Modeling and Optimization of Synchronous Behavior for Packaging Machines

Sathyamyla Kanthabhabhajeya, Bengt Lennartson

Automation Research Group, Department of Signals and Systems, Chalmers University of Technology, SE-41296 Gothenburg, Sweden (e-mail: satkan@chalmers.se)

Abstract: This paper proposes a new modeling solution for the synchronous behavior of packaging machines, and a strategy for maximizing the production rate based on a formal model. A common modeling platform is recommended to handle information exchange and to develop a collaborative workflow, in this paper involving mechanical design and software development. The modeling solution for the synchronous behavior is developed in SysML (Systems Modeling Language), being the common platform. Then a formal modeling language called Sequence Planner Language (SPL) is interfaced with SysML, to overcome some limitations of SysML. The synchronous behavior of the packaging machine is also developed in SPL, from which the optimization problem is defined. The result of the optimization shows that it is possible to improve the efficiency of packaging machines with new configurations compared to more conventional design. The proposed strategy is evaluated for a filling machine at Tetra Pak.

Keywords: SysML, Formal Modeling, Optimization, Synchronous behavior, Workflow

1. INTRODUCTION

The complexity in manufacturing industry has increased due to introduction of partial automation and enormous information exchange. Though technological developments are intended to provide a simpler manufacturing environment, it complicates the design and control of manufacturing systems. This complex design and control increases the design cost, time and maintenance. Design of complex manufacturing systems is often based on a virtual model of the system. A high-level abstract model of the system is then developed during the design stages. Abstract modeling, based on a high-level architecture, gives an overview of the manufacturing system. However, a methodology is required to use this model during different stages of the design. This model of the manufacturing system also needs to be accessed by different domains in the industry.

A workflow of any manufacturing industry therefore includes different domains working in a collaborative way. One example of such a collaborative workflow is illustrated in Figure 1, where mechanical design and software development are integrated, sharing common information. A more specific example, also illustrated in Figure 1, is control design and implementation in a PLC controller that need to be integrated with the physical system design, involving both plant dynamics, actuators, and sensors. In Kanthabhabhajeya et al. (2012), this collaborative workflow is related to more detailed development & information models, such as the V-model, the Waterfall model and Product Lifecycle Management.

The need to handle information efficiently, and to reduce the commissioning time, lead to the requirement of a common modeling platform, as described in the collaborative workflow. Automation in manufacturing industry uses different software applications for different domains. This necessitates manual intervention for exchange of information. Different domains in the manufacturing industry include process design, mechanical design, software engineering, electrical system design, environmental demands, production, sales and marketing, etc. The information from these domains is dependent on each other. The information is stored in different formats and in different software applications, thereby complicating the information exchange.

Therefore, Systems Modeling Language (SysML) (Friedenthal et al. (2012)) is proposed as a common modeling platform. Kanthabhabhajeya et al. (2012) explain how SysML fits well as such a platform in a collaborative

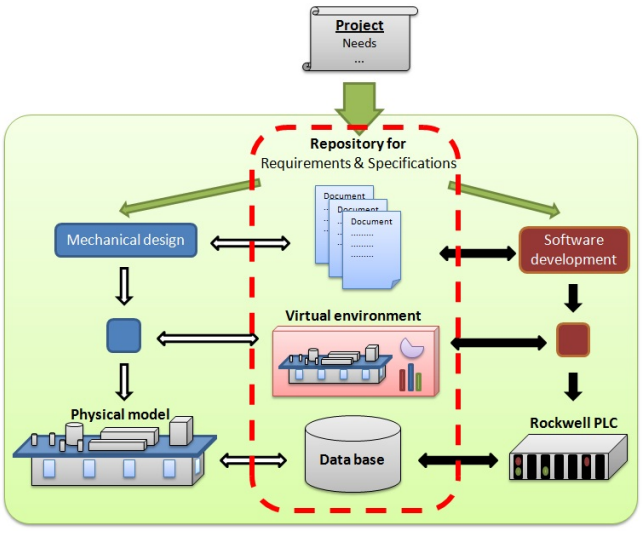

Fig. 1. Collaborative workflow including integrated mechanical design and software development.

workflow involving mechanical design and software development. SysML is also preferable as an interface between different applications, which helps to improve exchange of information between different domains. Furthermore, it is mentioned in Qamar et al. (2011) that SysML is a common modeling language between other modeling languages, because it supports strong relations between domain-specific models and generic system models.

SysML, being a graphical modeling language also has the possibility to model structural and behavioral constructs of a manufacturing system. The architecture of SysML includes, "structures to domain concept, mapping of domain concepts to language concepts, and instantiation and representation" as described in Friedenthal et al. (2012). Nine diagrams in SysML intend to satisfy the requirements of System Engineers and facilitate the construction of a graphical model of a manufacturing system, see OMGSpecification (2010). SysML has also been used as a modeling framework to describe the integration of a new execution system with an existing one, see Pietrac et al. (2011).

