Computers and Chemical Engineering 23 (2000) 1773–1782

Computers & Chemical Engineering

www.elsevier.com/locate/compchemeng

Block-oriented process simulation of solids processes

J.-Chr. Toebermann, J. Rosenkranz, J. Werther *, G. Gruhn

Technical University Hamburg-Harburg, Chemical Engineering I, Denickestrasse 15, 21071 Hamburg, Germany

Received 21 October 1998; received in revised form 1 December 1999; accepted 1 December 1999

Abstract

A block-oriented process simulation system has been developed for solids processing applications. The specific requirements which have to be fulfilled for the description of processes with particulate solids are outlined and their implications for design and implementation of a simulator are discussed. The prototype simulator SolidSim is presented and its features are demonstrated by soil-washing and gravel processing applications. © 2000 Elsevier Science Ltd. All rights reserved.

Keywords: Flowsheet simulation; Process simulation; Solids processing; Mineral processing; Soil-washing; Gravel processing

1. Introduction

Process simulation, also termed flowsheet simulation, is widely used in chemical engineering and at least application of steady-state simulators for complex fluid processes has become state of the art (Schuler, 1995). A few commercially available simulators have extensions to handle solids, but their respective capabilities are not at all as sophisticated as their pure fluid handling capabilities. Consequently, the application of process simulation is in solids processing not as familiar as it is in the fluid-oriented fields of chemical engineering. Moreover, process alternatives with solids are often ignored during process design, because useful tools for solids processing are missing within the usual CAPE[1]-environment (Beßling, Lohe, Schoenmakers, Scholl & Staatz, 1997).

Particulate solids are more intractable than other dispersed systems — gas–liquid or liquid–liquid — for various reasons and derivation of mathematical models is more difficult for solids processes than it is for fluid processes (Clift, 1996). In practice, design of solids processes is based strongly on experiments. For simulation purposes many alternative models with different degrees of sophistication and different application domains may exist for an apparatus or no model with an appropriate precision may exist at all. Solids processing always considers disperse systems. Important properties of such systems, e.g. particle size, are distributed properties. Such properties must be characterized by their whole distribution, because they may not be sufficiently approximated by characteristic values, e.g. a mean particle diameter (Rumpf, 1990; Peukert, 1996). Besides, the process under consideration may require a simultaneous handling of several solid properties. For example, in mineral processing the liberation state, in gravel and sand industry processes the fractional density and in environmental processes like soil-washing the fractional contamination must be considered along with the particle size distribution.

In block-oriented simulation the process streams connect blocks, which usually represent a unit-operation and its model. This approach is straightforward to a process engineer and such programs are easy to use (Schuler, 1995). Examples for commercial simulators are Aspen Plus, Pro/II and HYSYS in the chemical and petrochemical industry as well as USIM Pac and JK-SimMet in the mineral processing industry. A block reads data from its input streams and writes data to its output streams. A well-defined information structure stores the process stream data. In chemical engineering the information structure usually consists of temperature, pressure as well as components and their partial mass flows. This structure is well suited to fluid pro-

* Corresponding author. Tel. +49-40-428783039; fax: +49-40-428782678.

E-mail address: werther@tu-harburg.de (J. Werther)

[1] Computer aided process engineering.

cesses, but due to the lack of particle characterization features it is not sufficient for solids processes. An additional particle size distribution may be defined for example in Aspen Plus (Aspen Technology, 1988). The number of size fractions and the size limits are user-defined, but are the same for each stream[2] in a simulation. Additional data fields may be provided for each process stream by the simulator on user-demand. These data fields are not supported in any other way by the simulator and are therefore only meaningful for user-supplied subroutines. However, these data fields enable the definition of user-defined properties. Jones (1984) has used this feature for the simulation of an oil extraction process from shale with carbon content as particle property in addition to the particle size. However, including and organizing further solids features is rather complicated, makes the simulation inflexible and requires special expert knowledge. Broussaud (1988) as well as Napier-Munn and Lynch (1992) give reviews about process simulation systems which have been developed in mineral processing. These programs use just the particle size distribution. More recent developments for special purpose simulators have a more extended stream structure to represent further, e.g. assay, data (JKTech, 1998). Nevertheless, the information structure remains fixed, so no user-defined attributes may be used. Very recently, Hill and Ng, 1997 presented a prototype program using a general framework for discretized population balance equations and their transformation. So far they also used particle size distribution as the only particle property.

In equation-based process simulation the user specifies all model equations, which are simultaneously solved by the program. Examples of commercial simulators are SpeedUp and gProms. All process stream variables are variables of the equation set. The information structure for a process stream is therefore not limited by program internal reasons. However, standard models of process streams are available for a comfortable user interaction. Any particle feature may be included by appropriate equations. Barton and Perkins (1988) used SpeedUp to solve mineral processing problems. They either used a predefined discrete particle size distribution or distinguished between particle types with characteristic values. Pantelides and Oh (1996) used gProms to solve a crystallization process. A distributed parameter system was defined for the particle size distribution. The discretization of the particle size distribution was therefore independent of the model equations and distribution moments were calculated by gProms. On the contrary in SpeedUp user-supplied subroutines were necessary for the same purpose (Barton & Perkins, 1988). In principle, equation-based simulation systems might therefore be appropriate tools for solids processes, but the high effort for set-up and evaluation of the equation system as well as the required expert user knowledge is a disadvantage. Consequently such programs seem to be more relevant for special problems to be treated by experts (Schuler, 1995).

