Requirements Analysis of Using Object-Orientation in Filling Machine Systems

Peter Wodzynski

LUND
UNIVERSITY

MASTER THESIS in Automatic Control 2013

Supervisor: Peter Lindberg, Tetra Pak

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

ISSN 0280-5316
ISRN LUTFD2/TFRT--5924--SE

© 2013 by Peter Wodzynski. All rights reserved.
Printed in Sweden by Media-Tryck.
Lund 2013

Acknowledgements

Firstly I would like to thank Anders Olsson, for introducing me to Niklas Svanberg, the core automation manager of Tetra Pak Carton Bottle.

Niklas gave me the opportunity to write my master thesis at Tetra Pak and for this I am a very grateful.

I would also like to thank all the people I was able to interview at Tetra Pak, for they were all very friendly and supportive.

Last but not least I would like to thank my supervisor Peter Lindberg, who has been an inspiration and an excellent mentor to me. Thank you so much for the countless hours of help and all the laughter.

Abstract

The use of an object-oriented approach in software engineering has proven to be successful for many years, but in the fields of mechanics and automation it has been ignored for long. The Tetra Pak Carton Bottle automation team believes that a switch from a function-oriented architecture to an object-oriented one would be of benefit not only to them but also any department which is involved with development, production, and maintenance of filling machines.

Many product projects of today make the mistake of having its participants jumping straight into the implementation part of the project while skipping the work of first defining requirements of what the product shall accomplish. People in general like to start by doing the *how* of the project without first focusing on the *what*.

But if the needs of other product concerned are overlooked then there is a risk of an important part being left out, perhaps leading to a weaker product. This master thesis focuses on the *what* part of a development project where an object-oriented architecture shall be deployed for Tetra Pak's development, production, and maintenance of machine systems. During the master thesis requirements from various stakeholders were defined based on interviews and discussions held with various Tetra Pak departments.

The result, the requirements, show that Tetra Pak has several areas which would be benefited by an object-oriented architecture by making Tetra Pak's machines more modular and the maintenance of them simpler.

Contents

1. Intro

The use of an object-oriented approach in the field of software has been successful for many years. However, in the field of automation it has hardly been used at all, it is just recently that certain automation system providers have made a switch to an object-oriented architecture to gain the benefits it has brought the software world. The company Tetra Pak is still one of the companies that has not chosen to use a supplier with an object-oriented automation system.

Tetra Pak has been a technology driven company for many decades, by using new technologies which they would base their filling machines on. The company has just recently switched over to a requirement driven approach. Before a new project goes into the implementation phase it shall have requirements defined of what the new product shall accomplish i.e., the guessing game of creating a new product and assuming that someone will want it has been removed.

And as such, before an object-oriented architecture can be implemented, a stakeholder analysis must be made. This will determine the pros and cons of today's architecture and how an object-oriented architecture could be of help. The analysis should include the departments of Tetra Pak that would be affected by an object-oriented architecture and not just the automation department. And thus the decision if an object-oriented architecture should be implemented will be based on requirements from all concerned parts. If an object-oriented architecture would appear to be beneficial for more than the automation team of Tetra Pak Carton Bottle then it would be all the more reason to implement one.

1.1. Aim

The aim of this work is to define stakeholder requirements based on stakeholder inputs, requirements which can be fulfilled if Tetra Pak implemented an object-oriented architecture.

1.2. Methods

Every requirement defined in this thesis shall originate from stakeholders voices. The stakeholder voices shall be collected through interviews and thereafter be translated into stakeholder requirements. Stakeholders are personnel working with development, production, and service of Tetra Pak's Carton Bottle machine systems.

1.3. Report Structure

The report is divided into two sections. The first section, chapters 2-6 describes the results of choosing a less performing automation system, the A6 machine platform which is the main area the analysis is based on, the finding of stakeholders, how the requirement elicitation process was carried out, and the principles of object-oriented programming. The second section, chapters 7-12 shows the core of this analysis, the stakeholder requirement analysis. The structure for every stakeholder follows the three points:

1. Conclusions with related requirements
2. Stakeholder inputs collected from interviews and discussions
3. An object-oriented analysis of the stakeholder inputs

After the second section the overall conclusions of this master thesis are stated.

2. Tetra Pak's Automation History

2.1. Machines before the PLC

In the mid 1970's all of Tetra Pak's machines where based on large mechanical axes that were driven by asynchronous motors. The mechanical axis' job was to run various camshafts were the cams performed the mechanical work such as lifting and sealing packages. In addition to the mechanical work there were also various electrical functions which had to be carried out. These electrical functions were performed using mechanical logic relays. A button press could switch on a logical relay and in turn, an electrical function would start. The electrical functions were often encapsulated in large mechanical boxes that were mounted on the filling machines. For example a box could contain a PID regulator, that is to say, that by 1970's machine objects could modularly be mounted and replaced at will.

In the mid 1970's Tetra Pak was delivering three machines types and each type consisted of many different versions. One was the very known and successful Tetra Brik machine, which produced brick looking packages in various sizes. The second machine was the Tetra Rex machine that produced packages with gable tops, also in various sizes. The Tetra Rex machine is today a part of the Tetra Pak Carton Bottle product company. The third one was the Tetra Classic machine that produces tetrahedron looking packages. The two larger package producing machines, Tetra Brik and Tetra Rex had at this time a capacity of five to six thousand packages per hour. The three machine systems are still in use as of today.

2.2. The PLC

It was not until the late 1970's that Tetra Pak began using PLC systems and the first one came from Allen Bradley, which was later acquired by Rockwell Automation. The PLC replaced all existing mechanical relays and at the same time provided flexibility for the users to easily create and run programs. To make it possible for the PLC system to interact with the machine's components an encoder was connected to the mechanical axis, the encoder's task is to signal the current angular position of the axis to the PLC program.

Over time, the machine systems requirement on capacity increased, the machines should go faster. Around the mid 1980's the Tetra Brik and Tetra Rex machines had been developed and improved greatly and they ran like clockwork while also being extremely robust. The machine capacity had at that time reached seven thousands packages per hour and it was around this time that the aseptic Tetra Brik machines became a huge success. Specifically TBA 8, TBA 9 and the simpler version TBA 3 were Tetra Pack's big successes.

Figure 1 Production line using 11 TBA8 filling machines at Pascual, Spain

As Tetra Pak was constantly raising the machine requirements together with new and more advanced technologies being invented, the need for a faster PLC system arose. Tetra Pak's PLC system provider at the time could not deliver a good enough one, and neither did one exist on the market. Eventually Tetra Pak asked the company SattControl, which today is part of ABB, to create a new high performing PLC system. SattControl took on the task and created the PLC system TPMC, Tetra Pak Machine Control. The new system was brought in use in the late 1980's, and it was a unique system that only Tetra Pak had access to. Already at that time, Tetra Pak had a fast scan which could scan the input- and output-signals of the PLC with a speed of one millisecond. The fast scan was used for critical functions which had to be performed quickly and precisely, such as the pulse which sealed the packages.

2.3. Tetra Top - Moulding a top

In the late 1980's the Tetra Top machine system was developed, a machine that produced a carton package that looked like a bottle but with a plastic moulded top. A plastic wafer that was pulled off by the consumer resembled the lid of the carton bottle, and it was later on that Tetra Pak added a plastic cap onto of the lid. The moulding device for the Tetra Top and its fast cycle time was considered impossible by Tetra Pak's suppliers, as the top had to be moulded, heated and cooled very fast, repeatedly. However, Tetra Pak pulled it off and the Tetra Top was considered the future.

2.4. A unique product

Tetra Pak sold as well as leased machines to their customers, often with a leasing cost so high that the actual machine was paid off after one to two years. Tetra Pak could use such a strategy because they had a unique product that nobody else in the world had. The aseptic machines allowed customers to aseptically package products such as milk and juices with a shelf life of over six months and to be stored at ambient temperatures. Therefore, storages could be filled and deliveries could be made when it suited the customer, as opposed to pasteurized products that had a refrigerated shelf life of only ten days.

2.5. Cost focus

Hans Rausing and Gad Rausing were sons of Ruben Rausing and they were the owners of Tetra Pak in the late 1980's. It was Hans Rausing who was the operative owner and ran the company while Gad Rausing chose to be more in the background. Hans put large emphasis on cost, just as Ruben did in his time, and therefore many of Tetra Pak's projects never reached the market as they would increase the cost of the packaging material. Due to this the customers of Tetra Pak often thought that Tetra Pak was not a company focused on the development of new technologies. But in reality they were just not released to the public customers.

2.6. The new servo technology

1987 was the year Tetra Pak started using servomotors. Servomotors were a new and unique technology as the servos could run a motors axis to a certain position with ease, as opposed to the more difficult to position induction motors. The servomotors were used for the more critical and precise functions. Back then they were delivered in large boxes with all the components separated and had to be assembled. It was at later time that the different function blocks of the servo motor were encapsulated and assembly was no longer needed. Shortly after, during 1990 a new multi axis system for servomotors was created. The new system made it possible for several servomotors to be run synchronously and thereby perform fast and precise functions together while less programming was needed from the machine program implementer.

2.7. Importance of the automation system

During 1990 a project to create a filling machine which was divided into two parts was underway. The first machine would produce ready to fill packages which would then be delivered to the second machine. The second machine would be located at the customer and would sterilize, fill, and seal the delivered packages. This concept was developed as a response to competitors because they had such systems underway and Tetra Pak wanted to offer a similar system for their customers as well. The machines were from the beginning of the project designed to use multi axis servo systems, later on a decision to use single axis system instead was made. As a result the developers had to rethink their design and they were forced to use a sequential programing style instead of a synchronous one. As developers had to create extra functions to take care of communication between the servomotors, the required machine cycle went up by 10%, thereby lowering the capacity of the machine by 10%.

Another project that started in the 1990's was a project to construct a Tetra Rex machine, the TR10 machine. This project deviated from the standard cheaper automation system that Tetra Pak was using at the time. It used a better performing PLC from the company Ge Fanuc, as opposed to the slower model which was Tetra Pak's standard. By using a VMEbus with the better PLC, servo drive systems, and a HMI system they could achieve very fast communication between the systems. The automation platform was one of the best performing systems that Tetra Pak had, however the project was cancelled due to other factors.

A project to once more create a new TR machine was started in 1994. This project was to develop the TR18 machine, its principle was to take the best from the TR8 and the TR10 machine systems and to create the rest. It was supposed to have a capacity of 14,000 packages per hour. They reused the VME communication bus, the HMI system, and servo system from the TR10 machine but downgraded the PLC to the cheaper and less performing model. By choosing the weaker model the system lost its rapid communication in the VME bus and problems were experienced in the interface between the systems. The resulting machine could not achieve the target capacity, this was the second time Tetra Pak had chosen a weaker automation system.

In the early 21 century Tetra Pak started working on an aseptic carton bottle machine called A5 which was supposed to pack premium products that had been enriched with additives, omega-3 oil for instance. Around this time the company Atlas Copco AB sold of their division responsible for their multi axis servo systems which they had acquired previously from Socopel. Atlas Copco only kept a small team to support what was already on the market. The A5 team was therefore tasked to evaluate which automation system should be used instead and a system from the company B&R Automation was chosen. With the new automation system the A5 team succeeded in creating Tetra Pak's best machine systems from an automation point of view. With modules that could be tested and run stand-alone and the use of module plates the machine was up and running after a three week start-up phase, the shortest ever in Tetra Pak's history. The A5 platform was meant to be an additive for costumers that want to sell premium products, but during a restructuring of Tetra Pak the target of the machine was changed. The target was now to acquire 20% of Tetra Pak's market. But the package production cost of the A5 machine was considered too high for the new target and the system was therefore cancelled before it ever reached the market.

In 2006 the Impact project was underway, the projects goal was to construct a carton bottle packaging machine but with a lower package production cost than the A5 machine system. During the Impact project the decision to use a weaker automation system was made once again, for the third time. The principle to take the best from the previous systems and build the rest was used once again, the TT3 and A5 machine systems were used. Before the Impact projected had started a corporate decision had been made to start use the automation system supplier Rockwell Automation for Tetra Pak's machine system. This decision downgraded the B&R automation system the A5 machine used. With the new weaker PLC system the A6 machine was left with slower performance and without functionality, which is today desired by customers, such as remote troubleshooting and possibility for inline quality control.

Tetra Pak has underestimated the automation system needs on several occasions, which has left them with lower performance machine systems. Today the speed and complexity of machine systems are growing rapidly and becoming a larger part of the industry worldwide. The automation system is thus becoming of more importance and cannot be ignored. If Tetra Pak continues to overlook the automation systems in terms of performance and functionality then the customers will choose a different packaging system supplier.

3. Filling machines

3.1. Selling package material

Tetra Pak's business model is to make money by selling packaging material to customers who use it in their filling machines, however Tetra Pak has also chosen to manufacture filling machines. Obviously, Tetra Pak has an income from selling machines but the largest income comes from the packaging material, as the customer continuously buys it for his filling machine, especially since a filling machine can be operational up to 20 years. Tetra Pak creates an advantage for them self by choosing to also manufacture machines. It leads customers to more likely choose Tetra Pak instead of a supplier whom only delivers one part because of the additional security. A customer who buys machine and packaging material together from Tetra Pak has the security of always turning to Tetra Pak whenever a problem arises, as opposed to having two different suppliers which will most likely blame each other.

3.2. Aseptic package

Tetra Pak's largest income comes from the package material that the customer buys and uses in aseptic machines. The aseptic machines fill packages in a sterilized environment, where sterile components are not

exposed to an unsterile, bacterial zone. The packages are filled with UHT products, as those products have been momentarily heated up to high temperatures which reduce bacteria and spore.