In this paper, the modeling platform based on SysML is used for integrated mechanical and software design, focussing on the packaging industry. To obtain a common model, it is required to understand the complexity of the packaging industry. Industrial automation in the last two decades has replaced the mechanical shafts, cams and motors, with electric drives and servo motors controlled by Programmable Logic Controllers (PLCs) within this industry, see Bassi et al. (2011). However, this transition still imitates the former mechanical behavior, by replacing the mechanical parts with mechatronic devices.

Mathematical models of motion profiles are defined to control the servo motors in master-slave relations. The motion profiles are defined by choosing an acceleration profile fitted to the position points of the servo motors against a shaft angle $(0-360°)$. The motion profile defines velocity curves for the slave motors and hence generates a synchronized motion between different behaviors. Both mechanical and software design need a common model to represent this synchronous behavior. A modeling solution representing this synchronous behavior is therefore developed in this paper based on SysML.

Though SysML acts as a common modeling platform in the workflow, it is only a semi-formal language. Its structure is not suited for formal verification, as argued by Bassi et al. (2011). Hence, a related but more formal modeling language, called Sequence Planner Language (SPL), is interfaced with SysML for verification of the model developed in SysML. The framework of SPL is based on sequences of operations and also includes a formal relation between products and processes, see Lennartson et al. (2010) (where SPL is referred as SOP - Sequences of Operations). Kanthabhabhajeya et al. (2012) classify the advantages of SPL over SysML and the need for this interface.

The behavioral functions of the machine are formally defined using SPL. This formal mathematical model also includes necessary information for optimization. Furthermore, SPL adds to the value by providing visualization of different projections of the same model such as product view, resource view, operator view, etc. An interface between SPL and SysML is therefore implemented, and the results are presented in Kanthabhabhajeya et al. (2013).

The main contributions of this paper are as follows: A modeling solution for the synchronous behavior of a packaging machine using SysML is proposed. This modeling solution is analyzed in a case study of a filling module of the TR/28 machine, from Tetra Pak, a global company delivering packaging solutions. In addition, this paper develops an SPL model of the complete TR/28 filling machine. An optimization problem, to maximize the production rate, is then formulated by the SPL model and solved using existing solvers. The result of this optimization shows that it is possible to improve the efficiency of the TR/28 machine compared to the current configuration of the machine.

2. SYSTEMS MODELING LANGUAGE

SysML, as described in Section 1, is a general purpose modeling language, extended from UML and has nine diagrams to represent the model of a system. These diagrams are shown in the SysML taxonomy in Figure 2, which also differentiates between the diagrams taken directly from UML and those partly or completely added as new ones. In a broader perspective, Structural diagrams are used to represent a system composed by its parts, interconnections, properties and constraints. Behavioral diagrams are used to describe the flow of parts, relations between user and model, and also description of functions with state transitions. In this section, a few of the diagrams used in the case study model (Section 4), are described in detail, c.f. Friedenthal et al. (2012).

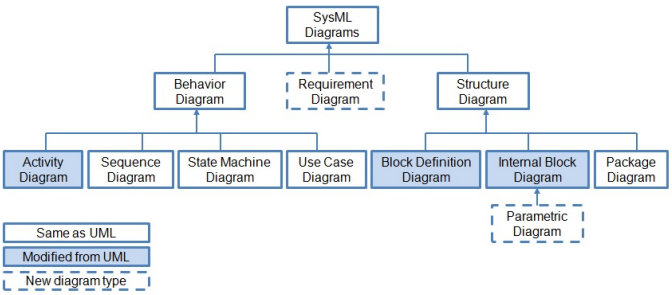

Fig. 2. SysML Taxonomy.

2.1 Structure Diagrams

Among the structure diagrams in SysML, the Block Definition Diagram (BDD) is used to represent the system's hierarchy by defining composition and/or classification of modules as well as parts of the system. The BDD includes blocks, to which the connections are exhibited with composite association, reference association or generalization path. A block defines the modular unit of a structure, with uniquely identifiable instances. A block in a BDD has compartments to add different properties like Parts, References and Values. As the name suggests, the Parts compartment includes the parts that compose the respective block. Reference property of a block is used to describe whether the current block has any reference to another block. The Values compartment includes the value

properties of a block, and hence relates to the physical properties of the real system.

With the overall description of a system in a BDD, the details of parts, item flows, interconnections, allocations, etc, are defined in an Internal Block Diagram (IBD). Avoiding the details on IBD, the Parametric Diagram (PD) in SysML relates the model with the real system and supports engineering analysis. This diagram coordinates the constraint and value properties of the blocks defined in BDD. A constraint property is added either as a separate constraint block or in the compartment of the respective block in BDD.

PD defines the details of binding connections between the added constraints and their parameters, along with the values of the block. This feature used in the modeling of the synchronous behavior of filling machines in Section 6. Time dependent constraints need a time property, which can be expressed explicitly as a reference clock property with its unit and quantity defined. This is also applied in the synchronous modeling in Section 6.

Finally, we observe that the Package diagram of SysML helps to define the hierarchy of the model of the system.

