Informatikprojekt

Project of:	Simon Günter I.Nr. 96-102-777 Email:sguenter@iamexwi.unibe.ch
Projectdate:	February to May 1999
Projectname:	Trademark Application
Contactperson of the University:	Prof. Dr.Oscar Nierstrasz Email:Oscar.Nierstrasz@iam.unibe.ch
Contactperson of the Swiss Federal Institute of Intellectual Property:	Matthias Günter Email:Matthias.Guenter@ipi.ch

Table of contents

1 Requirements

1.1 Goal of the Project

Development of a tool with the following functionality:

1. Allows the user to enter a complete trademark application(national and international part) for the Swiss Federal Institute of Intellectual Property in a comfortable way. The user must have the possibility to input all the information supported by the SGML-file format for the application. The project has not the goal to create a perfect tool with input controlling functions, but just to provide the possibility to put all the necessary information in and to make only rudimentary checks of data consistence.
2. Allows the user to store the inputted data in a SGML-file conform to the format of the Swiss Federal Institute of Intellectual Property.
3. Allows the user to load a SGML-file, which must be conform to the format of Swiss Federal Institute of Intellectual Property, and provides the user the possibility to modify the data. Of course the meaning of the data should not change in this process.
4. Allows the user to store the data which was inputted by the user or loaded from a SGML-file as a HTML-file. The printed HTML-file should be a trademark application in the ISAF (International Standard Application Format).

1.2 Purpose of the Project

The Swiss Federal Institute of Intellectual Property has already an application (EP-Easy) for creation of the national part of the trademark application(as file or as paper). An upgrade for the same program which also includes the international part is being build, but it's unknown when this application will be available, so this project is a backup solution, if the deployment of the other application will be delayed to long.

1.3 The usage of the application

The application will be used mainly in the Institute for database testing purpose, but in a second step (not part of the project) the application could be upgraded (if necessary) and deployed to external user.

2 Design Decisions

2.1 Possible systems for Implementation of the Application

1. HTML files on a server by the Institute. Using cookies to store data on the local drives of the user.
2. HTML files stored on the local drives of the user. Using cookies to store data on the local drives of the user.
3. HTML files and CGI scripts on a server of the Institute.
4. HTML files and CGI scripts stored on the local drives of the user.
5. Standalone application installed at the user side

2.2 Detailed description and advantages/disadvantages of the implementation types

1) To make a trademark application the user contacts the server of the IPI(by using a normal webbrowser) and start a special page. With normal HTML forms the user enters the data and this is send to the server. For storing the data on the client side the server produces a cookie which contain the data (A cookie is a piece of data stored by an User agent of the clients webbrowser, which was instructed by a origin server to do so. If the same page for which the cookie was generated is loaded, the data in the cookie is automatically transferred to the contacted server.)

   Advantages:
   A) User don't have to install anything.
   B) Easy to implement and to adapt(only HTML used and changes only by the server)

   Disadvantages:
   A) Some browser only support cookies of limited size (e.g. 4kB), some attachment like colour picture could be very large, therefor this file can not be embedded in the SGML file and must be separately sent to the Institute (e.g. as a attachment of email).
   B) The cookie is stored by an User agent, so it's difficult to retrieve the data in the cookie by another application(e.g. if the applicant want to send the SGML-file by email, this person has to know the filename of the cookie and even if he or she find the file it could been in a special format and contain additional information for the User agent).
   C) Extern SGML-file (which were not produced by the application) must be loaded by a special function and then stored as a cookie.
   D) Needs an origin server.
   E) Security Problem: Trademark applications are sensitive data. Normal transfer by TCP/IP is not safe, special precautions are perhaps needed..
   F) Quotation of Internet Draft „Applicability Statement for HTTP State Management“:
   "The purpose of the HTTP State Management is to allow an HTTP-based service to create stateful „sessions“ which persist across multiple HTTP transaction."
   Cookies are a part of State Management, precisely one task of the State Management is managing the cookies.
   One goal of the project is to create a SGML-file that „survive“ the creation „session“,

so even if it's technical possible to use cookies it would not be the intended use of the cookies.

