Rule Engine

Øystein Eriksen
Andreas Smogeli Leite

Master of Science in Computer Science
Submission date: June 2007
Supervisor: Tor Stålhane, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Problem Description

A prototype of the Rule Engine was developed during the In-Depth Study, autumn 2006. It is now going to be developed further with more controls, both logical and validation rules. The rules have its background in the rule set "Kontroll og validering i ny EPD" (Control and validation in new EPD). The rules operates on product data. These are products with many properties of different kind, and are connected in different ways.

The project is to develop the Rule Engine further. The Rule Engine is a validation system, With the Control and validation in new EPD rule set as background, where you can build and change both logical controls and valiadtion rules without having any programming skills. The system must include a validation page where a user can validate the product data with rules. The users of the system must be able to build rules and validate products by using the rules they have built, without making a new version of the system. The system must use AJAX to give the user an immediate feedback on the validation. The system must be tested on a selection of users to check that the robustness and user-friendliness is satisfied. This applies for administrators and regulare users.

Focus area: robustness, user-friendliness
Technical requirements: ASP.NET, C#, AJAX, SQL Server 2005

Assignment given: 15. January 2007
Supervisor: Tor Stålhane, IDI

Abstract

This project is a study of the development of the Rule Engine, which is a validation system for quality assurance of product data used in the grocery business. The authors was asked by Cogitare AS to develop the Rule Engine. A system where users without programming skills can build rules and validate product data. The main quality attribute focus is robustness and user-friendliness. A survey has been used by the authors to be able to explore if our objectives have been achieved and to identify further work. The questionnaire has been conducted on students and software developers.

Preface

This report is a result of our study in TDT4900 Master thesis. This is a part of the 10th semester at the Norwegian University of Science and Technology (NTNU)

We would like to thank our university supervisor, Tor Stålhane for guidance and feedback. We also would like to thank our employees at Cogitare AS, Rune Kvisten and Geir Aakvik, and fellow students for their participation in the survey, which was valuable to our project.

Trondheim, June 11, 2007

Øystein Eriksen *Andreas Smogeli Leite*

Contents

List of Figures

List of Tables

1 Introduction

This chapter is an introduction to our project work. Here we elaborate the motivation, problem definition of the project and a short summary of the rest of the chapters in this report.

1.1 Motivation

In the summer of 2006 the authors worked with the consulting engineering company, Cogitare AS[1]. This project is a further development of the prototype Rule Engine we developed in the in depth study [1]. For our master thesis at NTNU the management at Cogitare AS, Rune Kvisten and Geir Aakvik, made an assignment for us. The assignment was to develop a rule engine, for validating products in an existing production system. The task is to develop a website which easily register new rules, administer rules and validate products. The existing system is covered in Chapter 2.2.

To know that our project can be used by Tradesolution gives us an extra motivation to deliver a system of as high quality as possible.

1.2 Problem definition

To create a software system of high quality you have to fulfill the quality attributes given by the stakeholders. The project requirements is based on "Control and validations in new EPD" (2.3). This implies that the system must cover logic controls and validation rules. Robustness and user-

[1]http://www.cogitare.no/

friendliness are the main quality attributes. When the development of this software system is completed, it must be tested on real users. A survey will provide us with important information pertaining to whether the main quality attributes have been fulfilled. It will also give answers to positive and negative aspects of the system.

1.3 Report outline

This section describes an outline of the rest of the report, a short summary of the contents of each chapter will be given.

Chapter 2 - Prestudy

Chapter 2 gives a short introduction to the company Tradesolution which uses the current validation system. The chapter also describes the existing validation system, the use of XML and survey.

Chapter 3 - Method

Chapter 3 describes the methods behind developing the Rule Engine and the user survey.

Chapter 4 - Rule Engine

Chapter 4 describes the Rule Engine. Focus is on how the system is build up and how it works from the users point of view.

Chapter 5 - Development method

Chapter 5 describes the method used to develop the Rule Engine and some examples to achieve a satisfactory level of user-friendliness by using AJAX.

Chapter 6 - Software system test

Chapter 6 presents the software system testing and the results of the test.

Chapter 7 - Possible hazards

Chapter 7 presents a risk analysis on what can go wrong and which consequences this have when using the Rule Engine.

Chapter 8 - Survey

Chapter 8 presents the results of the survey. We describe our interpretation of the result and how we will use the results.

Chapter 9 - Evaluation

Chapter 9 gives an evaluation of this work and what we have achieved.

Chapter 10 - Our contribution

Chapter 10 describes our contribution to the system. The chapter consist of a description of the Rule Engine. In addition it describes how the system is tested.

Chapter 11 - Conclusion and further work

Chapter 11 presents conclusion and further work.

2 Prestudy

The prestudy uses the result of the prototype Rule Engine [1]. This chapter gives a short introduction to the company Tradesolution which uses the current validation system. The chapter also describes the existing validation system, the use of XML and survey.

2.1 Tradesolution

The Rule Engine is developed for Tradesolution. Tradesolution is owned by the industry. Their purpose is to maintain and manage central register and databases in Norway. Their customers and collaborating partners are considerable actors within industry, trade and service.

The major type of business for Tradesolution is to maintain the EPD-base, which contains information about products distributed by the grocery industry, and manage EPD Sjekkpunkt, which is a measure and control service [2].

The EPD-base is a product database for exchange and quality assurance of information about products which is distributed and sold between suppliers and convenience chains in Norway. The convenience chain is fast moving costumer goods, kiosk, gas and service business, hotel, restaurants and catering [2].

2.2 Existing system

Tradesolution has a system that control data that exists in their database. This system is based on Visual Basic code. Figure 1 illustrates an overview of the existing system.

The existing system is working fine but it need to be more flexible regarding rules that are used to control data. Each time a new rule is made, someone have to code this in Visual Basic and make a new version of the system before the rule can be used. This is not efficient and takes a lot of time and the need of programming expertise is large since you are dependent on someone who knows the programme.

The rules control that all the data is correct. For instance, the maximum temperature of a product cannot be lower than the minimum temperature of the same product. This is a rule that is used when the data is being controlled. The main object of this master thesis is to make a system where the rules are not hard coded into the code, but are inserted by an administrator. The rules must be saved so they are easy to access and edit.

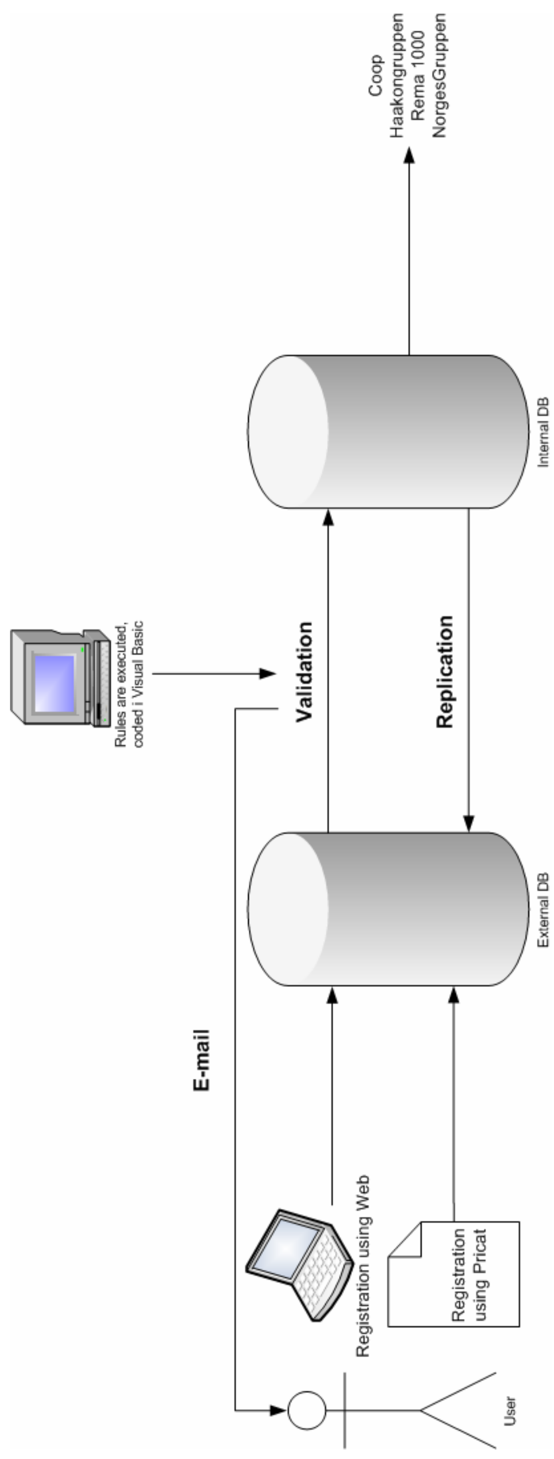


Figure 1: This is an overview over the existing system.

2.3 Control and validations in new EPD

Control and validations in new EPD, from now on CVE, is a document which contains all the rules that the Rule Engine is based on. Tradesolution has developed this document to have an overview of all the rules. These rules validate the data in their database before they are transmitted into the internal database. The CVE consist of 168 rules.

The rules in the document are divided into two groups, validations and logical controls. Each group is divided into the four pack levels shown in Figure 2. The rules in CVE is based on fields in the database. E.g the GLN owner[2] has to be the same for all pack levels, and there also has to exist a customer which has an active subscription. This control is in the validation group. Another example is the width of the product package. This field has both validation controls and logical controls. The validation controls that if a product is registered, this field must be filled with a number. This rule applies to all pack levels. There are several logical controls on the width field. One that applies for consumer packaging (Figure 2), says that the value cannot be greater than the greatest of height/width/depth of the retailer packaging (Figure 2).

To make a rule engine based on the rules in CVE, we had to find what the rules had in common and then group the rules. We concluded that the rules could be divided into two groups. The groups are compare and format.

[2]GLN owner is a number which is connected to the company who owns the product

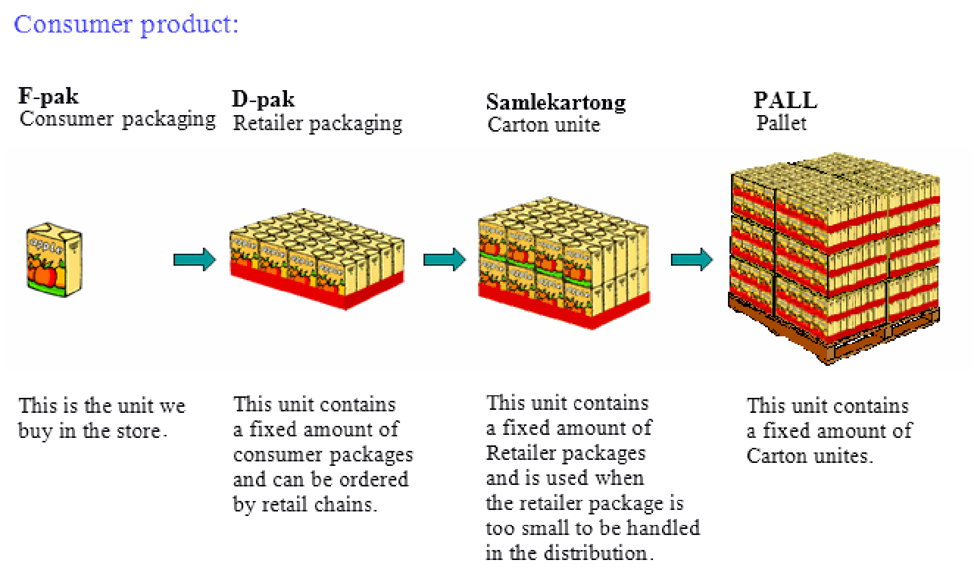

Consumer product:
F-pak
Consumer packaging
D-pak
Retailer packaging
Samlekartong
Carton unite
PALL
Pallet
This is the unit we buy in the store.
This unit contains a fixed amount of consumer packages and can be ordered by retail chains.
This unit contains a fixed amount of Retailer packages and is used when the retailer package is too small to be handled in the distribution.
This unit contains a fixed amount of Carton unites.

Figure 2: The different pack levels in a product set

The compare group consists of rules that have the expression 'the value cannot be greater than', 'the value cannot be greater or equal than', 'the value must be greater than', 'the value must be greater or equal than', 'the value must be the same as' and 'the value must different'. This group is always connected to fields with numbers and expression like 'net_weight * numbers_of_consumer packaging_in_retailer_packaging cannot be greater than gross_weight_to_retailer_packaging'.

The format group consists of rules that has something to do with the format of the value. This can be 'the value must contain 8 numbers'.

The Rule Engine prototype

A prototype of Rule Engine was developed in a depth study project done by us [1]. The prototype was made to see if it is possible to make a rule engine based on rules that exists in an xml-file. The system that was made is shown in Figure 3. The goal in the depth study was to execute the rules on the

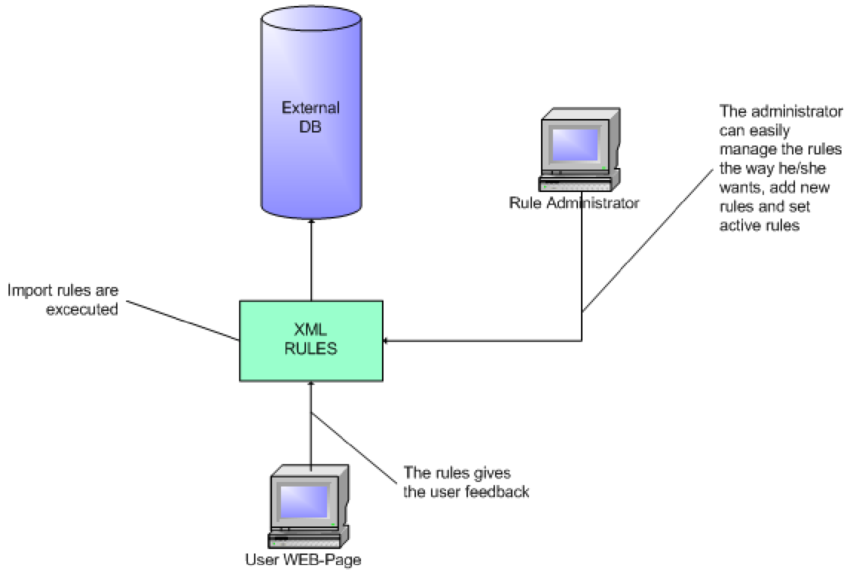

External DB
The administrator can easily manage the rules the way he/she wants, add new rules and set active rules
Rule Administrator
Import rules are excecuted
XML RULES
The rules gives the user feedback
User WEB-Page

Figure 3: A overview over the prototype.

website where the users register their products. By doing this you can be sure that the data that is added to the external database are correct and the control between external and internal database is unnecessary. For the new Rule Engine the focus has changed. As shown in Figure 1, we have been made aware of a new method to register products. Users that do not use the website to register their products can send their data to Tradesolution which add the data to a pricat-file[3] and the system read the data from the

[3]Pricat is a file type which can be opened by using Excel, contains information about

pricat and add them to the external database. This leads to insecure data and introduce the need for better control.

The prototype was limited to control that an expression is equal to another expression. The new Rule Engine have to be more flexible in declaring different rules. It have to handle that an expression can be more or less to another expression. It also have to control the format on the data.

2.4 Rule Engine based on XML

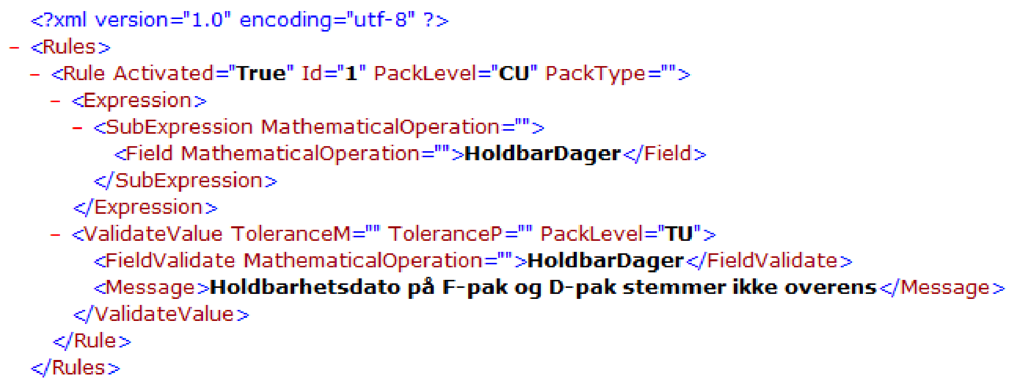
In the depth study of the prototype Rule Engine, the authors made an conclusion on whether to use XML or not:
"XML provides far better user friendliness and you do not need programming skills to make and understand XML files. The depth study shows that stored procedures are more difficult to develop and less intelligible than Extensible Markup Language" [1].

The Rule Engine creates rule tags in the rule.xml, declared from user input. Figure 4 shows an example of a declared rule. The prototype only covered numerical and integer fields with mathematical operations.

2.4.1 Structure of the prototype rule.xml

<Rules>
This is the start element of the XML-file. The file may contain many rules, between the root element <Rules> and the end of the root element

products.



```
  <?xml version="1.0" encoding="utf-8" ?>
- <Rules>
  - <Rule Activated="True" Id="1" PackLevel="CU" PackType="">
    - <Expression>
      - <SubExpression MathematicalOperation="">
          <Field MathematicalOperation="">HoldbarDager</Field>
        </SubExpression>
      </Expression>
    - <ValidateValue ToleranceM="" ToleranceP="" PackLevel="TU">
        <FieldValidate MathematicalOperation="">HoldbarDager</FieldValidate>
        <Message>Holdbarhetsdato på F-pak og D-pak stemmer ikke overens</Message>
      </ValidateValue>
    </Rule>
  </Rules>
```

Figure 4: Example XML-file from the prototype Rule Engine

<**/Rules**>.

<**Rule**>

This tag contains all the information about one rule and has four attribute values and two sub elements.

- **Activated** - tells the programme whether the rule is active (true), or not (false).

- **Id** - unique identification of the rule.

- **PackLevel** - level of packaging the rule has an expression for.

- **PackType** - type of packaging we has an expression for. This can be empty if we want it to pass for all pack types.

<**Expression**>

This element contains the expression of the rule, and contains the following elements:

- <**SubExpression**> This element can contain many fields.

- <**Field**> This element contain a numeric- or integer field from the database product table. The **MathematicalOperation** attribute contain the selected numeric operator the user selects. This attribute is logically always empty in the last field.

<ValidateValue>
This element defines the rules validation value. The element has three attributes:
- **ToleranceM** - if the rule has a measured limit of tolerance.
- **ToleranceP** - if the rule has a percentage limit of tolerance.
One rule can only have either a measure- or percentage limit of tolerance, or no tolerance at all.
- **PackLevel** - pack level of where the rule will validate the expression.

<FieldValidate>
This element has the same structure as the <Field> element. The FieldValidate is the element to validate the Field expression.
<Message>
This element specifies the error message of the rule.

The main challenge for the extended Rule Engine is to cover more types of rules. For instance, more and less comparison for validation, declaring different formats to validate. To achieve this the XML-file must be extended. The content to modify will be the <Expression> and <ValidateValue> tags. The content of these tags will be different between different rules.