Simulation of solids processes has its special problems. An appropriate, generally applicable and easy-to-use tool for such processes is not yet available. It is the aim of the present work to identify special objectives and requirements and tackle the observed problems. For ease of use a block-oriented, sequential–modular approach was chosen. In order to use the available commercial block-oriented simulation systems for solid processing, extensions and enhancements in the process stream as well as in the model library would be necessary. User models may be incorporated with some effort, however, the same extensibility is not true for the process stream, due to the usually strong coupling between models and data-fields for process streams. Even sophisticated interfaces like OLE-interfaces[3] suffer from this strong coupling. User variables, e.g. a particle size distribution, may be easily added to a process stream, but the user must supply the code to handle its additional data at each standard model, i.e. even at a simple splitter. As a consequence of such considerations, a new process simulation system, called 'SolidSim' (Gruhn, Rosenkranz, Werther & Toebermann, 1997; Werther, Gruhn, Toebermann & Rosenkranz, 1997), with focus on solids specific tasks and just basic features in fluid handling and simulation framework has been developed. To provide a flexible and modular software design, an object-oriented implementation (Booch, 1994; Rumbaugh, Blaha, Premerlani, Eddy, & Lorensen, 1991) using C++ was chosen.

2. The SolidSim simulation system

2.1. Layout of the simulator

The simulator has a conventional block-oriented layout. Particularly designed for solids processes is the core of the simulator, which consists of unit operations and streams together with the associated model library and database. In the actual prototype version necessary simulation framework features are still largely missing and the program interfaces are just rudimentary implemented. Nevertheless, additional tools for data reconciliation and parameter optimization are integrated with SolidSim to enable an automated calibration of

[2] More precisely they are the same for a substream, but they may be different in different substreams.

[3] OLE: object linking and embedding.

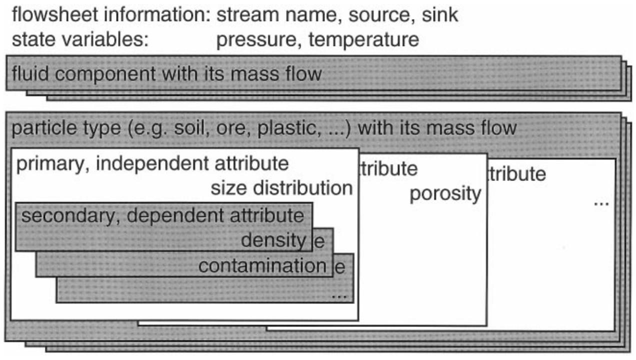
flowsheet information: stream name, source, sink
state variables: pressure, temperature
fluid component with its mass flow
particle type (e.g. soil, ore, plastic, ...) with its mass flow
primary, independent attribute
size distribution
porosity
secondary, dependent attribute
density
contamination
...

Fig. 1. Information structure for material stream representation.

model variables to experimental data. The models which are presently implemented are given in Table 2.

2.2. Process stream definition

A block-oriented simulation requires a well-defined information structure for a process stream, because otherwise a block might write data as temperature and another block might interpret the same data as pressure. Crucial points for the information structure are the organization of information exchange between blocks and process streams as well as how much information should be available within a process stream.

In addition to the usual information – temperature, pressure and flow of each fluid component – the particulate matter must at least be characterized by a discretized particle size distribution. Several processes may require additionally solid attributes to further characterize the disperse state or to characterize other solids properties. These attributes are also characterized by discretized distributions. The solids attributes may depend on each other, but due to practical reasons, i.e. measurement problems, just a two-dimensional characterization is reasonable. However multiple two-dimensional characterizations are possible (Toebermann, Werther, Rosenkranz & Gruhn, 1999). Because of its practical importance, a particle size attribute is always present and does not depend on another attribute. Otherwise any attribute and any dependency structure is supported. Different solid phases are independently characterized using ''particle types''. A particle type has its material flow and is characterized by its attributes, which define its properties. So a particle type is in some sense a solid equivalent to a fluid component. The complete information structure is given in Fig. 1 where size distribution and particle porosity are presented as examples of independent attributes. Particle density and contamination are listed as examples of secondary, i.e. dependent attributes. It should be noted that the particle density could as well be an independent attribute. The proposed structure has the advantage of a structuring of information levels, by defining general, fluid specific and solid specific information, respectively. The particle characterization is essentially a general tree structure with just two levels implemented by linked attributes' collections. Collection and attribute behavior are implemented separately to decouple software design. If necessary, this approach may be easily extended to more-dimensional characterizations.

