chemical Engineering 17 (Suppl. 1), S263–S285.

Biegler, L.T., Grossmann, I.E., Westerberg, A.W., 1997. Systematic Methods of Chemical Process Design, vol. 13. Prentice-Hall, Upper Saddle River, NJ, USA.

Bohnet, M., 1969. Neuere Untersuchungen über die Trennwirkung und den Druckverlust von Hydrozyklonen. Verfahrenstechnik 3 (9), 376–381.

Borchardt, J., 2001. Newton-type decomposition methods in large scale dynamic process simulation. Computers and Chemical Engineering 25, 951–961.

Bradley, D., 1965. The Hydrocyclone. Pergamon Press, Oxford.

Braun, B., 1989. Theoretische und experimentelle Untersuchungen des Einflusses der Feststoffkonzentration und der Partikelgrössenverteilung auf das Trennverhalten von Hydrozyklonen. Dissertation, Universität Braunschweig.

Brenan, K.E., Campbell, S.L., Petzold, L.R., 1989. Numerical Solution of Initial-value Problems in Differential-algebraic Equations. North-Holland, New York.

Chianese, A., Karel, M., Mazzarotta, B., 1995a. Nucleation kinetics of pentaerythritol. The Chemical Engineering Journal 28, 209–214.

Chianese, A., Karel, M., Mazzarotta, B., 1995b. Growth kinetics of pentaerythritol. The Chemical Engineering Journal 28, 215–221.

Deufhardt, P., Hairer, E., Zugk, J., 1987. One-step and extrapolation methods for differential-algebraic systems. Numerical Mathematics 51, 501–516.

Gahn, C., Mersmann, A., 1999. Brittle fracture in crystallization processes, Part A. Attrition and abrasion of brittle solids. Chemical Engineering Science 54, 1273–1282.

Garcia-Osorio, V., Ydstie, B.E., 2004. Distributed, asynchronous and hybrid simulation of process networks using recording controllers. International Journal for Robust and Nonlinear Control 14 (2), 227–248.

Gilles, E.D., Holl, P., Marquardt, W., 1988. The dynamic simulation of complex chemical processes. International Chemical Engineering 28, 579–592.

Grund, F., Ehrhardt, K., Borchardt, J., Horn, D., 2003. Heterogeneous dynamic process flowsheet simulation of chemical plants. In: Jäger, W., Krebs, H.-J. (Eds.), Mathematics—Key Technology for the Future. Springer, Berlin, pp. 184–193.

Helget, A., 1997. Modulare Simulation verfahrentechnischer Anlagen, VDI-Fortschritt-Bericht 251, Reihe 20. VDI Verlag, Düsseldorf.

Henning, M., Vinoski, S., 1999. Advanced CORBA Programming with C++. Addison-Wesley, Reading, MA, USA.

Hulburt, H.M., Katz, S., 1964. Some problems in particle technology. A statistical mechanical formulation. Chemical Engineering Science 9, 555–574.

Liu, Y.C., 1983. Development and analysis of coordination algorithms for interacting dynamic systems. Ph.D. Thesis, Case Western Reserve University, USA.

Liu, Y.C., Brosilow, C.B., 1987. Simulation of large scale dynamic systems—I. Modular integration methods. Computers and Chemical Engineering 11 (3), 241–253.

Marquardt, W., 1991. Dynamic process simulation—recent trends and future challenges. In: Arkun, Y., Ray, W.H. (Eds.), Chemical Process Control CPC-IV. CACHE Publications, Austin, pp. 131–180.

Neesse, T., Schubert, H., 1975. Modellierung und verfahrentechnische Dimensionierung der turbulenten Querstromklassierung. Chemische Technik 27 (9), 529–533.

O'Meadhra, R., Kramer, H.J.M., van Rosmalen, G.M., 1996. Crystallization kinetics of pentaerythritol. Journal of Crystal Growth 166, 1046–1052.

Pantelides, C.C., 1996. An advanced tool for process modelling, simulation and optimization. Presented at CHEMPUTERS EUROPE III, Frankfurt.

Petzold, L.R., 1983. A description of DASSL: a differential/algebraic system solver. In: Stepleman, R.S. et al. (Eds.), Scientific Computing. North Holland, Amsterdam, pp. 65–68.

Plácido, J., Guardani, R., Seckler, M.M., Derenzo, S., Giulietti, M., 2002. Simulation of a pentaerythritol industrial DT crystallizer. Proceedings of 15th International Symposium on Industrial Crystallization. AIDIC, Milan, pp. 1371–1376.

Randolph, A.D., Larson, M.A., 1988. Theory of Particulate Processes. second ed. Academic Press, San Diego.

Rosen, M., Curtis, D., 1998. Integrating COBRA and COM Applications. Wiley, New York.

Scharwaechter, H., von Wedel, L., Yang, A., Marquardt, W., 2002. A tool integration framework for dynamic simulation in process engineering. In: Proceedings of 15th IFAC World Congress on Automatic Control, Barcelona, Spain, 21–26 July, 2002.

Schopfer, G., Yang, A., von Wedel, L., Marquardt, W., 2004. CHEOPS: a tool integration platform for chemical process modelling and simulation. International Journal on Software Tools for Technology Transfer 6, 186–202.

von Wedel, L., Marquardt, W., 1999. CHEOPS: a case study in component-based process simulation. In: Malone, M.F., Trainham, J.A., Carnahan, B. (Eds.), Foundations of Computer-Aided Process Design, No. 323 in A.I.Ch.E. Symposium Series, A.I.Ch.E., pp. 494–497.

Westerberg, A.W., Hutchinson, H.P., Motard, R.L., Winter, P., 1979. Process flowsheeting. Cambridge University Press, Cambridge.