2.5 Survey

One of the objectives to this master thesis is to carry out a survey. The reason to do this is to get important answers for the quality of finished system. Important questions are for instance: Has the system achieved good robustness and user-friendliness? What could be different and better? This is done with a survey on the real users of the system, in our case students and software developers.

A survey was also carried out in the depth study of the Rule Engine. This was conducted as interview and the result of it can be summed up as follows: "The mutual agreement about the system is that it is incomplete. The system cover implementation and realizes the main purpose of this product but it has a long way to go before it is completed. When it comes to robustness the interviewee was very satisfied. They could not save a rule that is not valid and the rules that were made act just like expected. The user-friendliness was not so good. It was easy to understand how to make a rule but the system gave little feedback. When making a rule they got no feedback when it was saved, and when saving values of a new product you cannot see if the rule was executed. The system has to cover more so the administrator can do more, like removing and editing rules."[1]

We gained much experience from this survey. One important answer was that user-friendliness must be in focus this time. The preparation of the new interview is important, including the follow-up questions. One object to improve this time is a wider selection range of subjects to get more opinions.

3 Method

This master thesis consist of developing the Rule Engine and carry out a survey on subjects. This chapter will describe the methods used in this work.

3.1 Software Development Methodologies

The Rule Engine is the software system that was developed in this master thesis. When developing a system like this, it is important to use good software development methodologies. For more information about advantages by using a software development methodology see Appendix B. The methodology that is used in this study is the waterfall development, see Figure 5. The waterfall methodology is just one of many methodologies. The reason for using the waterfall instead of others is that we think that this is a good way to develop a system. This is because we have stable requirements and the technology is known. We also wanted to see if it is actually possible to make a good system using this methodology.
The waterfall methodology flows smoothly over the classical phases[4] and it is the most common methodology. It is nice idea to do development in this way but it is unrealistic to follow this method throughout the project. When doing the waterfall methodology you do one phase at the time and when you are finished with that phase you do not look back.

From the beginning of this study we have used a plan based on the waterfall

[4]For information about the classical phases see Appendix A

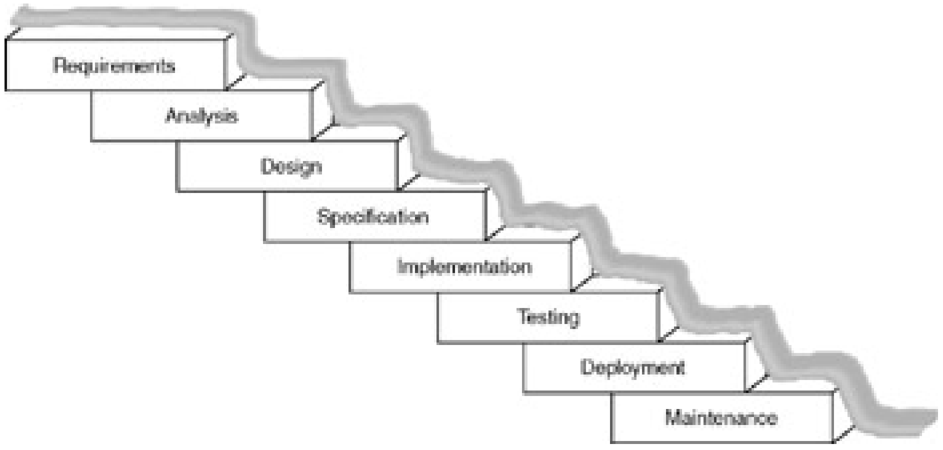


Figure 5: Waterfall development

methodology. We have worked through every step and followed the time scale all the way to the testing phase. After the testing phase we depart from the waterfall methodology. We wanted to use our test results to improve our system. To achieve this we had to do the implementation phase over again. After this we did the rest of the waterfall methodology phases. This worked well and the result is good because of the method we have used.

When coding the system we have used both paired programming and programming alone. Paired programming is when two people sit together and programme, one is typing and they are discussing with each other. In this way we think the programming is easier and the code is better. We have not measured it, but when coding gets complex we feel the code is better and faster because of the paired programming. Also the programming alone method is used. The system have been divided into a number of parts and when the parts are smaller and not complex to develop, the programming alone method works just fine.

3.1.1 Development tools

In this section the development tools used in the project will be described. The choice of development tool was never an issue between us because both of us have been working with the same tools and they are well-known to us.

ASP.NET

ASP.NET is a server-side technology for developing Web applications based on the Microsoft .NET Framework. Instead of being interpreted by the client, server-side code (for example, the code in an ASP.NET page) is interpreted by the Web server. In the case of ASP.NET, the code in the page is read by the server and used dynamically to generate standard HTML/JavaScript/CSS that is then sent to the browser. As all processing of ASP.NET code occurs on the server, it is called a server-side technology. As Figure 6 shows, the user(client) only sees the HTML, JavaScript, and CSS within the browser. The server (and server-side technology) is entirely responsible for processing the dynamic portions of the page. [3]

C#

C# is an object-oriented programming language on the .NET platform and designed for improving productivity in the development of Web applications. C# boasts type-safety, garbage collection, simplified type declarations, versioning and scalability support, and other features that make developing solutions

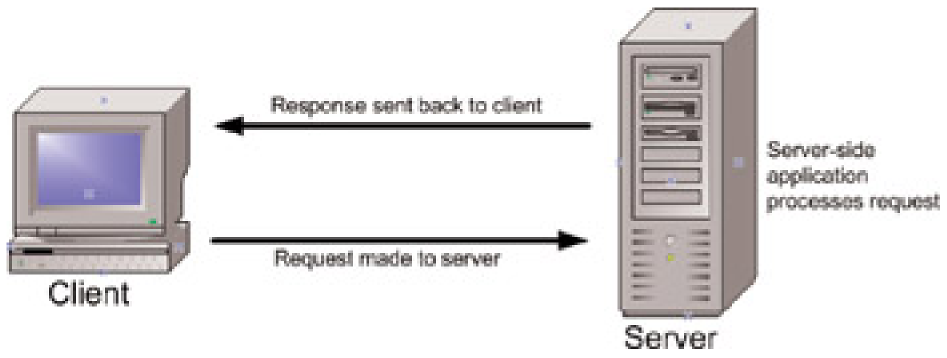

Response sent back to client
Request made to server
Server-side application processes request
Client
Server

Figure 6: The Web server is responsible for processing the server-side code and presenting the output to the user (client)

faster and easier. [3]

SQL Server 2005

SQL Server 2005 is a comprehensive database software platform providing enterprise-class data management and integrated business intelligence (BI) tools. The SQL Server 2005 database engine provides more secure, reliable storage for a relational database format or XML.

3.1.2 ASP.NET AJAX in the Rule Engine

"Microsoft ASP.NET AJAX enables developers to create Web pages that include a rich user experience with responsive and familiar user interface elements. ASP.NET AJAX provides client-script libraries that incorporate cross-browser ECMAScript (JavaScript) and dynamic HTML technologies, and it integrates them with the ASP.NET 2.0 server-based development platform. By using ASP.NET AJAX, developers can improve the user experience and the efficiency of Web applications" [4].

3.2 Survey method

As mentioned in Chapter 2.5, one of the objectives of this master thesis is to carry out a survey on students and software developers. "Survey is a set of standardized questions about a theme on a selection of people. The objective is to gather data through interviews or questionnaire. Surveys are used in public as well as the private industry" [5].

We have experience with conducting interview in the depth study at NTNU [1]. The strength of an interview is its flexibility. The interviewer can quickly clear up misunderstandings and it is also more motivating for the participants. The interviewer can make follow-up questions to get more information for participants.

For our master thesis there will be 11 subjects to survey. These subjects are located in different areas and this makes it difficult to carry out interviews. Therefore, a questionnaire will be used this time.

3.2.1 Questionnaire

"Questionnaires is the most structured of the survey techniques. It can be done in many ways: postal surveys, e-mail surveys, questionnaires that is handed out and picked up, group filling in and questionnaire in combination of interview" [5].

Postal surveys are used most often. The form is sent to a selection of subjects. The participants receive the questionnaire and a letter which describe the purpose of the questionnaire and how to fill it in. When the questionnaire is finished the respondent return the questionnaire [5].

The Internet is used as a medium to spread questionnaires, and will be used by us. The questions is sent by e-mail and the respondent return the answers by e-mail. The e-mail contains a link to the questionnaire and you can answer directly on the homepage. Doing it in this way it is easy and quick to answer the questionnaire. The answer is registered directly to a computer and the results can be analyzed quickly [5]. For more on how we conducted our survey see Chapter 8.

Strength and weakness of questionnaire

The most important strength of the questionnaire is that it is cheap to use. Big investigations can be managed by few persons. The interviewees can answer the questionnaire when it is convenient. In addition problems and misunderstandings are difficult to sort out. The main disadvantage is that the motivation for the respondent is little and this leads to fewer answers to the questionnaire.

3.2.2 Test of the questionnaire

We developed a questionnaire based on the requirements of the system. To make sure that this questionnaire was intelligible it was tested on some fellow students. We interviewed them about how they understood the questionnaire. Some small adjustments on formulations were done after the test.

4 The Rule Engine

In this chapter we describe the Rule Engine. Focus is on how the system is build up and how it works in practice from the users point of view. The purpose of the Rule Engine for Tradesolution is bipartite. Primarily it is to validate product data between an external database and an internal database. The reason for this is to make sure that all product data is correct when customers and suppliers insert their product data. In this Rule Engine project we confined to only one server. So there will not be any data transmission done between servers, only validation of data on one server. Secondary, there is a great amount of data attached to each product, and new products are inserted all the time. A software used to maintain this, has to be user-friendly and robust. The Rule Engine makes it possible to add new-, modify- and delete validation rules without having any knowledge of programming.

The Rule Engine is divided into three main parts:

- Validation - This is the main page. Here the user can select rules to validate selected products in the database.
- Administration - An administrator user can activate or deactivate rules, copy and modify rules and choose server connection.
- Create rules - Contains wizard to guide the user through the steps needed to create a new rule, with help texts.

In the following sections the Rule Engine and its tasks will be described in more details.

4.1 The Rule Engine overview

In this section the UML class diagram, database model and an overview of the new Rule Engine system is described.

4.1.1 The Rule Engine UML class diagram

In this section we cover the class diagram for the Rule Engine. The class diagram is illustrated in Figure 7.

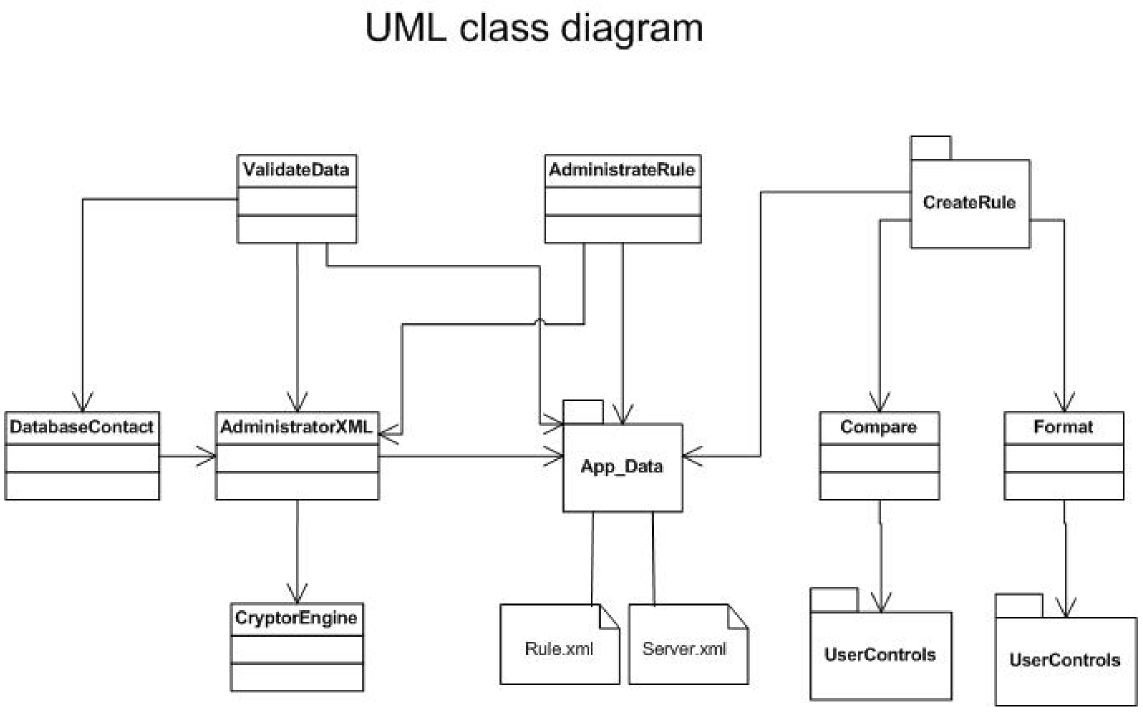


Figure 7: The Rule Engine class diagram.

The software system consist of the following classes:

- ValidateData - see Section 4.2 for more information.
- AdministrateRule - see Section 4.3 for more information.

- CreateRule - see Section 4.4 for more information.
  - Compare
  - Format
- DatabaseContact - Responsible for establishing connection to a database, and return SQL queries.
- AdministratorXML - Responsible for modifications done to the App_Data XML-files (Rule.xml and Server.xml).
- CryptorEngine - Responsible for encrypting and decrypting passwords for database connection.
- App_Data - contains the declared rules in an XML-file and a declared server connection in another XML-file.
- UserControls - folder which the "Format" and "Compare" classes uses, contains several "Web User Control" classes. These classes contains the contents to the wizard steps for creation of rules.

4.1.2 Database model

An intro to the EPD-base was given in Section 2.1. This is the product database used for exchange and quality assurance of information about products which is distributed and sold to suppliers and convenience chains in Norway.

Since the development of the prototype Rule Engine in autumn 2006, the stakeholders was not pleased with the feedback given when rules where executed. The system lacked a complete overview of tasks performed by the software system. To solve this problem we decided to develop a log of what has been done to the system. The log contains the rules that has been executed, and the products that passed the rule, and those that did not. To achieve this we had to create new tables to the database to store the log. The table "RuleEngineValidateSet" stores data from one validation execution. One validation execution may contain many rules and many products. We developed a table "RuleEngineValidateData" where we store this data, see Figure 8. Now the software system can give the user feedback and a complete overview of all validations done in the system.

4.1.3 Overview of the new Rule Engine system

Figure 9 illustrates the new system. The Rule Engine is now installed on a server. The Rule Engine can be operated by a user through the Internet and perform all the tasks described in this chapter. As the diagram indicates, the Rule Engine is connected to the database. The Rule Engine use product data from the database and validates the data. The results of the validation is stored in the database, see Figure 8.

Let us take a look back at the old system found in Figure 1. The main difference is the way data validation is done. In the old system the rules were hard coded into the system. In the new system the Rule Engine replaces the hard coded rules and thus makes the system more user-friendly and flexible. In addition, it eases the maintenance of the system.

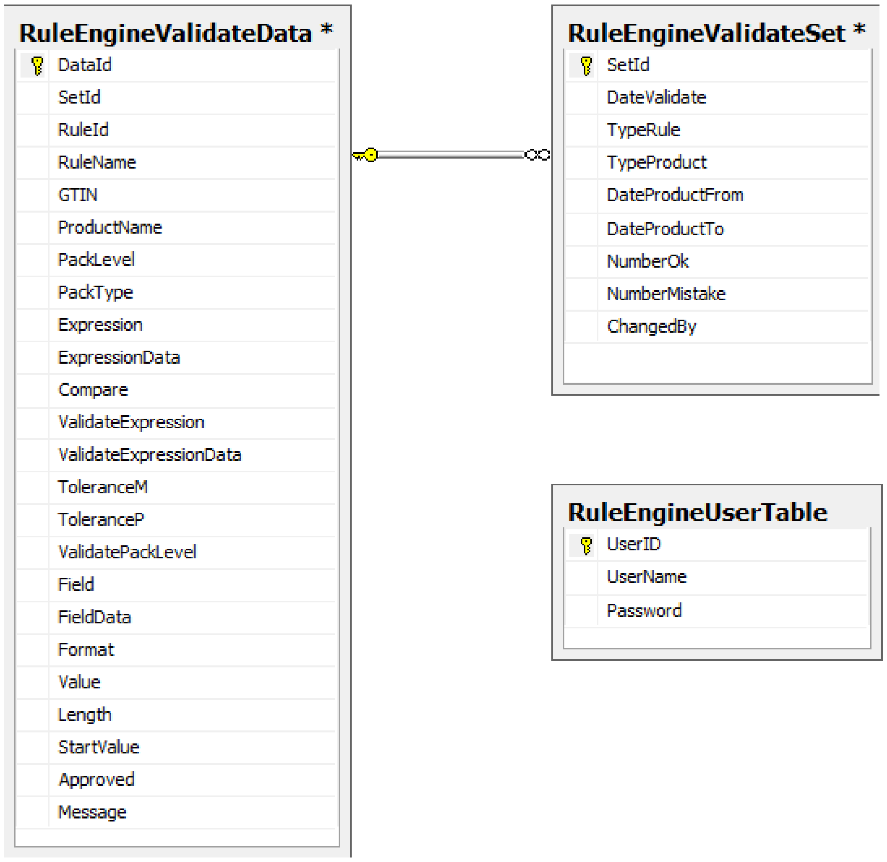


Figure 8: New database tables created.

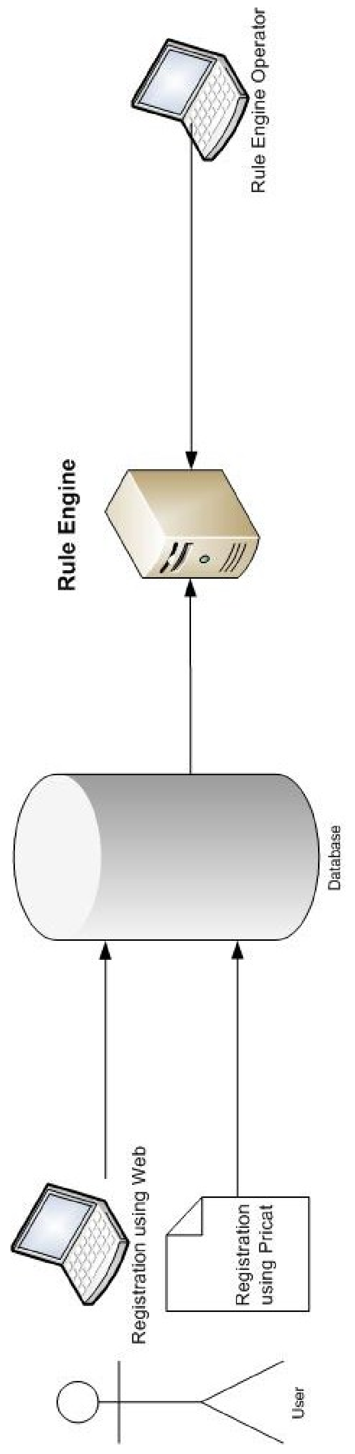


Figure 9: An overview of the new system.

4.2 Validation

Validation is the first page that is shown when you enter the Rule Engine web site after log in. This page is where you run the validation and can see everything that have been done earlier. You have two choices under the "Validering"-tab, "Validering" and "Logg".