To illustrate the object-oriented implementation the class diagram for the process stream information structure showing the mains associations and a small part of

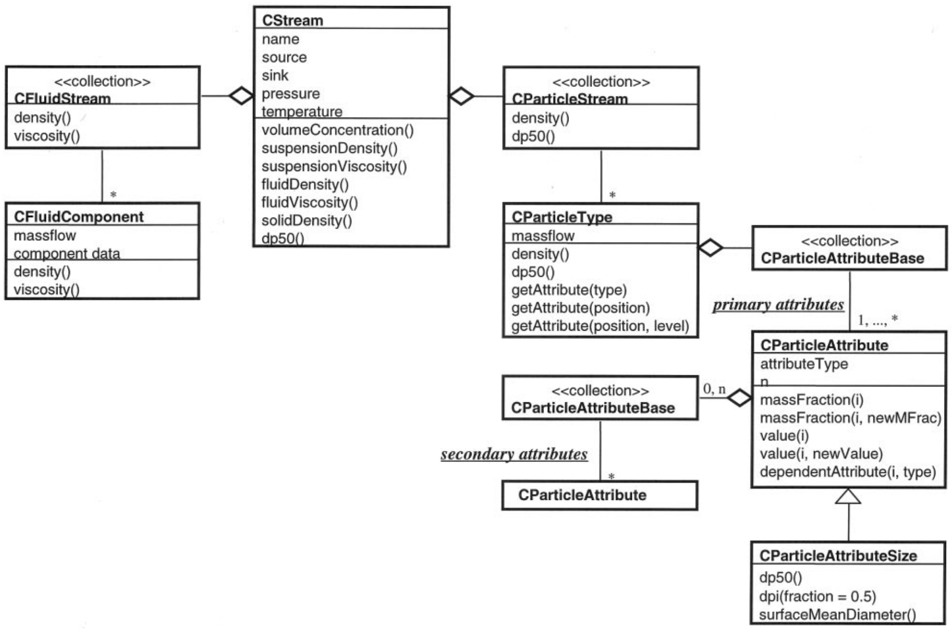

Fig. 2. UML-class diagram for the process stream information structure.

the inheritance structure is given in Fig. 2 using UML-notation[4]. A material stream object CStream contains stream name, flowsheet connectivity information, pressure and temperature as well as two collections for fluid components and particle types. It provides operations for the state variables and for solid–fluid-system properties like solids volume concentration and suspension viscosity. In addition, the interface provides access to all fluid and basic solid functionalities. CFluidComponent has its mass flow and access to its pure component data for thermodynamic and physical properties. CFluidStream is a collection of CFluidComponent-objects and has functions for properties of mixtures. Because of the focus on solids processing fluid system functionalities are just basically implemented. For example, phase equilibria are not considered. State variables are contained in CStream, but otherwise CFluidStream and CFluidComponent implement the "classical" fluid stream. Fluid system features may therefore be extended in a common way. In principle, it would be possible to do without CStream and to use CParticleStream as a sub-object of an existing fluid stream representation, if functionalities for suspension properties and a reference to the super-object, the (fluid) stream, is added to CParticleStream. But the proposed structure has — as already mentioned — the advantage of well-structured levels of data and operations. CParticleStream is a collection of CParticleType objects and has operations to condense solid information, e.g. calculating the mean particle diameter of all solid phases. CParticleType has its mass flow and the attribute information. It manages access to the attributes and ensures the integrity of the complex structure. CParticleAttributeBase is a collection, which helps in organization of the general tree structure. CParticleAttribute has a discretized distribution and, if necessary, references to dependent particle attributes. A distribution is stored as two vectors, one vector with the mass fractions and the other vector with the corresponding value, e.g. upper particle diameter of a size interval. If the vector length degenerates to one, just a mean value is stored. Because of the importance of particle size characterization CParticleAttributeSize is derived from CParticleAttribute and such a CParticleAttributeSize object is always the first element in the primary attribute's collection of a CParticleType. Its specialized and extended interface offers among other operations to calculate $d_{p,50}$ or the surface mean diameter.

This particle characterization is very flexible. Any attribute may be dynamically created as soon as required. So it is not necessary to predefine an attribute structure or have an otherwise fixed attribute structure. For example the particle density may be defined as a particle size-dependent attribute, as an independent attribute or not defined as an attribute at all. Furthermore, this definition may be different for various process streams and may also be changed by a unit-operation calculation. The flexible structure entails an information decoupling between streams and blocks. A block cannot assume anything about the actual internal information structure of the stream, but due to the object-oriented implementation it does not need to know anything. It just requests a specific information and the stream object is responsible for returning the information according to its actual state or for returning an error if the information cannot be retrieved. Any stream information may be investigated, modified or created by the interfaces of CStream, CParticleType, CParticleAttribute and CParticleAttributeSize.

An example may illustrate the sophisticated particle characterization features. One fundamental operation in mineral processing is liberation of the valuable minerals from the waste gangue minerals. An appropriate representation of a mineral with two components A and B according to the liberation model of Wiegel (Lynch, 1977) is sketched in Table 1. Particle size is the only one independent attribute. To describe the state of liberation two size-dependent attributes are used, one for the mean volume fraction of B in a composed particle and the other for the volume fraction distribution of pure A, pure B and composed particles.

