Dirk van der Linden*

Implementing ISA S88 for a discrete process with the Bottom-Up approach

1. Introduction

Manufacturing operations can be generally classified into one of three different processes: discrete, continuous, and batch. On October 23, 1995, the SP88 committee released the ANSI/ISA-S88.01-1995 standard to guideline the design, control and operation of batch manufacturing plants [7]. The demand of the user's for production systems with a high flexibility and a high potential of making product variants, became important. Recipe based production was introduced, but it lacked clarity. In some way, this was caused by the fact that experts on different fields have to cooperate, but think in another way. Sometimes they use the same words but mean something else. Process engineers, for example, focus on how to handle the material flow to meet the specs of the end-product. Control system experts focus on how to control equipment [2]. To improve the cooperation of both groups, the SP88 committee had isolated equipment from recipes. This provides the possibility of process engineers to make process changes directly, without the help of a control system expert (reducing the setup-costs). This provides also the ability of producing many product-variants with the same installation (increasing the target market). Expensive equipment can be shared by different production units (enough reducing the production costs).

Diversification of products provides a faster action on the market. A fast evolution of international financial interests causes production sites to be transferred quite often from one financial group to another. In some cases, this leads practically to a multi-vendor environment within the same mother company. From a technical point of view, the best choice for letting equipment interoperate in a multi-vendor environment is the OPC standard [8–9]. Since 1996, the OPC specifications became a very successful communication standard on the field of automation. Interoperability became a lot easier and cheaper. On the other hand, integration is not achieved with software interoperability alone. The availability of raw data needs to be expanded with data to information conversion. Many production systems are built with custom made interfaces and data conversions. On short term, the end user is satis-

* Hogeschool Antwerpen, Department Industrial Science & Technology, Division Electromechanics, Antwerp, Belgium

fied with a working solution, but on the long term, the maintenance of this kind of solutions can be a headache. While companies are moving across the world-market, staff members are also. When maintenance is depending on specific staff members and documentation, the use of standard terminology and data models will be an improvement. Transfer of knowledge in case of a company adoption or a staff adoption becomes a key point.

By structuring maintainability improves. Maintenance can be organised more efficient because the identification of an exception can be linked to the most suitable break- down service. Moreover, the taking in service of a new installation or expansion can be planned more efficiently and the spin up time can be reduced. By using standard models and terminology, documentation becomes easier and the end users can better define their needs.

Another cost reducing possibility of an S88 structured installation is the ability of sharing expensive equipment. Several production units can share the same shared equipment following the physical model of S88.

Because of some new legislation and higher demands of production efficiency in general, the requirements of data collection became more important. Plantwide information management systems integrate all the process information islands and make this information available throughout the plant to support decision making at operational, management and executive levels. Raw data can be converted into information, information can be converted into knowledge and knowledge can be converted into profit as a result of better informed and smarter business decisions. The use of S88 data models facilitate the conversion of raw data to information.

The SP88 committee was focussing on batch control, but made the standard universal enough to make it suitable for other process types [4]. Large series in discrete production and a low diversity of end-products in continuous processes did not requested flexibility in the nineties, but recent evolutions changed this. Implementation of S88 for discrete processes are on the moment of the writing of this paper rather low. We wanted to know what points of interest are vital for discrete processes in comparison with batching. For this purpose, we realised a S88 implementation on a lab-scale discrete process with 2 robots, 8 conveyors and 4 pneumatic lift systems devices.

2. The application

The equipment of this application includes two robots which are repositioning wooden (letter-) blocks, and a rectangular conveyor system transporting those blocks on pallets. On every corner of the conveyor system there are lift-systems, because the conveyors are not running exactly at the same level (Fig. 1).

The user has to prepare a production run by placing an amount of blocks on the storehouse (right side of Fig. 1), and one or two pallets somewhere on the conveyors. Further, the user can start the application by choosing a recipe and to give a start command on an HMI or SCADA screen. The recipe driven application will use the pallets to move blocks from the storehouse (right) to the formation robot (left).