The bacterial proof material together with UHT threated products and aseptic filling method produces an aseptic package which can last up to six months in an unrefrigerated environment. Aseptic packaging is a good choice for countries with hot climates as well as for those that are in short supply of certain products as they can be transported without the usage of refrigerated containers.

3.3. A6

Carton bottle

The A6 is Tetra Pak's latest released aseptic machine system. It is a machine used for packing UHT milk in one-litre aseptic carton bottles. The carton bottle is made of cardboard and has a moulded plastic top. It has a large screw cap that makes pouring easy without splashing. The carton bottle is designed so that the shape and size suit both children and adults. The machine is the first aseptic carton bottle in the world and it was created as a response to the PET bottle being sold in southern Europe. Therefore, the target was warm countries like Portugal, Spain and Italy, where the cost of refrigerated transport and storage is expensive.

The machine

The A6 is a two-sided machine where each machine side produces five thousands packages per hour. The machine production uses the following physical material: packaging material, strip, plastic granulate, capped necks, and a product. The A6 machine is Tetra Pak's largest machine system, it has two floors and is over 15 meters long, see Figure 2. Although it is larger than the Tetra Brik machine, it is not more complex. The technologies to create an aseptically sealed carton bottle take up more space than the ones used in the aseptic Tetra Brik machine. The A6 machine system is constructed of 11 different modules produced at various module suppliers where they are also test ran, the modules are thereafter shipped to Tetra Pak Carton Bottle in Lund where they are docked together and further tests are performed before the whole machine is shipped to the customer.

Figure 2 The complete Tetra Evero A6 Aseptic filling machine

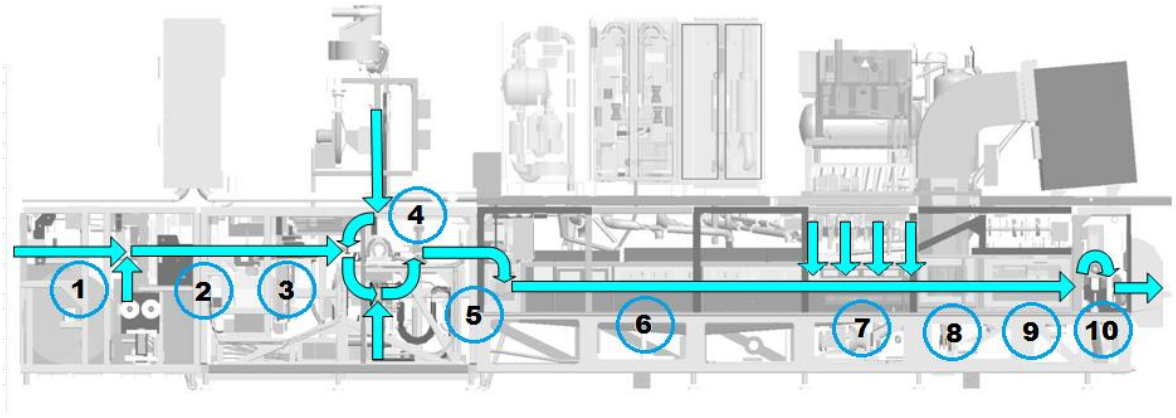


Figure 3 The material and product flow of the A6 machine

Machine flow

1. Package material feeding and Strip application

The material flow starts in the first module, called ASU & Cutting. The module supplies the machine with packaging material and strip. Both materials are sealed together by melting one edge of the packaging material. The strip is applied to protect the product, preventing bacteria in the fibers to reach the product.

2. Cut into sheets

The package material with the applied strip is cut into sheets.

3. Sleeve forming

The sheet is formed to a sleeve by squeezing it around a mandrel and sealed the sides using induction.

4. Top Moulding

The sleeve is pulled off from the forming mandrel onto a mandrel wheel. On the mandrel a capped neck is placed and prepared. The mandrel wheel indexes once and the sleeve is now pointing downwards, from below a so called hotrunner is lifted up to the sleeve which moulds a plastic top on the sleeve. The moulding forms a plastic cover between the capped neck and the sleeve, thereby binding the two together and creating a Ready-to-Fill (RTF) package.

5. Transport into SFS

After one more indexing the RTF pack is transferred into the aseptic chamber and positioned bottom-up. The package has at this point been transferred from the RTF part of the machine to the SFS, Sterilize-Fill-Seal part.

6. 3 step sterilization

Each RTF pack goes through a three stage sterilization process. During the first stage the RTF pack is pre heated to prevent condensation during the second stage. At the second stage the RTF pack is gassed and sterilized with hydrogen peroxide. In the third stage the hydrogen peroxide gas is vented away and neutralized by a catalyst converting the gas to water and oxygen. Once the hydrogen peroxide has been vented away then non-sterile air must be prevented from reaching the inside of the RTF pack until it has been filled and sealed. Sterile air is therefore blown downwards from the top of the aseptic chamber, pressing the non-sterile air downwards. A virtual barrier is formed where the upper part of the chamber is sterile and the lower part is non-sterile.

7. Four stations filling

Once the sleeve has been vented empty of hydrogen peroxide, it is ready to be filled. Four stations are used to fill the RTF package with one litre of a product. The partial filling is to reduce foaming, as excess foaming may prevent the RTF pack from being aseptically sealed.

8. Sealing

The filled but open container is indexed forward, with a motion profile that does not spill the milk. At the sealing station the open sleeve is squeezed together and sealed using induction heating, thus creating an aseptic sealed package.

9. Fold and form

At the end of the aseptic chamber the bottom of the filled pack is formed. A cylinder presses the bottom down so that two flaps are formed. The plastic on the inside of the flaps is melted with hot air and finally the flaps are folded against the underside of the bottle. The aseptic carton bottle is now complete.

10. Date marking

The aseptic carton bottle is now complete, and an arm lifts the bottle out of the aseptic chamber and places it upright on a conveyor belt. The bottle is now transported out of the machine, through the date marking equipment and to the distribution equipment where it will be wrapped in film or put in cardboard boxes.

4. Tetra Pak Development Plan and Stakeholders

Tetra Pak has from when it was founded until 2005 been a company that has based its products and project on ideas brought up by its owners and employees. More of a bottom-up principle was used in their development process during this time. By starting their projects based on ideas, which often were technologies, and then to work their way up to a complete product which was mostly a filling machine. Today, Tetra Pak works differently. They base their projects more on whether they can satisfy a market need. They define clear and specific requirements of what a new product must meet before they begin the actual development of the product's concepts. The principle is now therefore top-down, from having the overview mapped out first and then to continue further down into smaller parts.

This master thesis is the first one in a series of master theses. The work way in this master thesis will be performed at a very abstract level to set up requirements for Tetra Pak's architecture in development, manufacturing, and maintenance of their machine systems. In a top-down project the requirements are established first, before the concept that will fulfil them is developed. However, this thesis will already from the start have a concept in mind, an object- oriented architecture concept. The next thesis in line shall examine how a potential object-oriented architecture could look like.

4.1. V-model

One of the models Tetra Pak uses for their development projects is the V-model. It is an general project model that shows at which detail level the activities take place. The model is visualized with a graph and when the level of detail is observed while time passes a V is formed, hence the name V-model.

Tetra Pak's V-model was used in this thesis as a tool to find out what the personnel of Tetra Pak Carton Bottle work with, what roles they have in the development, manufacturing, and maintenance of Tetra Pak machine systems, and last to find potential stakeholders. Figure 4 on next page shows Tetra Pak's V-model and where each respective department that has been interviewed works.

At the beginning of the project, on the highest and very abstract level of the V-model is where the work that handles the system as a whole is performed, such tasks as project planning, market surveys, requirements elicitation, and establishing of functionality. At this level the task is to find out what a product should be capable of and not how it should be implemented. The input to the first activity System Requirement Analysis are the stakeholder inputs, gathered from for example market surveys. The system engineers working at this level are responsible for translating stakeholder inputs to requirements, a stakeholder input, could for example come from a customer who thinks that his transport costs from location A to B of filled milk cartons is too high and shall be reduced. System engineers could translate the input to a technical requirement, stating that transport trucks shall be fully filled with packages with minimum space wasted. Such a requirement is still a very abstract one and will be further broken down. To fill a truck fully means the packages on the bottom shall hold up a minimum weight of X kilos.

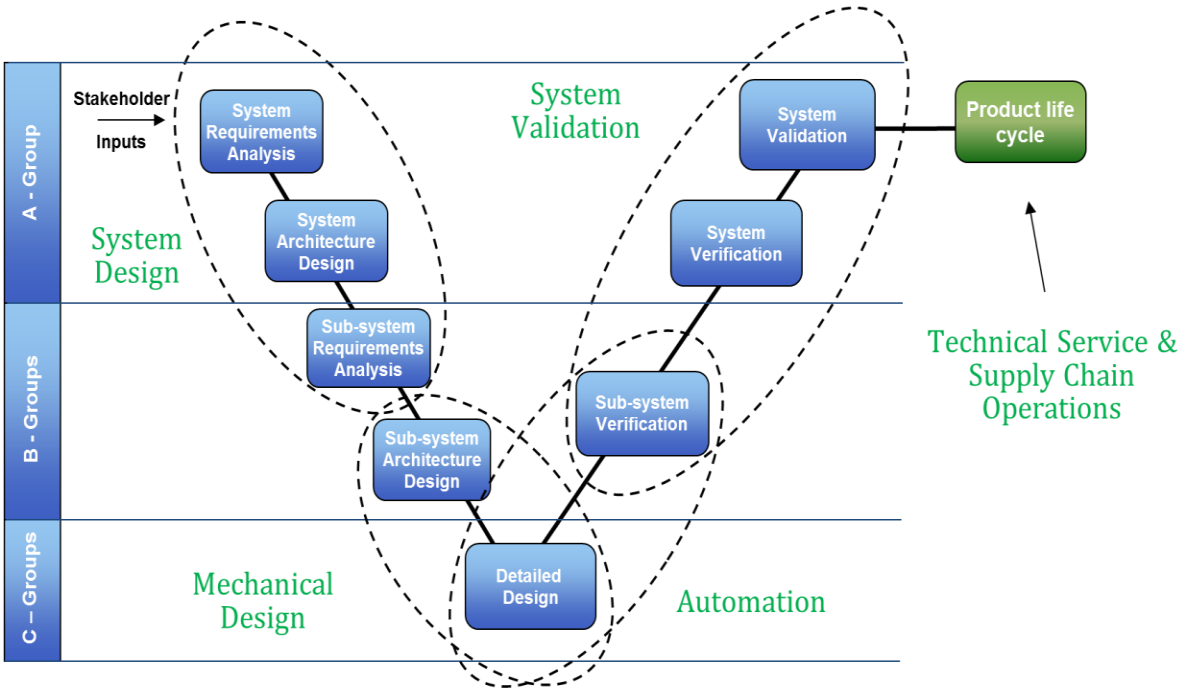


Figure 4 Stakeholders marked in the V-model

At the B-level the whole system is divided into subsystems that several parallel working B-groups have responsibility over. Here the requirements are broken down once more into sub-system requirements and each B-group receives their respective requirements. The B group responsible for the package material must now use a certain supplier that uses high quality of cardboard and that the width of the material is within a certain limit to ensure the weight will not crush other packages. Another task at this level is the finalization of which concepts the machine should use, which enables the mechanical designers to start their work.

At the bottom of the V-model, the subsystems are broken down into functional groups called C-groups. The two departments working mainly at this level are automation and mechanical engineers. They are responsible for producing solutions that will fulfil the sub-system technical requirements. The mechanical engineers start drawing their solutions in CAD programs while automation engineers create electrical diagrams and machine programs.

Once the C-group solutions have been implemented the work to close the loop starts, the System Validation department takes over from here. They are responsible for organizing tests and keeping track of test machines located in-house at Tetra Pak. At the B-group level the chosen concepts are verified by making sure the sub-system technical requirements for each B-group are fulfilled.

Back at the highest level the whole machine is verified by verifying the highest level of the requirements. The maximum weight a package can support is tested and if it can support X kilos it means the technical requirement has been fulfilled. The system verification is followed by a system validation, this activity is carried out to make sure the raw inputs from stakeholders have been fulfilled. This is done in the correct and real environment, a truck is loaded full with milk packages and is driven the path from A to B. If all milk cartons are intact then the raw stakeholder input to reduce transport costs has been fulfilled. If the test ends in failure then the translation from the stakeholder input to the technical requirement has been made incorrectly. The system verification tests are used to verify a problem while system validation tests are used to verify if the correct problem was solved.

Once all the stakeholder inputs are validated the machine system is released to the market and the project has reached its end. Now Supply Chain Operation and Technical Service take over. SCO is responsible for building and testing new machine orders before they are sent to the customers, while TS is responsible for the machine once the customer receives it. TS installs, services, and maintains machines until they are scrapped.

5. Object-oriented Principles

The following chapter describes the object-oriented programming principles (Halladay & Wiebel, 1993) that the master thesis analysis was based on.

5.1. Classes

A physical object in reality is created from some sort of a blueprint and it works the same way in the object-oriented software world. A class acts as a blueprint from which software objects are instantiated and just like in real life where multiple objects of the same type are created from one blueprint. In the same sense there can be multiple software objects instantiated from one class.

Classes are created to help users solve various problems, but instead of recreating a solution every time a programmer needs a new program they instead import classes from software libraries, classes which together will help solve the programmes problem.

The classes which are stored in software libraries do not have to be created by a program implementer, classes and executable programs are stored separately. As a result the classes and programs are maintained independently from each other, making it easy to update and trouble shoot the classes. But with the advantage of a class update propagating to every program that uses the class, without the need for updating the executable program.