2.3. Design of the model library

In solids processing disperse state and other solid attributes influence the performance of a specific apparatus in a very complex way. In addition, geometry and throughput may have major influences. Due to this complexity and due to tradition often empirical modeling approaches are used and even physical models may have major empirical corrections and parameters. Accordingly, many models with different degrees of sophistication and different application domains will exist for an apparatus. On the other hand generally applicable models for some apparatus are still lacking, e.g. for spiral concentrators. The quality of solids processing models is often not satisfactory, i.e. model accuracy is relatively poor or models do not define a complete transformation of input-to-output streams. For example, many hydrocyclone models calculate cut size, but not separation sharpness or fines bypass. Modeling the whole behavior of a process or an apparatus often requires a combination of models for sub-processes. These sub-models may differ widely in their modeling approach, e.g. in comminution a rigorous particle stress model may be combined with a short-cut model — e.g. just an empirical separation efficiency curve — for an internal classification. Taking into account empirical or semi-empirical model derivations the user must some-

[4] UML: unified modeling language.

Table 1
Example of a particle characterization for a mineral processing application

Primary attribute		Secondary attributes			
Size distribution		Distribution			Mean value
Size interval (μm)	Mass fraction (weight%)	Within size interval			
		V (%)[a]			$C_{B,AB}$ (%)[b]
		A	B	AB	
–53	18.0	0.74	0.22	0.04	0.489
53–106	10.0	0.71	0.19	0.10	0.475
106–212	17.0	0.66	0.15	0.19	0.453
212–425	18.0	0.55	0.09	0.36	0.408
425–850	17.0	0.28	0.02	0.70	0.316
850–1400	20.0	0.09	0.00	0.91	0.259

[a] V, volume fraction of pure A, pure B and composed particles
[b] $C_{B,AB}$, volume fraction of B within a composed particle.

times adjust model parameters to a specific application and therefore user-support in estimating parameter values would be desirable.

In summary, a model library should meet the following requirements and objectives:

- provide as far as possible only physically well-based models with complete definitions of the transformation from the input to the output streams;
- provide shortcut and rigorous approaches for different levels of simulation purposes, e.g. in an early screening of process variants a user-supplied separation efficiency curve may be sufficient while for process optimization a rigorous model may be required;
- if necessary, provide alternative models to cover different application domains with acceptable precision
- provide database support for the user when choosing appropriate values for empirical constants

The field of particle technology is characterized by a vast variety of models. A structured model selection is therefore required for a useful model library. Moreover, a comfortable interchange of models for an apparatus must be made possible, which enables the appropriate model for the actual application and the required precision to be chosen.

The design of the model library is sketched in Fig. 3 for the example of hydrocyclones. In the left branch specialization of models take place, from a general model via using a general separation efficiency curve model to a specific separation efficiency curve. In the right branch specialization of apparatus take place, from a general unit operation via a general cyclone-defining geometry to a hydrocyclone-defining operational features. A combination of these branches leads to a specific and useable model. Combination may be done using association or multiple inheritance. If alternative models are not required for an apparatus, model-specific features are added in the apparatus specialization, so the right branch only exists. The intermediate objects are used for the design of the model library only and just the last objects are available to and of interest for the user. Objects and their functionalities are reused within the model library by inheritance or delegation, while polymorphism still allows specialization of an operation. Consequently the code is redundancy-free, i.e. an operation is only coded and needs only be maintained once. The library design is flexible, enables subsequent and incremental development and supports the software maintenance. It results in a modular design, where user-level models may be added through simple model combination. For example, the internal structure for a comminution apparatus is given in Fig. 4. For a given apparatus a specific particle stress model is combined with optional internal classifications. A pre-classification result for already

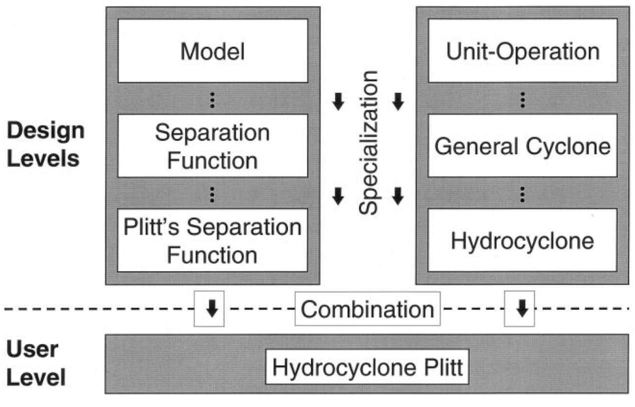

Fig. 3. Model library structure.

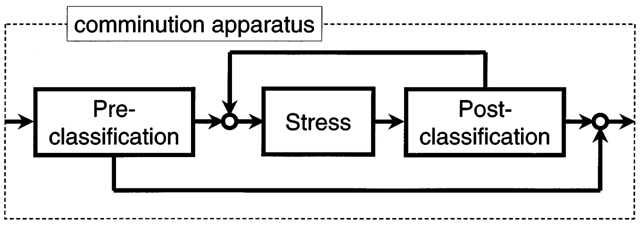

Fig. 4. Internal structure of a comminution apparatus.

fine particles in bypassing further particle stress. A post-classification result in an internal recycle loop for particles, which are in spite of the performed particle stress still too coarse. Several combinations of particle stress models and internal classification result in non-linear equation systems. In such a case the user model is implemented as a combination of the necessary stress, classification and mixing models and if required a tear-stream model for an internal recycle loop. The calculation routine of the user model then just organizes the necessary iterations for the solution.