Under "Validering" you can run the rules that are made towards data in the database. On the left hand side you choose the rule(s) you want to run, and on the right hand side you choose the products you want to run the rules on. Figure 10 shows the left hand side and in Figure 11 the right hand side is shown.

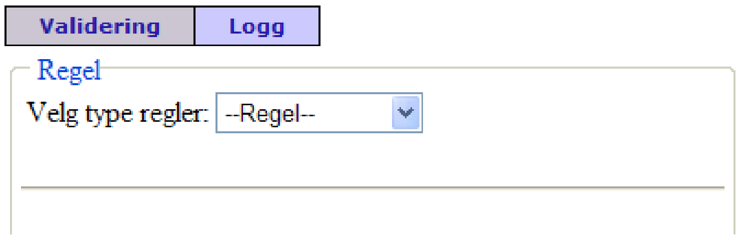


Figure 10: The left hand side of the validation site (Choose rules).

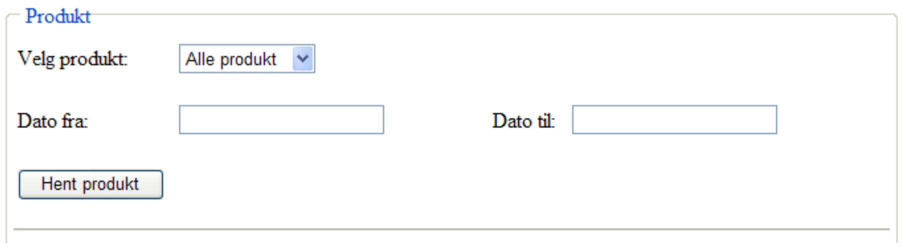
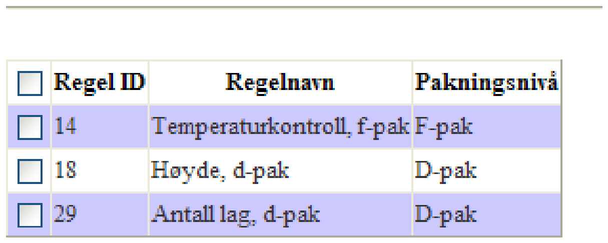


Figure 11: The right hand side of the validation site (Choose products).

When there is a lot of rules it is not always practical to display all rules that have been made. When choosing rules you have three alternatives: "Alle regler" (All rules), "Sammenligning" (Compare) and "Format" (Format). When the rules are displayed (see Figure 12) you have the possibility to choose all the rules that are displayed, or just some of the rules. This can be done by using the check box and check the rules you want to use.



☐	Regel ID	Regelnavn	Pakningsnivå
☐	14	Temperaturkontroll, f-pak	F-pak
☐	18	Høyde, d-pak	D-pak
☐	29	Antall lag, d-pak	D-pak

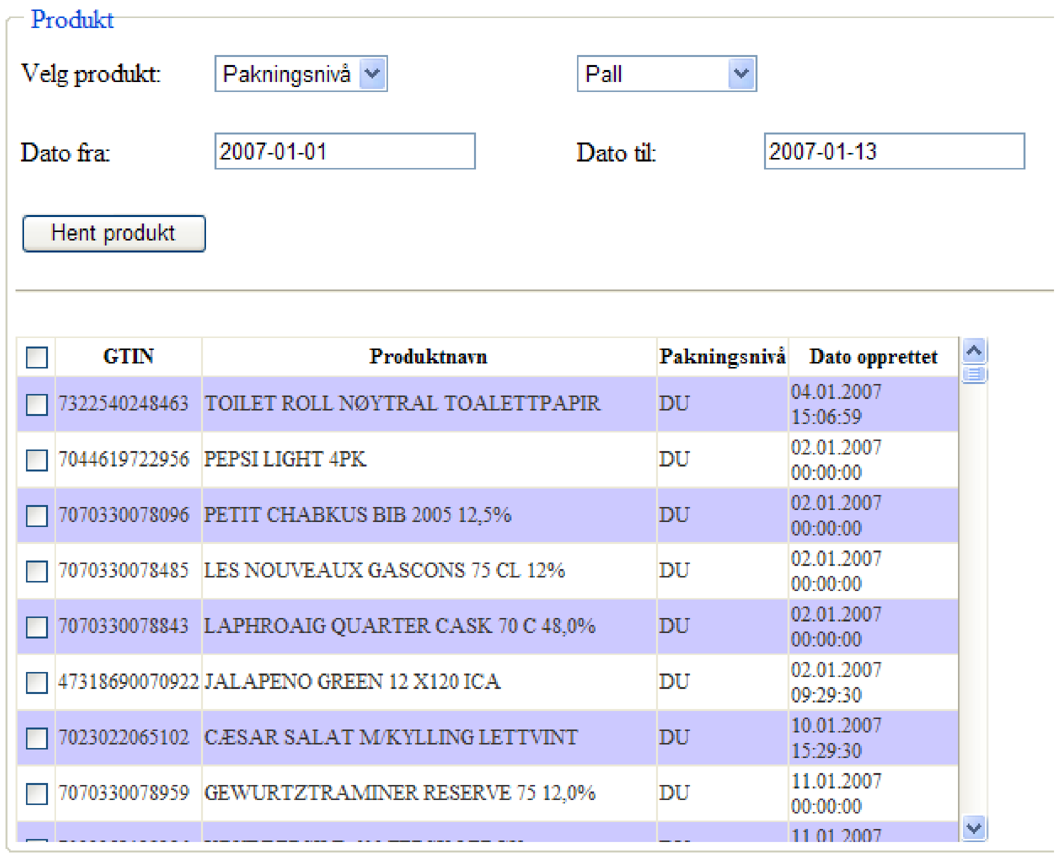
Figure 12: Rules displayed.

In the database there will be many thousand products and it is not necessary to show all of them. We can choose between "Alle produkt" (All products), "GTIN" (Product number in database), "Pakningsnivå" (Pack level) and "Produktsett" (Product set), see Figure 2. You can choose from date and to date[5]. When choosing "GTIN" a text box for the product number will pop up. This will display the whole product set for this product. If you choose "Pakningsnivå" you get four more options. These options are the four pack levels, see Figure 2, and just the pack level you choose will be displayed. "Produktsett" is an option to display the product that has the highest level in a set. When choosing "Produktsett" all products in the set you choose

[5]Date in this context is the product registration date.

will be validated.

An example of how the product can be displayed is shown in Figure 13.



Velg produkt:	Pakningsnivå	Pall	
Dato fra:	2007-01-01	Dato til:	2007-01-13

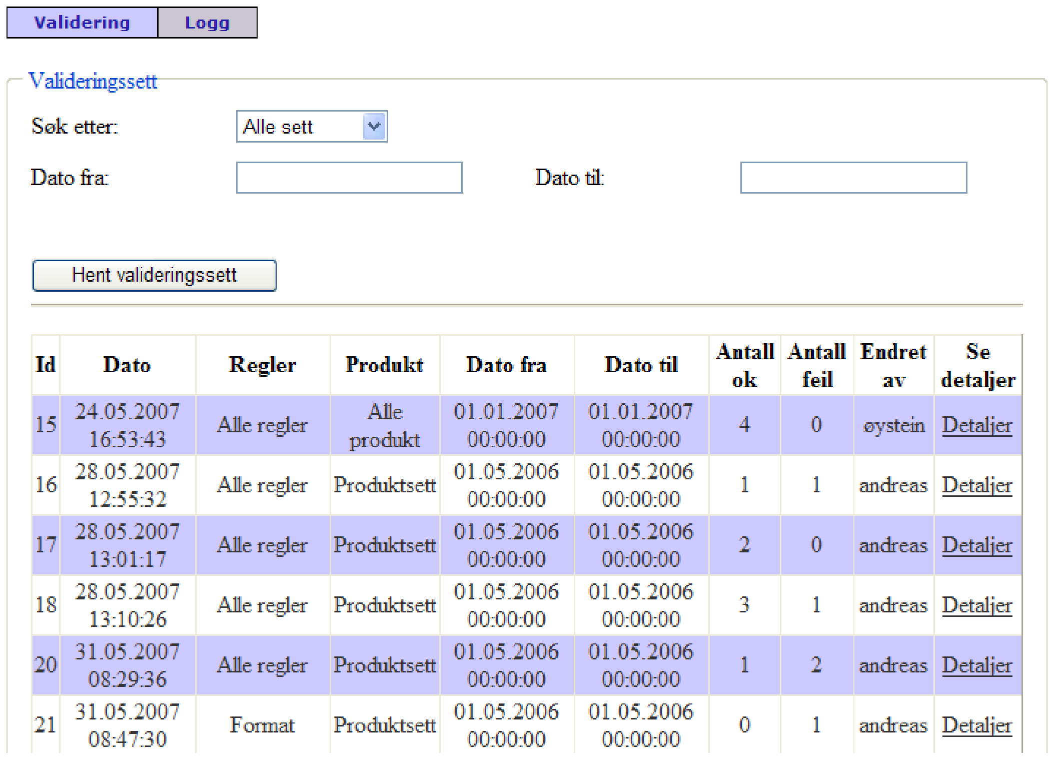
☐	GTIN	Produktnavn	Pakningsnivå	Dato opprettet
☐	7322540248463	TOILET ROLL NØYTRAL TOALETTPAPIR	DU	04.01.2007 15:06:59
☐	7044619722956	PEPSI LIGHT 4PK	DU	02.01.2007 00:00:00
☐	7070330078096	PETIT CHABKUS BIB 2005 12,5%	DU	02.01.2007 00:00:00
☐	7070330078485	LES NOUVEAUX GASCONS 75 CL 12%	DU	02.01.2007 00:00:00
☐	7070330078843	LAPHROAIG QUARTER CASK 70 C 48,0%	DU	02.01.2007 00:00:00
☐	47318690070922	JALAPENO GREEN 12 X120 ICA	DU	02.01.2007 09:29:30
☐	7023022065102	CÆSAR SALAT M/KYLLING LETTVINT	DU	10.01.2007 15:29:30
☐	7070330078959	GEWURTZTRAMINER RESERVE 75 12,0%	DU	11.01.2007 00:00:00
☐				11.01.2007

Figure 13: Products displayed.

After choosing rules and products you want to validate, you click the button "Kjør validering" to validate the products. The system will then validate all the rules that are checked toward all products that are checked and make a log in the database. After the validation is done you will be redirected to where you can see the log that was generated.

Under the "Logg"-tab you can see all information about the validation that has been made. Every time you validate products a new validate set will be inserted into the database. The first thing you have to do on this page is to choose what to display. You can choose between "Alle sett" (All set), "Sammenlignings-regler" (Compare rules), "Format-regler" (Format rules), "Antall Ok" (Number Ok) and "Antall feil" (Number mistake). Like the validation page this page has "from" and "to" date. These dates indicate when the validation was done. "Sammenlignings-regler" and "Format-regler" displays sets where these rules are used. When choosing "Antall Ok" and "Antall feil" two boxes will pop up. You have the possibility to insert a number and choose if approved number or failure number is higher, less or equal to your number. For instance, display sets where number mistakes is higher than 0. This will display all the sets where something is wrong.

When you have made your choice and the validation sets are displayed, you can click on "Detaljer", see Figure 14. You will then be redirected to a new page which contains an overview of the combination of rules and product that was validated in this set, see Figure 15. The information is rule name and product name plus set id, rule id, gtin, pack level and pack type. The purpose of this page is to get an overview over rules and products. The overview will contain one row for each record in the database since the validation of each product must be done for all the rules that was selected for validation. As you can see in Figure 15 the column "Godkjent" (Approved) will have one of two colours, red or green. This indicates whether the validation was approved (green), or rejected (red).



Søk etter: Alle sett

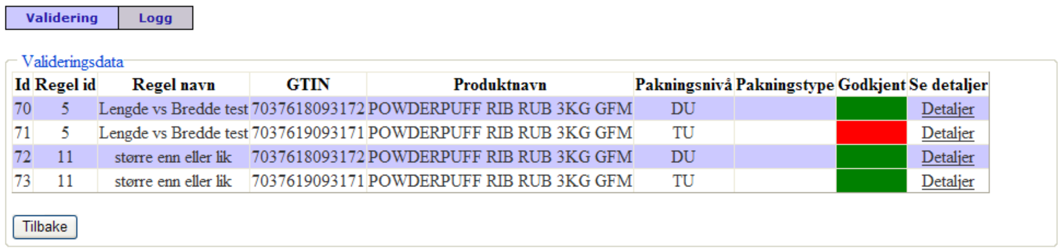
Dato fra:

Dato til:

Id	Dato	Regler	Produkt	Dato fra	Dato til	Antall ok	Antall feil	Endret av	Se detaljer
15	24.05.2007 16:53:43	Alle regler	Alle produkt	01.01.2007 00:00:00	01.01.2007 00:00:00	4	0	øystein	Detaljer
16	28.05.2007 12:55:32	Alle regler	Produktsett	01.05.2006 00:00:00	01.05.2006 00:00:00	1	1	andreas	Detaljer
17	28.05.2007 13:01:17	Alle regler	Produktsett	01.05.2006 00:00:00	01.05.2006 00:00:00	2	0	andreas	Detaljer
18	28.05.2007 13:10:26	Alle regler	Produktsett	01.05.2006 00:00:00	01.05.2006 00:00:00	3	1	andreas	Detaljer
20	31.05.2007 08:29:36	Alle regler	Produktsett	01.05.2006 00:00:00	01.05.2006 00:00:00	1	2	andreas	Detaljer
21	31.05.2007 08:47:30	Format	Produktsett	01.05.2006 00:00:00	01.05.2006 00:00:00	0	1	andreas	Detaljer

Figure 14: Sets displayed in log.

The Figure 16 will be shown when you click on "Detaljer" in Figure 15. Here all the information related to the validation between rule and product will be displayed. The seven rows at the top, from "Id" to "Pakningstype" will always contain information. The rows that come after this are split into two groups, - one for compare rules and one for format rules. The fields "Uttrykk", "Uttrykk med data", "Forhold", "Valideringsuttrykk", "Valideringsuttrykk med data", "Toleransemål", "Toleranseprosent" and



Id	Regel id	Regel navn	GTIN	Produktnavn	Pakningsnivå	Pakningstype	Godkjent	Se detaljer
70	5	Lengde vs Bredde test	7037618093172	POWDERPUFF RIB RUB 3KG GFM	DU		🟩	Detaljer
71	5	Lengde vs Bredde test	7037619093171	POWDERPUFF RIB RUB 3KG GFM	TU		🟥	Detaljer
72	11	større enn eller lik	7037618093172	POWDERPUFF RIB RUB 3KG GFM	DU		🟩	Detaljer
73	11	større enn eller lik	7037619093171	POWDERPUFF RIB RUB 3KG GFM	TU		🟩	Detaljer

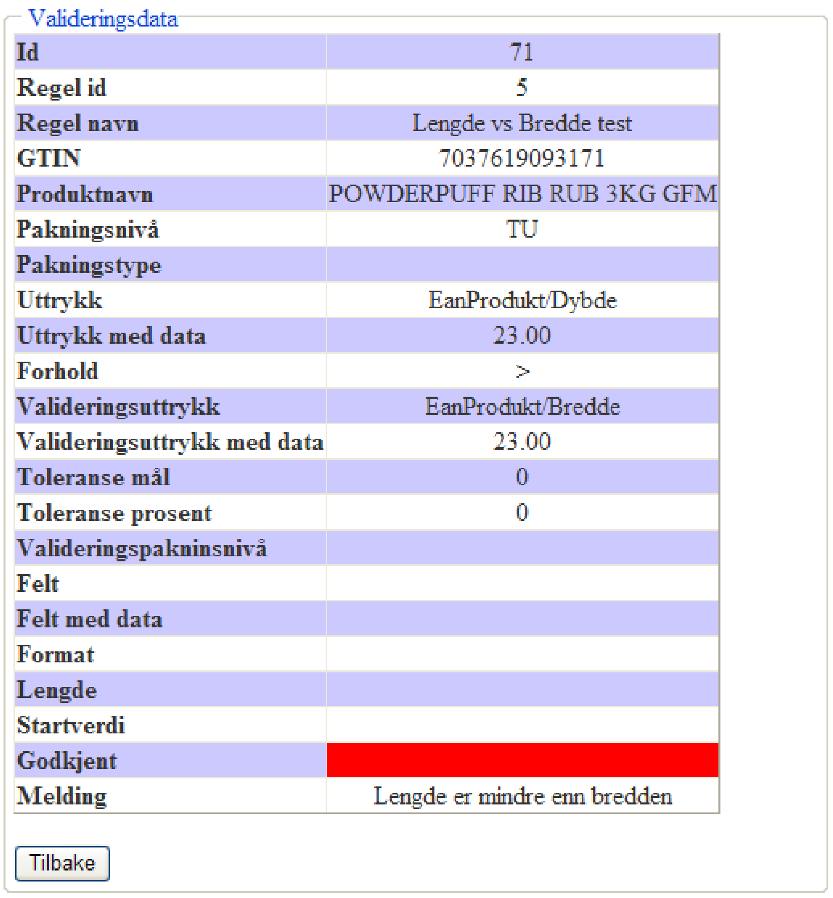
Figure 15: Overview of rules and product.

"Valideringspakningsnivå" will be used when the rule is a compare rule. The rest of the fields belongs to the format rules, except "Godkjent", which indicates if the validation is approved or not, and "Melding" which contains the error message if validation was rejected. Information about each field can be found in Appendix E.

4.3 Administration

On the Administration page you have the possibility to change, delete and copy rules. You will also get an overview of all the rules that are available. Information about the server you want to connect to can also be changed here. You can choose between "Regler" (Rules) and "Server" in the administration page.

"Regler" gives you the possibility to see all rules that are available. You have to choose between "Sammenligning" (Compare) and "Format" since we have two types of rules, see Figure 17 to see the overview. In every text box that are displayed, you can change the rule. In this way, when you have become a more experienced user, you do not have to go through all the steps in the



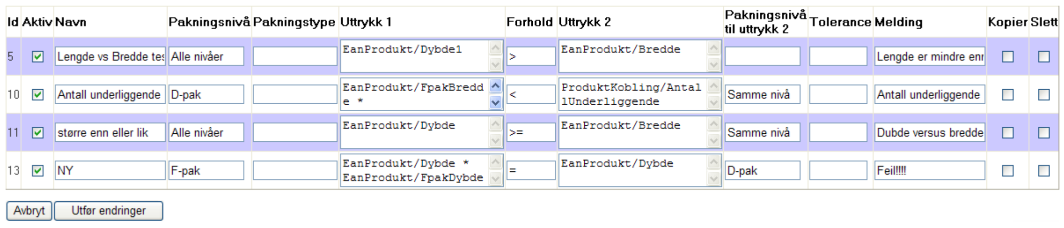
Valideringsdata

Id	71
Regel id	5
Regel navn	Lengde vs Bredde test
GTIN	7037619093171
Produktnavn	POWDERPUFF RIB RUB 3KG GFM
Pakningsnivå	TU
Pakningstype	
Uttrykk	EanProdukt/Dybde
Uttrykk med data	23.00
Forhold	>
Valideringsuttrykk	EanProdukt/Bredde
Valideringsuttrykk med data	23.00
Toleranse mål	0
Toleranse prosent	0
Valideringspakninsnivå	
Felt	
Felt med data	
Format	
Lengde	
Startverdi	
Godkjent	🟥
Melding	Lengde er mindre enn bredden

Figure 16: Details of what has been done in the validation.

wizard, shown in Section 4.4. Users can copy and change the rules directly. This require knowledge about the Rule Engine and about the database.