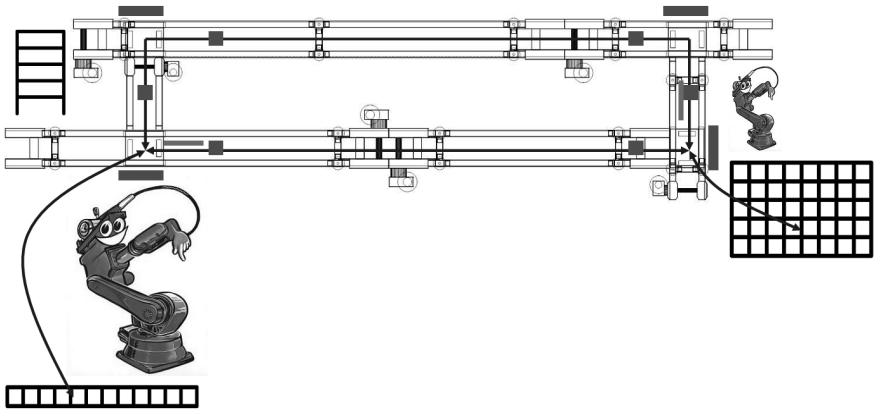

Fig. 1. The material flow represents the transport of blocks from the storehouse (right) to the formation robot (left) where the blocks are placed in a specific order

2.1. S88 models

The S88 standard provides a set of models (Fig. 2). We concentrate on the procedural model, the physical model and the process model.

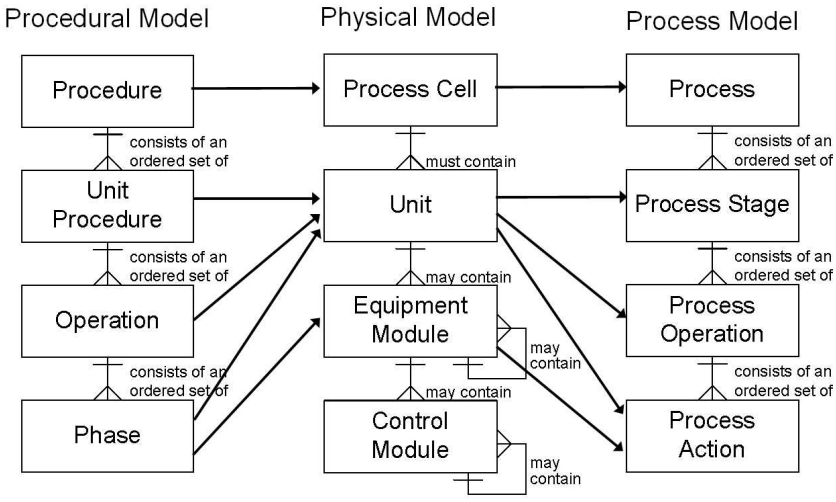

Fig. 2. The relation between procedural, physical and process model

Although the first target was batch control, the design of the models is universal, so that they can be used for continuous and discrete processes as well. A clear terminology makes communication between vendors, systems integrators and end-users better. With this

terminology, less misunderstandings are possible during discussions about the processing requirements. Data structures helps to integrate solutions in a multi-vendor environment. Parameter guidelines makes configuration of batch solutions easier.

S88 is not a compliance standard. It allows partly implementation, different levels of flexibility and complexity. In practice, this means that a lot of the features of a software product do not always have to be used. Vendors are often product-oriented, but to serve the end user best, you better be application-oriented. Structuring is good, as long as it serves the application. Besides, the majority of commercial software products has a licence policy which is based on the amount of tags. So implementing things the end user will not use, is not only a waste of time, it is also detrimental to the overall licence cost.

For our robot and conveyor application, we choose to use the procedure model, the lower parts of the physical model and the recipe model. We left the process model and the control activity model as a background theory, because these models are less significant when we focus on the difference between S88 implementation on discrete processes and more classical batch processes. The relation of the physical model and procedural model is a key point when we discuss the Bottom-Up approach, and informative we give the relation with the process model.