The principle of an object-oriented system where the executing program and its classes are kept separate eases the use of software libraries. Libraries can have already created classes that are ready for use. A user can then look through the library to find the items he sees would fit to solve his problem and import them to his program. The programmer does not even have to understand how they are implemented, he simply imports ready solutions and uses them right away.

5.2. Object

A software object is a representation of something, for example a lamp, a servomotor, a PID regulator and so on. Each object has two properties, attributes and methods. The attributes describe the status of the object while methods are used to change status. A lamp represented by a software object has one attribute, if the light is on or off, and two methods which are, turn on the lamp and turn off the lamp.

The object stores and keeps track of all attributes while it exposes the methods to the executing program. When an object is instantiated from a class, the user must specify the value of attributes, how much and what attributes should be specified is determined by the class implementer. The attributes are stored in variables within the object and the object itself is stored in a variable which is tracked by the executing program.

The source code for a class can be written and maintained independently of other classes. This means that object-oriented software is modular, the program objects can be plugged out and in independently. Another object can easily replace incorrectly functioning objects, it is an analogy to repairing a mechanical problem in real life. If a hose breaks then only the hose is replaced and not the whole machine.

5.3. Encapsulation

In an object-oriented system the methods and attributes are stored together. They are both encapsulated in an object and the encapsulation can be used to prevent unauthorized attribute modification of objects, the attributes can have one of three safety declarations. A variable with associated attributes can be declared as public, private or protected. A private variable can only be accessed within the object. Other objects and applications that wish to use these variables have to do it through public methods, methods that can be implemented to disallow invalid modification of the variable. Private variables together with public methods are one of the major purposes of object-oriented programming.

A public variable can be accessed and modified from anywhere in a program. Public variables can result in unsafe programming and should mostly be avoided. They are more suited for small programs that are perhaps only implemented by one person.

The third type is protected variables. A protected variable can be accessed by the object and by other classes that inherit from the class that was used to instantiate the object.

The actual encapsulation of an object-oriented system causes the workspace to look a lot smaller and less complex, as the code is implemented in a variety of smaller units, often referred to as information hiding.

Figure 5 shows attributes and methods being encapsulated in a class, from which a programmer instantiates four objects of the same type.

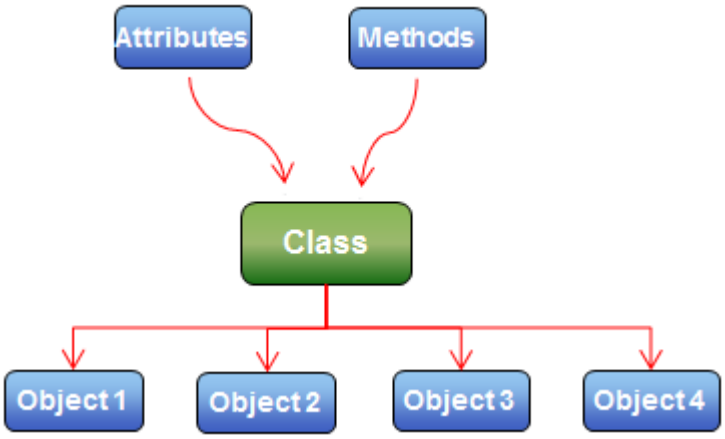


Figure 5 Classes, objects and encapsulation are principles of object-oriented programing

5.4. Inheritance

Several objects of a type will often have some elements in common with each other. Cars, trucks, and motorcycles are vehicles that share features but also have those that separate them from each other. In object-oriented systems, a class can inherit attributes and methods from another class so that the common elements do not have to be implemented more than once. The new class will have access to everything that is not declared private in the parental class. The programmer will then implement the methods he sees are missing in the new class.

If a programmer searches through a class library but does not find a suitable class for his program then he can choose to inherit the methods from a class that at least have some beneficial methods. In the new inheriting class he implements the lacking features. As more classes inherit from the same parental class, a hierarchal class library composed of similar classes will be formed. The further down in the hierarchy, the more detailed are the classes. In some programming languages, such as Java, all classes are subclasses of a core class. In Java the core class is called Object, all classes can thus be followed back to the original Object class. In such programming languages the hierarchy is one large library where every object share a is a kind of relationship with the original parental class.

Inheritance can also be a way for companies to protect their software, by preventing other companies access to the source code but still letting them be able to inherit from the their classes. The customers can then further implement methods which they considered missing while also using the selling's company's methods.

5.5. Polymorphism

In most programming languages the compiler needs to know exactly what functions the program will use. However, the polymorphism principle in object-oriented programming allows the programmer to create programs that do not have precisely set which objects it will use during compile time. The program will try to look up a used method and bind it to the correct object during run time instead. This simplifies the programming of interactive applications where users specify during runtime what objects the application should consist of. Polymorphism is implemented by using classes that share a common interface, classes that have methods named the same but that differ in functionality.

6. Requirement elicitation process

The figure below shows the chain of developing an object-oriented architecture. The target of this thesis was to accomplish the first activity in the chain, which is a requirement elicitation process.

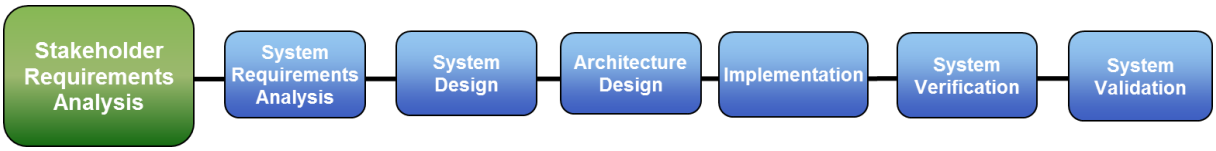


Figure 6 Object-oriented development chain

Requirement elicitation is the process of collecting requirements from stakeholders, stakeholders being for instance, users, customers, and financial personnel. The word elicitation is used instead of the word gathering because collecting requirements cannot be done by simply asking the stakeholders for them. Stakeholders direct requirements will be limited due to them not being completely sure of what is necessary, they might not posses the needed competence, have problems communicating needs that are assumed to be obvious, or they provide requirements that are ambiguous. Requirements elicitation is instead performed with interviews, stakeholder observations, workshops, user and marketing surveys, prototyping, and team discussions (Bittner & Spence, 2003).

For the master thesis the requirement elicitation processes was done by interviewing the stakeholders. The first few interviews were carried out to gain an understanding of Tetra Pak's development, production and service of their machine systems but also to understand the principles of requirement elicitation by interviewing System Design engineers.

After comprehending the basic, at least a part of what Tetra Pak is and its 60 year old history, more interviews were carried out. In these interviews the task was to collect raw stakeholder inputs, called voices which were documented by paper and voice recordings.

The next step was to comprehend the voices and to use them to find connections to a possible object-oriented architecture by comparing them to the principles presented in the previous chapter. All voices where an object-oriented architecture would help were translated into stakeholder requirements and by using concrete examples it was shown how they can be upheld.

The stakeholder requirements are the main points of the thesis, as they are the output, the result. The output requirements will be used by the next thesis worker in line, by whom they will be translated into technical requirements.

7. Automation

7.1. Conclusions and requirements

The automation methodology Tetra Pak uses today for their machine systems is function-oriented, it is a methodology that was created a long time ago to suit electricians when the shift from using physical logic relays to PLC processors took place. It was designed for simple and small machines without complex functions. Today after over 30 years Tetra Pak still uses it, even though new and better systems are available. As a result a lot of the new functionality that developers would like to implement is skipped because the automation system lacks functionality. Therefore creating more complex functions is more time consuming than it should be.

As Tetra Pak's programs are becoming larger and more complex with time they also become more difficult to understand and maintain when a standard function-oriented system without any extensions is used. Developers have it difficult to become familiar with other developers' programs and to update them due to less structure, large size and lack of standardized interfaces. The programs are structured using a large list where interactions between the items are difficult to spot as a hierarchy structure is not possible to use.

The automation system used by Tetra Pak Carton Bottle does not have support for software libraries. Without software libraries it is difficult to reuse program solutions. They use add-on instruction blocks instead, the blocks encapsulate code and can be imported to programs. However, these blocks lack the capacity to define concrete methods. Methods have to be coded into logical signals, making the usage of add-on instruction blocks clumsy. Another disadvantage of using blocks is that they take extra computing time compared to if the code was entered directly into the program.

An object-oriented automation system would help Tetra Pak with the limitations they are faced with today. Class libraries would help reusing software solutions and reduce development time. Using classes would also help service engineers install updates out in the field as they would import classes instead of manually entering code into routines. This would in turn limit how much a service engineer can change in a machine program and this helps the developers as they want to keep the machine programs as standard as possible. The encapsulation principle is another benefit that would help create a hierarchal program structure by encapsulating data and methods together, making it easier to understand the program and its functions.

The requirements that follow below are my interpretations of the inputs collected from the system engineer Peter Lindberg and the automation engineer Ulf Svensson.

Automation Requirements

- Software development time shall be reduced
  - Software shall be reusable
  - Multi instance functions shall only have its code inserted once
  - Basic functions of a specific component type shall only be implemented once
- In field modification of programs shall be kept to a minimum
  - Software code installation shall be reduced to a minimum
  - Program deviations shall be kept separate from the original program
  - Software implementations shall be encapsulated
- Program understanding shall be easy for anyone
  - A hierarchal program structure shall be used
  - Complex functions shall be encapsulated
- Software program shall be protected
  - Variables shall be encapsulated with its functions
  - Private variables shall be accessed using safe methods

7.2. Stakeholder input

7.2.1. PLC Programming

The programs of today are created on a PC and downloaded to the PLC processor. The standard for PLC program from before was that five different programming languages were available. While today many other systems offer other programming languages instead of the standard five, Tetra Pak's automation system does not support any new languages.

Ladder diagram programming is the most used language at Tetra Pak. It is a graphical programming language that looks like a ladder, with two vertical sticks and several rungs. Each rung contains a number of logical gates and a coil. If the logic conditions of the gates are true then the coils output is set high. See Figure 7 for a ladder diagram example. This programming method was developed to fit electrical technicians. They created the machine logic using logical relay diagrams and a ladder diagrams were made to visually mimic them. This made the electricians understand the new programming language without the need to learn a lot. But ladder diagram language is very old and was designed for simple machines, yet Tetra Pak continues to use this language even when the market offers new and better programming languages.

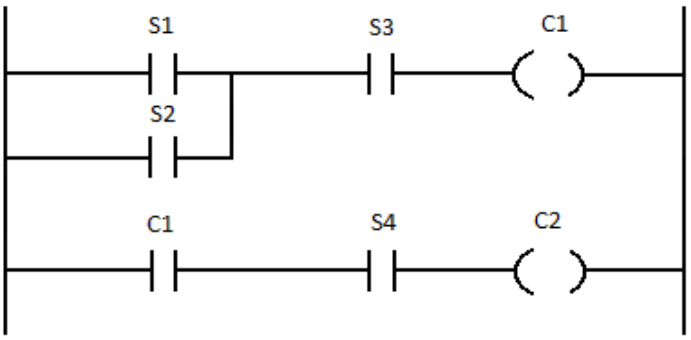


Figure 7 Example of a ladder diagram with two rungs

The two other programming languages Tetra Pak uses are, structured text which is a text-based high- level programming language and the language sequential function chart which is a graphical language that uses logical conditions to jump between steps and each step has an associated action. The three programming languages Tetra Pak uses complement each other and can be used in the same application.

7.2.2. System Manufacturer Rockwell

Tetra Pak uses the automation system supplier Rockwell for their machines. Rockwell is a stable, global company that provides a wide range of automation components. Tetra Pak buys equipment such as PLC-processors, servomotors, and frequency converters from them. Unfortunately, Rockwell's focus is not on the faster machines such as Tetra Pak's filling machines as they only account for a small portion of Rockwell's customers. As a result Rockwell lags behind what Tetra Pak's development would want. For instance, the A6 machine operates using seven PLC processors.

7.2.3. Program Structure

The machine program of the A6 machine system is built up by a number of tasks where each task consists mainly of a program which runs a set of routines. Figure 8 illustrates a part of the A6 machine program. The Machine Controller is a task used to synchronize the module programs and it contains each machine step. The module programs are tasks consisting of routines and in the routines all the PLC code can be found. For example, a routine can manage alarms, machine parameters or a servomotor.

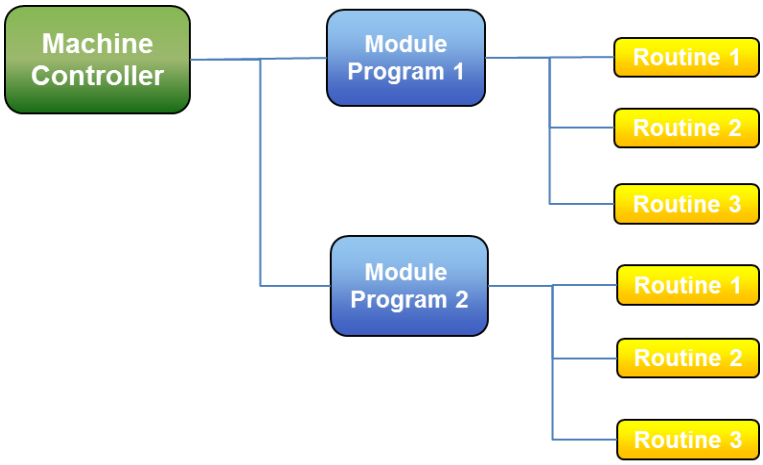


Figure 8 Rockwell's automation program structure

7.2.4. Variables

Two types of variables are used to store data in the routines of the A6 machine, global and local variables. Global variables can be accessed by all programs within a processor while local variables are only accessible by the program that created them. The global variables can be modified throughout the programs and can therefore be unsafe if they are not handled properly. The function-oriented architecture does not provide methods to protect the variables from erroneous modification. There have been occasions where machines have stopped working because engineers have modified the value of global variables to suit their own programming style. Another error source is the inability to use constants in today's A6 machine program. The workaround is to use global variables which have values assigned to them during an initialization phase, but without proper variable protection, errors happen.