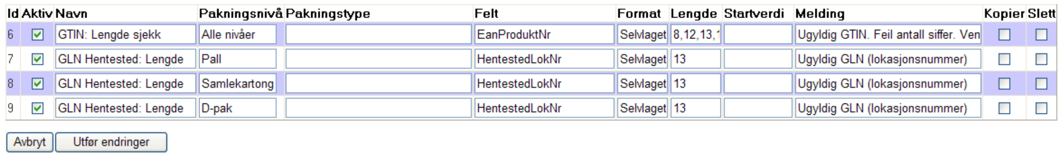
Another possibility in the Administration page is to enable and disable rules. By clicking on the check box under "Aktiv" (Activated). When a change has been done by the user, the button "Utfør endringer" (Execute changes) must be activated. Also, when removing and copying rules you have to push this button to execute the changes.



Id	Aktiv	Navn	Pakningsnivå	Pakningstype	Uttrykk 1	Forhold	Uttrykk 2	Pakningsnivå til uttrykk 2	Tolerance	Melding	Kopier	Slett
5	✓	Lengde vs Bredde tes	Alle nivåer		EanProdukt/Dybde1	>	EanProdukt/Bredde			Lengde er mindre enn	☐	☐
10	✓	Antall underliggende	D-pak		EanProdukt/FpakBredde *	<	ProduktKobling/Antal lUnderliggende	Samme nivå		Antall underliggende	☐	☐
11	✓	større enn eller lik	Alle nivåer		EanProdukt/Dybde	>=	EanProdukt/Bredde	Samme nivå		Dubde versus bredde	☐	☐
13	✓	NY	F-pak		EanProdukt/Dybde * EanProdukt/FpakDybde	=	EanProdukt/Dybde	D-pak		Feil!!!!	☐	☐

Figure 17: Overview of Compare rules.

Administration of format rules works in the same way as compare rules. Figure 18 depicts the Format rule.

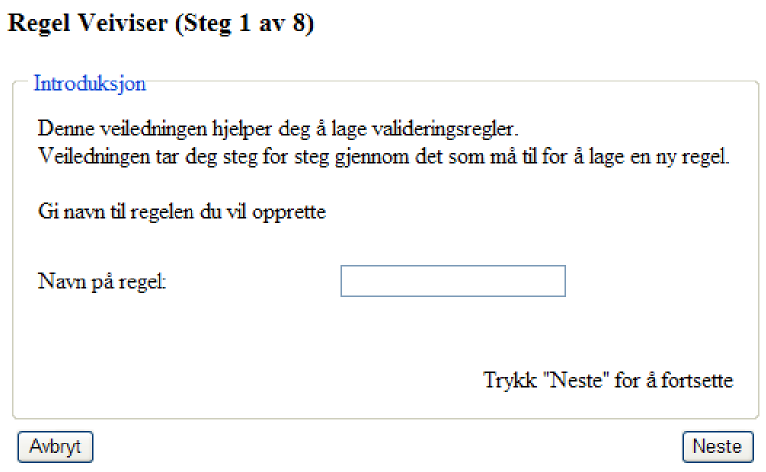


Id	Aktiv	Navn	Pakningsnivå	Pakningstype	Felt	Format	Lengde	Startverdi	Melding	Kopier	Slett
6	✓	GTIN: Lengde sjekk	Alle nivåer		EanProduktNr	Selvlaget	8,12,13,1		Ugyldig GTIN. Feil antall siffer. Ven	☐	☐
7	✓	GLN Hentested: Lengde	Pall		HentestedLokNr	Selvlaget	13		Ugyldig GLN (lokasjonsnummer)	☐	☐
8	✓	GLN Hentested: Lengde	Samlekartong		HentestedLokNr	Selvlaget	13		Ugyldig GLN (lokasjonsnummer)	☐	☐
9	✓	GLN Hentested: Lengde	D-pak		HentestedLokNr	Selvlaget	13		Ugyldig GLN (lokasjonsnummer)	☐	☐

Figure 18: Overview of Format rules.

4.4 Create rules

Under the "Lage regel" tab you have two choices, you can create "compare rules" or "format rules". To build a rule you have to go through a wizard. Both types of rules are based on the same wizard. The introduction is the first step in both wizard, see Figure 19. Below we show a simple explanation, of how to create a rule. In the first step you can type a name for the rule, see Figure 19. At the bottom of Figure 19 there is two buttons. "Avbryt" (Abort) is used to abort and go to the first step in the wizard, which is introduction. This button will be present in all the steps so you can abort at any time. The "Neste" (Next) button is used to get to the next step in the wizard, after typing a name for the rule.


Regel Veiviser (Steg 1 av 8)

Introduksjon

Denne veiledningen hjelper deg å lage valideringsregler.
Veiledningen tar deg steg for steg gjennom det som må til for å lage en ny regel.

Gi navn til regelen du vil opprette

Navn på regel:

Trykk "Neste" for å fortsette

Figure 19: Step 1: The introduction to create a rule.

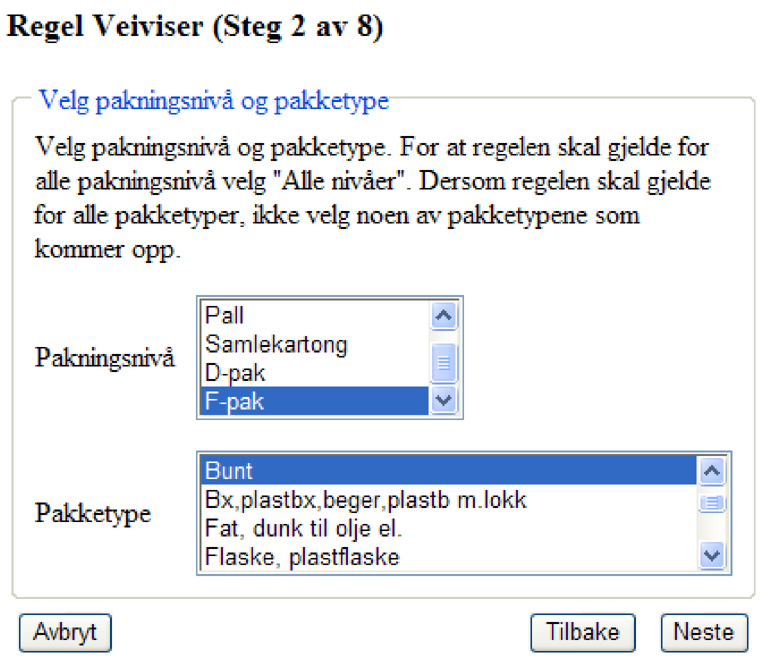
On the right hand side of step 1 in the wizard, you will see the same as in Figure 20. This is an overview of what you have done earlier in the wizard. It is empty in the first step since we have not done anything yet.

Figure 20: Step 1: Overview of the rule.

Step 2 in the wizard is to choose pack level and pack type for this rule, see Figure 21. You must decide if the rule is for all levels or just one level. If you want the rule to available for all pack types in a pack level you do not need to choose a pack type. At the bottom of step 2, see Figure 21, a new button is shown. The "Tilbake" (Back) button goes back one step in the wizard. This step is the same for both compare rule and format rule.

In the overview we can see that the rule name is shown, see Figure 22. In this step you select a field or fields from the database, see Figure 23.

If you want to make an expression you can select more than one field, but you must have mathematical signs between the fields.



Regel Veiviser (Steg 2 av 8)

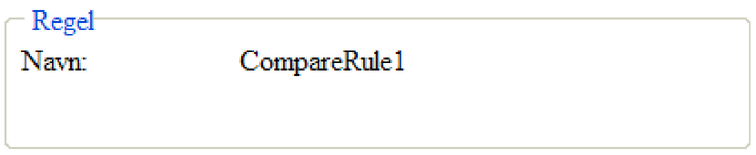
Velg pakningsnivå og pakketype

Velg pakningsnivå og pakketype. For at regelen skal gjelde for alle pakningsnivå velg "Alle nivåer". Dersom regelen skal gjelde for alle pakketyper, ikke velg noen av pakketypene som kommer opp.

Pakningsnivå: Pall, Samlekartong, D-pak, F-pak (selected)

Pakketype: Bunt (selected), Bx,plastbx,beger,plastb m.lokk, Fat, dunk til olje el., Flaske, plastflaske

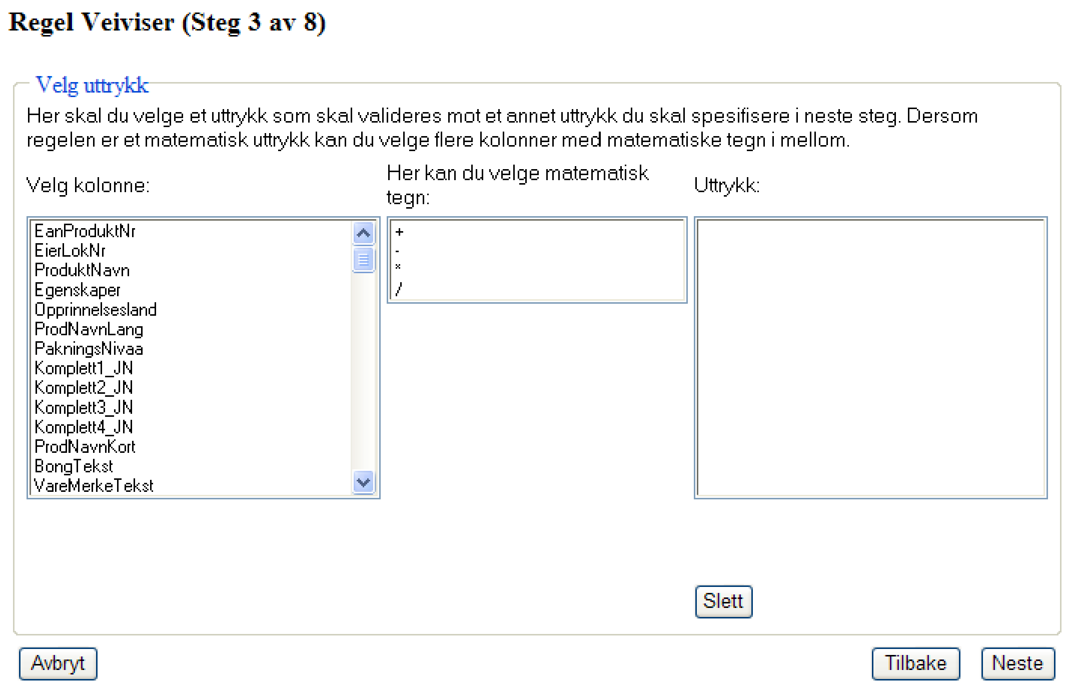
Figure 21: Step 2: Choose pack level and pack type.



Regel

Navn: CompareRule1

Figure 22: Step 2: Overview of the rule.


Regel Veiviser (Steg 3 av 8)

Velg uttrykk

Her skal du velge et uttrykk som skal valideres mot et annet uttrykk du skal spesifisere i neste steg. Dersom regelen er et matematisk uttrykk kan du velge flere kolonner med matematiske tegn i mellom.

Velg kolonne:

EanProduktNr
EierLokNr
ProduktNavn
Egenskaper
Opprinnelsesland
ProdNavnLang
PakningsNivaa
Komplett1_JN
Komplett2_JN
Komplett3_JN
Komplett4_JN
ProdNavnKort
BongTekst
VareMerkeTekst

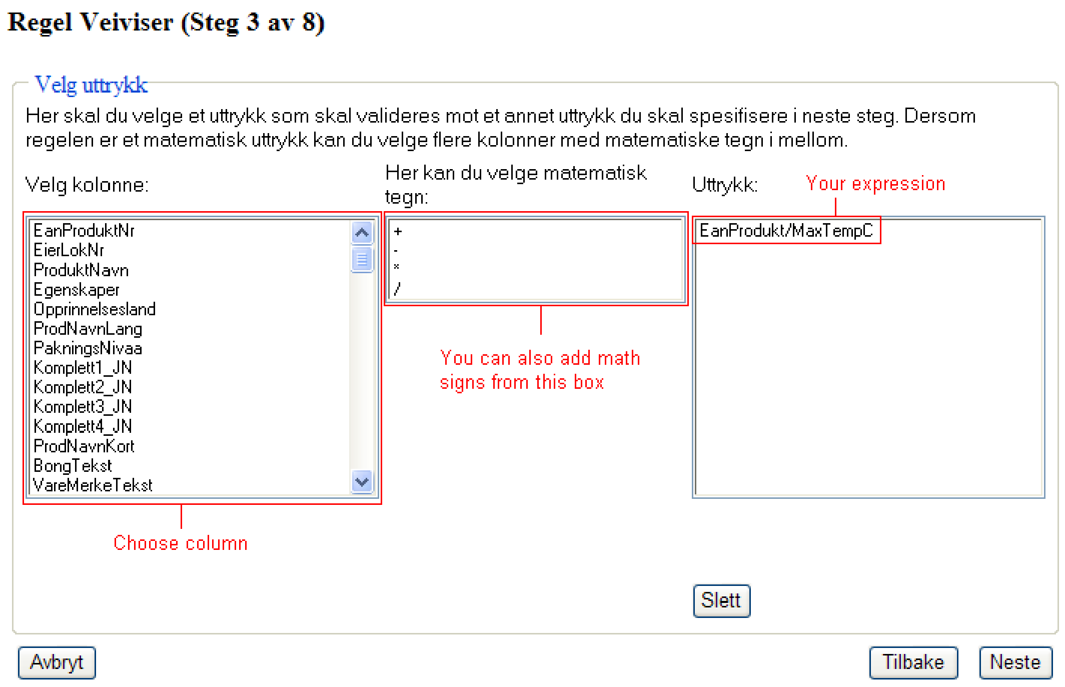
Her kan du velge matematisk tegn:

+
-
*
/

Uttrykk:

Figure 23: Compare rule, step 3: Make expression.

All the steps that will be shown here can be seen in Figure 24. All the columns in the "Eanprodukt" table is displayed in the text box on the left hand side. When choosing a column the table name and column name will be displayed in the expression box. If you want more than one field you can add mathematical signs from the appropriate box. In this way you can build your own rules based on the columns in the database. If you want to delete something that you have chosen, you can use the "Slett" (Delete) button. This will delete the last inserted field in the expression box.



Regel Veiviser (Steg 3 av 8)

Velg uttrykk

Her skal du velge et uttrykk som skal valideres mot et annet uttrykk du skal spesifisere i neste steg. Dersom regelen er et matematisk uttrykk kan du velge flere kolonner med matematiske tegn i mellom.

Velg kolonne:

EanProduktNr
EierLokNr
ProduktNavn
Egenskaper
Opprinnelsesland
ProdNavnLang
PakningsNivaa
Komplett1_JN
Komplett2_JN
Komplett3_JN
Komplett4_JN
ProdNavnKort
BongTekst
VareMerkeTekst

Her kan du velge matematisk tegn:

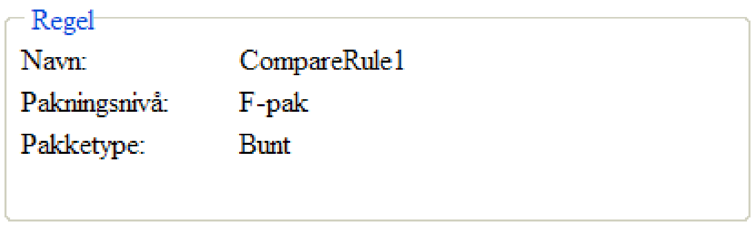
+
-
*
/

Uttrykk:

EanProdukt/MaxTempC

Figure 24: Compare rule, step 3: The steps on making an expression.

In the overview on the right hand side we see that pack level and pack type have been added, see Figure 25.



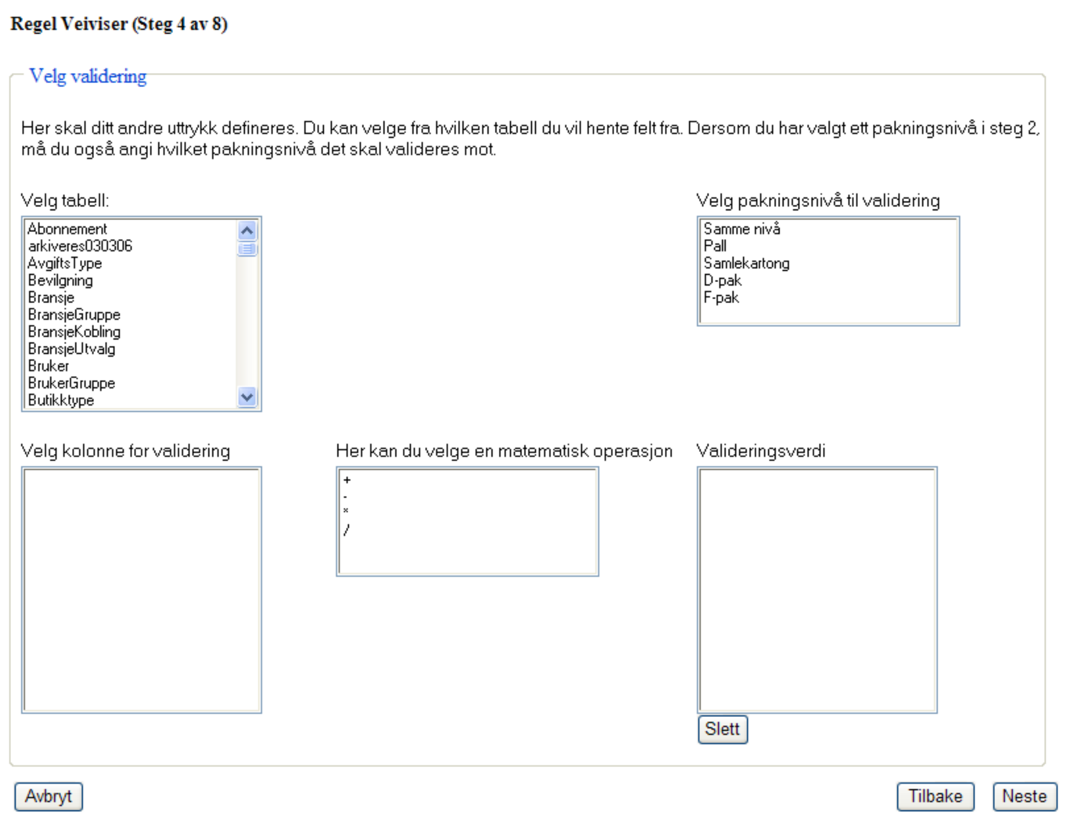
Navn:	CompareRule1
Pakningsnivå:	F-pak
Pakketype:	Bunt

Figure 25: Compare rule, step 3: Overview of the rule.

Step 4 is like step 3, see Figure 26. This is the validation expression. This step works just like step 3. The only ting that is different is that you can select a table you want and the pack level of the validation expression. In this example we will choose table "EanProdukt" and the column "MinTempC". Pack level is "F-pak".

In the overview on the right hand side, we see that the expression has been added, see Figure 27.