System integrators who are not aware of S88 often apply some intuitive kind of programming structure, because this can help the maintainability and reproducibility on short period. For large companies this may even develop into internal standards, which can have also very efficient models. Nevertheless the S88 models have the advantage that they are universal, which makes the necessary training of new employees shorter if they already know the basics of the standard. The models of S88 are also very qualitative because very much sectors and companies are represented in the SP88 committee. This heterogeneous group can make standard models more generic than a developer team in only one company.

2.2. Bottom-Up approach

In the ISA S88.01 standard the descriptions of the physical model and the procedural model are hierarchical, the highest level first, the lowest level last. When building a new factory from scratch this Top-Down approach might be correct. When one starts with existing machines and equipment, and the aim of the project is an enhancement of the control concepts and software, the Bottom-Up approach has many advantages. When one has to re-use existing equipment, a Top-Down approach might end in the necessity of the development of custom device drivers. In the worst case, the result of expensive software is a printout, and the operator will have to convert the abstract recipe to equipment commands. When using a Bottom-UP approach, the development has to deal right from the beginning with device drivers. The developer can choose standardized interfaces and gateways like OPC servers to access existing equipment. Besides the development can stop at several levels, and let the process run on an automatic, partly-automatic or manual way. The Bottom-Up approach is a guideline which parts of the S88 models should be used in a small application [3].

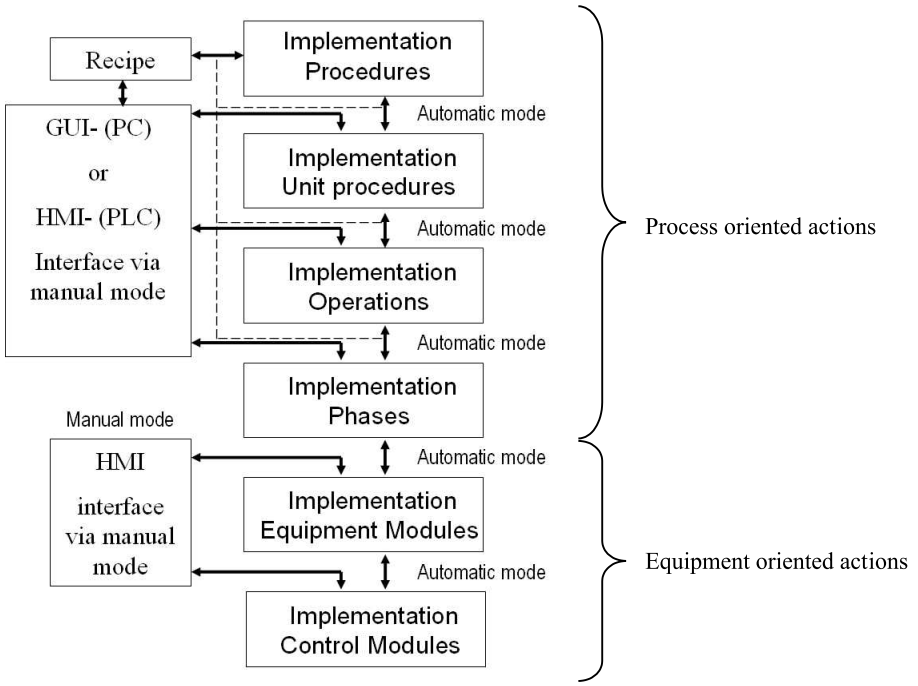

Fig. 3. Recipe driven automatic mode and manual mode as an alternative during development following the Bottom-Up approach

The application described in this paper is developed with the Bottom-Up approach. At every level there is an interface to operate the system manually. Every level is also a foundation for future implementation of higher level objects. The lowest level will be discussed first, the highest level last.

At the lowest level we have control modules. In the first place, control modules are device drivers. In this application, they also provide robust methods of device control to improve maintainability, take-in-service time, troubleshooting, and exception handling. These robust methods include automatic and manual modes, simulation mode, permissions and alarms (Fig. 3). In general, one should implement exception handling on the level where the exceptions are occurring [1]. Hardware exceptions like security curtains, mechanical obstructions etc are handled with the permission interface parameter on control module level. We implemented 5 types of Control Modules (CM). Each of them has a similar set of interface parameters.