2.2 Behavior Diagrams

Among the behavior diagrams in SysML, the Activity Diagram (AD) involves actions to represent a particular behavioral function of the system. The actions are interconnected according to the flow of items and flow of control in the real system. The AD of SysML is influenced by Petri Nets, see Reisig (1992), which involve tokens generation and consumption to represent the flow of control through an executable action. An action in an AD needs at least one token to be consumed at the input to execute, and generates new token(s) at the output. The relation between actions is expressed by fork-join, decision-merge nodes.

An action can be described in detail with a State-Machine Diagram (STM) including states, transitions and events. This diagram is similar to any classic representation of behavioral modeling with states and transitions. As observed in SysML taxonomy, there is no hierarchical structure between behavioral diagrams, but in the actual usability of the STM diagram, it is typically initiated by an action in an AD.

The Sequence Diagram in SysML describes the interaction between the system and the user, while the Use Case Diagram describes the functionality of a system with respect to different users' goals of a system. These two diagrams represent users as actors who interact with the system. The latter define in detail the outcome that the actor needs to achieve.

Finally, the Requirement Diagram in SysML lists the specification of the system that needs to be modeled. This diagram includes text, tables and keywords to add meaning to the requirement list. The case study in this paper involves the BDD, PD, AD and STM of SysML to generate a model of the system, which is later analytically executed.

3. SEQUENCE PLANNER LANGUAGE

The Sequence Planner Language (SPL) is a graphical modeling language that models hierarchical operations and sequences of operations, formally defined by automata extended with variables, see Lennartson et al. (2010). By introducing this type of formalism and reusable information, there exists a potential to obtain automated design, optimization and correct implementation of embedded control functions. The SPL, its graphical structure and properties are now explained by an illustrative example, presented in Figure 3.

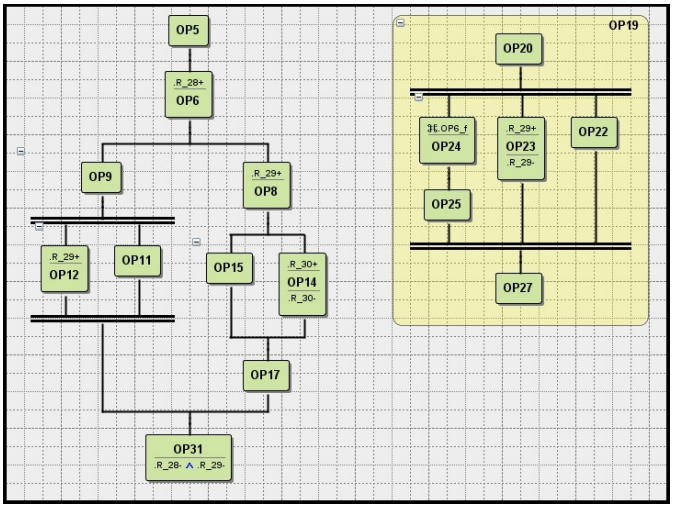

Fig. 3. Illustrative example of an SPL.

Three-state operation model An SPL operation is a three-state model, represented as an automaton with variables, as shown in Figure 4. The three state model defines whether the operation is currently in its *intial, executed or final* state. An operation O_k moves from its initial state O_k^i to its execution state O_k^e when the precondition $C_k^{\uparrow}$ is satisfied and the event $O_k^{\uparrow}$ occurs, and it moves from its execution state O_k^e to its final state O_k^f when the postcondition $C_k^{\downarrow}$ is satisfied and the event $O_k^{\downarrow}$ occurs.

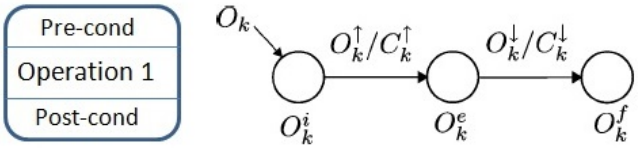

Fig. 4. Three state model of an operation.

Self-contained operations Though this formal model has a graphical structure similar to Sequential Function Chart and Grafcet, as can be seen in Figure 3, every operation is *self-contained*, since it holds all necessary information to determine its own state but also, together with other operations, to determine their relation to each other. The relation between a number of operations is often a straight sequence, but may also include other relations, see below.

For example, $OP24$ in Figure 3, contains information on its pre-condition that $OP6$ has to be finished, but also

from the graphical relations that $OP20$ must be completed before $OP24$ can be executed. This allows the operations to be grouped and viewed automatically from different perspectives, see more below on visualization.

Resources & Operations Some of the operations include allocation of necessary resources. SPL uses simple variable notations to book and unbook the resources. In Figure 3, $OP12$ books the resource $R29$ in its pre-condition (R_{29}^{+}) and unbook it (R_{29}^{-}) after executing the operation $OP31$. Similarly, $OP23$ also requires the same resource $R29$, as $OP12$. This implies that operations $OP12$ and $OP23$ cannot be executed at the same time, provided that the availability of the resource $R29$ has capacity one. This expresses the mutual exclusion between the two operations, due to the shared resource $R29$.