7.2.5. Add-on Instruction blocks

To reuse functions without rewriting the same code in multiple program sections function block are used and they are called add-on instructions in the Rockwell system. The add-on instructions encapsulate code and are called on by routines.

The encapsulation provides a security for the variables by limiting the access way to them. The blocks are exposed to the routines by interfaces. Through the interfaces the routines use the add-on instruction functions, but only logical signals can be sent and received by the add-ons. Concrete methods are not available and the developers instead have to encode methods into the logical signals. However, with large add-ons and many methods, the technique becomes very clumsy and is therefore rarely used.

The add-on instructions are stored in the PLC processors memory cells and each processor has their own memory set. Tetra Pak has chosen that each processor must have the same set of add-on instructions available. Since the A6 machine system needs seven PLC processors, an add-on instruction must be imported into all seven processors since a software library for add-on instructions cannot be used.

Using add-on instructions brings the advantage to be able to reuse solutions but there is also a downside to using them. Two disadvantages are that machines cannot be online during add-on instruction importing, and to call on add-on instructions from routines takes much longer than if the code was directly implemented in the routine. The Rockwell automation system that Tetra Pak uses today interprets the code once every scan cycle instead of compiling it once before runtime.

A project that Tetra Pak currently conducts is to develop a new graphical interface for the HMI screen. The HMI screen is used to control and read the status of the machine and data is therefore transmitted regularly between the HMI system and the processors. As there is a lot of different program sections that need to transmit data the add-on instructions would be put to good use. However, the excessive use of add-on instructions caused the processor usage to increase by over 20%. They solved the problem by scheduling the add-on instruction calls to not call for each add-on instruction each cycle, thereby reducing the execution load but also reducing the data resolution.

Rockwell's automation system puts little focus on indirect data accessing and they take much longer time than a direct memory access. The consequence is that programmers use less *for* loops with indirect indexing and instead write out the same function several times using direct memory access. This destroys any modularity the code could have, reducing the reusability of functions.

Many of Tetra Pak's Carton Bottle machines are two-sided. Two parallel sides produce filled packages, some machine parts are even further divided from two parallel lines into four parallel lines. For example, there is a machine part with four simultaneous sleeve forming stations. In order to control all four stations at the same time, the program made to run a station has to be copied three times and each copy need to have its signals adjusted. Tetra Pak Carton Bottle has a strict naming convention for signals, depending on which function and

station the signal is connected to. Some of the signals can therefore be renamed easily by using the simple "Find and Replace" method but the rest need to be found by a programmer and manually renamed.

7.2.6. Similar Add-on Instructions

A frequently used electrical component by Tetra Pak machines is the frequency converter. Its role is to transform the grid power frequency to a suited frequency. The converters are coupled to induction motors and control the speed of the motor, the speed is proportional to the converters output frequency. A converter has basic functions such as target frequency, alarm limits and trajectory settings of both the acceleration and decelerations ramps.

Tetra Pak uses two different add-on instructions to control their frequency converters. These two have the same basic functions but they also have their peculiar functions. The actual software implementation of these two frequency converters has been made once in each add-on instruction. Even though there are large similarities between these two, all the basic functions have been implemented twice. It is just an addition of an extra signal in the second block. Using two add-on instructions means an update of a basic function will have to be inserted twice.

7.2.7. Manual intervention in programs

The Tetra Pak product companies develop and maintain machine programs but it is the service engineers that are tasked with installing and updating them on machines out in the field. The developers are responsible for the programs and do not want the service engineers to change around in them but to instead report any problems or possible improvements to the developers. The developers want to keep the machine programs as standard as possible and at the same time make sure solutions reach out to all machines. However, during new machine installation and machine updates the service engineers sometimes have to modify the programs. For example, there is a risk that the program needs to be modified due to the customer's external equipment. If the customer has a product supply system that differs from Tetra Pak's standard then the program interface between the system and the filling machine will differ.

Another occasion where a service engineer might have to modify a program is during rebuilding kit installations, a kit is a collection of mechanical, electrical and software updates. The developers provide what the service engineers are supposed to update or add. Since kits are released continuously and not all of them are installed on each machine it makes the machine programs differ from one machine to another. This brings a risk that the installation manual will not match the program and the service engineer will have to search through the programs to find the correct sections. Because of the differences in the machine program the service engineers cannot use a simple find and replace tactic. The service engineer has to use his knowledge to update the machine correctly. There have been cases where service engineers have made changes on his own initiative, as he believes his solution is better than the one provided by the developers.

As mentioned before, if a service engineer discovers a bug or a section for an improvement then he should notify the product company and they will assess whether the program should be updated or not. The engineers should not perform any updates on their own initiative, if they do not follow these guidelines new errors could be introduced which will be difficult to locate for others. Another disadvantage is that the knowledge will not reach the product company and therefor prevent other machines from receiving the update. Mostly the service engineers do not have to know how the updates work, as they are there to install them and not to modify them. However, a situation that an engineer has to trouble shoot and understand the update is when an update does not perform as planned. On those occasions it is important that the machine program has been kept as original as possible without any undocumented changes.

7.2.8. HMI

All Tetra Pak machines are equipped with a HMI system to let machine operators and engineers to control the machine. For the HMI to carry out the correct action, communication between the HMI system and the machine system is required. However, both systems of the A6 machine system are very nestled into each other, meaning that an upgrade in the machine programs will require an update in the HMI program as well. For example if a new alarm is added to the machine system and not to the HMI system then there is a risk that the machine operator will stand clueless if the machine stops due to the new alarm. As the HMI system has not been updated, the triggering alarm will not be presented on the HMI screen. As the alarm is not displayed on the screen the technician cannot acknowledge it, and any attempt to start up the machine will not work.

7.3. Object-oriented architecture analysis

The A6 machine program is structured fully when it comes to modules and their functions. However, the depth level is currently very low. The routines and its functions should be broken down further as the routines are large and contain much of the logic. For example, a servomotor routine contains alarm functions, exceptions and safety logic. Those are the functions that mechanically operate the servomotor and the functions that evaluate when and how the servomotor should be used based on the state of and in which steps the machine is in. All these functions and the associated variables are located within the same routine.

When developing a routine, the developer is mostly interested in a few functions but still all the other functions and variables are there and which he has to take into consideration as to not break the them or even break other routines. If all functions and their variables were encapsulated in smaller classes the programmer could worry less about introducing bugs and just focus on his work.

7.3.1. Class Library

An object-oriented program consists of an executable program that imports classes and instantiates objects from them. The classes are independent items that encapsulate functions and variables together which makes them easier to store in class libraries. The libraries helps developers reuse software solutions, instead of implementing an already existing function from scratch the program creator imports a class with the same function from the library. The program creator can then straight away take use of the class as it has already been tested before.

Similarly, as a programmer can assume that the classes he imports are working correctly so can a service engineer assume that an update of a machining program will work correctly as he imports an updated and tested class. A benefit with using classes is the lack of manual intervention in the programs when updating, which reduces the errors a service engineer can introduce. If an error does happen during start up the service engineer can worry less about if he inserted the update correctly and instead focus on troubleshooting the functionality.

7.3.2. Programming guidelines and encapsulations

Having a proper and organized structure in a function-oriented program is evidently important because today's automation solutions can be very large with massive amount of variables and functions. The growing program size makes them difficult to understand and maintain. Therefore, proper programming guidelines should be established and followed. But without proper encapsulation the safety of routines and functions cannot be guaranteed. A programmer, who creates a splendid routine that is extremely structured, will ease the understanding for others. However, there is nothing that stops someone else from ruining the splendid routine by mistake from another routine.

7.3.3. Hierarchy

The routine level of today's A6 automation system is the deepest program level. If encapsulation of functions is sought for then they have to be located in new routines, but on the same level. It is possible to call on another routine from a routine. However, this creates a long list of routines where it is difficult to see which belongs to

which. In an object-oriented program the structure is hierarchal, which is appealing as a user can immediately understand which object consists of which, see Figure 9 for an example. The creation of depth levels is another advantage that is made easier to implement with the help of encapsulation. Figure 9 shows an example of how a routine that moulds a plastic top could be broken down into further levels in an object-oriented program. Each object in the figure is created from a class, for instance the two hotrunner objects are created from the same hotrunner class. Each object contains the methods and attributes that are declared in the corresponding class. Attributes in objects can be kept private and changing them can then only be done through public methods, methods which can disallow incorrect modifications of the attributes. For instance, Hotrunner 1 will need to use a public method of needle cylinder 1 to access one of the needles private attributes.

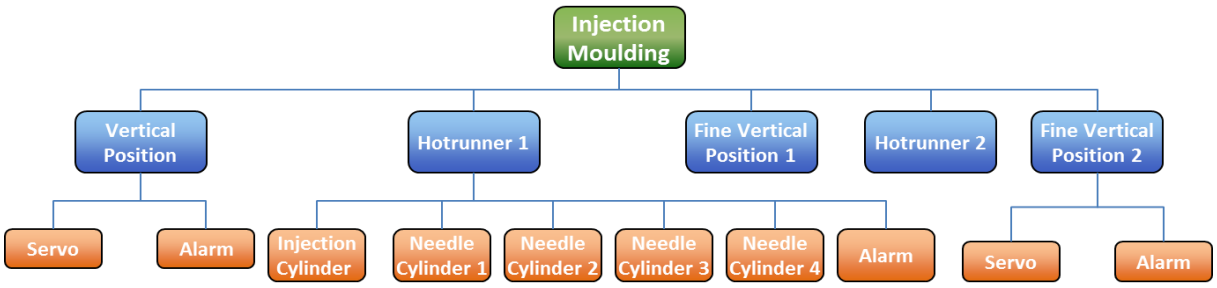


Figure 9 Hierarchy over the A6's Injection Moulding module.

7.3.4. Reusability

The reusability can also be seen in Figure 9, there are two Hotrunner objects per Injection Moulding unit but only one Hotrunner class is implemented and maintained. Any updates made to the Hotrunner class will be reflected in both Hotrunner objects. In today's machine program an update to the Hotrunner has to be entered not twice but four times, as the machines has two sides and each side has an Injection Moulding unit. In the same way will the software of the Injection Moulding units only be maintained once, as the Injection Moulding unit is also an object instantiated from an Injection Moulding class.

7.3.5. Reusability, inheritance and local adaptions

The reusability in the previous section is easily done because the exact same objet is needed in many program sections. However sometimes two functions are needed that are not exactly the same but are still similar. In object-oriented architecture this is solved by using inheritance. A new class is created by inheriting everything from a parental class. The new class will retain all the methods from the parental class, and the developer can then implement any changes into the new class. The advantage of using inheritance is that any change to the parental class will automatically transfer to the new class as well. Inheritance creates a hierarchy within the class library where the top level will present a class with standard functions while deeper levels show classes with methods more specific to one subtype component. Inheritance could be used for the two types of frequency converters to remove the duplicate work needed to implement and maintain them.

Another area where Tetra Pak could use reusable classes and inheritance is for their various machine products. Many of their filling machines have similarities, and often when a new filling machine is developed the technologies from previous machines are reused. Instead of writing completely new programs for a new machine they could be created by inheriting one of the top level classes of a possible Tetra Pak's class library. For instance, both the A6 and Tetra Top machines use an injection moulding unit, and the units have big similarities in appearance and functionality. A basic moulding unit class could be created from which both the Tetra Top and A6 machine inherit functionality.

The developers of Tetra Pak product companies want to limit the amount of code that the service engineers can modify in the machine programs to keep programs as original as possible. Classes and inheritance can reduce it and at the same time keep original classes and adaptions separated. The developer classes can be kept unchanged and instead the adaptions are inserted in local adaption classes which inherit from the original

developer classes. In the new classes the local adapted methods can overwrite the original methods while leaving the remaining original methods intact. The advantage is that the adaption class will receive any update made to the original class as long as it has not been overwritten.

7.3.6. HMI

Today's HMI system is programmed to specifically look through certain register and check if alarm bits have been set to high. For every added bit to the registers the HMI system has to be updated with a new alarm. In an object-oriented program the alarm trigger process would be handled easy by using an object list. Whenever a new alarm is triggered the object would be moved to the list. The HMI system could thereafter communicate with PLC that tracks the list. The PLC would transmit data to the HMI system whenever the HMI asks for the next triggered alarm and once the list is empty then the PLC would know that all the alarms have been taken care of. For such a system to function the transmitted information between the HMI system and the PLC processor would have to be standardized

8. Automation Machine Variants

8.1. Conclusions and requirements

Tetra Pak's goal is to offer their customers the machine functionality that they want. To do so Tetra Pak creates and maintains various machine variants of their machine systems. With time, Tetra Pak created so many machine variants that the additional cost of maintaining them became too large. And so Tetra Pak decided to limit the amount of machine variants, for instance the Tetra Top machine variants were merged into a new single machine variant. The new variant received the newer and more advanced functions from the previous machine variants. The solution was both positive and negative for the customers, those who wanted a machine with more flexibility could receive one for a cheaper price while the price went up for the customer who needed a simpler machine.

One of the reasons for the costs of managing machine variants is the program updates. Each machine variant has a different master program. If a common function between the machine variants is modified then each of them must be updated and tested. The automation updates are inserted manually into each program and the tests are to make sure the insertion was done correctly, as the actual functionality of the update has been tested on one machine previously. The tests are performed using customer machines and are very expensive. The larger the update is the more difficult it is to find a customer with a suitable machine and sometimes the tests have to be split and carried out at multiple locations which causes additional costs. The help of standardized interfaces and modularity of an object-oriented architecture would help Tetra Pak create one master program to reduce the amount of field testing that is needed.