In Figure 1 we can see a rectangular material flow of the pallets. It is logic to divide this rectangle in 4 sides. All the CMs of every side are combined in one Equipment Module (EM). Whereas we want to use this rectangular material flow both clockwise and counter clockwise, it is not obvious to which side of the corner the lift CMs belong. The easiest way do deal with this, is defining 4 extra EM for the lift equipment. The key point of an EM is providing features by combining CMs, whereas CMs are in the first place device drivers. So

a CM can be commanded on two ways. First in manual mode by the operator, and second in automatic mode by the EM from which the CM is member of.

Following the models on Figure 2, starting from the EM level, an interaction between the physical model and the procedural model is possible by implementing phases.

There is a conceptual difference between the basic control functions [5] of the physical elements and the procedural elements (Fig. 3). The functions of EMs and CMs are *equipment oriented*. You can for example start and stop a conveyor without any pallets on it. On the contrary, a procedural element is focusing on material flow (*process oriented*). If the target of a procedural element is to move a pallet, there will be an exception if the pallet has not reached its destination in a reasonable period of time.

In this context the lift phases are doing more than just going up from level 0 to level 1 and vice versa, they are also taking in consideration whether there is a pallet on the lift or not. If the lift is empty, the phase concerned is preparing the reception of a pallet. If the lift is full, the phase concerned is actually moving a pallet.

Figure 2 indicates that phases can be defined at EM level or at unit level. Operations are always at unit level. Before we discuss the operations, we should explain where the units are.

We have the storage unit, which includes a short conveyor EM and two lift EMs. We have also the formation unit, with a similar short conveyor EM and two lift EMs. We have chosen these names because it is the storage unit's task to load and store wooden blocks from a storehouse. On every side of these cubical blocks, a letter is printed. The task of the formation unit is forming a word with these letter-blocks.

In between these two units we have the long conveyors, which are common resources. Both units can use them to receive or send pallets.

In Figure 4 it is clear that every unit also has a robot included. These robots are equipment, but we don't call them EM because they do not fit in the S88 standard. These robots are controlled by a custom controller, custom software and have a custom interface. The custom software includes phases, but not accessible via an S88 suggested interface. In theory, it is possible to re-engineer the robot software and re-program the control software, but in practice this is not realistic and not economic. Following the bottom-up approach, these robots do have a sort of control modules (axes, grasper), they do have a kind of equipment modules (modes, statuses), and a kind of phases (teached positions and sequences). In this application we have encapsulated these custom features and we provided an S88 interface on operation level.

On this level, it is allowed to use recipe parameters, so we added some formula parameters (the from- and to- coordinates of the blocks). In this situation, it is a good idea to protect the equipment against bad formula parameters. If a robot places a block on another block's coordinate, then the robot crashes. To prevent this, the PLC can maintain a memory storing occupied positions. When a formula parameter is sending a block to an occupied position, the operation switches from automatic mode to semi-automatic mode. In semi-automatic mode the system is asking for a confirmation. If the operator acknowledge, this probably means he changed the position of some block manually, and then the operation

just can proceed. If the operator does not acknowledges, the operation is changed to the aborting state, where the block is left at his original position, and the operator reset the operation. As an alternative, it should be possible to validate recipes, but this is against the objectives of S88. A recipe should be abstract, without taking care of the limitations of the equipment.

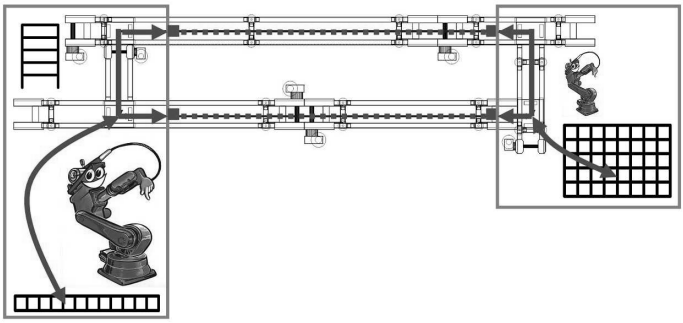

Fig. 4. Storage unit (right) and formation unit (left) are handling the material flow independent

On the unit procedure level, the option to send or receive a pallet with the front conveyor or back conveyor is chosen by a formula parameter whereas on operation level this was chosen by a different procedure (operation). For the robots, a sequence of operations will handle enough blocks to form a word or a part of a word.