In step 5 you have to decide the relation between the expression and the validation expression. In Figure 28 you can see that the expression is displayed in the box on the left hand side and the validation expression is displayed in the box on the right hand side. Between them is a combo box which consist of six values: "more than", "more than or equal", "less than", "less than or equal", "equal" and "different". In our example we want "MaxTempC", which is the maximum temperature, to be higher than "MinTempC", which is the minimum temperature.



Regel Veiviser (Steg 4 av 8)

Velg validering

Her skal ditt andre uttrykk defineres. Du kan velge fra hvilken tabell du vil hente felt fra. Dersom du har valgt ett pakningsnivå i steg 2, må du også angi hvilket pakningsnivå det skal valideres mot.

Velg tabell:
- Abonnement
- arkiveres030306
- AvgiftsType
- Bevilgning
- Bransje
- BransjeGruppe
- BransjeKobling
- BransjeUtvalg
- Bruker
- BrukerGruppe
- Butikktype

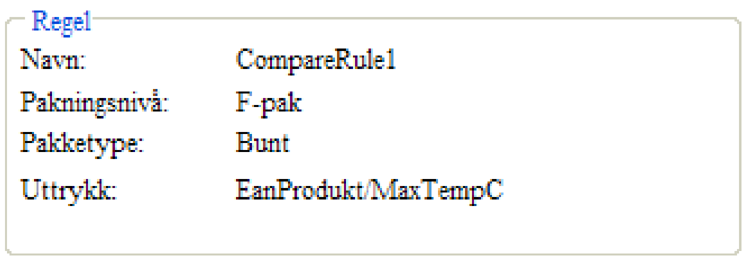
Velg pakningsnivå til validering
- Samme nivå
- Pall
- Samlekartong
- D-pak
- F-pak

Velg kolonne for validering

Her kan du velge en matematisk operasjon
- +
- -
- *
- /

Valideringsverdi

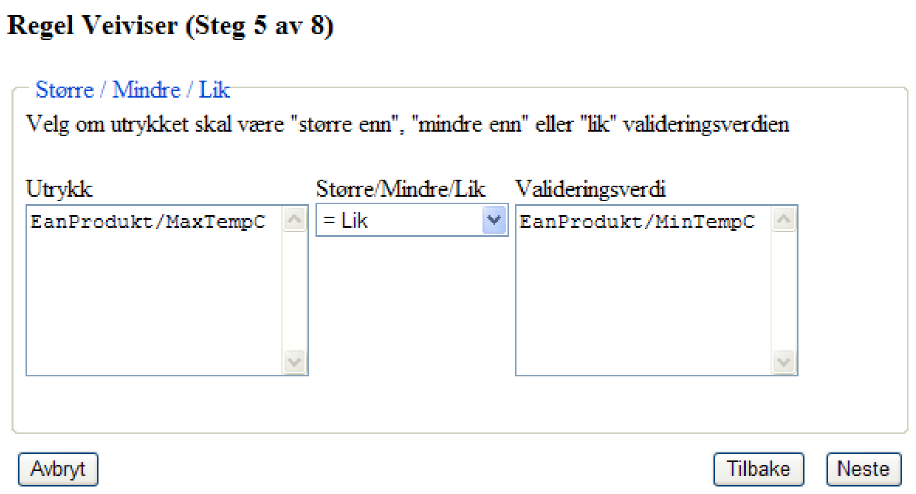
Figure 26: Compare rule, step 4: Make validation expression.



Regel

Navn:	CompareRule1
Pakningsnivå:	F-pak
Pakketype:	Bunt
Uttrykk:	EanProdukt/MaxTempC

Figure 27: Compare rule, step 4: Overview of the rule.


Regel Veiviser (Steg 5 av 8)

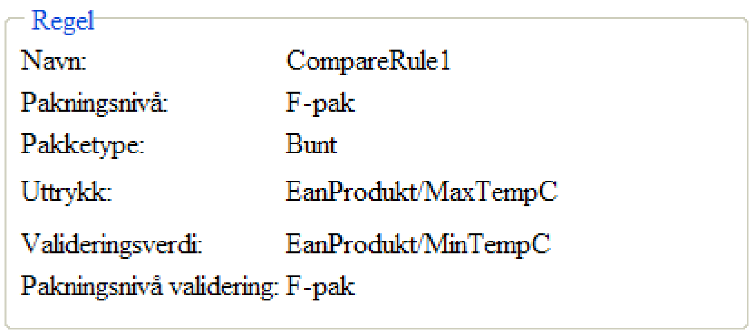
Større / Mindre / Lik

Velg om utrykket skal være "større enn", "mindre enn" eller "lik" valideringsverdien

Utrykk	Større/Mindre/Lik	Valideringsverdi
EanProdukt/MaxTempC	= Lik	EanProdukt/MinTempC

Figure 28: Compare rule, step 5: Decide the relation between the expression and the validation expression.

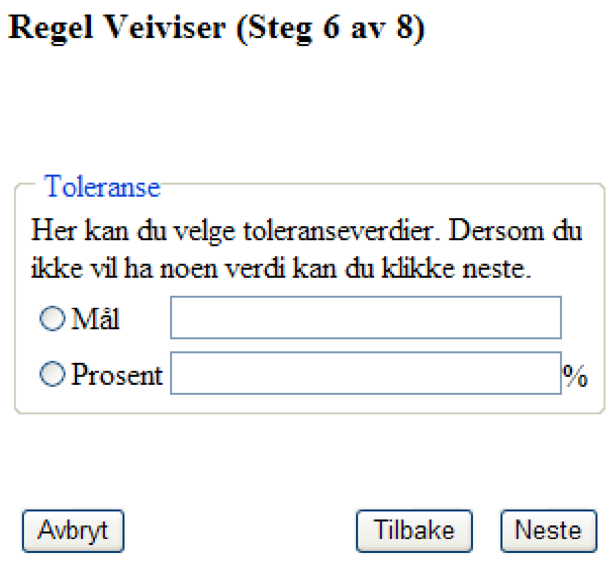
In the overview on the right hand side we see that the validation expression and validation pack level have been added, see Figure 29.


Regel

Navn:	CompareRule1
Pakningsnivå:	F-pak
Pakketype:	Bunt
Uttrykk:	EanProdukt/MaxTempC
Valideringsverdi:	EanProdukt/MinTempC
Pakningsnivå validering:	F-pak

Figure 29: Compare rule, step 5: Overview of the rule.

If the rule should be accepted by a variance, a tolerance value can be added in step 6. You have the possibility to add either measure or percentage tolerance, see Figure 30. In our example we do not add any tolerance because we never want the maximum temperature to be less than minimum temperature.


Regel Veiviser (Steg 6 av 8)

Toleranse

Her kan du velge toleranseverdier. Dersom du ikke vil ha noen verdi kan du klikke neste.

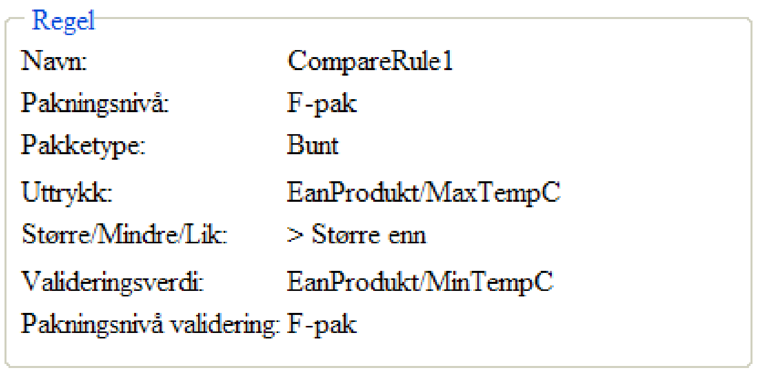
○ Mål
○ Prosent %

Figure 30: Compare rule, step 6: Tolerance.

In the overview on the right hand side we see that the relation have been added, see Figure 31.

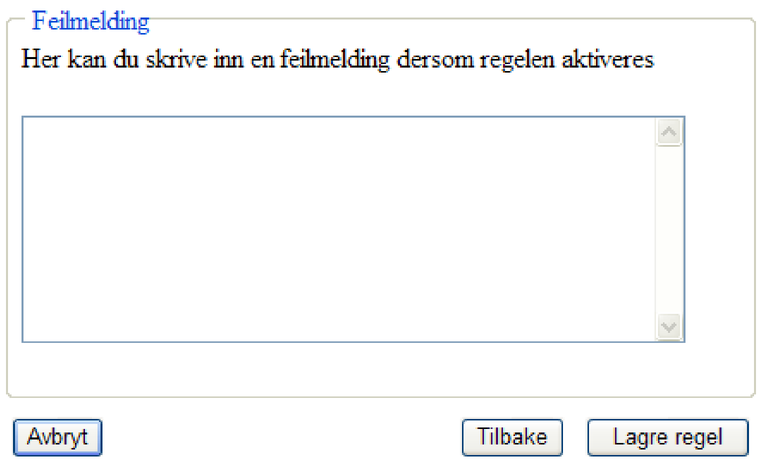
In step 7 you add the failure message that will be displayed when this rule is rejected, see Figure 32. The rule definition is now finished and the last thing to do is to save it in XML format. Clicking the "Lagre regel" (Save rule) button will save the rule.

When the rule is saved, an overview on the right hand side of the rule is displayed in step 8, see Figure 33.


Regel

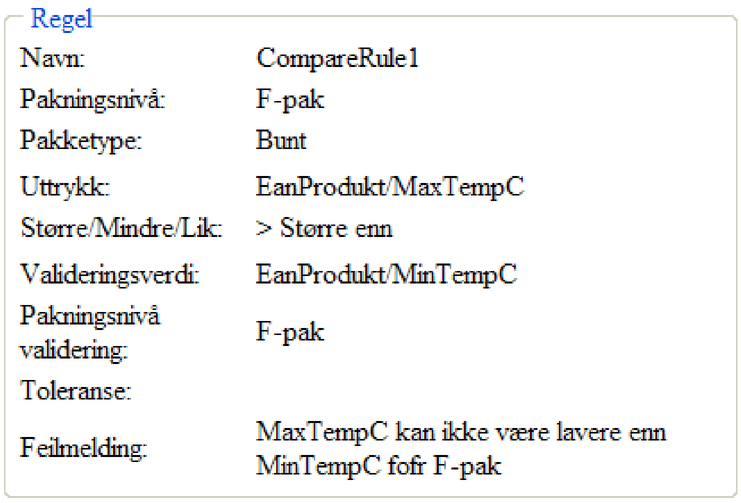
Navn:	CompareRule1
Pakningsnivå:	F-pak
Pakketype:	Bunt
Uttrykk:	EanProdukt/MaxTempC
Større/Mindre/Lik:	> Større enn
Valideringsverdi:	EanProdukt/MinTempC
Pakningsnivå validering:	F-pak

Figure 31: Compare rule, step 6: Overview of the rule.


Feilmelding

Her kan du skrive inn en feilmelding dersom regelen aktiveres

Figure 32: Step 7: Failure message.



Regel

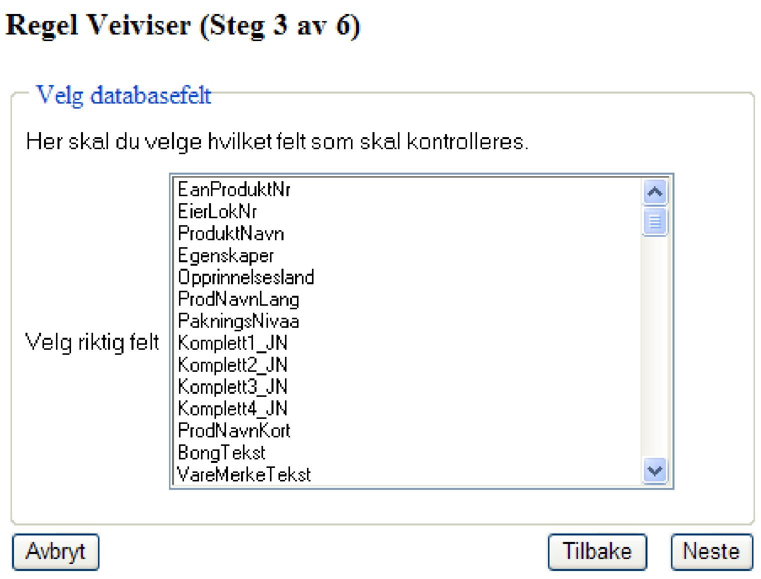
Navn:	CompareRule1
Pakningsnivå:	F-pak
Pakketype:	Bunt
Uttrykk:	EanProdukt/MaxTempC
Større/Mindre/Lik:	> Større enn
Valideringsverdi:	EanProdukt/MinTempC
Pakningsnivå validering:	F-pak
Toleranse:	
Feilmelding:	MaxTempC kan ikke være lavere enn MinTempC fofr F-pak

Figure 33: Step 8: Overview of the rule you just have made.

Now lets have a look at the Format rule. The two first step is the same as in Compare rule, see Figure 19 and 21. The third step are different and shown in Figure 34. In this step you have to select a database field which going to be controlled. The database field that are shown is from table "EanProdukt" in EPD-basen (2.1).

In the overview on the right hand side we see that the name and tack level and pack type have been added to the rule, see Figure 35.

Step 4 in the Format rule is where you decide the format of the rule, see Figure 36. You can choose between "J/N" (Y/N) and "Selvlaget" (User defined). "J/N" means that if the database field is something it has to be either "J", "N" or "NULL". "J" and "N" indicates yes and no. "NULL"



Regel Veiviser (Steg 3 av 6)

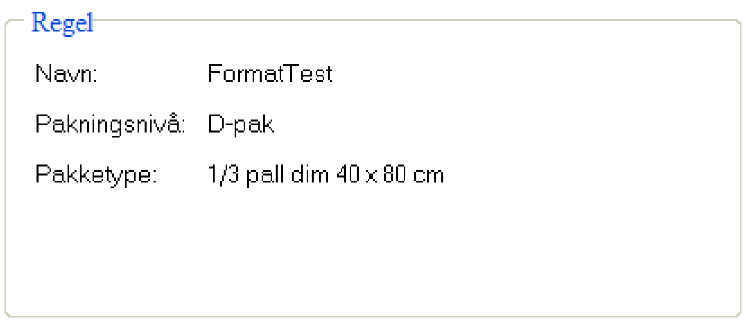
Velg databasefelt

Her skal du velge hvilket felt som skal kontrolleres.

Velg riktig felt

EanProduktNr
EierLokNr
ProduktNavn
Egenskaper
Opprinnelsesland
ProdNavnLang
PakningsNivaa
Komplett1_JN
Komplett2_JN
Komplett3_JN
Komplett4_JN
ProdNavnKort
BongTekst
VareMerkeTekst

Figure 34: Format rule, step 3: Select a database field.

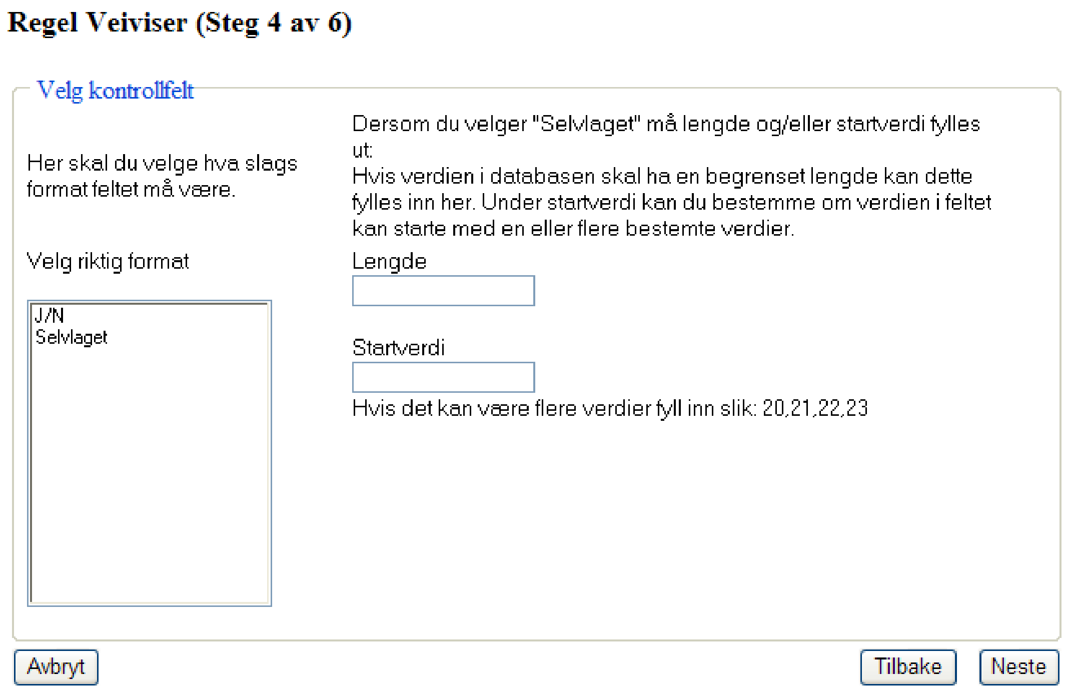


Regel

Navn:	FormatTest
Pakningsnivå:	D-pak
Pakketype:	1/3 pall dim 40 x 80 cm

Figure 35: Format rule, step 3: Overview of the rule.

means that it is empty. If you choose "Selvlaget" you can define you own rule by using length and/or start values. E.g. "EanProduktNr" has to have the length 8, 12, 13 or 14 to be correct. In the length text box we type "8,12,13,14". The same is for the start value. Some rules require that the database field must begin with e.g. two specified numbers. We then type this numbers into the start value text box in the same way we did for the length.


Regel Veiviser (Steg 4 av 6)

Velg kontrollfelt

Her skal du velge hva slags format feltet må være.

Dersom du velger "Selvlaget" må lengde og/eller startverdi fylles ut:
Hvis verdien i databasen skal ha en begrenset lengde kan dette fylles inn her. Under startverdi kan du bestemme om verdien i feltet kan starte med en eller flere bestemte verdier.

Velg riktig format

J/N
Selvlaget

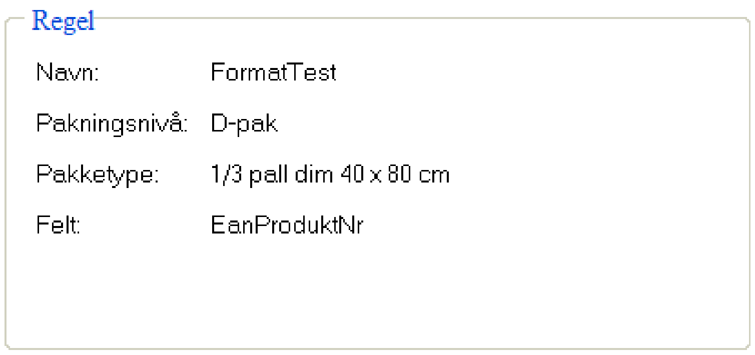
Lengde

Startverdi

Hvis det kan være flere verdier fyll inn slik: 20,21,22,23

Figure 36: Format rule, step 4: Select format.

In the overview on the right hand side displays the database field chosen, see Figure 37.