The requirements that follow below are derived from my interpretations of the inputs collected from the automation engineer Johan Helander.

Automation Requirements

- Organization work of variants shall be reduced
  - Only one master program shall support all variants
  - Variant functionality shall be kept separate from master program
- Field test duration and amount of tests shall be reduced
  - Program code intervention shall be kept to a minimum
  - Only one verification test of manual code intervention shall be performed
- Technical support effort of variant maintaining shall be reduced
  - Update installation time shall be reduced
  - Local machine deviations shall not increase maintenance time of machines

8.2. Stakeholder input

8.2.1. Machine Variants

Tetra Pak offers a wide range of machine systems. How far a machine system has been developed is shown with a development step, starting at the number 100 which increases by 100 for each new development step. The step is increased when a significant change has been made to the machine system, for instance when the Tetra Top machine system started using servomotors instead of a mechanical axis the development step was increased to 1800. Once a new development step has been released to the public, then Tetra Pak will stop selling machines in the previous step. However they will continue to support some of the older ones. A machine system in one development step can have many variants. The different machine variants are developed and sold to customers who are looking for additional functionality for their machine. One variant of the Tetra Top machine offers the customers the possibility to fill two products simultaneously by using two product tanks instead of one. A new variant is created by copying the current development steps hardware and software and inserting the new functionality into the copy. Each machine variant has therefor its own basis, which increases Tetra Pak's costs significantly by having to administrate all machine variants separately.

8.2.2. Testing functions

Before a machine update can be released it has to be tested on a customer's machine. An in-house machine will detect less errors, for instance errors cause by machine wear are less likely to be detected. Tetra Pak tries to test as many updates as possible at one location to limit costs. But it is not always possible as machines out in the field have settings, adaptions, and machine variants that differ. If the service engineers are looking to test function F and G and they cannot find a machine with both the functions activated they will have to run two field tests. The testing plan is to run one field test occasion for as many functions as possible to make sure they work and then to install them in all the needing machines. This modular thinking becomes risky as more variants and functions are released. Making sure an updated function works correctly in one configuration of a variant does not conclude it will work equally in another variant and its configuration. Testing every possible variant and setting configuration is however not economically feasible.

8.2.3. Release trains

After an update has been approved by a field test it has to be entered manually into the master programs. If the update concerns a common function between the different variants then it has to be entered into respective master program. The master programs are updated continuously and after a set of updates has been accumulated then additional field tests are performed, one for each machine variant. These field tests are to make sure the master program was updated correctly and that the updates function together. Obviously with more variants more field tests have to be performed, tests which are very expensive.

8.2.4. Tetra Top machine system

The Tetra Top machine system is an excellent example when it comes to machine variants, as it has been out on the market for over 20 years. The development step of today's sold Tetra Top machines is in the 2000 step. However, Tetra Pak still supports and maintains machines as far back as the 1700 step.

8.2.5. The Tetra Top 1800 development step

The Tetra Top 1800 step machine had a new function called XH, extended hygiene, which increased the shelf life of certain products. The 1800 Tetra Top machine was released before the time when Tetra Pak started using servomotors and so a new variant of the 1800 step was created which used the servo technology. As the development continued a new function called OSO was invented for the Tetra Top machine, a one-step top opening device instead of the normal two-step one. The customer requested the new OSO function for their 1800 Tetra Top machines and thereby two additional variants were created. Tetra Pak now needed to run four field tests whenever they wanted to update the 1800 Tetra Top machines. Creating new variants is nothing that is planned and it is often a result when new technologies emerge.

8.2.6. Too many variants

The 1800 development step and its variants is just one example out of many possible ones. The Tetra Top machine system has had a lot of variants created, so many that over time the cost to organize, test, and maintain them became too large. Tetra Pak chose to merge and eliminate many of the available machine variants to reduce the cost for Tetra Pak. They merged the better and more advanced functions of the previous machine variants into a new development step. If they had not chosen to reduce the amount of machine variants available then the Tetra Top machine system would have 80 different variants today, an amount too expensive to maintain.

As of today, there are not any variants of the Tetra Evero machine system, but new features are being developed, for example, to be able to fill oxygen sensitive additives such as omega-3 or a one-step opening system for the lid.

8.3. Object-oriented architecture analysis

Today's machine programs are designed to fit one specific machine development step or machine variant which makes it difficult to reuse programs. When a new variant is created, Tetra Pak completely copies the current systems implementations and implements the changes in the new copy. The drawback is that it creates several independent machine programs that have to be updated on their own. The programming code has to be entered manually into each program, even if it concerns a common and identical function.

8.3.1. Modularity

Tetra Pak is trying to handle the testing and updating of machine variants in a modular fashion. They insert a function into one program and run tests on it, and afterwards they copy and insert it into the other machines which also run the same function. They hope the function will work correctly and that there will not be a mismatch between the machines caused by machine differences. However, their current function-oriented automation system lacks encapsulation and therefore requires additional work compared to an object-oriented system to be able to function as a modular system. The encapsulation of data and functions into classes makes the different program sections operate more as standalone units which are more resistant to errors caused by out of function interference. The ability to import classes would also remove much of the manual code insertion that is need today when machine variants are maintained, which would also reduce error sources.

8.3.2. Modularity – One master program

Another advantage that comes from the modularity of an object-oriented system's classes is the separation of the executable programs and its classes. As the executable programs are used more as shells where classes are imported to, the classes dictate what should be run while the executable program decides when it shall be run. This could be used to only have one master program as a basis where all the different machine variant classes are imported to. This would reduce the spending's on administration and maintaining machine variants.

9. Supply Chain Operations

9.1. Conclusions and requirements

Tetra Pak spends a huge amount of money on system managing their new machines before they are shipped to customers. A part of the large expense comes from a supply chain step that every machine has to go through, a final assembly step at Tetra Pak in Lund which happens between the module suppliers and the customer. Here the machines are assembled and tests are run before they are delivered to the customers. The work performed at this step adds no value to the functionality of the machines, it is just a step to make sure the machine will be delivered to its customer without the possibility for larger malfunctions.

The work performed at the Tetra Pak product companies' costs a lot more than if it was done at the module suppliers. Therefore Tetra Pak wants to outsource as much as possible of the work to module suppliers, meaning the work that it is not possible to entirely eliminate. Another reason to outsource machine tests is the importance of time, in-house at the Tetra Pak product company the machine is very near delivery to the customer and a malfunction may delay the machine as well as cause additional overtime costs. Test running at module suppliers where the needed competence and spare parts are available would yield less costs and delay for Tetra Pak. A mean to lessen the supply chain cost is to make the machine modules testable and runnable stand alone and thereby making it possible for the module suppliers to test the machine parts instead.

The requirements that follow below are derived from my interpretations of the inputs collected from the supply chain engineers Thomas Wåhlin and Oscar Cruce, and the production engineering & QA manager Anders Hansson.

Supply Chain Operation Requirements

- Module supplier shall deliver modules directly to the customer
  - Modules shall be tested and ran standalone
  - Modules shall use their final electrical components at the module supplier
- Electrical component failures shall be discovered early in the SCO chain
  - Electrical component burn in phase shall start at the module supplier
- System tests and electrical component configurations shall be outsourced to module supplier
  - Teaching and calibration of electrical components shall take place at module suppliers
  - Module suppliers shall test their modules using the modules final electrical components
- Supplier electrical cabinet components shall not become obsolete
  - Module suppliers shall test their modules using the modules final electrical components
- Work orders shall have their automation part synced with the mechanic part
  - The master program shall support all types of machine variants and kits
- Work orders shall not delay machine delivery
  - The master program shall support all types of machine variants and kits
  - Software update shall be installed safely

9.2. Stakeholder input

A new machine order has to go through many steps before it reaches the customer. Figure 10 shows the supply chain of Tetra Pak's various machine systems. The main areas of Supply Chain Operation of Tetra Pak Carton Bottle are module suppliers and system assembly at Tetra Pak in Lund.

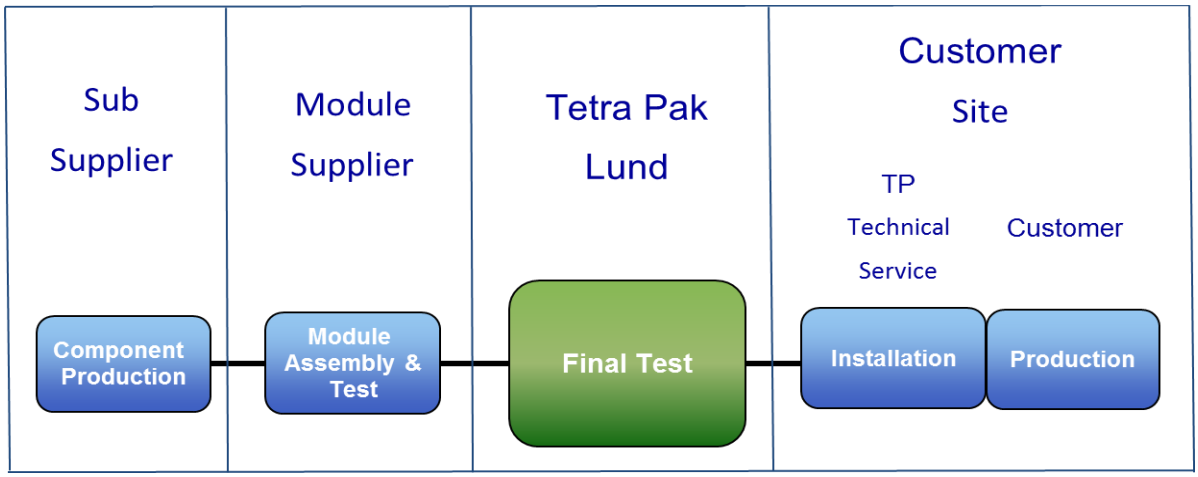


Figure 10 Supply chain of Tetra Pak's machine systems

9.2.1 Module supplier

Overall, when Supply Chain Operation receives an order for a machine they contact their module suppliers who are responsible for delivering assembled and tested modules to Tetra Pak. In turn the module supplier contacts their suppliers with orders for machine components. Some of the component suppliers are chosen by Tetra Pak to guarantee a specific price, quality and system developer. For example, Tetra Pak uses Rockwell Automation as their system supplier of PLC and automation components and so shall the module suppliers do.

9.2.1.1 Assembly

Once the machine components arrive at the module suppliers, the suppliers start to assemble modules and prepare them for testing. Various tests are then performed on the modules before they can be shipped to the Tetra Pak product company in Lund.

9.2.1.2 Testing

Each module supplier runs smaller tests to calibrate the more coarse settings of the module. A few electrical tests are run and mechanical configurations are made, essentially all the tests and configurations that are possible to make without the remaining modules. The testing takes place at the module suppliers but it is Tetra Pak who owns, pays for, and develops the tests and they have their own employees at the module suppliers.

Testing - Electrical cabinets and their components

To be able to run and control the modules each module supplier has an electrical cabinet with electrical components. The electrical PLC processors have the task to control the module but are also used to simulate adjacent but not present modules. Simulators are also used for module suppliers that produce more than one module as is not worth the time or money to connect modules together.

Tetra Pak has chosen to deliver the final electrical cabinets and electrical components directly to the Tetra Pak production companies instead of delivering them first to the module supplier. The suppliers must therefore use custom-made cabinets, which stay with the supplier throughout orders. Over time, the machine modules receive updates that the cabinets have to reflect and therefore new components replace the old ones while

others are removed as the module no longer needs them. The replaced and removed components are then discarded as they are no longer needed.

Testing - Calibration of electrical components

Tetra Pak's electrical cabinets contain a wide range of electrical components. E.g. the more standard components: solid-state relays, I/O nodes and fuses but also the more advanced ones such as servo amplifiers, frequency converters, vibration generators and induction generators. Electrical components control various machine functions, for example when a solid-state relay receives its control signal it switches on and leads power through the relay to a thermal element. For instance, one of the functions an element has is to together with an extruder to melt plastic that will be used in tops.

Many electrical components have to be calibrated together with their mechanical counterpart to properly operate. Servo amplifiers and solid-state relays are examples of such. A servomotor needs to be rotated to its mechanical zero position and the offset value has entered into the machine program. The resistance of a load that is connected to a solid-state relay varies from load to load and the current passing through the solid state has to be entered into the machine program to help detect errors.

Since the module supplier electrical cabinets are stationary and do not follow with the module to their next destination, the configurations will have to be redone once more at the Tetra Pak product company.

Testing - Electrical component grouping

The grouping of the electrical components in the five electrical cabinets of the A6 machine is based on component type and module. All the solid-state relays from several modules are embedded on the same rack in one cabinet. The grouping is done the same way for most electrical components. A module of the machine carries out a bigger function of the machine, for example the ASU & sheet cutting module, sleeve forming, moulding tops and so on. However, the division of the electrical components in the electrical cabinets do not follow this modulated principle.

9.2.1.3 Module shipping

Each supplier ships their module to Tetra Pak Carton Bottle in Lund once the tests are complete. Kostwein is one of the A6 machine module supplier and they produce four out of the total eleven modules. They are responsible for the ASU/CUT unit, the supply unit, the RTF machine part and the capped neck supply unit. Tetra Pak's goal is to deliver these modules straight to the customer and thereby skip the test phase at Tetra Pak in Lund. They want to achieve this by testing the four modules as one, by docking them together which essentially creates half of the complete machine as opposed to testing one module at a time.