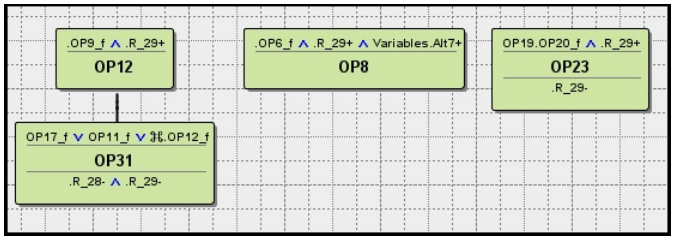

Fig. 5. Resource view for resource R_{29} based on the SPL in Figure 3.

Relations The identified relations between different operations, which can be expressed graphically in SPL, can be divided into five categories. Sequence, Parallel, Alternative, Hierarchy and Arbitrary order are the relations between operations that define the flow of processes/products in an SPL model. An operation that can only be started after another operation is finished exhibits a sequence relation between the two operations. $OP8$ and $OP9$ in Figure 3 are alternative operations, where only one of them is executed, provided the corresponding pre-condition is satisfied. The parallel relation between $OP11$ and $OP12$ represents that both operations need to be executed and be in their final state, before the following operation $OP31$ begins. A highest hierarchy is represented by operation $OP19$ in Figure 3 with $OP20$ being the initial operation of the sequence within $OP19$. The operation $OP19$ is considered to be completed first when $OP27$ has reached its final state. The arbitrary order combines the parallel operation and mutual exclusion, where operations are executed in parallel but not at the same time and in arbitary order. A detailed description of the relations between operations in SPL is available in Lennartson et al. (2010), Bengtsson (2012).

Visualization SPL allows the sequences of operations to be projected into different views with respect to resource, product or other perspectives such as safety, involvement of operator, etc. The example presented in this section, uses different resources for executing some of the operations. The product view of this sequence of operations is shown in Figure 3. SPL has also the facility to view the sequences of operations with respect to a particular resource as shown in Figure 5. The operations related to resource $R29$ are $OP12$, $OP8$, $OP31$ and $OP23$. These operations include their relations to each other, but also to other operations in their pre-conditions. Such different views can be automatically generated, see Bengtsson (2012), based on the self-contained operation information mentioned above.

SysML - SPL Interface For SysML, being a common modeling platform, but semi-formal, the behavioral part of the model is interfaced with SPL. The required information from the behavioral part of an SysML model is mapped across SPL, using built-in methods and additional algorithms. The purpose of this interface is to make use of the benefits of SPL over SysML, c.f. Kanthabhabhajeya et al. (2013)

4. FILLING MODULE OF PACKAGING MACHINE

The filling module of a packaging machine (Tetra Rex TR/28) is modeled in SysML and SPL in Kanthabhabhajeya et al. (2012). This filling module of the TR/28 machine uses a *Lift* and a *Pump* to perform the operation of filling a carton. The SysML model of this filling module is now extended with the motion profile along with synchronous behavior of the machine. The filling module is divided into two sections: One section includes the lift and the servomotor for the lift, while the second section includes the pump and valves for filling the liquid in the carton. The cartons are fed to the filling module through an indexing conveyor. The *Lifts* move the cartons up. When lowering the cartons, the *Pumps* fill the liquid. The movements of the *Lifts* and *Pumps* are synchronized such that the filling process begins as soon as the lift along with the carton starts to move down. The filling module with the lift moving down and the pump filling the liquid are shown in Figures 6 and 7.

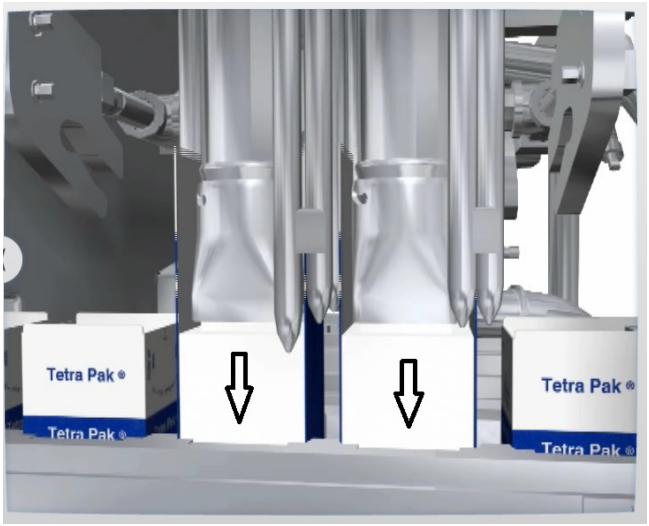

Fig. 6. Filling module of packaging machine, see TetraPak-Engineers (2007).

On a more detailed level the lift is defined with an UP and DOWN movement. The lift needs to move UP $250mm$ in $100ms$, while it takes $400ms$ to move DOWN. The acceleration stays negative (i.e, below $9.81m/s^2$) while the lift is moving DOWN. This negative acceleration makes sure that the carton follows the lift. The cyclic behavior of this machine is considered to have a fixed machine cycle time of $1000ms$.