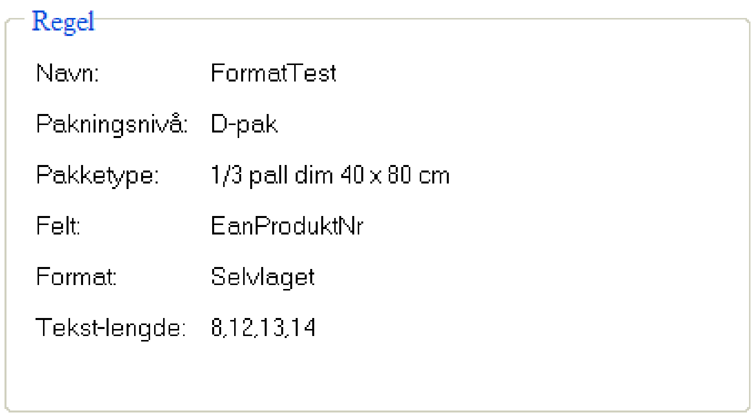


Regel

Navn:	FormatTest
Pakningsnivå:	D-pak
Pakketype:	1/3 pall dim 40 x 80 cm
Felt:	EanProduktNr

Figure 37: Format rule, step 4: Overview of the rule.

Step 5 is the same as step 7 for Compare rule, see Figure 32. Here you can type in the error message. In the overview on the right hand side we see what the format of the rule have been, see Figure 38.

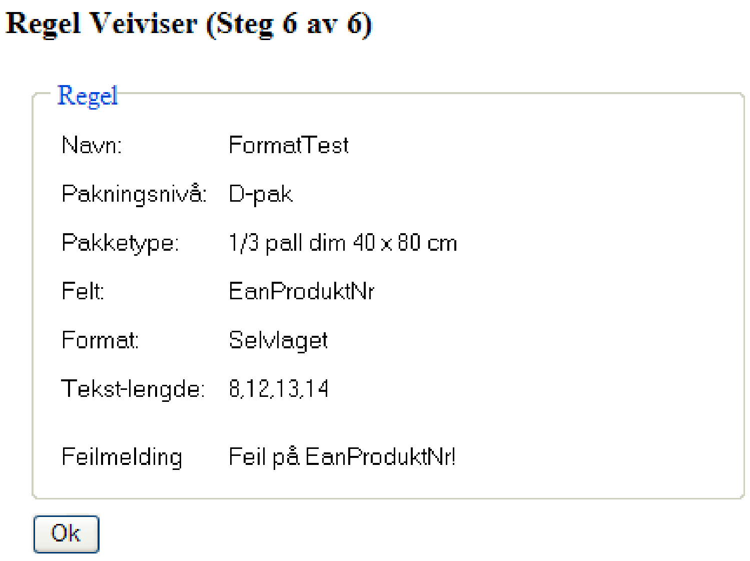


Regel

Navn:	FormatTest
Pakningsnivå:	D-pak
Pakketype:	1/3 pall dim 40 x 80 cm
Felt:	EanProduktNr
Format:	Selvlaget
Tekst-lengde:	8,12,13,14

Figure 38: Format rule, step 5: Overview of the rule.

Step 6 is the last step where you can see the rule you have build, see Figure 39.


Regel Veiviser (Steg 6 av 6)

Regel

Navn:	FormatTest
Pakningsnivå:	D-pak
Pakketype:	1/3 pall dim 40 x 80 cm
Felt:	EanProduktNr
Format:	Selvlaget
Tekst-lengde:	8,12,13,14
Feilmelding	Feil på EanProduktNr!

Figure 39: Format rule, step 6: Overview of the rule.

5 Development method

In this chapter we describe the method used to develop the Rule Engine and some examples to achieve a satisfactory level of user-friendliness by using AJAX.

5.1 Development of the Rule Engine

In the beginning of the development of the new Rule Engine, we used CVE, see Section 2.3. All the rules in CVE was divided into groups. The developers could place the rules into two groups, comparison and format rules. This made the development much easier when we generalized the rules. The new system was based on two main groups and the development started.

One of our main objective was to develop a system with good user-friendliness. With that in mind the developers made a wizard the user could use when a rule is made. This wizard take one step at a time and explain what each step implies. For more information about the steps see Section 4.4.

It is important for users of the system to have an overview over what kind of rules that are made. The developers created an administration page, see Section 4.3, where users can see and edit the rules that are built. In order to make the system user-friendly we introduced the "copy rule" functionality. For more experienced users this will be a a good method. You can drop the wizard and copy a rule and edit the new rule in any way you like. In this way you save a lot of time and frustration when the wizard steps are superfluous.

The most important and new development is the rule validation part. The user can select rules and products he want to validate, see Section 4.2. The validation get information about tables and columns in the rule.xml file. The data in the database are controlled with other data in the database. We can also control the format, such as the length and start values of a database field.

When developing the Rule Engine we have focused on user-friendliness and robustness. To reach the robustness objective we have connected the system to database field in the EPD-base (2.1). In this way the rules made from the wizard must be correct. Further we have tested each method that have been developed. If there have been any problems with a method, the problem have been fixed before a new part of the system was developed. In this way the robustness has been a main objective for the developers.

When the Rule Engine was finished, we wanted to test the system on the real users to get feedback on the system's functionality. The users used some time to test the system and gave us feedback by using a survey we made, see Section 8. It is important to get information about what can be done different and what we can do better in the future. We used this information to improve the system. The aspects we did not complete we have added as aspects to do in the next version of the Rule Engine.

5.2 AJAX in the Rule Engine

To achieve a satisfactory level of user-friendliness in our project we decided to use ASP.NET AJAX. We used the following AJAX controls:

UpdatePanel Control

"ASP.NET UpdatePanel controls enable developers to build rich, client-centric Web applications. By using UpdatePanel controls, you can refresh selected parts of the page instead of refreshing the whole page with a post back. This is referred to as performing a partial-page update" [4]. This control is used in almost every page. For example it is used in the wizards. When users click on "Back" or "Next" only the area of the wizard is updated, nothing else on the page. Also used in Administration and Validate page.

UpdateProgress Control

"The UpdateProgress control provides status information about partial-page updates in UpdatePanel controls" [4]. This control is used in the Validation page. If the users requests a lot of products in the Validation page, the user can see the update progress and it is a indication that the system is working. Also used when getting rules in the Validation page.

Calendar

"Calendar is an ASP.NET AJAX extender that can be attached to any ASP.NET text box control. It provides client-side date-picking functionality with customizable date format and UI in a pop up control. Users can interact with the calendar by clicking on a day to set the date. In addition, the left and right arrows can be used to move forward or back a month. By clicking on the title of the calendar you can change the view from Days in the current month, to Months in the current year. Another click will switch to Years in the current Decade. This action allows you to easily jump to dates in the past or the future from within the calendar control" [4]. This control is used in the Validation page, where users can select "to" and "from" date, date in this context is product creation date.

FilteredTextBox

"FilteredTextBox is an extender which prevents a user from entering invalid characters into a text box" [4]. This control is used in the wizard step where the user can specify a tolerance value. The text box only allows a numeric value.

6 Software system test

An important step when developing a software system is the system test. This chapter presents the software system testing and the results of the test.

6.1 Software testing and quality attributes

Software testing is the process used to help identify the correctness, completeness, security, and quality of developed computer software. Testing is a process of technical investigation, performed on behalf of all stakeholders, that is intended to reveal quality-related information about the product with respect to the context in which it is intended to operate. This includes the process of executing the program with the intent of finding errors. In this context it is the debugging on the software system to find errors, not gaining confidence in the system, this will be done in the survey.

We comprehend quality an absolute value, it is value to some person. In our project user-friendliness and robustness are the most important quality attributes for the stakeholders. ISO 9126 [6], define then quality attributes as follows:

- Usability - A set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users. The attributes are:
  - Learnability
  - Understandability

  - Operability

- Robustness - A quality of being able to withstand stresses, pressures, or changes in procedure or circumstance. A system, organism or design may be said to be "robust" if it is capable of coping well with variations in its operating environment with minimal damage, alteration or loss of functionality.

6.2 White-box testing

The method used for software system testing in this project is white-box testing. White-box or logic-driven testing, permits us to examine the internal structure of the program. This strategy derives test data from an examination of the program's logic. White box testing includes analyzing data flow, control flow, information flow, coding practices, and exception and error handling within the system, to test intended and unintended software behavior. White box testing can be performed to validate whether code implementation follows intended design, to validate implemented security functionality, and to uncover exploitable vulnerabilities [7].

White box testing requires access to the source code. Testing of the software system has been executed in parallel with the development of the system. When one system part has been finished developed, it has been tested. The system has been divided into parts that can be tested individually. Based on the test results, further work may be required to reach an acceptable level for each part to reach the wanted system quality.

6.2.1 Test plan

Before the development of the system started, we defined a test plan, see Table 1. In this test plan we divided the system into parts and the goals to be achieved for each part of the system.

System parts	*Goals*	*Description*
Create rule	Save to XML	Save the right values and structure
	Get data from database	Get right information from the database
	Wizard steps	Show selected values when navigating the wizard
Administrate rule	Edit rule	Change saved rule data
	Copy rule	Copy existing rule data to a new rule
	Delete rule	Delete existing rule
	Server settings	Configure server data
Validation	Display rule	Show selected rules from XML
	Display product	Show selected products from the database
	Validate compare rule	Control that validation is done correctly
	Validate format rule	Control that validation is done correctly
	Validation of product	Validate correct product
	Log	Show validation data

Table 1: Table over system test goals

6.2.2 Test case specification

For each of the goals in the test plan we specified a test case. The cases are listed below. For a detailed description of each case, see Appendix C. Below we also give an indication of whether the test was a success or a failure the first time tested.

We choosed these parts to test because they are critical for the system to work properly.

6.2.3 Test results and supplementary work

The tests here will only be shortly described, for more information see the Appendix C. In the following we will look at the tests that failed and describe the supplementary work done after the tests.

C.1 **Wizard steps** - Failed
When the 'Back' button in the wizard was pressed some of the data was lost. The reason for this was some unfinished coding. To fix this, we had to save data in variables.

C.2 **Get data from database** - Success
Getting data from the database was a success each time we tested.

C.3 **Save user data to XML** - Failed
The problem occurred when we tried to save expressions as 'less then - <' or 'greater then - >' in the XML file. These symbols are interpreted as start-

and end tags in XML. All XML files has certain by-passing code for these symbols, '<' is '&lt;' and '>' is '&gt;'. This was not a direct problem, we just had to include these special symbols in the validation logic.

C.4 **Edit rule** - Success
Editing rules saved the right values each time we tested.

C.5 **Copy rule** - Success
Copying rules copied the right rule each time we tested.

C.6 **Delete rule** - Success
Deleting rules deleted the right rule each time we tested.

C.7 **Server settings** - Success
Changing the server setting connected correctly to defined server each time we tested.

C.8 **Display rule** - Failed
When the user get all the rules listed, we also got the rules that were inactive. This problem was easy to solve in the code, we now check whether the rule is active or not before it is displayed.

C.9 **Display product** - Failed
The problem was that the stakeholder wanted the highest level in a product set in the data grid. This made it a bit difficult since all products in the

set should be validated, when a product set is chosen. This was solved by adding the whole product set to the grid view where we set the underlying products not visible to the user.

C.10 Validate compare rule - Failed

The initial 'greater then' and 'less then' symbol did confuse us a bit. Did we mean 'greater then or equal to' or only 'greater than'? This failed when we have two equal values. The solution was to bring inn new symbols, now we have: '<', '=<', '>'. '=>', '<>', '='. Now there is no doubt what these symbols means.

C.11 Validate format rule - Success

The validation results was correct each time we tested.

C.12 Validation of products - Success

When validating, only selected products was validated each time we tested.

C.13 Log - Success

The log displayed the validation results each time we tested.

7 Possible hazards

In this chapter we will present all the elements that can go wrong and which consequences this have when using the Rule Engine.

A problem could be that the system stores all the rules in one XML file. The system depends on this file and if it is removed in any way you have to create another file from scratch. To avoid this problem we made sure that the system takes a backup of the XML file. The backup file is updated each time a rule is build or changed. In this way we have a correct backup all the time and if the original XML file should be removed, the system just copy the backup file and you got a new original file without loosing any rules.

Another problem that has been discovered is that you can make a rule that will make no meaning at all. There are actually two ways of doing that. One way is to select wrong database fields when creating rules. You select what ever you want even if the database fields have nothing to do with each other at all. This is a problem when a user do not know the EPD-base (2.1). The consequence is that you get some strange result in the Log that make no sense to the users.

The other way to make a rule with no meaning is when you change the rules. In all the text boxes in the Administration page you can write what you want. It is no control that check that e.g. the database fields is correct. By saving field names that not exist in the database you have a rule that will be rejected all the time. Also here the consequences is that you get some strange results in the Log.

The Rule Engine is a web page and one or more users can be logged on at the same time. This will not be a problem since we use a common XML file and database. The only problem that could appear is that you create the same rule. This is not a big problem since you can delete one of the rules in the Administration page.

Since this edition of the Rule Engine does not involve writing data to the EPD-base, there is nothing of value that could be lost. The only thing we are writing to database is the Log, and this data is never removed from the database.

Other things than what we have mentioned here is no concern when using the Rule Engine.

8 Survey

In this chapter we present the results of the survey. We describe our interpretation of the result and how we will use the results.

8.1 Survey method

When we started to plan the survey we made some choices about important factors like selection of people, when the survey should be complete and what kind of survey that should be done.

We will use the survey to get feedback of possible issues about the system. Also we like to get feedback of the system's user-friendliness and robustness. The results from the survey will be used to improve the system, and suggestions for further work.

We selected students and software developers to their survey since we believed that we would get most correct feedback by using people that have knowledge about software developing and systems. For each participant we gave an introduction to the Tradesolution and the EPD-base (2.1). It is easier to give this introduction to people who know the basics about developing software and databases than if they do not. The selection consist of nine students and two software developers.

To get feedback on the whole system the authors waited until the system was finished before they completed the survey, see Chapter 3.2. The questionnaire can be found in Appendix D.1.

Before the survey was complete the authors made some assumptions. The participants should have knowledge about the EPD-base, pack levels and pack types of products, how the Rule Engine works and what kind of information that is registered about products. To meet this assumptions, the authors used 30 minutes before the survey to give the participants a short introduction to Tradesolution and the EPD-base. After the introduction the participants got 60 minutes to test the system and answer the question in the questionnaire. The authors was accessible the whole time in case the participants had any further questions.

8.2 Results

The complete collection of results of the survey can be found in Appendix D.2. To achieve the quality requirements of the system, the following must be fulfilled:

Question 1,2 and 7 belongs to user-friendliness. Our requirement is that if 7 out of 11 have average or higher rating on a question, the question is fulfilled. In order to achieve sufficient user-friendliness all the questions must be fulfilled.

The rest of the questions belongs to both user-friendliness and robustness, this depends on the participant's answers. If 7 out of 11 answers are positive for a quality requirement on a question, then the requirement for this question is fulfilled. Below is a summary of the participant's answers.

The survey had, all in all, 11 persons participating. The participants asked a lot of questions, especially after the introduction. Questions like 'What it pack levels', 'What it pack types' and 'Which field from the database can I choose' were asked by 4 of 11 participants. It seems that the introduction we gave was not good enough. The person who is going to use Rule Engine has to know the answers to this questions. The person also has to know what Tradesolution (2.1) is, and have knowledge about the EPD-base (2.1). We experienced that there is need for good knowledge about the EPD-base. We recommend about 6 months of experience with registration of products and administrative work to the EPD-base before using the Rule Engine.

8.2.1 Questionnaire

Below the questions with a summary of the answers is presented. See the Appendix D.1 for a complete overview of the answers.

Question 1 - *Was it easy to understand the wizard for "compare rules"?* The participant's answers are shown in Figure 40. Most of the participants thought that the wizard for compare rules was easy or very easy to understand. We can see that one person had problems with the wizard but the rest thought it was good. The interpretation of this is that the wizard is a good way to build a compare rule.

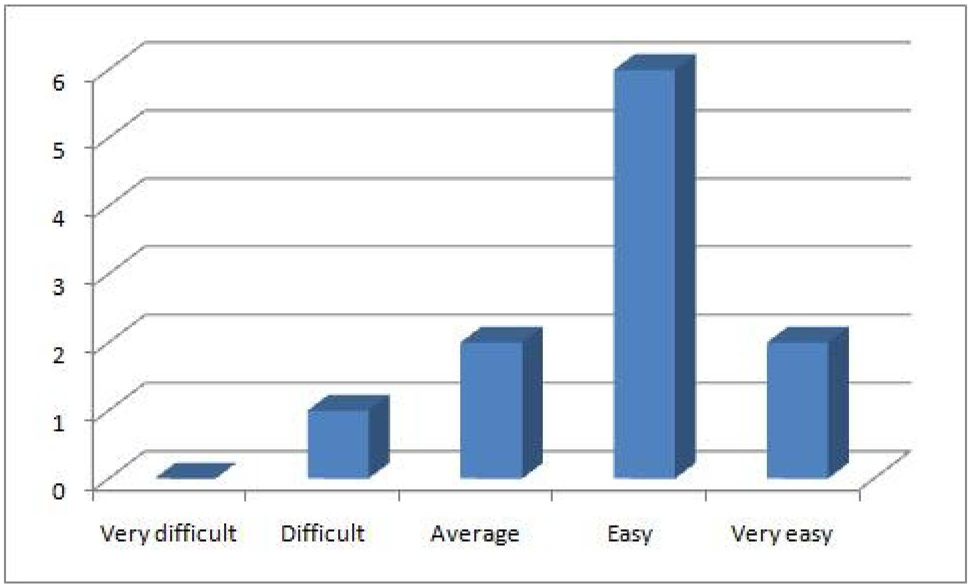


Figure 40: Result question 1

Question 2 - *Was it easy to understand the wizard for format rules?*

Also here most of the participants liked the wizard steps. Only one person did not like it at all.

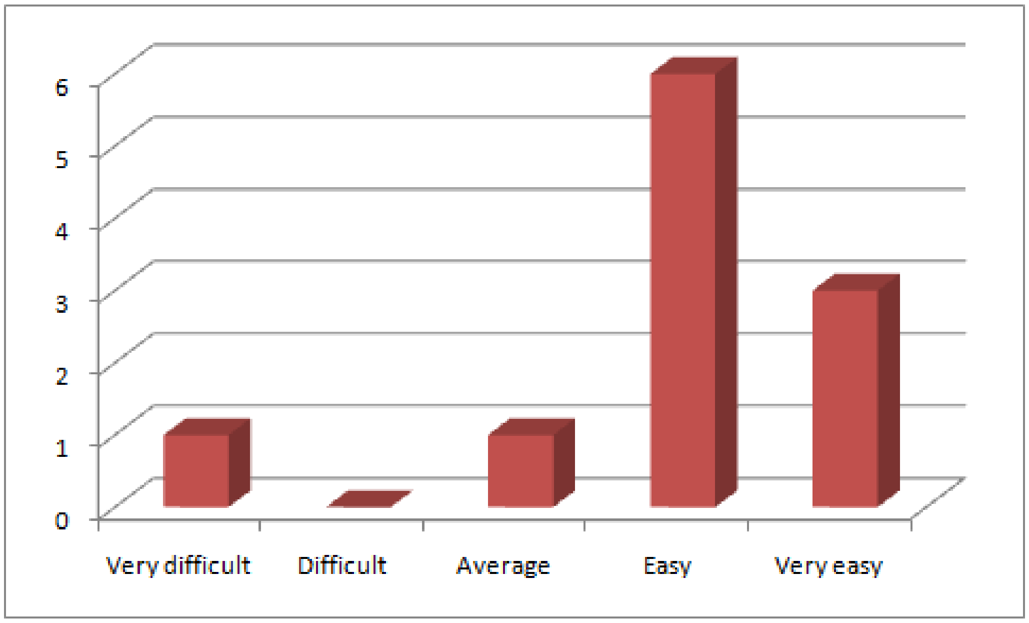


Figure 41: Result question 2

Question 3 - *Is there something missing from the wizards?*
The answers show that the subjects wanted more descriptions, explanations and help options in the wizards. There seems to be a gap between this question and question 1 and 2. In question 1 and 2 subjects seems to like the wizard. We think this is because the answers to the oral questions we received, at the beginning of the survey, cleared up some of the unclear elements. If the subjects shall use the system alone they need more guidance.