9.2.2 System Supplier, Tetra Pak Carton Bottle Lund

9.2.2.1 Assembly

Tetra Pak starts assembling the machine once the modules start arriving at Tetra Pak Lund, a process that takes about 15 days for four engineers to complete. The modules are docked together by following the interfaces between the modules. The interfaces are clear but differ from module to module as standard ones are not used. During docking all the cables and pipes are mounted on the machine. The cable connecting work alone takes a very long time as all five electrical cabinets are mounted and require wiring between them and the machine.

9.2.2.2 Start-up Phase

During the start-up phase, the machine is prepared to run a light production, one where quality is not checked. The machine preparation starts with a safety test of the emergency system, followed by opening valves for systems that provide ice water, drinking water, steam, air and hydrogen peroxide tanks. All settings made for the electrical components at the module suppliers must be repeated once more, but this time with the new

cabinets and electrical components delivered by Tetra Pak Processing. At the end of the start-up phase a small production shift is ran, to verify that all coarser settings have been made. The start-up phase is followed by a fine tuning of the machine in the machine performance step.

9.2.2.3 Machine Performance

During the fine-tuning of the machine performance step the functions of the machine are reviewed in order to ensure package quality. Test engineers take out samples of the packages after each machine step and perform tests on them to find deficiencies. Every found error is corrected by adjusting the machine parameters and a new test is run.

9.2.2.4 System Performance

System performance is the last step performed on a machine before it is packed and sent to the customer. During this step, three validations are carried out to ensure the machines efficiency is high enough. During these three validations, no adjustments that affect the integrity of the package should be done. If however, adjustments need to be made that affect the packaging integrity then there is a risk that the entire validation process must be repeated. The test engineers focus mainly on the MTBF, mean time before failure and MME, mean machine efficiency values during the system performance process.

9.2.2.5 Handing over the machine

Supply Chain Operations hand over the machine responsibility to the technical service team once they have completed their tasks and shipped the machine to the customer. Technical service engineers then begin to assemble and prepare it for production before finally handing it over to the customer.

9.2.2.6 Outsourcing

Tetra Pak wants to move tests and calibrations that are done at Tetra Pak Carton Bottle in Lund to the module suppliers instead, as the labour is more expensive than what it would be if the tests were handled at the suppliers.

Some of the points to why outsourcing tests should be done are that the suppliers keep a bigger stock of spare parts while machines which are tested at Tetra Pak are just about to be delivered to their customers and time is of essence, and any detected errors at this stage mostly indicate overtime work. Supply chain operations want to outsource as much as possible to their suppliers. Tetra Pak is a system supplier, however all the system work taking place at Tetra Pak does not add any value to the machine, it is just a way to become certain the machine is operating in a good manner before being shipped to the customer.

9.2.2.7 Work orders

Since the machine systems are continuously being improved, new update orders are issued for machines located both at customer and for those under construction by Tetra Pak and the module suppliers. A work order is an update to the mechanical, software or electric part of the machine. It is Supply Chain Operation that is tasked to perform work orders if the machine has not been shipped to the customer. The principle is that work orders should be performed as early as possible in the supply chain to minimize the costs and machine delivery delay. Tetra Pak sends out a request to the module supplier asking if they could install it. If they do not have the time, the job falls to the installation engineers in-house at Tetra Pak. A machine located at Tetra Pak Lund implies it is very near delivery to the customer, meaning it is very important that work order installation is done quickly and correctly. One of the troubles supply chain operation faces today with work orders is the dependency between mechanical updates and automation program versions. For example, if the machine program is running an old version that does not support a recently updated mechanical function then the program will have to be updated manually by an automation engineer if the mechanical update is installed. While if the machine program was of the latest version then the install engineers would only have to activate the program section that controls the recently updated mechanical function.

9.3 Object-oriented architecture analysis

9.3.1 Modular structure

Tetra Pak handles the mechanical aspect of the TPA6 machine system in a very modular fashion. A6 machine parts are produced, tested, maintained, and docked together in a modular fashion. Even the machine program is fully structured in a modular way, where a couple of small module programs represent a mechanical module. However, the electrical components organization in the electrical cabinets does not follow the modular structure.

9.3.2 Component plates

A direction Tetra Pak could use to reduce their in-house costs would be to use a more object-oriented architecture for their electrical cabinets. Tetra Pak Processing could mount electrical components on a component plate and then deliver the plates to module suppliers. Depending on the size of a module, a number of plates would contain its electrical components. The electrical cabinets would instead have empty slots where plates with electrical components can be plugged into.

This could create a standard for the electrical cabinets and the plates, making them cheaper to produce. Tetra Pak Processing is today responsible for producing and testing electrical cabinets. They make sure the cabinets that arrive at Tetra Pak have had all their components and the cable connections tested. If Tetra Pak chooses to use component plates then Tetra Pak Processing could test each one of them so when they arrive at the module supplier they are ready for use. The supplier would plug them into their empty cabinets and start using them immediately to test their respective module. Once the tests are finished the component plates would be shipped together with the modules to Tetra Pak Carton Bottle in Lund for final testing.

By sending plates to module suppliers the actual mechanics will be tested and run with the correct set of electrical components and thereby skipping those test at Tetra Pak Carton Bottle. Another advantage of component plates would be that electrical component are calibrated with their counterpart only once and at the module suppliers.

9.3.3 Burn in phase

Using component plates would mean that the electrical components are tested at an early stage in the production chain, and with an earlier burn in phase the chance to catch defect components increases. Electrical components have an estimated life length and those that do not live through their full life length mostly break down very early. Catching defect components earlier in the process leads to fewer machine delays as the module suppliers have a wider stock of spare parts than Tetra Pak Carton Bottle has.

9.3.4 Cabinet overview

A more modular architecture of electrical cabinets would make the overview of them easier. With a simple glance inside the cabinets, the engineer would understand which electrical components belong to which module and function. Depending on how large the module plates are and when an update is to be performed, an engineer could either remove the whole plate and install a new one or remove the plate, move to a better work bench and make the modification on it before plugging it back into the cabinet. If the machine is close to customer delivery a quick replacement of the entire module plate might be favourable.

9.3.5 Automatic configuration

Since a component plates would be linked to a specific function of a module then the plates could be used as a configuration tools for the machine and its program. When the machine detects a component plate it would understand that the associated mechanical function shall be checked for and the machine program configured. For example if the machines detects two component plates and each of them is used to control a product tank then the machine should configure itself to run two different products, one on each machine side. With standardized interfaces and the modularity of an object-oriented system the machine program would configure

itself by importing a two-tank class and then to instantiate a two-tank object from it. Today a change in the cabinet and mechanical setup requires a manual update of the program.

9.3.6 Dependencies between mechanics and software

An Object-oriented architecture is a good mean to remove the dependencies between mechanical updates and software versions. Today the new and updated functionality is inserted into new master program versions and the programs are thereafter released at various occasions, often months apart. When a mechanical part is installed then a supply chain engineer will activate the program section associated with the mechanical update. The trouble in the approach is when mechanical updates are installed on older versions which lead to extra work for automation engineers. In an object-oriented approach the implementation of functions is placed in software classes which are then imported to the programs. If focus was put on the machine programs to use standardized interfaces there would not be a need for automation engineers to manually update an older program version. They would instead import the new automation update in form of a class, a class that shares the same interface as the version before.

10. System Validation

10.1. Conclusions and requirements

The system validation team is responsible for organizing tests of new and updated machine functions. To do so Tetra Pak keeps a number of test machines available to the developers in-house. When the A6 machine system was still in the project phase there were up to five A6 machines located in the test hall of Tetra Pak Carton Bottle. Since the test machines are large investments for Tetra Pak it is important that they are utilized as fully as possible. But there are cases were test results have to be discarded, for instance, because of machine configuration mismatches. The A6 test machines do not use a versioning system as it is not economically feasible. A drawback is that it is more difficult to spot if a finished test has been rolled back to its pre-test configuration. The system validation wants to reduce the time it takes to organize and carry out tests when using the test machines. An object-oriented architecture would help do so by using standardized interfaces and its modularity. It would allow function tests to be performed regardless of the current machine program versions and also reduce the manual work that is needed when machines are being restored back after a test.

Another System Validation interest is the test rigs, which are used to test and verify functions. For the System Validation team it is important that mechanical, electrical and software solution implementation are transferable between test rigs and machines with ease. For example, software solution should need as little translation of code and virtual signals as possible. The transferring of implementation happens continually during development and a modular system with standardized interface could help a great deal, both hardware and mechanical. For instance, the whole injection moulding unit of the A6 machine can be disconnected and rolled out from the machine. It would save much time if the IM unit of an IM test rig could be rolled out and into a test machine without the need for mechanical rebuilding work.

The requirements that follow below are derived from my interpretations of the inputs collected from the system engineers Fredrik Mattson and Robert Hansson.

System Validation Requirements

- Test Machines shall be used more efficiently
  - Master program manual code intervention shall be kept to a minimum
  - Software implementations shall be stored separately from the test machines
- Time spent on transferring rig solution to machine shall be reduced
  - Software shall be compatible between simulator, rigs and machines
  - Software shall be reused between smaller and bigger rigs
  - Software shall be compatible when using different component models of the same type

- Validation tests shall be fast and safe
  - New software shall be installed using minimal code intervention
  - Complete and partial software restoration shall be possible with minimal work
  - Partial software restoration shall be made with minimal chance for errors
- Test machines configurations shall be managed accurately
  - Machines configuration shall be tracked using versioning

10.2. Stakeholder input

10.2.1. Machine verification

The employees of the System Validation department are responsible for organising tests of new and updated machine functions in-house at Tetra Pak's product companies. System Validation keeps track of the machines and their configurations so that when a party wishes to run a test they will know what functions are installed and the version they use.

An update that is up for testing can for example be the result from a field request, where a service engineer has sent in a bug report. The product company of Tetra Pak will make the decision if they wish to prioritize the bug, if they decide to do so then several steps will need to be fulfilled before an update is released. First out is a test readiness meeting held by the System Validation team to discuss a possible test. Some points of the meeting are, whether personnel with the correct knowledge will be present, if relevant requirements have been made, and will the correct test method be used. If the test is approved by system validation then the party will receive a time slot for a test machine to run their verification. After the test a signoff meeting is held, where the requirements are looked through to verify if they have been solved. Here it is decided if any missed requirement should be retested. If the change is considered critical then Tetra Pak will want to have it released and installed quickly, in such a case a kit and work orders are created

10.2.2. Version dependencies

One thing that can delay a test is the test machines configuration status and version. The functions that are present and can be enabled on a test machine are determined by the machine programs version. A newer version will have more functions available which can be enabled, For instance, a new function A must have automation kit B installed and enabled. But the master program of the test machine does not support kit B, an automation engineer will then have to manually implement kit B before the verification test can be run.

10.2.3. Test machine restoration

For the System validation team to accurately keep track of test machine's configuration it is important that the test machines have test updates installed correctly and that they are removed properly after the test. A complete rollback of a test machine is easily done with a backup import but a partial rollback requires some additional work. To test a new function the machine program will most likely have to be updated in several sections. In a partial restoration a number of such sections will have to be restored to pre-test condition while others will be left intact. Since the restoration work is done manually then there is a larger risk for human error. For instance, in a previous test occasion it happened that an alarm that was supposed to be turned back on was left disabled. All the tests ran on the injection moulding unit at later occasions did so with the alarm disabled and it was first when a new machine was deployed at a customer site that the fault was discovered, as the new machine kept triggering the alarm.

10.2.4. Updating master program

After a successful function test the master program will have to be updated with the functions. The responsible automation engineer will have to insert the function manually and the updated master program will have to be tested to make sure it was updated correctly.

10.2.5. Rigs

Tetra Pak uses test rigs when they develop new machine systems. Test rigs are used to test specific functions of a machine. For instance, The A6 team has a test rig called one-shot rig and it is used to test if it is possible to mould a plastic top with the characteristics that make it suitable for an aseptic machine. It is called one-shot because it cannot run continuously and needs to be reset after every moulded top, therefore this rig cannot be used for testing repeatability characteristics such as thermal load on the mechanical components. Once a function has been tested and approved using a rig it has to be validated in the actual machine. The concepts used in the test rig will be used for the machine design. The test rigs are also used to continuously improve and update machines and any improvements made using a rig will need to be transferred to a test machine before it can be released. It is therefore important that the transferring from rig to machine goes as smooth as possible with minimum time spent.

10.3. Object-oriented architecture analysis

One of system validations concerns is the restoration of test machines after testing, it is important to carry it out correctly so that the configuration and status of the machine can be tracked. Knowing the configuration will keep errors due to configuration mismatch low.

A complete restoration after a verification test is done easily by restoring a pre-test backup. However, a partial restoration is difficult to do. Using an object-oriented architecture the restoration could be handled by restoring classes back to pre-testing state while leaving the remaining. This is opposed to what is done today where restoration is done by locating rungs and restoring them manually.

10.3.1. Versioning

Another advantage of an object-oriented approach to the system validation team is the ease of versioning. In today's architecture the routines are updated by manually inserting program code into them. It is done this way because it is difficult to anticipate which and in what order updates will be installed, and simply replacing whole routines with new ones can erase local adaption or update functions it should not. An object-oriented system eases these points as local adaptions can be taken care of by using special made local adaption classes which either replace the original class or inherits from it. Functions without adaptions will be updated by importing and using new classes.