The pump filling stroke is required to move $150mm$ in $400ms$ to fill the required product in the carton, while the lift is reaching the bottom level. The pump profile is relative to sustain the dip level (as shown in Figure 6) constant. The return stroke for the pump is at the desired position before the next cycle begins, as shown in Figure 7. This filling module has a batch size of two cartons, i.e, two cartons are filled at the same time. These cartons are filled with two lifts and two filling pumps following the exact motion profiles, respectively.

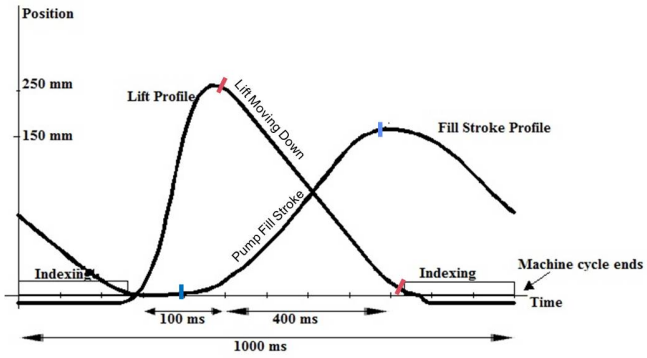

Fig. 7. Lift-fill profiles of filling module, see TetraPakEngineers (2007).

An *Indexing Conveyor* transports the cartons to different modules through the machine. The machine cycle determines the indexing and halting time of the conveyor. As seen in Figure 7, *Indexing* (moving the conveyor) occurs during the beginning and end of the filling module. This cycle is repeated with the next two empty cartons. The cartons are filled when the conveyor is halted. The duration of indexing the conveyor depends on the batch size of the filling module. The halting duration of the conveyor will not change, with the assumption that the number of resources (*Lift*, *Pump*, etc) proportionally increase with respect to the batch size. Based on the SPL model of the filling module, this model is extended by including the other modules of TR/28 like Folder, Bottom Sealing and Top Sealing.

5. MOTION PROFILE PRINCIPLES

Earlier generations of packaging machines were composed of mechanical shafts, cam wheels and indexing gearboxes, as shown in Figure 8. Motion control of these machines were managed by controlling the motion of the main shaft with respect to the design of the cam wheels and gearboxes. The human interface to these machines, in normal operation procedure, was only used to start/stop the main motor, while the position of the shaft and cam wheels were read by the encoders and fed as inputs to a PLC. The necessary input/output events were triggered by the PLC with respect to the position of the shaft. It was the cam wheel that delivered the necessary position and velocity of the respective moving parts in the machine.

In modern packaging machines, the principle of the old mechanical system is duplicated with a single master and many slaves designed by servo motors. The main shaft with different cam wheel profiles is introduced as mathematical models in virtual environment. In motion control

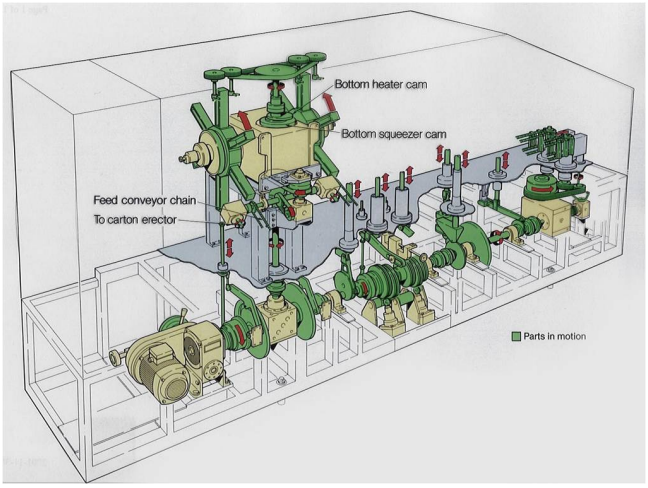

Fig. 8. Packaging machine with mechanical solution, see TetraPakEngineers (2007).

systems, controllers in closed loops are used to control the velocity and position of machines in order to generate desired motions, see Nguyen et al. (2008). Polynomial equations representing the cam wheel profiles are used for control purposes. Different slaves from multiple axes are controlled with respect to their particular profiles. There is no feedback control in normal operation, hence the open loop motion control is expected to perform in a correct way. Error handling is taken care of by the controller that performs an exception event when a fault is induced.

Motion control is a key concept in many manufacturing applications, where the precision of position and velocity of moving parts is critical. Nguyen et al. (2008) states that s-curve velocity profiles have the tendency to reduce vibration and achieve better precision. A pulse and sinusoid are combined to form the acceleration profile for Tetra Pak. Desired position points, depending on the machine, determine the velocity profile, considering the specified acceleration profile. These defined curves have their respective polynomial equations to be used for motion control of the slaves.

A mathematical model of the profiles of *Lift* and *Pump* (as shown in Figure 7) is determined and used for extending the model of TR/28 in SysML. Although the behaviors of the *Lift* and the *Pump* is controlled individually, the synchronized behavior of both parts yield the required outcome. The velocity profile is utilized for controlling the respective moving parts of the machine. The modeling of this synchronous behavior in SysML is explained in Section 6.