The object-oriented model library implementation is illustrated by hydrocyclone models, Fig. 5. CUnitOperation defines the common interface for all unit operations. This enables, for example, the simulation control to initialize, call the calculation routines and converge the flowsheet without knowing the specific apparatus and model of any unit operation. No implementation is given for most of the operations at this stage (indicated by func() = 0). CUnitOperation1to2 implements the connection of the unit operation to its streams for one inlet and two outlet streams. Its initialization routine checks whether all ports are properly connected and its calculation routine initializes the outlet streams with the inlet stream. This assignment violates the mass balance, but it secures transportation of all stream information through the unit operation by setting and saving attributes, which are unused by the unit operation. Otherwise unused attributes would not be written to the outlet stream(s) by the unit operation in any way. Derived classes are then responsible to balance mass flow and change appropriate attributes. CGeneralCyclone implements cyclone geometry and its initialization routine checks the validity of the actual geometry. CHydrocyclone adds operational features of a hydrocyclone, e.g. calculation of maximum pressure drop. More significantly, it references a CHydrocycloneModel object. The main part of the calculation is delegated to this object or more precisely to one of the derived classes, which implement specific hydrocyclone models. This construct allows an easy and even dynamic exchange of models. CHydrocycloneModel is indirectly derived from CUnitOperationModel, which defines a common interface similar to the CUnitOperation interface. But the former interface is intended for an internal access within the model library, while the latter interface is accessed from all program parts. The intermediate derivation step — CParticleSeparationModel — defines operations to retrieve a specialized 'view' of a stream, i.e. vectors for mass flows, nodes and grades, according to a separation property, e.g. size, settling velocity or density, and calculation of the outlet streams after performing separation on the specialized view. To support models that calculate parameters of a predefined separation function a CSeparationFunction is associated with CParticleSepa-

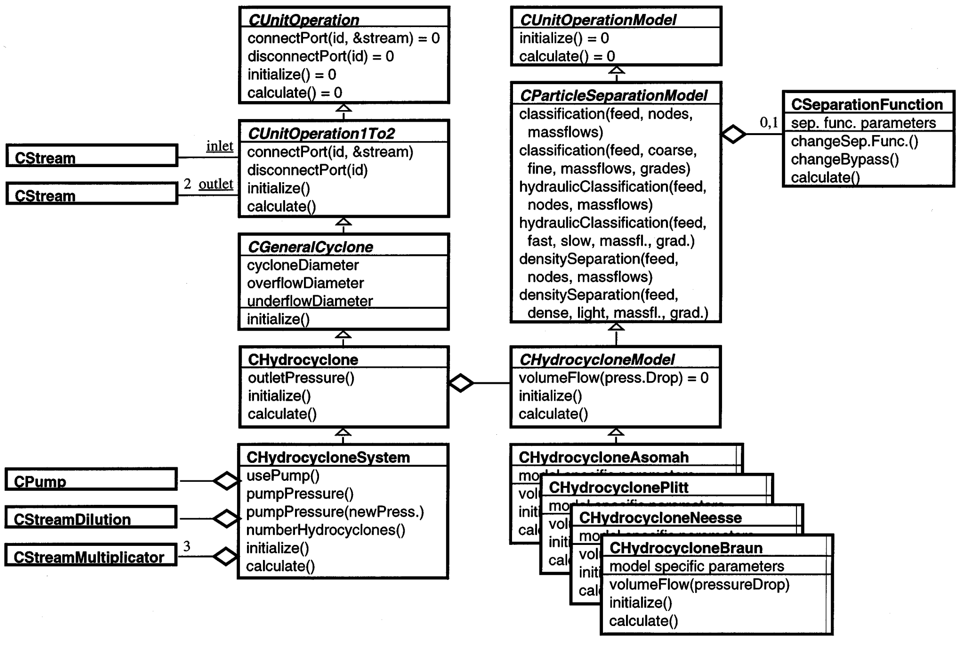

Fig. 5. UML-class diagram for the hydrocyclone implementation.

Table 2
Currently implemented models

Solid–solid separation and solid–fluid separation	
Separation function kit for classification, hydraulic classification and density separation: separation functions after Lilge/Plitt, Lynch/Rao, Neeße/Schubert, Plitt, Austin/Luckie, modified Plitt for screens (asymmetric) fish hook after Roldán-Villasana/Williams/Dyakowski	
Hydrocyclone: Asomah/Napier–Munn, Braun/Bohnet, Neeße/Schubert, Plitt	
Aerocyclone: Muschelknautz/Trefz, Mothes/Löffler, Lorenz/Bohnet	
Comminution	
Jawcrusher:	Gap size, Forssberg, size-mass-balance
Conecrusher:	Gap size, size-mass-balance with parameters by Whiten
Rollcrusher:	Gap size, size-mass-balance with parameters by Austin
Ball mill:	Bond, Size-Mass-Balance with parameters by Austin
Impact crusher:	Bond, size-mass-balance with parameters by Austin
Liberation in soil-washing	
General and high pressure water jet:	Wilichowski/Werther
Others	
Mixer, splitter, stream multiplication, stream dilution, pump, tear stream	