10.3.2. Transferability

The rigs used by Tetra Pak can be likened to building blocks, where blocks are used together to construct a large house. The rigs are used the same way, small rigs are developed to verify smaller functions, and as development continues larger rigs are constructed that use the same type of hardware as the smaller rigs. However, this is not always the case, for example the one-shot rig did not use the same extruder type as the machine and another smaller rig was used for testing the correct extruder. The code transition from one rig to another is therefore important and it would be handled with ease using an object-oriented system. Object-orientation puts a lot of focus on standardized interfaces, were a program can use any model of an extruder as the interfaces are kept the same. Not only would an object-oriented approach help the transition between rigs, but also between rig and machine. The rigs are just a mean to test various technologies before implementing them in the machine and standardized interfaces would help a lot by simplifying the transferring of solutions from the test rigs to the machine, both hardware and software solutions. For instance, the whole injection moulding unit of the A6 machine can be disconnected and rolled out on wheels from the machine. It would save much time if the IM unit of an IM test rig could be rolled out and into a test machine without the need for mechanical rebuilding work.

11. Technical Service

11.1. Conclusions and requirements

Tetra Pak is very keen to release new machine systems on the market as quickly as possible, in order to secure new customers and to make money. Thereby the machine systems are developed in a fast pace and in most cases they do not achieve their specifications when they are released. In order to bring the machines up to their specifications a huge amount of money is put into technical support, money which is spent on kit installation on newly released machine systems. A possible action to reduce the cost could be to reuse as much as possible from previous machine systems, which would reducing development time and the quality would be kept.

Another concern is the rebuilding kit quality. As the quality is not high enough it happens that kits bring trouble to the machine owner. For instance, there have been cases where the kit did not work, the wrong kit was delivered, or that the machines performance was reduced. These points and that customer machines may need to be stopped for longer periods during kit installation can lead to customers not wanting to install them. As a result the local Tetra Pak companies have a deadline of nine months to install new kits. The local Tetra Pak companies will try to plan in kit installations with customers, but often it is too difficult to do so as newer kits will be released over the nine month period. Some kits will become obsolete and others will not need to be fully installed. It will become difficult to utilize the scheduled time and so Tetra Pak will instead try to squeeze in kit installations between production runs. To plan a kit installation is more difficult than the actual installation.

Some of the performed work during installations adds no value, for example, the automation part of a kit has to be manually inserted into the machines program. This action adds no value to the machine but instead creates a potential error source which has in the past led to malfunctions, causing more work to be required to troubleshoot the machine. Manual code implementation also opens up the ability for the service engineer to slip in a personal solution. Individual modification of machine programs will cause problem for the next engineer that has to maintain the machine as he will have a more difficult time understanding the machine.

The requirements that follow below are derived from my interpretations of the inputs collected from technical liaison Fredrik Larsson and rebuilding kit coordinator Christofer Malmros.

Technical Service Requirements

- Software updates shall be safe and installed fast
  - Software code shall be installed with minimal manual code intervention
  - Software updates shall take minor time
- Machine downloads shall not interfere with local adaptations
  - Software updates shall not overwrite program deviations installed on previous occasions
  - Local adaptations shall be kept separately from the original program
- Machine configuration status shall be traceable
  - Versioning shall be used
- Software errors shall not ruin other program sections
  - Variables shall be encapsulated
- Quality of released machine systems shall be increased
  - Solutions shall be reusable
- Local company support shall be reduced
  - Software updates shall be preloaded automatically
  - Software shall be restorable
- Malfunction errors shall be traceable
  - Variables and functions shall be encapsulated

11.2. Stakeholder input

11.2.1. Technical service and kits

One of Technical Services tasks is to install rebuilding kits. A kit is a collection of mechanical, software and electronic parts used to update a specific machine system and they are installed on machines located in customer factories. The kits are divided into various categories, for example a mandatory correction kit can concern areas such as legal requirements, safety or critical errors. Tetra Pak installs and pays for the installation of kits but it is up to the customers to decide if and when they want a kit to be installed.

11.2.2. Fast development of machine systems

Tetra Pak is very keen to release new machine systems to the market as quickly as possible, to secure new customers and to start making money. Thereby the machine systems are developed in a fast pace and in most cases they do not achieve their specifications. As a result a huge amount of kits are released to bring up a machine system to its specifications. For example, still after a year of field testing the A6 machine system there are several updates released per week to it. All the support spent on maintaining and installing kits is huge expense for Tetra Pak, and even though the A6 machine system is in need of many updates it is still one of the best machine system releases Tetra Pak has ever done.

11.2.3. Customers and kits

Since kits are released to improve machines which are already located at customer sites, it is up to the customer to decide if and when he would want the kit to be installed. The customers' main goal is of course to use his machine to produce packages. Obviously a customer will not want to install a kit if his production has to stop and if he sees no problem with his machine.

For smaller kits it can be enough to momentarily stop the production to install a kit and then resume the production. While for bigger kits the machine may need a cold start, from a cold start it takes five hours to bring the A6 machine back to production. And that is without counting the time that is needed for the actual kit installation.

A kit installation is therefore nothing that makes a customer happy. Another reason why a customer would not want to install a kit is if it is a bug-fix kit. For example, a very unlikely exception is found that will cause the machine to stop if it occurs. A kit that eliminates the exception is developed and released. This kit will not improve the machine from the customer's point of view and he might hesitate to let it be installed and especially so if he has experienced a bad kit before.

Since the quality of the kits is not high enough then the planning of a kit installation is more difficult than the actual kit installation. Hence, why the installation deadline for a newly released kit is nine months, no matter if it concerns are legal demands or small bug fixes.

11.2.4. Kit installation planning

A released kit is the end result of a long chain that might have started from a low performing function or a technicians request for improvement. Once a kit has been verified in-house it is released in the personal information viewer as a technical bulletin and Tetra Pak's local companies around the globe subscribe to bulletins. They are responsible for installing the kits while it is a Tetra Pak product company that develops kits and acts as a global support unit. When the local companies receive a kit bulletin, they know it is time to contact the customers to plan installations. The local companies are paid by the product company for the kit material and to perform the installations, and a nine month deadline marks the point when the Tetra Pak product company will stop paying for the kit installation.

11.2.5. Local Adaptions

The local Tetra Pak companies are also responsible for local adaptions. They are needed because of local factory and machine deviation. Machine deviations are often due to physical differences in the machine while factory deviations can be customers who uses external systems that differs from Tetra Pak's standard. In both cases the machine programs needs to be adapted to handle the deviations.

Each machine differs from the next one because of local adaptations and variety of installed updates and it is up to the local Tetra Pak companies to keep track of them. Machine adaption lists are used to document hardware and software updates and adaptions. These lists are used when entire machine programs are downloaded to the PLC processors, as they overwrite the existing program and all its adaptions. The service engineers will thereafter use the lists to insert all the adaptions again.

11.2.6. Kit installation

The technical bulletin contains information about the newly released kit. A brief description of the problem and the associated solution is described, how many service engineers are required, how long the installation will take, and last how large level and which type of expertise is needed. The bulletin contains a very detailed installation manual on how to rebuild the machine and update the automation program.

In the installation manual the needed code for the automation part of the update is present, usually in several screenshots. They display what should be entered and at which rung of the program. As the service engineer has to insert the code manually the risk for error is larger while it also gives service engineers the ability to

insert their own individual solution instead of the one provided. Individual implemented solutions create confusion for other service engineers, as they will first have to figure out the machine program before they can install an update or troubleshoot a machine error. The individual solutions add to the already differing machine programs.

11.2.7. Release trains

As time goes by, developers implement new functions and update the old ones, both hardware and software functions. To reduce the money Tetra Pak spends on installing updates they choose to gather up updates continuously in new machine program version. Tetra Pak continuously sets dates months in advance of when the new versions shall be released and it is during the installment of new versions that the whole machine program is updated and adaptions are overwritten. On a previous A6 machine update installment it took the crew four hours to install the new versions and another four hours to put back the local adaptions.

11.3. Object-oriented architecture analysis

A major problem with today's rebuilding kits is the quality of them, they take a long time to install and do not always work fully. Sometimes they even reduce the machine efficiency, or wrong kits are delivered to the machines. All these factors contribute to the customer's decision as to whether or not to delay a kit installation. An object-oriented architecture would help reduce some of these problems.

11.3.1. Importing classes

An object-oriented architecture would simplify the update process by making it more automatic. An object-oriented program imports classes from a software library. The classes are used by the program to instantiate objects which together form a functioning program. Installing the automation part of a machine update using an object-oriented architecture would be handled by uploading updated classes to a local machine library and then having the service engineer select which classes the program should use. The need for manually inserting code into programs would be reduced to near nothing or totally eliminated depending on the automation system capabilities. This would help reduce the time of installing kits and made them safer. Elimination of manual program updates would also reduce individual program solutions and it would help keep the machine programs more original and easier for service engineers to understand.

A usage of object-oriented polymorphism for Tetra Pak could be to preload automation updates on machines without the need for service engineers. A central software library could be established by Tetra Pak which machines at customers can automatically use to download classes from and into their local machine class library. With a high performing automation system the automation program could import the new classes from the local library. Whenever the machine is not being used the program would compile the new classes and have them ready for use. A machine operator could thereafter select if he wishes to instantiate objects from the new classes or keep the old. It could also be used for mechanical updates, the system would preload the new classes and once a service engineer has installed the mechanical parts he would select the new classes. Polymorphism makes it possible as the machine program does not need to know which classes will be used during runtime as long as they are present during compilation.

11.3.2. Separation of programs and functions

A way to deal with machine program adaptions being overwritten would be to export and import routines before and after a complete machine program download. This approach would however stop the parts that have not been adapted from receiving the new versions updates.

The object-oriented principle inheritance would solve the previously mentioned problem. Inheritance would separate the developer made classes from the local adaptions made by service engineers. Adaptions would be created in new and separate classes which inherit their methods from the developer made classes. Any inherited methods that need to be changed would be overridden and modified, and kept in the new classes.

Using inheritance during a complete machine program update would let the new updates propagate into the adaption classes and at the same time keep the overridden methods.

11.3.3. Versioning

It is difficult to trace updates in today's automation system as the machine program is updated by entering the code manually. Any deviation from the developer instructions will reduce the traceability of knowing which program version is installed. An object-oriented system would facilitate versioning, as each update will be stored in a new class and each class will have a version number associated with it. Therefore, it will be easy to verify which version of the machines program is being used because manual code insertion is not necessary. An error coming from the usage of an incorrect class will be easy to trace by controlling its version number. Restoring the machine program to an old version will also be done with ease, by instantiating objects from the previous the class version.

11.3.4. Encapsulation

11.3.4.1. Protected data

Encapsulation of an object-oriented program can be used to protect the attributes that are stored inside the classes. If the attributes are declared private then they can only be accessed within the class or by the class methods which have been declared public. The public methods can be implemented so that they reject incorrect values or to make the variables read only when read from outside the class. Private variables would reduce the risk of were a service engineer can incorrectly or by mistake modify a variable, and especially for global variables which are used through the whole program. There are not any obstacles to limit if an service engineer can modify global variables. Encapsulation of attributes and methods also prevents the service engineer from entering programming code in the wrong class as it will not have access to attributes of other classes and a compile error will be given if he tries.

11.3.4.2. Hiding functionality, easing the trouble shooting

A class encapsulates both attributes and methods which make it easier to troubleshoot potential errors. A service engineer who has traced the cause of the malfunction into an object will only be presented with the methods and attributes belonging to the object. Methods and variables belonging to other objects will be hidden. When the malfunctioning object uses methods belonging to other objects the engineer will not need to understand how the methods are implemented but instead focus on the answer they give. If the answer is correct then he can disregard the other object as it is working correctly while an incorrect answer means he is troubleshooting the wrong class and should instead switch to the second one.

11.3.4.3. Overview

Encapsulation moves a lot of the program code into classes and thereby breaking down the program into smaller building stones which together with the hierarchal structure of object-oriented architecture makes it easier to have a good overview of the program and knowing which object consists of which object.

12. Technical Service – Condition monitoring

12.1. Conclusions and requirements

Today's time based maintenance model brings Tetra Pak unnecessary cost because machine components are replaced without fully utilizing their life length, either too early or too late. Tetra Pak wishes instead to change to an event based model, where maintenance orders are generated based on machine health. To do so they will implement a central decision support system where the decision to replace components shall be taken regardless of service engineer's age, education or knowledge. The central system will have algorithms implemented to decide when maintenance orders shall be generated, the algorithms will use raw data collected from Tetra Pak's machines.

Using a central instead of a local machine system will lessen the already heavily loaded automation system. But as the machines are becoming more complex and the requirement for better performance increases the usage of external systems is just a temporary solution. One drawback with the central system is that the data collected from a machine cannot be used in real time as feedback to the machine program to improve the machine efficiency. An object-oriented system could help the already large machine programs, as implementing an event based monitoring system on the machines would cause the program size to increase greatly. The monitoring logic often requires more code than the actual control logic of a component. The monitoring logic could be hidden within monitoring classes and thereby reduce the complexity and size of the machine programs.

The requirements that follow below are derived from my interpretations of the inputs collected from the development engineer Per Jureus.

- Updating effort of customer machine to a condition based model shall be kept to a minimum
  - Software code shall be installed with minimal manual code intervention
  - Software updates shall take minor time
- Implementation time of a condition based maintenance model shall be kept to a minimum
  - Reusability of self-diagnostic functions shall be possible
  - A condition monitoring library shall be used
- Complexity of machine program shall be reduced
  - Condition monitoring functions shall be encapsulated

12.2. Stakeholder input

Technical Service is responsible for maintaining Tetra Pak's machines located at customer sites. Today's maintenance model is based on time. 500th, 1000th and 3000th hour maintenance's are carried out by the local Tetra Pak marketing companies. They replace machine components which are likely to not survive till the next maintenance. Which component shall be replaced is decided solely on the life length estimation of them. This model brings unnecessary costs and Tetra Pak wants to introduce a better one. They are currently running a project to develop a system that should be capable of predicting when a machine component is about to fail.