Motion Profiles - Optimization For the *Indexing Conveyor*, simplified motion profiles are determined with respect to the *Batch Size* (S) and *Machine Cycle time* (mc). The position curve decides the duration of the displacement of the conveyor and halting time of the conveyor. This duration is directly proportional to the number of cartons in the respective batch and the width of each carton. A trapezoidal velocity profile generates a linear and a constant slope. A proportion of time called *formfactor* (ff) that falls as the constant velocity is determined by experience at Tetra Pak. This can also be considered as an

optimization problem, which we consider for future work. In this paper, the optimization problem, maximizes the *Production Rate* (PR) with constraints on the *Acceleration* (a). Replacing the trapezoidal velocity profile with more exact velocity profiles in the optimization of the indexing conveyor will also be considered for future work.

6. MODELING SYNCHRONOUS BEHAVIOR IN SYSML

The SysML model for the filling machine in Kanthabhabhajeya et al. (2012) includes block diagrams of the filling module along with the flow of items between different parts. An activity diagram describes the sequence of flow of actions that are necessarily performed by the blocks of the filling module. In this paper, the synchronous behavior among the elements of the module, is also taken into account. The model of the filling module is also divided into two sections as upper and lower sections, similar to the actual machine (as described in Section 4). The main filling module, along with the pump, the supply unit with valves together with the tank assembly, form the upper section, while the lower section includes the carton lifter with lift, gripper and cleaning module.

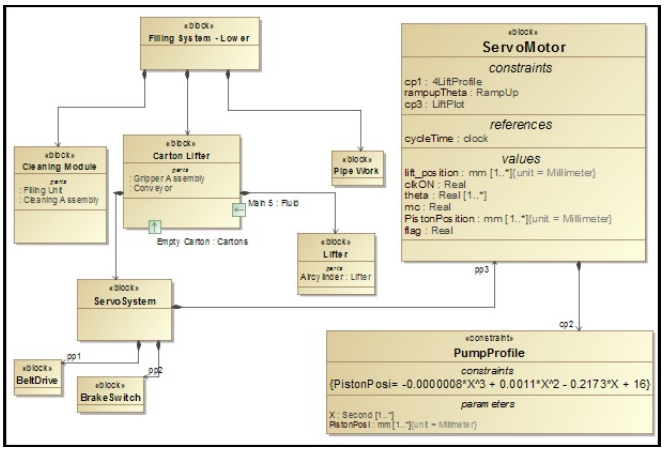

Fig. 9. BDD of filling module, including constraint properties of ServoMotor.

Constraint properties The motion profiles of the *Lift* and the *Pump* in the filling module, as given in Section 5, are represented by a 3^{rd} order polynomial equation as shown in Figure 9. In the lower section of the filling module model, a constraint block is added to include the constraint property of the Lift profile as described in Section 2. This constraint block describes the 3^{rd} order polynomial equation of the respective motion profile. This constraint property is composed to the ServoMotor, which is a part of the block CartonLifter. This ServoMotor is responsible for the control of the Lift's speed and/or position with respect to the pre-defined motion profile. The necessary parameters along with the value types, of the constraint property are listed in the *ServoMotor*. The parameters such as time and liftPosition are supplemented to the ServoMotor for the lift profile. These two parameters are defined as vectors of size varying from 1 to n. The defined vectors store the respective values of liftPosition at its respective times. The unit millisecond (ms) and millimeter (mm) are used to denote time and liftPosition. In the same way, the profile for Pump is modeled with a constraint property, which defines its polynomial equation and the respective parameters.

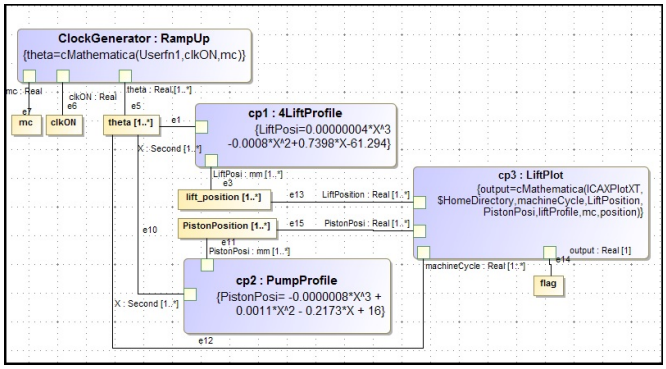

Fig. 10. PD of filling module, including LiftProfile, PumpProfile and ClockGenerator for synchronization.

Synchronization of Lift and Pump profiles As described in Section 2, the constraint property is a mechanism for engineering analysis and these constraint properties need a Parametric Diagram (PD), where the binding connectors connect the properties and their respective parameters. The PD for the filling module, with the Lift and the Pump profile constraint properties, is shown in Figure 10. This diagram defines the connection between the constraint properties, *cp1:LiftProfile, cp2:PumpProfile* and the constraint for the clock generation. The value typed parameters required for these constraints are denoted along with the binding connectors, connecting the parameters to respective constraint properties. The constraint property for the clock generation model uses the default *cMathematica* property of constraint block from SysML. The clock generates ticks that are fed to both *cp1 & cp2* profiles synchronously. This mechanism in PD is the key in the synchronization of the Lift and Pump profiles.