Question 4 - *Did you manage to edit a rule in the administration window?*

The main impression here is that editing a rule can be a unsafe process, 5 of 11 subjects pointed this out. There is no control that the rule actually works after editing a rule. Some pointed out that it is difficult to get an overview of expressions when they are long. Apart from this problem, 6 of 11 thinks that editing a rule went well.

Question 5 - *Did you manage to copy a rule in the administration window of your own?*
All of the 11 subjects managed to copy one or more rules, and this was easy-to-understand and worked as expected.

Question 6 - *Did you manage to delete a rule in the administration window of your own?*
Everybody managed to delete a rule. One of the subjects pointed out that copy has its own button, but for delete there is a button 'Perform editing'. Functionality of copying and deleting should be done in the same way.

Question 7 - *Do you think that the validation window gives you a good overview of how the validation is done?*
The majority of the subjects thinks the validation window is easy-to-understand or very easy-to-understand 10 out of 11. 2 subjects felt that it has average understandable and one person thinks it is below average, see Figure 42. The interpretation is that we achieve our goal of user-friendliness.

Question 8 - *Did you manage to run a rule on selected products?*

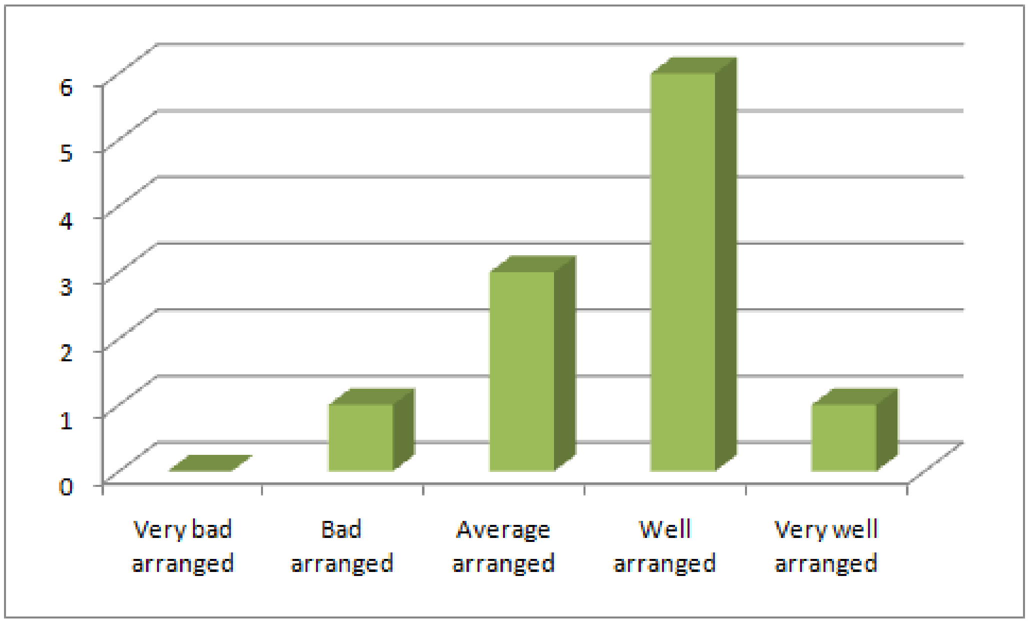


Figure 42: Result question 7

Some subjects had start up problems, and some had difficulties understanding pack levels and pack types. One subject pointed out we have different descriptions on pack levels for rules and products. When displaying products we use the symbols 'TU', 'DU' and 'CU'. For rules we describe the pack levels with words. We will fix this by changing the symbols with words to describe pack levels.

The problem with editing rules described from question 4 with no control when editing a rule, strikes again when validating a rule that contains error.

Question 9 - *Was the result of validation as expected?*

There seem to be no complaints from the subjects on this question, since the log sums up the results from the validation.

Question 10 - *Was the log generated from the validation simple and easy to understand?*
The subject was pleased with the way the log was build, with details from each validation. 2 subjects pointed out that this log might get big when we for example validate 1000 products at once. We might therefore split up the log in page tabs when a large amount of products is validated at once.

8.2.2 Survey results

Results from the survey gave us important information about the system's strengths and weaknesses. This was used to make the system better and also as suggestions for further work. The elements to improve in our system was:

- Effective use of the system is dependent on good user knowledge. We may therefore need better help functions.
- After editing rules there should be a control or validation to confirm that the rule is correct and that it makes logical sense.
- Editing and copying should be done the same way, not with two different buttons.
- The pack level should be presented in the same way in both grid views when displaying rules and products. Now there is two different ways to display pack level.

- When there is a lot of products validated, there should be a page counter to give the user a better overview of validations.

Below is a short summarize of success or failure of the two quality requirements. For more information see Appendix D.2.

Quality requirements achieved for user-friendliness:

Question 1) 10 out of 11 - success.

Question 2) 10 out of 11 - success.

Question 3) 7 out of 11 - success.

Question 4) 10 out of 11 - success.

Question 5) 11 out of 11 - success.

Question 6) 10 out of 11 - success.

Question 7) 10 out of 11 - success.

Question 8) 9 out of 11 - success.

Question 9) 11 out of 11 - success.

Question 10) 8 out of 11 - success.

Quality requirements achieved for robustness:

Question 3) 11 out of 11 - success.

Question 4) 6 out of 11 - failed.

Question 5) 11 out of 11 - success.

Question 6) 11 out of 11 - success.

Question 8) 8 out of 11 - success.

Question 9) 10 out of 11 - success.

Question 10) 11 out of 11 - success.

Robustness failed in one area, and this requirement is not met. The user-friendliness requirement is met.

9 Evaluation

In this chapter we will give an evaluation of this work and what we have achieved.

9.1 The software system

The quality focus for this project has been user-friendliness and robustness. Below we sum up what is good and what is not so good with this system.

We have developed a rule engine where you can build and edit rules and validate product data. The Rule Engine function correctly and in the way it was planned. The system is user-friendly because you can use the wizard step to build rules. When the user get more experience, he or she can use the copy/edit function to do this faster. To achieve user-friendliness we also used AJAX, which gives a better user experience.

Robustness is achieved through different functions. We created a backup function which saves the rule.xml file each time a change has been done in the file. If the system discover that the file is missing, the backup is restored and no data is missing. Another function is that we require inputs in all important steps in the wizard. This way the structure and the data of the XML file will always be correct. In addition all the tables and fields a user can select comes from the database. To achieve the robustness requirement in the Administration page, we added a function that controls all the important fields when you edit values to a rule.

User-friendliness is a difficult goal to measure. We used a survey to check if the users found the system to be user-friendly or not. There are opinions of this, but most of the participant in the survey meant that this goal was achieved, see Section 8.

For further development we suggest that the use of prolog is introduced to avoid the problem that could appear when users build rules where the fields do not have any connections. Also another server should be integrated to the system for transmitting data between the external and the internal database.

9.2 Survey

A survey was completed for this project (8) to get feedback from potential users. When the Rule Engine prototype was developed, a visiting interview was completed [1]. In this project we decided that a questionnaire (3.2) was better than a visiting interview, since we wanted to include many more persons this time.

When the questions for the questionnaire were made, we tested the questions on students and software developers. We did this because we wanted to check if our test persons interpreted the questions in the same way we did, and that everybody had the same understanding of the questions.

The questionnaire has given us important feedback and we used the information to improve the Rule Engine, see Appendix D.2.

10 Our contribution

This chapter describes our own contribution to the project. The chapter consists of a description of the Rule Engine plus how the system is tested with white-box testing and a survey in form of a questionnaire.

10.1 The Rule Engine

The development of the Rule Engine has been the main result of our project. The ideas and development started with the basis work of the Rule Engine prototype in the autumn 2006 [1]. In our master thesis we have used the prototype as a basis and developed it further. We decided to give the Rule Engine a completely new look to make it more user-friendly. The idea of XML based rules was used and developed further from the prototype. We have increased the number of types of rules a user can declare. The idea to produce a log after validation was also realized.

The EPD-base used by the Rule Engine was not developed by us. This is an existing database for product information, see Chapter 2.1. We developed some new database tables in three different areas of the system, see Figure 8.

10.2 Survey and tests of the Rule Engine

After commissioning the Rule Engine, we used white box test to test the software system. This was done after completion of each of the system parts.

The test results can be found in Chapter 6.

Further testing was done through client participation and a survey. We used questionnaire as survey method. 11 people participated on our survey and the results gave us important answers of the system quality and suggestions of improvements. See the results in Chapter 8.
We improved the following areas of the system:

- Copying rule button was removed and the function is moved to the "Execute changes" button, the same as the delete function works.
- We added a control function which controls all important fields when a user edit a rule.

11 Conclusion and further work

11.1 Conclusion

In this project we have developed a Rule Engine, a software system for validation of products. Users can build their own rules in the system without having any programming skills. After the system was developed, we carried out a survey. According to the participants of the survey the system was a partial success. User-friendliness was good but the robustness can not be classified as a complete success. The results was used to improve the system. The finished system works as expected and has achieved sufficient user-friendliness and robustness.

11.2 Further work

The following is a list with aspects that has come to our knowledge, through the development and the survey, that should be included in the next version of the Rule Engine:

- The system is dependent on good user knowledge. We therefore need better help functions.
- The pack level should be presented in the same way both when displaying rules and products. Now there is two different ways to do this.
- When there is a lot of products validated, there should be a page tabs in the Log to give the user a better overview of the validations.

- Prolog should be introduced to avoid the problem that could appear when users build rules where the fields do not have any connections.
- Integration with a new server to transmit data between the external and the internal database.

References

[1] Andreas Smoglie Leite and Øystein Eriksen. TDT 4735 Software Engineering Depth Study - Rule Engine. 2006.

[2] Tradesolution. *http://www.tradesolution.no/*.

[3] Zak Ruvalcaba. *Build Your Own ASP.NET Website using C# & VB .NET*. SitePoint, 2004. ISBN:0957921861.

[4] The Official Microsoft ASP.NET AJAX Site. *http://ajax.asp.net*.

[5] Kristen Ringdal. *Enhet og mangfold*. Fagbokforlaget, 2001. ISBN:9788276745696.

[6] ISO 9126 - International Standard For The Evaluation Of Software.

[7] Glenford J. Myers. *The Art of Software Testing, Second Edition*. John Wiley & Sons, 2004. ISBN:0471469122.

[8] Mike O'Docherty. *Object-Oriented Analysis and Design: Understanding System Development with UML 2.0*. Professional Engineering Publications, 2005. ISBN:0470092408.

A Classical phases

The software development consist of many phases, according to [8]. The phases are requirements, analysis, design, specification, implementation, testing, deployment and maintaince.

Requirements is about discovering what we are going to achieve with the new software and has two aspects, business modeling and system requirements modeling.Business modeling is about understanding the context in which our software will operate and system requirements modeling means deciding what capabilities the new software will have and writing down those capabilities.

Analysis means understanding what we are dealing with. It is important to know about relevant entities, their properties and their inter-relationships before we can design the solution.

Design phase is where we work out how to solve the problem. The system is broken down to logical subsystems and physical subsystems. In this phase you also decides how machines will communicate, chooses the right technologies for the job and other things that is important for the system.

Specification is a clear, unambiguous description of the way the components of our software should be used and how they will behave if used properly. The sort of statement we make during the specification phase is ŚIf the shop assistant object is logged on, it can ask the store object for today's special offers; in return, it receives a list of products, sorted in alphabetical orderŠ.

Implementation phase is where the code is made.

Testing phase is when the hole system is developed and it is ready to be tested. We test against the system requirements.

Deployment is the phase where we install the new software with the end users and give them the training they need so they can use the system.

Maintaince is after the deployment is done and the end users start to use the system. There will be a lot of bugs and things to take care of.

B Software development methodologies advantages

According to [8] there are many advantages by using software development methodologies:

- A methodology can help to impose discipline on the coding effort.
- Going through even the basic steps of a methodology increases our understanding of the problem, improving the quality of our solution.
- Writing lines of code is only one of the many activities in software development: performing some of the other activities helps us to spot conceptual and practical mistakes before we commit them to source code.
- At every stage, a methodology specifies what we should do next, so we're not left scratching our heads, thinking ŚOkay, what now?Š
- A methodology helps us to produce code that is more extensible (easier to change), more reusable (applicable to other problems) and easier to debug (because it has more documentation).
- Improved chances of delivery on time and within budget.
- Better communication between users, sales people, managers and developers: A good methodology is based on logic and common sense, so it will be easy for all participants to grasp the basics; thus, we have a more orderly development, with less scope for misunderstanding and wasted effort.

  A good methodology will address at least the following issues:

- Planning: Deciding what needs to be done.
- Scheduling: Mapping out when things will be done.
- Resourcing: Estimating and acquiring the human, software, hardware and other resources that are needed.

- Work flows: The subprocesses within the wider development effort (for example, designing the system architecture, modeling the problem domain and planning the development effort).
- Activities: Individual tasks within a work flow, such as testing a component, drawing a class diagram or detailing a use case, too small or indefinable to be a work flow in their own right.
- Roles: The parts played by personnel within the methodology (developer, tester or sales person).
- Artifacts: The products of the development effort: pieces of software, design documents, training plans and manuals.
- Education: Deciding how to train personnel, if necessary, to fulfill their required roles; deciding how end users (staff, customers, sales people) will learn how to use the new system.

C Test cases

C.1 Wizard steps

Wizard steps	*Results*
Executor	Øystein - format wizard, and Andreas - compare wizard
Date	13.04.2007
Stimuli	Users should be able to navigate the wizard and see earlier steps
Expected response	None of the data specified in the steps should be lost
Observed response	When the 'Back' button is pressed in the wizard, some of the chosen data in no longer there
Evaluation	Failed

Table 2: Test of wizards

C.2 Get data from database

Get data from database	*Results*
Executor	Øystein and Andreas
Date	07.03.2007
Stimuli	The system should get the right data from the database
Expected response	Displaying the right data from the database when navigating the wizard
Observed response	Success in 10 of 10 executions
Evaluation	Success

Table 3: Test of database queries

C.3 Save user data to XML

Save user data to XML	*Results*
Executor	Øystein and Andreas
Date	20.04.2007
Stimuli	The system should save the specified user data in correct XML structure
Expected response	After completing the wizards, data is saved to XML
Observed response	At first this looked like a successful test, but we later discovered that for example '<' and '>' expression values had to be saved in another was because of the formatting in XML.
Evaluation	Failed

Table 4: Test of user data saved to XML

C.4 Edit rule

Editing rules	*Results*
Executor	Øystein
Date	12.04.2007
Stimuli	The user should be able to modify and update rules
Expected response	When editing a rule the changes should be saved
Observed response	Success in 10 of 10 executions
Evaluation	Success

Table 5: Test of editing rules

C.5 Copy rule

Copying rules	*Results*
Executor	Øystein
Date	12.04.2007
Stimuli	The user should be able to copy a rule
Expected response	When copying a rule, the exact same rule should be copied, only with new rule id
Observed response	Success in 10 of 10 executions
Evaluation	Success

Table 6: Test of copying rules

C.6 Delete rule

Copying rules	*Results*
Executor	Øystein
Date	12.04.2007
Stimuli	The user should be able to delete a rule
Expected response	When deleting a rule, it should be entirely removed from the XML file
Observed response	Success in 10 of 10 executions
Evaluation	Success

Table 7: Test of deletion of rules

C.7 Server settings

Copying rules	*Results*
Executor	Øystein
Date	02.05.2007
Stimuli	The user should be able to define server settings
Expected response	When changing server setting, the system should connect to the new defined server
Observed response	Success in 10 of 10 executions
Evaluation	Success

Table 8: Test of server settings

C.8 Display rule

Display rules	*Results*
Executor	Andreas
Date	04.05.2007
Stimuli	The user should be able to display rules on the validation page
Expected response	When selecting 'All', 'Format' or 'Compare' rules, they should be displayed
Observed response	The rules are all displayed, but the test fails because of rules that are not active is also displayed
Evaluation	Failed

Table 9: Test of displaying rules

C.9 Display product

Display products	*Results*
Executor	Andreas
Date	04.05.2007
Stimuli	The user should be able to display products on the validation page
Expected response	Products should be displayed after different filtrations
Observed response	The products are displayed, but some problems with product sets
Evaluation	Failed

Table 10: Test of displaying products

C.10 Compare rule validation

Validate compare rules	*Results*
Executor	Andreas
Date	11.05.2007
Stimuli	The system should be able to validate products from compare rules
Expected response	Validation results should be after the logic of rules
Observed response	Seems ok, but there is an issue when there are for example greater than '>' and the two values specified in the rules are equal, the validation fails
Evaluation	Failed

Table 11: Test of compare rule validation

C.11 Format rule validation

Validate compare rules	*Results*
Executor	Andreas and Øystein
Date	11.05.2007
Stimuli	The system should be able to validate products from format rules
Expected response	Validation results should be after the rules logic
Observed response	Success in 10 of 10 Executions
Evaluation	Success

Table 12: Test of format rule validation

C.12 Validation of products

Validate compare rules	*Results*
Executor	Andreas and Øystein
Date	18.05.2007
Stimuli	The system should validate the correct (selected) products
Expected response	Validation is only done to the selected products
Observed response	Success in 10 of 10 Executions
Evaluation	Success