rationModel, which provides a separation efficiency curve kit. If a model calculates the grade efficiency at each node without assuming a predefined separation efficiency curve, such an association does not exist. CHydrocycloneSystem is derived from CHydrocyclone and aggregates a CPump, a CStreamDilution and three CStreamMultiplicators objects. This adds usual auxiliary features to the hydrocyclone resulting in a more comfortable model to the user (Toebermann et al., 1999).

3. First applications of SolidSim

3.1. Soil washing

Soil washing is a well-established method for remediation of contaminated soils. As much as possible of the soil is decontaminated to an extent that allows a reutilization or at least a disposal on easy terms. The highly contaminated residue is further processed or disposed as hazardous waste. Pre-treatment steps are followed by wet liberation where due to input of mechanical energy contaminated parts are partially liberated from the bulk of the material and transferred into the fines and the lights fraction. Less contaminated fractions are then separated from the highly contaminated fractions by classification and sorting; e.g. screens, hydrocyclones, flotation devices and spiral concentrators are used.

Crucial for the success of soil-washing is the contamination within different soil-fractions and the separation performance. An appropriate stream representation has at least one independent attribute, i.e. particle size, and particle size-dependent attributes for the contaminants and the particle density. They are necessary to describe the distribution of the contaminants within the size fractions and to predict the performance of usual separation steps, e.g. hydrocyclones and density separators. An example for a soil contaminated by mineral oil is given in Table 3. The dependent attributes are described by their mean value within a size interval, because determination of a distribution within each size interval was not possible (contamination by mineral oil) or required to much effort (particle density).

The examined pilot-scale soil-washing plant is illustrated in Fig. 6 and consists of a high-pressure water jet as wet liberation device, a wet screen and up to three hydrocyclone in series connected via the underflows.

Table 3
Example of a particle characterization for a soil-washing application

Primary attribute		Secondary attribute	
Size distribution		Mean value within size interval	
Size interval (μm)	Mass fraction (weight %)	Contamination by mineral oil (mg/kg dry matter)	Density (kg/m^3)
–45	4,70	73052.4	1957.2
45–63	0,82	20618.3	2405.8
63–100	2,53	8344.2	2542.2
100–180	11,07	1265.1	2628.2
180–355	41,70	376.8	2639.4
355–630	29,44	376.8	2639.4
630–2000	7,98	1184.3	2629.2
2000–4000	1,76	2449.4	2613.4

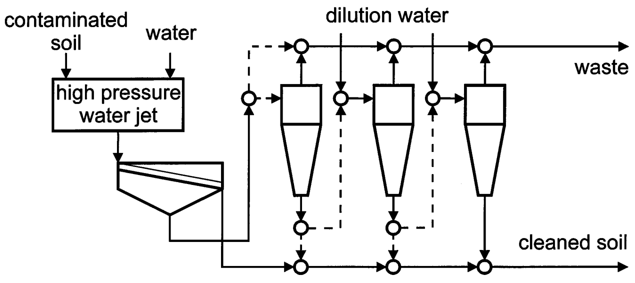

Fig. 6. Layout of pilot-scale soil-washing plant.

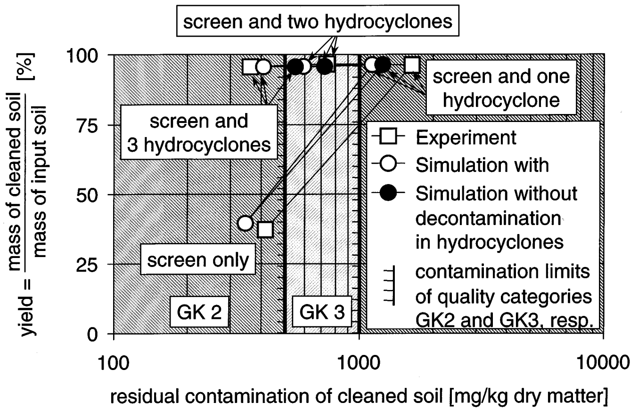

Fig. 7. Comparison of simulation and experimental results for soil NMS.

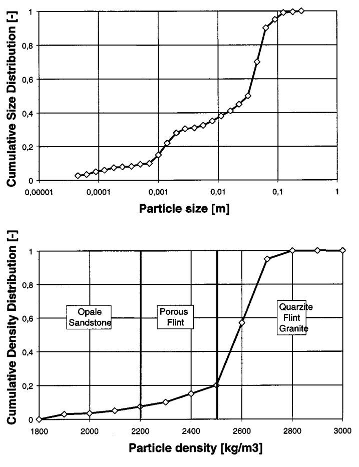

Fig. 8. Dispersity properties of raw gravel.

Correspondingly flowsheets with apparatus parameters given by Wilichowski and Werther (1996) were simulated with the soil characteristics given in Table 3.