7. SPL MODEL OF TR/28

The complete TR/28 packaging machine from a behavioral construct is modeled in SPL. The modules of the machine such as *Bottom Sealing, Filling* and *Top Sealing* are modeled as operations on the highest hierarchical level. The complete SPL model of the machine is given in Figure 11.

Module operations The operations of each module are modeled within their respective module hierarchy. Cartons are the products of the TR/28 machine, represented as variables in the SPL model. The products flow one after the other, through the operations of the model. The *Filling* module handles two products with redundant resources working at the same time. A buffer variable, waiting for two products at the *Filling* module is defined in the SPL model. The *Bottom Sealing* module includes operations related to the sealing process of carton sheets. A five arm *Mandrel* is the resource, carrying one folded carton in each arm and performing the operations *Heating, Folding, Squeezing* one after the other for each sheet. Each sheet fills the position of the empty arm of the *Mandrel* and executes the sequence of operations as mentioned before.

The respective operations of the *Filling* and the *Top Sealing* modules are also defined in the SPL model. The *Filling* module includes operations *Charge Stroke, Lift Up, Lift Down and Fill Stroke*; where *Lift Down & Fill Stroke* are synchronized by parallel operations in the SPL model. The *Lift* and *Pump* are booked as resources for respective operations in the *Filling* module. With the help of *Indexing Conveyor*, cartons are transported to the *Filling* module by every two steps. The operations of the *Indexing Conveyor* are synchronized with other modules of the machine as mentioned in Section 5. The motion profiles of *Lift, Pump & Conveyor* are modeled by adding the relevant execution time of each operations which use these resources. The fact that operations such as *Filling* and *Top Sealing* are executed only when the conveyor is halted is taken care of by considering the conveyor motion profile. Hence, this SPL model includes the conveyor operations to be performed in parallel to other operations.

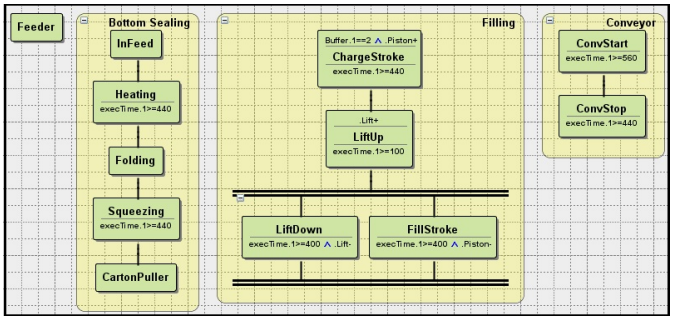

Fig. 11. SPL model of TR/28 packaging machine.

Resource View The SPL model of the TR/28 machine includes *Lift*, *Pump* and *Conveyor* as resources. These resources are booked and unbooked by corresponding operations. Projecting the sequences of operations from a resource point of view is visualized in SPL, as discussed in Section 3. Visualizing the sequences of operations with respect to the *Pump* is shown in Figure 12. The operations, *ChargeStroke* and *FillStroke* along with *LiftUp* followed by *LiftDown* are identified as relevant sequences of operations with respect to the resource *Pump*.

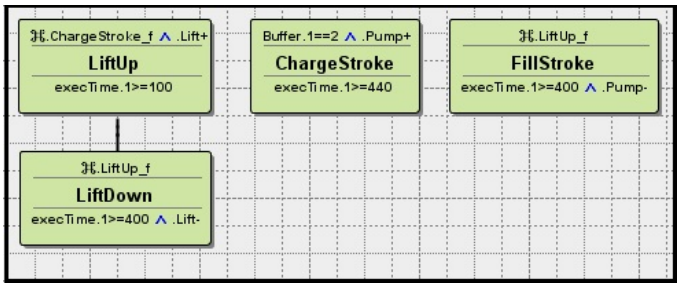

Fig. 12. Resource view of pump in SPL.

Optimization The three state SPL operation model holds a formal definition which is directly translated to an optimization model. A mathematical programming optimization model is developed with respect to the start and finish time of each operation in the case study. The relation between the operations is defined by including the minimal duration time of each operation. Summation of start time and minimal duration time of the previous operation gives the earliest possible start time of the current operation. This type of model can represent relations such as sequence, parallel, hierarchy, etc. Sundstrom et al. (2012) explain in detail the relation between an SPL model and an optimization model.

The optimization problem considered in this paper is to maximize the production rate (PR) of TR/28 with the constraint that the acceleration of the conveyor being less than $7m/s^2$. In the current case study, the cartons are being processed in batches of two as described in Section 4. In order to identify the optimial solution, the *BatchSize* and the *Machine Cycle* are varied from 1 to 10 and 0.6 to 1.3 s, respectively.