12.2.1. Performance limits and expertise

Tetra Pak wishes to introduce a new maintenance model, but one where the decision to replace a machine component will be the same regardless of the service engineer's age, education or knowledge. To do so a central decision support system shall be used, a system that will keep track of components breakdown limits. The central system will be implemented by an external company with a long experience working with the component types that Tetra Pak uses.

The breakdown limits will be based on Tetra Pak's long expertise in machine maintenance as they have serviced machines for over 50 years. The external company will use Tetra Pak's break down limits to implement various algorithms that will monitor Tetra Pak's machines. Once the system is up and running it will use data collected from Tetra Pak's machines to generate maintenance orders.

12.2.2. PLC - Central algorithms

The decision system will use raw machine data from Tetra Pak's machines. To collect data certain machine components will have to be equipped with monitoring equipment. However a lot of the data can be extracted from the PLC system, information that is unused today. Tetra Pak will have to implement functions to collect the data and send it to the central system.

12.2.3. Automatic maintenance

The new event based maintenance model is supposed to reduce Tetra Pak's costs of maintaining machines. By observing component status it shall be possible to more accurately estimate when a component is about to fail, thus increasing efficiency of the component by replacing it as late as possible without causing the machine to stop. The system will increase the machines performance attributes such as MME, MTBF, and mean time to recovery, MTTR which will please the customer by reducing thier production stops and Tetra Pak will have to deal with less customer claims due to malfunctions.

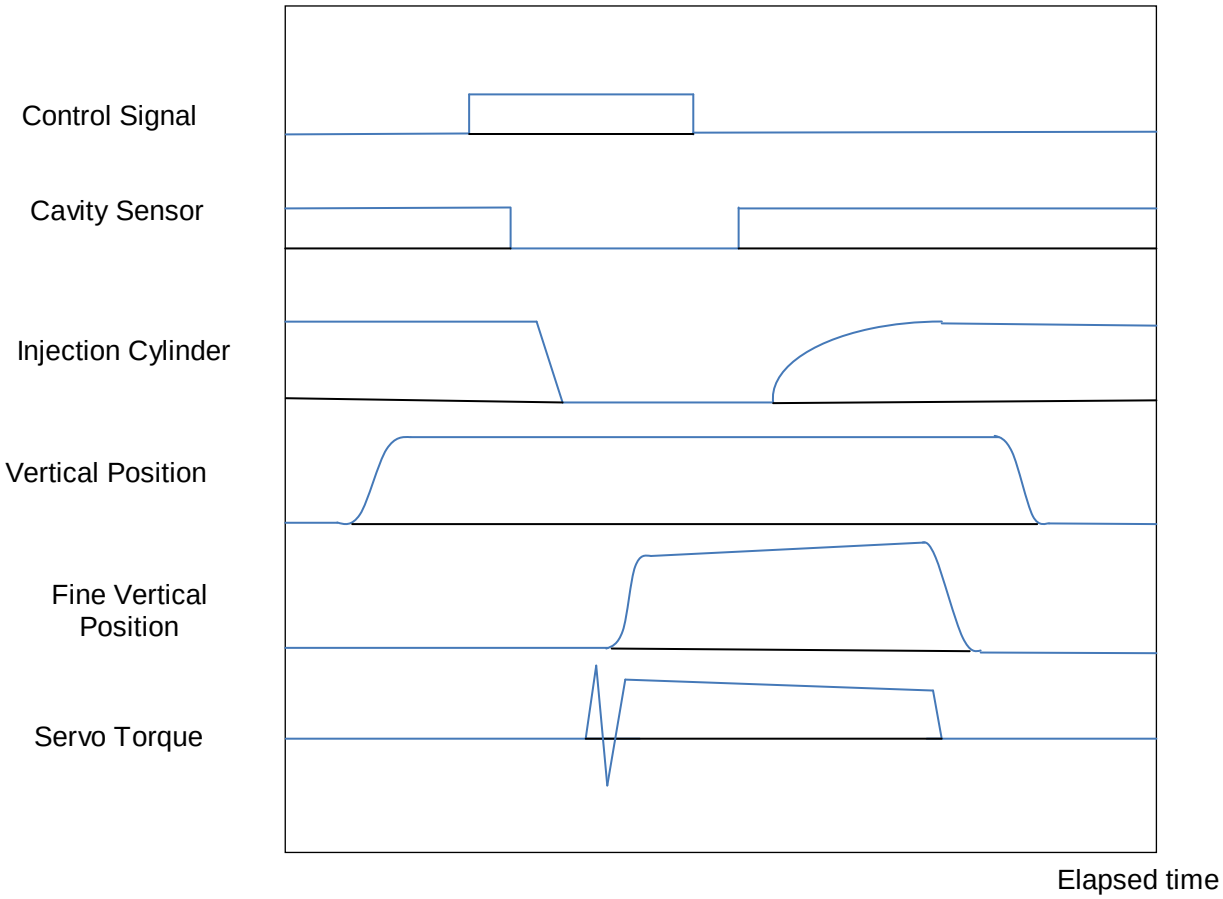
12.3. Object-oriented architecture analysis

12.3.1. Automatic control

The new condition monitoring system is going to use a one-way stream of information, from Tetra Pak's machine to the central decision support system. This setup will help Tetra Pak generate maintenance orders more accurately. But the setup omits a possibility where the data could be used to prolong the life length of mechanical components and not only to generate maintenance orders. For instance, when a pneumatic cylinder is getting old it might start to move slower, but instead of replacing the cylinder the monitoring system can be implemented to measure the new time it takes for the cylinder to move. The time will then be used in the program to make the cylinder start its movement earlier and still be able to fulfil its function. And it is first when the cylinder is approaching its maximum allowed movement time that a new maintenance order is generated. The automatic surveillance functionality would be kept on the machines which require the automation system to have a high performance.

12.3.2. Injection Moulding unit

The following section describes how an event based surveillance system could be used to monitor an injection moulding unit's health. Below follows a graph showing an injection moulding unit and its actions during a machine cycle. The unit moulds a plastic top on top of carton sleeves. First, a sleeve and a capped neck are prepared in a vertical position, and up to the sleeve a moulding device is lifted which injects hot plastic through four cavities, thereby moulding the sleeve and capped neck together.



The first curve shows the control signal, when it is turned high a signal is sent to the four needle cylinders, causing them to retract and thereby opening four nozzles in the moulding unit from which the hot plastic can stream out. Once the top has been moulded the control signal will go back to low.

The second curve shows one of the four sensors that monitor the needle cylinders. As soon as the needle starts moving the sensor signal will go low, the delay between the control signal and the needle movement is because the cylinder terminal has to pressurise.

The third curve shows the injection cylinder, once the needles cylinders are fully retracted the injection cylinder will extend and push out plastic through the four nozzles. After the top has been formed the needle cylinders will return to closed position and an accumulator will push new and hot plastic into the injection cylinder. The rising curve shows how the plastic slowly pushes the injection cylinder back.

The fourth curve shows the vertical position of the moulding unit. Once a sleeve is prepared above the unit a servomotor will start rotating which lifts the unit up to the sleeve. With a ready top the unit will be lowered to create room for the package to be removed.

The fifth curve shows the angular position of a servomotor that makes an eccentric movement, it is used to fine-tune the vertical position of the moulding unit and thereby press out the plastic. When the plastic top is cooling it shrinks and the eccentric movement secures he pressure between the top and the sleeve.

The final curve shows the applied servomotor torque of the fine tuning servo.

12.3.3. Torque monitoring

The only quality variable Tetra Pak monitors in the IM unit is the eccentric servomotor torque, it is used to determine if the plastic top weight is outside allowed limits. The torque of the servomotor is read at the end of each cooling cycle, right before the moulding unit is about to release and descend. If the read torque is too low it means that not enough plastic was used to mould the top, while a too large torque means that either too much plastic was used or that a needle slacked behind and caused an uneven distribution of the plastic. If the read value crosses the allowed limits, the machine will stop. However, the torque is monitored for product quality purpose and not for machine health purposes.

12.3.4. Condition monitoring library

Tetra Pak could implement monitoring algorithms in various classes, which together form a Condition Monitoring class library. A developer could thereafter import a class from the library to monitor the health of his component. For example classes with the functionality vibration measurement, filters, or simple functions as average value calculation. These classes could share a standardized interface to make them easy to understand and use. For instance, the constructor could be used to deliver signals for monitoring into the class and methods to receive if there have been variations from optimal usage. A reusable monitoring class library would reduce the development time and also simplify the updating work of machines to an event based maintenance system.

12.3.5. Large routines becoming larger

The advantage of using an object-oriented program and a condition based monitoring system is the use of encapsulation. In the example with the injection moulding unit several areas are suitable for monitoring component health. For instance, the time it takes for the needle cylinder to return to closed position after the close command has been issued by the control signal going back high. If the closing time of the cylinder would suddenly start to increase it could be an indication that it is about to break or that plastic has burned onto it and the additional friction makes it harder for the cylinder to close. In both cases a maintenance order might be necessary. This is just one example of many areas that could be monitored of the IM unit, and if Tetra Pak chose to implement surveillance of their components then the already large machine program would grow even larger. The encapsulation would help hide the new programming code in classes and thus reduce the size and complexity of the machine program, making it easier for both developers and service engineers to understand and troubleshoot programs.

13. Conclusions

The conclusion, based on the gathered requirements, is that Tetra Pak would benefit from a switch to an object-oriented architecture, however such a conclusion is made very vaguely. The aim of this master thesis was to collect stakeholder voices and to translate them into requirements. The main focus has not been to study how the requirements shall be fulfilled. Together with the lack of competence in other fields, where for example supply chain operation and technical support works, the disadvantages cannot be weighed versus the advantages to find an accurate conclusion. It is up to the second thesis team to make such a decision as they will be the one to design the *how* of the entire object-oriented project. However during the interviews and discussions with the stakeholders several clear areas have been found where an object-oriented architecture would benefit them.

Two of the largest advantages found were that object-oriented architecture would lead to modularity and simplicity, in an object-oriented architecture they both go hand in hand. Modularity would make machine systems easier to develop due to reusability of modules and programs. While at the same time production, testing, and service would become easier with the standardizations of module and program interfaces. The modularity of an object-oriented architecture leads directly to simpler work ways which would help reduce or remove much of the work Tetra Pak does that adds no value to the end product, for instance the reconfiguration of mechanical components done twice or the manual insertion of code during updates.

The disadvantage of an object-oriented architecture is the overhead cost, everything will initially be more expensive, as with any system that focuses on reusability. But once the object-oriented system kicks in, the costs should decrease by a large step, but it is up to the next thesis team to figure out a definite answer.

14. References

Books

Bittner, K & Spence, I. (2003) Use Case Modeling. Boston: Addison-Wesley. ISBN 0-20-170913-9

Halladay, S & Wiebel, M. (1993) Object-Oriented Software Engineering. Lawrence: R&D Publications Inc. ISBN 0-13-034489-3

Unprinted sources

Cruce Oscar, Supply Chain Engineer, Tetra Pak, Interview

Hamer Martin, Automation Engineer, Tetra Pak, Interview

Hansson Anders, Manager, Production Engineering & QA, Tetra Pak, Interview

Hansson Robert, Systems Engineer, Tetra Pak, Interview

Helander Johan, Automation Engineer, Tetra Pak, Interview

Ivarsson Conny, Automation Engineer, Tetra Pak, Interview

Jureus Per, Development Engineer, Tetra Pak, Interview

Larsson Fredrik, Technical Liaison, Tetra Pak, Interview

Lindberg Peter, Systems Engineer, Tetra Pak, Interview

Malmros Christofer, Coordinator Rebuilding Kit, Tetra Pak, Interview

Martinsson Mats, Development Engineer, Tetra Pak, Interview

Mattsson Fredrik, Systems Engineer, Tetra Pak, Interview

Sundberg Anders, Systems Engineer, Tetra Pak, Interview

Svensson Ulf, Automation Engineer, Tetra Pak, Interview

Wåhlin Thomas, Supply Chain Engineer, Tetra Pak, Interview

Yao Peter, Systems Engineer, Tetra Pak, Interview

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER THESIS

Date of issue
August 2013

Document Number
ISRN LUTFD2/TFRT--5924--SE

Author(s)
Peter Wodzynski

Supervisor
Peter Lindberg, Tetra Pak, Sweden
Karl-Erik Årzén, Dept. of Automatic Control, Lund University, Sweden (examiner)

Sponsoring organization

Title and subtitle
Requirements Analysis of Using Object-Orientation in Filling Machine Systems

Abstract
The use of an object-oriented approach in software engineering has proven to be successful for many years, but in the fields of mechanics and automation it has been ignored for long. The Tetra Pak Carton Bottle automation team believes that a switch from a function oriented architecture to an object-oriented one would be of benefit not only to them but also any department which is involved with development, production, and maintenance of filling machines.

Many product projects of today make the mistake of having its participants jumping straight into the implementation part of the project while skipping the work of first defining requirements of what the product shall accomplish. People in general like to start by doing the how of the project without first focusing on the what.

But if the needs of other product concerned are overlooked then there is a risk of an important part being left out, perhaps leading to a weaker product. This master thesis focuses on the what part of a development project where an object-oriented architecture shall be deployed for Tetra Pak's development, production, and maintenance of machine systems. During the master thesis requirements from various stakeholders were defined based on interviews and discussions held with various Tetra Pak departments.

The result, the requirements, show that Tetra Pak has several areas which would be benefited by an object-oriented architecture by making Tetra Pak's machines more modular and the maintenance of them simpler.

Keywords

Classification system and/ or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-57

Recipient's notes

Security classification