At simulation start all streams are declared, but only the stream structure of the feed is initialized with the soil data, with water as fluid component and with the fluid and solids massflows. The stream structure of all other streams is dynamically defined during the unit-operation calculations, that is a unit-operation newly defines the stream structure of its outlet streams. The high pressure water jet model modifies the fractional values of the contamination attribute, the screen model separates its solid inlet stream according to the particle size information and the hydrocyclones model separates its solid inlet stream according to the fractional settling velocities derived from the particle size and the fractional particle density information. At the mixing points the information of the inlet streams are merged. In this simple example only the mass fractions of the particle size and the fractional contamination must be calculated, because the attribute structure of the feed propagates through the flowsheet during the simulation and remains unchanged. In more complicated cases the merging may fail due to incompatible stream information. The simulator would then terminate and the user must solve the problem by redefining one or more feed streams.

A comparison of the simulation results and the experimental data, Fig. 7, shows that the decontamination potentials of the process alternatives were properly predicted and that the simulation results are reliable within the required accuracy for soil-washing. In this case three hydrocyclones in series are necessary for an acceptably high yield and a residual contamination of cleaned soil according to German-regulations, that is less than 500 mg/kg dry matter.

3.2. Industrial gravel processing

The production of sand and gravel for road and construction purposes involves a number of usually wet processing steps to meet standard product specifications and qualities. Besides screen classification to separate the different product sizes additionally washing and sorting processes are required to remove organic matter (wood and coal) and sometimes alcali-reactive impurities (minerals of low density like porous flintstone) from the natural bulk material. Furthermore cohesive silt and clay fractions may have to be separated from the coarse material as they negatively affect the concrete strength. Depending on the deposit also crushing of oversize material might become necessary to improve the yield of valuable fractions. To properly define the material properties for a simulation run several solids attributes are therefore necessary.

For example gravel from the northern or north-eastern part of Germany is characterized by high fractions of material larger than 63 mm as well as alcali-reactive impurities of porous flintstone and opale sandstone. Comparison of the different mineral species shows that the density of alcali-reactive material is significantly lower. Particle density therefore is an appropriate dispersity property to be used in the sorting process and its simulation. During the calculation of the sorting and classification steps particle size distribution as well as particle density distribution are relevant as shown in Fig. 8.

Fig. 9 illustrates the corresponding SolidSim process stream information structure suited to the description of this type of gravel. Besides the fluid component 'water' the single solids type 'gravel' is defined, which is characterized by the two distributed attributes 'particle size' and 'particle density' on the first level of the attribute structure. If necessary a 'particle size' dependent attribute 'fines fraction' is used on the second level to add information on the amount of very fine particles occuring as adhesive material in the coarse fractions.

When performing a SolidSim simulation the data stored in this stream structure are used to analyze if the product properties follow the specifications given by the German standard of aggregates for concrete (DIN4226, 1983). Besides several requirements regarding the shape and position of the particle size distribution (maximum mass undersize, maximum mass oversize within one size fraction) the standard especially restricts the amount of impurities. Fig. 10 shows an appropriate plant design for producing gravel and sand from the type of raw material described above. The synthesis problem has been solved by applying a structured set of heuristics that limit both process structure and choice of apparatus type.

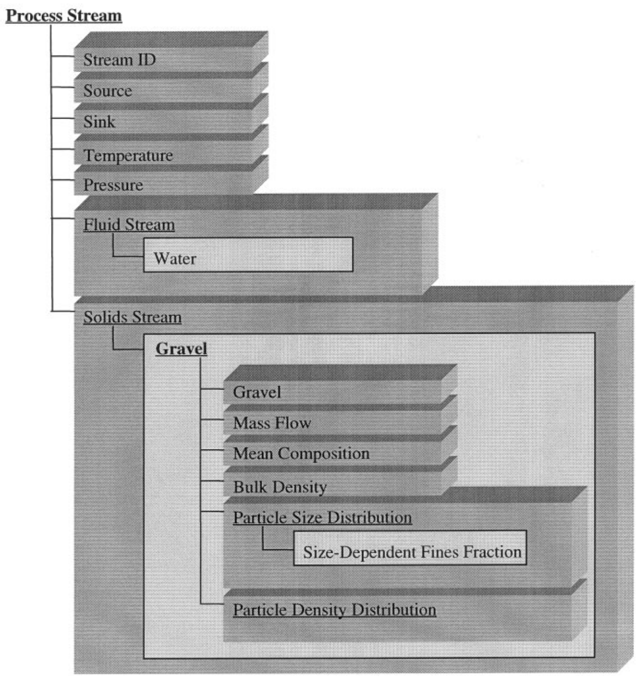

Fig. 9. Process stream information structure for gravel.