Objective Function:

$$\max PR$$
$$\text{s.t. } a \leq 7m/s^2$$

where

$$PR = \frac{3600}{mc} * N * S$$

N is number of Index Lines $= 2$,
S is BatchSize $= 1$ to 10,
mc is machineCycle $= 0.6$ to $1.3\,s$,
a is acceleration,
PR is Production Rate

The optimization model is developed in AMPL (A Modeling Language for Mathematical Programming) platform and the *Bonmin* solver is used to find the optimal solution. AMPL, is a powerful algebraic tool for linear and nonlinear optimization problems by utilizing different solvers, c.f. AMPL (2013). The nonlinearity in the optimization model is handled by *Bonmin*. By including the constraint for the acceleration, the optimal solution with production rate of 16600 cartons per hour is obtained at a *BatchSize* of 3 and *Machine Cycle* 1.3 s. The production rate of the current machine with *BatchSize* 2 and *Machine Cycle* 1 s is 12000 cartons per hour.

8. DISCUSSION & CONCLUSION

The need for modeling synchronous behavior raises from the built-in nature of packaging machines. Single master sending commands to a number of slaves, and the motions between slaves being synchronized are the traditions that the packaging industry still follows. The necessity for a common model to be shared between different domains creates a gap for synchronous behavior modeling. In this paper, a modeling solution for the synchronous behaviors using SysML is given and implemented with the case study of Tetra Pak TR/28 machine's filling module. The motion profiles are added as constraints to the respective resources. The resulting model is validated analytically and the results are documented in Kanthabhabhajeya (2013).

The entire TR/28 machine is also modeled in SPL, to make use of the advantage of the formal sequence planning language. The model is also projected with respect to one of the resources. An optimization model with objective function to maximize the production rate is developed with the support of the SPL model. The resulting optimal solution gets a maximum production rate, 16600 with batch size of 3 and machine cycle 1.3 s. This simple optimization

of the TR/28 machine shows that minor modification of an existing packaging machine may generate significant improvements. The optimization model will be extended to include more detailed motion profiles, as well as mutual exclusion between shared resources.

ACKNOWLEDGEMENTS

This work was carried out in collaboration with and support from Tetra Pak as well as the Wingquist Laboratory VINN Excellence Center within the Area of Advance, Production at Chalmers, supported by the Swedish Governmental Agency for Innovation Systems (VINNOVA). The support is gratefully acknowledged.

REFERENCES

AMPL (2013). A Modeling Language for Mathematical Programming. http://www.ampl.com/.

Bassi, L., Secchi, C., and Bonfe, M. (2011). A sysml-based methodology for manufacturing machinery modeling and design. *IEEE/ASME Trans. Mechatronics*, 16, 1049–1062.

Bengtsson, K. (2012). *Flexible Design of Operation Behavior using Modeling and Visualization.* Ph.D. thesis, Chalmers University of Technology, Sweden.

Friedenthal, S., Moore, A., and Steiner, R. (2012). *A Practical Guide to SysML - The Systems Modeling Language.* Morgon Kauffman OMG Press, 2^{nd} edition.

Kanthabhabhajeya, S. (2013). Analytical and Simulation validation of the synchronous model using sysml plugins. Report, to be published.

Kanthabhabhajeya, S., Berglund, J., Falkman, P., and Lennartson, B. (2013). Interface between sysml and sequence planner language for formal verification. *Proceedings of* 23^{rd} *Annual INCOSE International Sysmposium.*

Kanthabhabhajeya, S., Falkman, P., and Lennartson, B. (2012). System modeling specification in sysml and sequence planner language - comparison study. *Proceedings of 14th IFAC Symposium on Information Control Problems in Manufacturing Information Control Problems in Manufacturing.*

Lennartson, B., Bengtsson, K., Yuan, C., Andresson, K., Fabian, M., Falkman, P., and Akesson, K. (2010). Sequence planning for integrated product, process and automation design. *IEEE Transactions on Automation Science and Engineering*, 7, 791–802.

Nguyen, K.D., Ng, T., and Chen, I. (2008). On algorithms for planning s-curve motion profiles. *International Journal of Advanced Robotic Systems*, 5, 99–106.

OMGSpecification (2010). OMG Systems Modeling Language (OMG SysML). Technical report, Object Management Group SysML. URL `http://www.omg.org/spec/SysML/1.2/`.

Pietrac, L., Leleve, A., and Henry, S. (2011). On the use of sysml for manufacturing execution system design. *Proceedings of IEEE ETFA.*

Qamar, A., Wikander, J., and During, C. (2011). Modeling continuous system dynamics in sysml. *Proceedings of IEEE International Conference on Mechatronics.*

Reisig, W. (1992). *A Primer in Petri Net Design.* Springer-Verlag, New York, 1^{st} edition.

Sundstrom, N., Wigstrom, O., Falkman, P., and Lennartson, B. (2012). Optimization of operation sequences using constraint programming. *Proceedings of 14th IFAC Symposium on Information Control Problems in Manufacturing Information Control Problems in Manufacturing.*

TetraPakEngineers (2007). OM operation manual, Tetra Rex TR/28 XH. Technical report, Tetra Pak, Lund, Sweden.