In general, the commands become more abstract and some sequential logic is replaced by formula parameters.

At the top, procedures are used to control what is going to happen (which word is going to be formed, how many blocks are needed, how many and which blocks will be placed on a pallet, where the pallet is coming from, where it is going to, etc...)

Procedures are taking place at the process cell level. Since there are two units, two procedures can run simultaneous. An allocation system will ensure that one unit is working on one procedure at a time. Also, the use of common resources (large conveyors) is regulated by this allocation system.

2.3. Linking recipes to equipment control

The process model is abstract while the physical model is a real one. Therefore, the procedural model has both an abstract version and a real version. The recipe's procedural elements are abstract, the equipment's procedural elements are real . Which part of the procedural model is abstract and which part is real, is defined by the level where the recipe is acting on the equipment. The level of this link has a big influence on the flexibility of the system, but also on the complexity. The more flexibility, the higher the complexity. Developers should communicate with the end users, to know how much flexibility is needed, and realize an application which is not more complex then necessary.

The universal character of S88 allows implementations on discrete processes, but the speed of process control is crucial. Material handling on discrete processes quite often requires higher performance control in comparison with some slow (bio-)chemical processes in batch industry. Commercial phase-sequence software, that runs on a PC, is suitable for batching, but sometimes too slow to perform online phase control of a discrete process. This sometimes forces the link higher, because equipment procedural elements are typically controlled by a PLC, which has a faster performance.

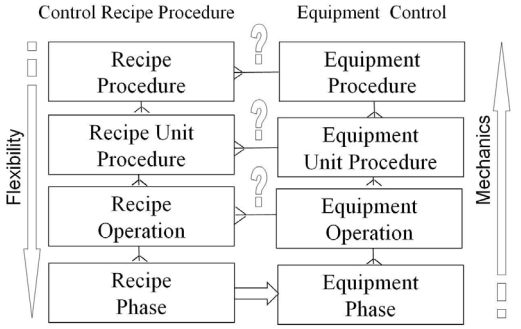

Fig. 5. The level of linking recipe and equipment control

S88 suggests that a batch management system (runs on a computer) and an equipment control system (runs on a PLC or DCS system) exist separately. The link of the recipe and equipment control is the connection between the PC application and the PLC application. For batching, the link between recipe and equipment control is made on phase-level. Typically control modules for batching are for example pumps, valves and temperature measurement, which can be combined to typically equipment modules like a material addition equipment modules (group of ingredient valves). Another example of an equipment module can be the combination of a pump and 2 valves (one before and another after the pump), which forms a material transfer equipment module. For a discrete process considered in this paper, it may be better to implement this link on a higher level (Fig. 5: on operation, unit procedure or even procedure level). This means in practice that procedural elements are implemented in a PLC, while a PLC is very common in factory automation (discrete processes). In former days, it was time-consuming to program a sequence in a PLC, but thanks to the IEC61131 standard [6] this has improved. With every PLC type that supports IEC61131 it is possible to program sequences with the SFC (Sequencial Function Chart) programming language on a straightforward way.

Figure 6 gives a detail of Figure 1. One EM is formed by 5 CMs. Note that the conveyor has two parts, with a separate motor each. The operator can control the motor CM manually by the parameters with prefix ‘M’(run command and direction), with the condition that the parameter ‘Mmode’ is true. In automatic mode (‘Mmode’ false), the parameters ‘Arun’ and ‘Adir’ are used, and controlled by the EM from which the CM belongs. When the operator uses for example a ‘Mrun’ command on the EM in manual mode, the EM controls both ‘Arun’ commands of the two motor CMs.