Table 13: Test of validation of products

C.13 Log

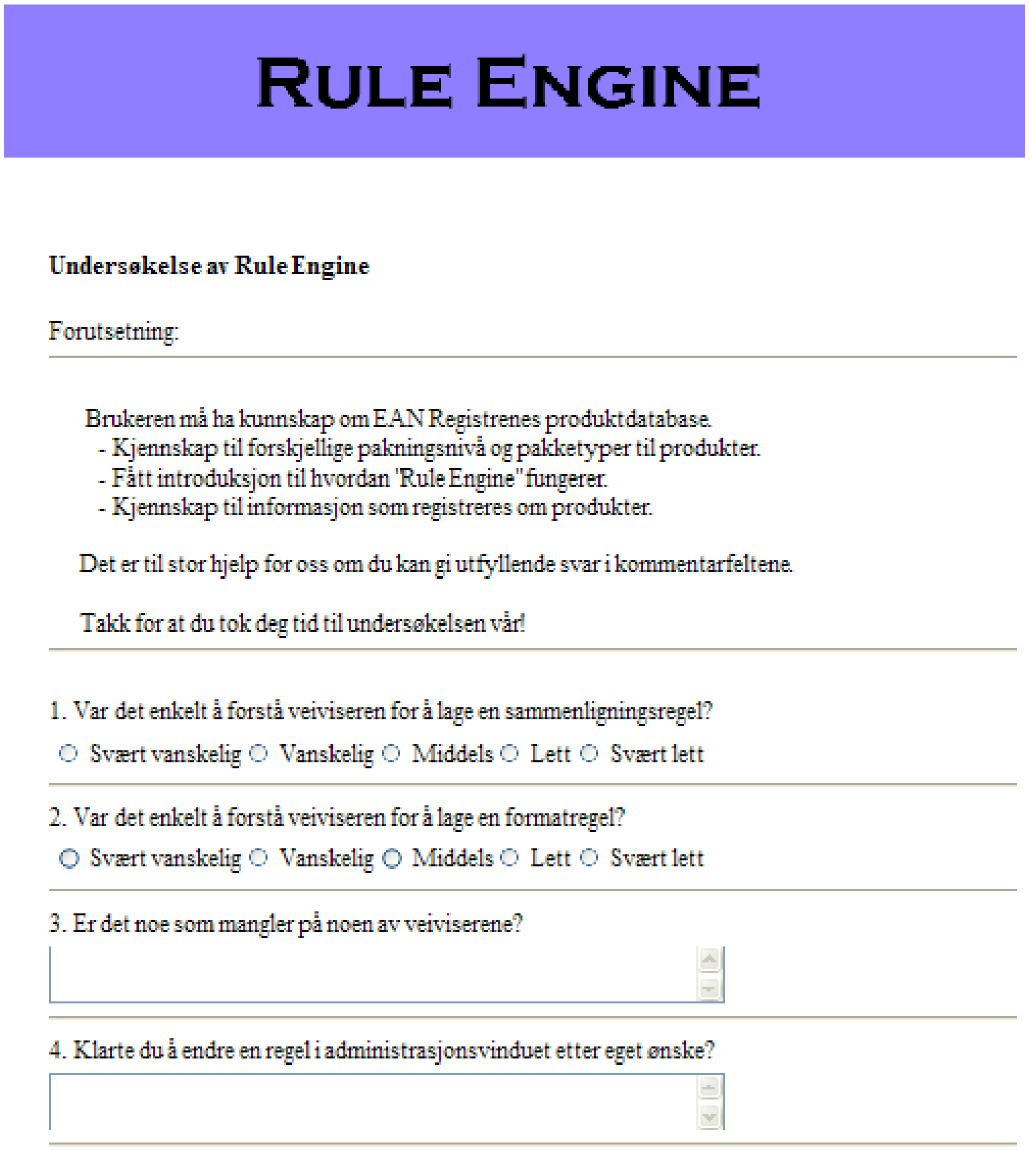
Test of log results	*Results*
Executor	Øystein
Date	18.05.2007
Stimuli	The system should display the right log from the validation
Expected response	The log shows the user the validation results
Observed response	Success in 10 of 10 Executions
Evaluation	Success

Table 14: Test of validation log

D Survey

D.1 Questionnaire

In Figure 43 and 44 you can see the questionnaire which was used for the survey of the Rule Engine.



RULE ENGINE

Undersøkelse av Rule Engine

Forutsetning:

Brukeren må ha kunnskap om EAN Registrenes produktdatabase.
- Kjennskap til forskjellige pakningsnivå og pakketyper til produkter.
- Fått introduksjon til hvordan "Rule Engine" fungerer.
- Kjennskap til informasjon som registreres om produkter.

Det er til stor hjelp for oss om du kan gi utfyllende svar i kommentarfeltene.

Takk for at du tok deg tid til undersøkelsen vår!

1. Var det enkelt å forstå veiviseren for å lage en sammenligningsregel?

○ Svært vanskelig ○ Vanskelig ○ Middels ○ Lett ○ Svært lett

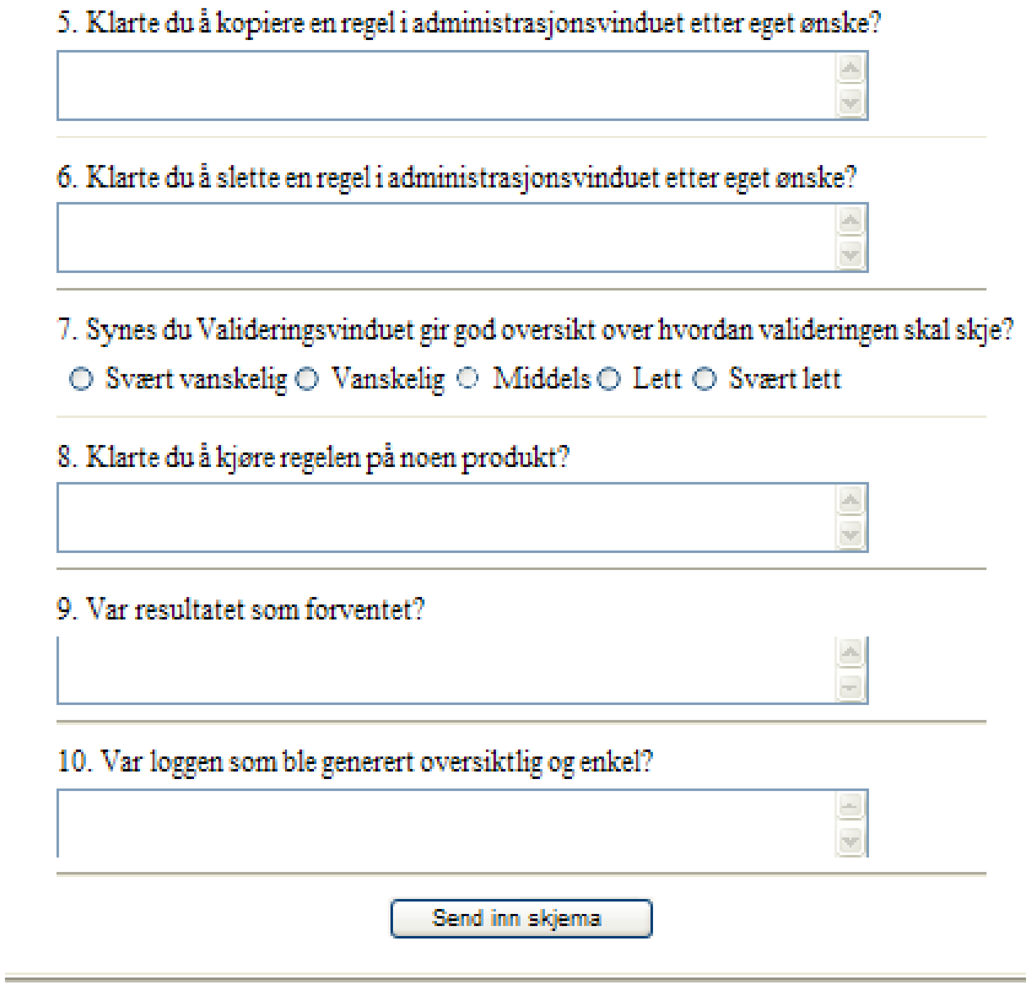
2. Var det enkelt å forstå veiviseren for å lage en formatregel?

○ Svært vanskelig ○ Vanskelig ○ Middels ○ Lett ○ Svært lett

3. Er det noe som mangler på noen av veiviserene?

4. Klarte du å endre en regel i administrasjonsvinduet etter eget ønske?

Figure 43: Questionnaire: question 1 - 4


5. Klarte du å kopiere en regel i administrasjonsvinduet etter eget ønske?

6. Klarte du å slette en regel i administrasjonsvinduet etter eget ønske?

7. Synes du Valideringsvinduet gir god oversikt over hvordan valideringen skal skje?

○ Svært vanskelig ○ Vanskelig ○ Middels ○ Lett ○ Svært lett

8. Klarte du å kjøre regelen på noen produkt?

9. Var resultatet som forventet?

10. Var loggen som ble generert oversiktlig og enkel?

Figure 44: Questionnaire: question 5 - 10

D.2 Answers

Here is the answers from the participants of the survey:

- Question 1
  i) Vanskelig
  ii) Middels
  iii) Lett
  iv) Lett
  v) Svært lett
  vi) Lett
  vii) Svært lett
  viii) Middels
  ix) Lett
  x) Lett
  xi) Lett

- Question 2
  i) Lett
  ii) Lett
  iii) Svært vanskelig
  iv) Svært lett
  v) Svært lett
  vi) Lett
  vii) Svært lett
  viii) Middels
  ix) Lett
  x) Lett
  xi) Lett

- Question 3
  i) Kan være en bedre beskrivelse av hvordan man skal lage reglene. Vanskelig å fortså når man ikke kan databasen godt nok fra før.
  ii) Nei, dersom jeg skal sette fingeren på noe må det være ekstra hjelpevindu med eksempler
  iii) Den for sammenligningsregel var lett å forstå men forsto ikke helt hva som var hensikten med formatreglene. Kunne vært litt mer

forklaring her.
iv) Kanskje en litt bedre forklaring til hva som menes med toleranse, selv om jeg fikk dette forklart muntlig.
v) Nei, de var enkle og grei. Gode forklaringer underveis.
vi) Wizarden gir god oversikt og god forklaring på hva som skal gjøres. Litt vanskelig til å begynne med men når man først forstår hvordan databasen er bygd opp gikk det bra.
vii) Veiviserene var enkle og grei. Kan bli kjedelig i lengden å lage alle regler ved å bruke disse stegene.
viii) De var helt ok. Ganske vanskelig til å begynne med men ble bedre etterhvert. Kan fortsatt ta med litt mer forklaring når man skal velg felt i databasen.
ix) Ikke i utgangspunktet, fint at regelen vises til høyre.
x) Flere operasjoner, spesielt $<$ og $>$ xi) Fungerte greit

- Question 4
i) Det gikk veldig bra. Hadde ingen problemer med dette.
ii) Det gikk greit.
iii) Endringen gikk bra men er fikk ikke de reglene jeg endret på til å fungere. Virker som systemet er litt følsomt for hva men skriver når man skal fylle inn databasetabell og kolonner.
iv) Ja, med det ble noe uoversiktlig når jeg hadde flere felter i det ene uttrykket mitt
v) Endring gikk fint.
vi) Klarte å endre, men kan være fort å gjøre feil her. Burde vært en form for sikkerhet så man må skrive noe som er korrekt i hvert fall.
vii) Klarte å endre, men feltene jeg skrev inn var feil. Her bør det være en kontroll.
viii) Dette klarte jeg, men vet ikke om det jeg skrev var riktig. Kunne godt vært en kontroll her.
ix) Ja, dette gikk greit. Men her kan jeg også lett ødelegge regelen om jeg taster inn feil
x) Ja
xi) Ja

- Question 5
i) Kopieringen gikk fint. Fikk også endret reglene til det jeg ville etter

at jeg hadde kopiert.
ii) Ja, dette var oversiktlig og greit
iii) Kopiering gikk fint.
iv) Ja, uten problemer
v) Kopiering var greit.
vi) Kopiering var greit.
vii) Ja
viii) Det gikk bra.
ix) Ja, kopierte regelen jeg hadde laget
x) Ja
xi) Ja

- Question 6
  i) Ja, det gikk fint.
  ii) Ja
  iii) Sletting gikk også bra.
  iv) Ja, uten problemer
  v) Sletting gikk fint. Men hvorfor er det slik at for sletting må man trykke utfør endringer og fro kopier er det en egen knapp. Dette kan vel gjøres likt.
  vi) Sletting var greit.
  vii) Ja.
  viii) Sletting gikk bra!
  ix) Ja.
  x) Ja.
  xi) Ja.

- Question 7
  i) Lite oversiktlig
  ii) Oversiktlig
  iii) Oversiktlig
  iv) Middels oversiktlig
  v) Oversiktlig
  vi) Middels oversiktlig
  vii) Oversiktlig
  viii) Oversiktlig

ix) Svært oversiktlig
x) Middels oversiktlig
xi) Oversiktlig

- Question 8
  i) Først rotet jeg litt med å finne frem til de reglene jeg hadde laget. Men etterhvert gikk det fint. Når jeg først hadde prøvd et par ganger fungerte det topp.
  ii) Jeg klarte å kjøre validering, skjønte ikke helt pakningsnivå på produkter, med CU og TU, osv
  iii) Synes det var enkelt å finn frem i valideringsvinduet. Fikk kjørt alle de reglene jeg lagde via wizarden.
  iv) Kjørte en regel mot ett produktsett, denne validerte kun ett produkt som regelen var beregnet på (med tanke på pakningsnivå), som forventet.
  v) Klarte fint å kjøre regel. Men er litt dumt at når man velger å hente fram produkt og mens man venter på at disse skal vises så prøvde jeg å hente frem regler og da stoppet systemet å hente produkt. Kan være litt irriterende.
  vi) Fikk kjørt de reglene som jeg laget i wizarden men ikke de jeg endret på. Kan være fordi jeg laget de feil. Men bør være noe som sier at reglene er feil. Ellers virket det greit.
  vii) Reglene jeg testa gikk ok.
  viii) Fikk ikke helt kontroll på hvilke regler jeg hadde laget. Var ganske mange regler der. Men klarte å kjør de jeg laget i veiviseren.
  ix) Kjørte regelen jeg laget mot et par produkter, det gikk greit
  x) Ja, men hadde trøbbel med format. Forsøkte å teste lengde på produktnavn, men den slo bestandig feil.
  xi) Ja.

- Question 9
  i) Resultatet var som forventet. Lagde regler som var godkjente og ikke godkjente og Rule Engine oppfattet det korrekt.
  ii) Ja, sånn omtrent
  iii) Resultatet var helt i tråd med det jeg forventet. Brukte endel tid på å finne hvilke verdier som ble brukt, men systemet gjorde det riktig.

iv) Ja, fikk opp logg som ga meg informasjon om resultatet
v) Resultatene var i tråd med det jeg trodde.
vi) Resultatene jeg fikk var slik jeg forventet.
vii) det ble riktige resultater.
viii) Så ut som det var riktig resultat. Forutsatt at den henter riktig verdi fra databasen da.
ix) Ja.
x) Nei, ikke på lengde av produktnavn.
xi) Ja.

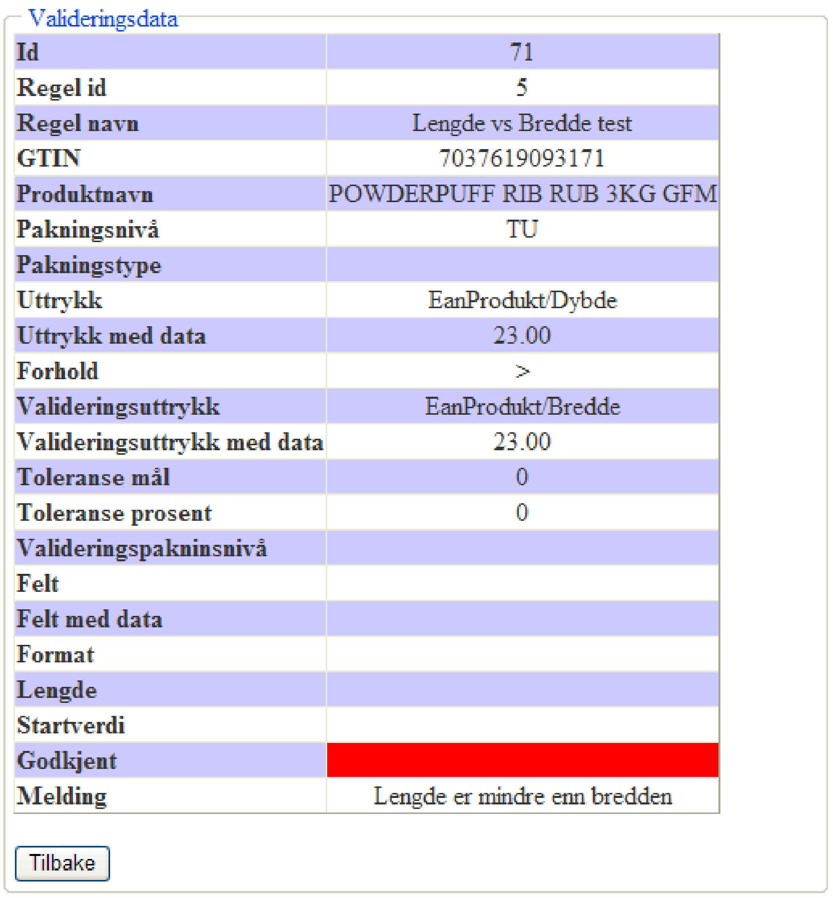
- Question 10
  i) Jeg synes loggen var veldig bra. Gir god oversikt og enkelt å bla seg frem og tilbake.
  ii) Denne var oversiktlig og grei.
  iii) Loggen var grei. Vil tro at det er enklere å gjøre om litt dersom man har f.eks 1000 rader i griden. Da kan det være lurt å endre litt på hvordan det vises.
  iv) Loggen var grei
  v) loggen var bra. Kan bli litt mange linjer nedover. Kan kaknskje deles opp i sider i stedet.
  vi) Loggen var vaeldig oversiktlig, bra at man kan se hvor mange som er feil i et sett. Kult med grønne og røde farge, sier i fra greit.
  vii) Loggen var ok. Ble veldig lang liste ettersom jeg kominerte mange regler og mange produkt.
  viii) Loggen var grei. kult at man kan se fler og fler detaljer. Bra at det starter mer oversiktlig.
  ix) Loggen var grei, her får jeg informasjon om hva som har skjedd i valideringen.
  x) Ja.
  xi) Ja, ok for de testene jeg kjørte.

E Detail view in Log

A explanation of the details view in the log is given in Table 15.

Field	*Explanation*
Id	The id if the record in the database
Regel Id	Rule id
Regel navn	Rule name
GTIN	"EanProduktNr", unique number of the product
Produktnavn	Product name
Pakningsnivå	Pack level from the rule
Pakningstype	Pack type from the rule
Uttrykk	The expression from the rule
Uttrykk med data	The expression with data from the database
Forhold	Relations
Valideringsuttrykk	The validate expression from the rule
Valideringsuttrykk med data	The validate expression with data from the database
Toleranse mål	Tolerance measure from the rule
Toleranse prosent	Tolerance percent from the rule
Valideringspakninsnivå	The validate pack level from the rule
Felt	The database field which being controlled
Felt med data	The value of the database field
Format	The format from the rule
Lengde	The length from the rule
Startverdi	The start value from the rule
Godkjent	Green if approved, red if rejected
Melding	The error message

Table 15: Explanation of Log detail, see Figure 45.



Valideringsdata

Id	71
Regel id	5
Regel navn	Lengde vs Bredde test
GTIN	7037619093171
Produktnavn	POWDERPUFF RIB RUB 3KG GFM
Pakningsnivå	TU
Pakningstype	
Uttrykk	EanProdukt/Dybde
Uttrykk med data	23.00
Forhold	>
Valideringsuttrykk	EanProdukt/Bredde
Valideringsuttrykk med data	23.00
Toleranse mål	0
Toleranse prosent	0
Valideringspakninsnivå	
Felt	
Felt med data	
Format	
Lengde	
Startverdi	
Godkjent	🟥
Melding	Lengde er mindre enn bredden

Figure 45: Details of what has been done in the validation.