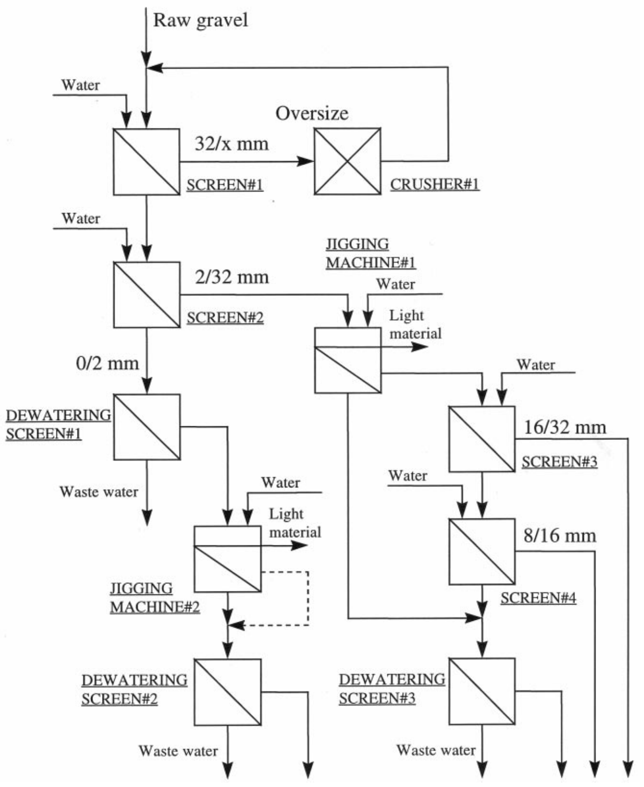

Fig. 10. Design of a gravel processing plant

4. Conclusions

Special requirements of solids processing for process streams and model library design were identified and their implications for the design of a block-oriented process simulator discussed. An appropriate simulator has been developed, which handles several solid properties simultaneously. Consequently it enables the simulation of complex solid processes, which is not possible with available commercial simulators. The results demonstrate that the simulator is an adequate and reliable tool. The object-oriented software design facilitates subsequent development, which is going to be extended in solid-specific areas.

Acknowledgements

The authors thank the Volkswagen-Stiftung, Hannover, Germany for financial support of this work.

References

Aspen Technology. (1988). *Aspen plus solids manual*. Cambridge.

Barton, G.W., & Perkins, J.D., (1988). Experiences with SpeedUp in the mineral processing industries, *Chemical Engineering Research Design* 66, 408–418

Beßling, B., Lohe, B., Schoenmakers, H., Scholl, S., & Staatz, H., (1997). CAPE in process design, *Computers & Chemical Engineering* 21; S17–S21

Booch, G. (1994). *Object oriented design analysis and design* (2nd ed.). Benjamin-Cummings, Redwood City.

Broussaud A. (1988). Advanced computer methods for mineral processing. In: E. Forssberg, *Proceedings of the XVI international mineral processing congress*, *Part A* (pp. 17–44). Elsevier, Amsterdam.

Clift, R. (1996). Powder technology and particle science. *Powder Technology*, *88*, 335–339.

DIN4226. (1983). Zuschlag für Beton. Beuth Verlag, Berlin (in German).

Gruhn, G., Rosenkranz, J., Werther, J., & Toebermann, J.-Chr. (1997). Development of an object-oriented simulation system for complex solids processes. *Computers & Chemical Engineering*, *21*, S187–S192.

Hill, P. J., & Ng, K. M. (1997). Simulation of solids processes accounting for particle-size distribution. *American Institute of Chemical Engineering Journal*, *43*, 715–726.

JKTech. (1998). Product information JKSimFloat and JKSimCoal. http://www.jktech.com.au/jktech.

Jones, G. L. (1984). Simulating the effects of changing particle characteristics in solids processing. *Computers & Chemical Engineering*, *8*, 329–338.

Lynch, A. J. (1977). *Mineral crushing and grinding circuits*. Amsterdam: Elsevier.

Napier-Munn, T. J., & Lynch, A. J. (1992). The modelling and computer simulation of mineral treatment processes. *Minerals Engineering*, *5*, 143–167.

Pantelides, C. C., & Oh, M. (1996). Process modelling tools and their applications to particulate processes. *Powder Technology*, *87*, 13–20.

Peukert, W. (1996). Trends in solid's process engineering, *Chemie Ingelheim Technische*, *68*, 1254–1263 (in German).

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., & Lorensen, W. (1991). *Object-oriented modeling and design*. Prentice Hall, Englewood Cliffs.

Rumpf, H. (1990). *Particle technology*. Chapman & Hall, London.

Schuler H., (1995). *Prozeßsimulation*. VCH, Weinheim (in German).

Toebermann, J.-Chr., Werther, J., Rosenkranz, J., & Gruhn, G. (1999). Flowsheet simulation of soil-washing processes, *Chemistry Engineering Technology* (in press).

Werther, J., Gruhn, G., Toebermann, J.-Chr., & Rosenkranz, J. (1997). A flowsheet simulation system for mineral processing and its application to soil-washing. In H. Hoberg, H. von Blottnitz, *Proceedings of the XX international mineral processing congress*, vol. 5. (pp. 661–671) Aachen, Germany.

Wilichowski, M., & Werther, J. (1996). Mathematical modelling of wet liberation, screening and hydrocyclone separation for the physical cleaning of soil contaminated by mineral oil. *Aufbereitungs-Technik*, *37*, 87–96.