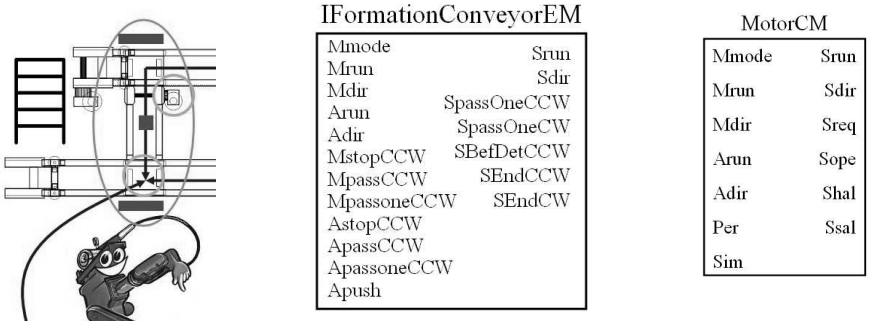

Fig. 6. Example of an EM: The Formation Conveyor Equipment Module, which is formed by 2 Motor CMs (circles), 2 rockerstop CMs (rectangles) and a stopgate CM (square)

In terms of communication the PLC programmer respects the interface parameters to communicate between the modules internally, and the SCADA programmer respects the interface parameters (OPC Items) to access the PLC subroutines (S88 modules).

3. Conclusions

Although the S88 standard is originally built for batching, it is universal enough for use in the three different types of processes. Because of this universal character, it is not smart to implement all the features and models, but to focus on an application-oriented design.

In batching applications, the link between an abstract recipe and the equipment is often on the phase level. For discrete processes it is better to make this link on a higher level. This is necessary to keep the recipe abstract, and to avoid interference with limits of the equipment. The higher the link, the less complex the recipe design will be. Also, lots of mechanical actions in discrete processes require a real-time character. The solution is to make the recipe link higher.

With the bottom-up approach, operating the installation is possible on every level. The higher the level, the less actions the operator needs to perform.

Exceptions can be handled on every level, but one should be aware of propagation. The best results are obtained by handling the exceptions on the level where they occur. Break downs will become more predictable and troubleshooting will become easier.

The efficiency of both recipe control and data captation improves with the use of standard data models, which prepares the conversion of raw data to information.

By combining the standards IEC61131, OPC and S88, it is possible to structure totally integrated projects, with cross-vendor interoperability and a close interaction of vendors, system integrators and end-users.

And last but not least, application-oriented thinking is important. Representatives of vendors think product-oriented, because it's their job to sell. Engineers who develop applications should use the S88 standards as a guideline, but always in function of what the end-user needs. The S88 standard should not be used as a target, only as a tool.

References

[1] Applying S88: *Batch Control from a User's Perspective*. Jim Parshall and Larry Lamb, Printed in the USA, August 2005, ISBN 1-55617-703-8.
[2] *S88 implementation guide: strategic automation for the process industries*. Darrin W. Fleming, Velumani A. Pillai, Printed in the USA, 1999, ISBN 0-07-021697-5.
[3] *Applying ISA S88 to Small, Simple Processes*. Clark Case, Rockwell Automation, World Batch Forum conference 13–15 nov 2006, Zemst, Belgium.
[4] *S88 for Researchers and Scientists*. Zofia Verwater-Lukszo, Delft University of Technology. White paper written for World Batch Forum June 2004.
[5] *Using Basic Control to Make Recipes Simple*. Francis Lovering, ControlDraw Ltd, World Batch Forum conference 13–15 Nov 2006, Zemst, Belgium.
[6] IEC 61131-1 Ed. 2.0 en: 2003, *Progammable controllers – Part 1: General information*.
[7] ANSI/ISA-S88.01-1995, *Batch Control, Part 1: Models and Terminology*. Printed in the USA, 1995, ISBN 1-55617-562-0.
[8] *OPC Data Access Custom Interface Specification 3.0*. OPC Foundation, March 2003.
[9] *OPC Fundamentals, Implementation, and Application, 3rd rev*. Frank Iwanitz, Jürgen Lange (Eds.), Printed in Germany, 2006, Hüthig GmbH & Co. KG Heidelberg, ISBN 3-7785-2904-8.