2) Same as 1., but here the server and the client are both on the user side. This means that the user have to install files on his or her local drives.

   Advantages.
   A) No security problems..
   B) Easy to implement.

   Disadvantages:
   A) Some browser only support cookies of limited size (e.g. 4kB), some attachment like colour picture for colour trademark could be very large, therefor this file can not be embedded in the SGML file and must be separately sent to the Institute (e.g. as a attachment of email). .
   B) The cookie is stored by an User agent, so it's difficult to retrieve the data in the cookie by another application(e.g. if the applicant want to send the SGML-file by email, this person has to know the filename of the cookie and even if he or she find the file it could been in a special format and contain additional information for the User agent).
   C) Extern SGML-file (which were not produced by the application) must be loaded by a special function and then stored as a cookie.
   D) Quotation of Internet Draft „Applicability Statement for HTTP State Management":
   "The purpose of the HTTP State Management is to allow an HTTP-based service to create stateful „sessions" which persist across multiple HTTP transaction."
   Cookies are a part of State Management, precisely one task of the State Management is managing the cookies.
   One goal of the project is to create a SGML-file that „survive" the creation „session", so even if it's technical possible to use cookies it would not be the intended use of the cookies.

3) To make a trademark application the user contacts the server of the IPI(by using a normal webbrowser) and start a special page. With normal HTML forms the user enters the data and this is send to the server. The server then store the data by the CGI-scripts in a special not protected directory on the local drives of the user.

   Advantages:
   A) User don't have to install anything.
   B) Easy to adapt (changes only by the server)

   Disadvantages:
   A) Security problem(Can be solved with some expense).
   B) Needs a server.
   C) The special directory of the user must be unprotected. This fact could be used by a third party.
   D) Possibly results in a complex programming structure which would be hard to adapt to changed requirements.

4) To make a trademark application the user starts a HTML file stored on his or her local drives with a webbrowser. With normal HTML forms the user then enters the data and the data is send to CGI-scripts which are also stored on the local drives. These scripts then save the data in a SGML file

Advantages:
A) No security problems.

Disadvantages:
A) Installation at the user side needed.
B) Possibly results in a complex programming structure which would be hard to adapt to changed requirements.

5) To make a trademark application the user install the necessary software for the standalone application on his or her local drives then he or she starts the application.

Advantages:
A) No security problems.
B) The GUI can be adapted for the requirements of the projects.

Disadvantages:
A) Installation at the user side needed.
B) normally more complex and larger then scripts.

2.3 Decision

For the Institute a high adaptability, the possibility to upgrade the system to high standards and a short Development time are important, so the representative of the Swiss Federal Institute of Intellectual Property Matthias Günter agreed with me that the fifth solution seems to be the most adequate.

3 Design Tools

3.1 Possible tools to create a stand alone application

1. No tools, plain programming languages.
2. IBM Visual Age for Java
3. Delphi

3.2 Short Description and Advantages/Disadvantages of the tools

1. Use of plain programming languages like C++ , Java, Pascal.. .
   Advantages:
   A) Cheap
   B) Easy to adapt and very robust.

   Disadvantage:
   A) GUI implementation very difficult

2. This tool support the visual creation of GUI and simplify the event handling and normal code creation. It is based on Java.
   Advantages:
   A) I have experience with programming in Java, so it's relatively easy for me to create additional code which can be integrated in this tool
   B) Java has become very popular in the last years.
   C) The Institute has no experience with this tool: This project will show if this tool is useful for implementing user interfaces and the additional logic.
   D) The next versions of VA for Java are likely to be backward compatible, that means that the program(s) produced by this project can be adapted with the new versions. The same is true for Java. (New Java versions were always compatible to older versions in the past).
   E) Suited for creation of GUIs and the additional logic.

   Disadvantages:
   A) The tool is not very mature, that means that there could be some bugs and incompatibilities in the tool.
   B) The Institute has no experience with this tool: That means that an adaptation of the program(s) to new requirements will be difficult.

3. Delphi is a fourth generation programming language which has been improved several times and has been adapted to new knowledge in software and hardware. It is therefore both top of the art and very mature.
   Advantages:
   A) Very mature, unlikely to have bugs.
   B) The Institute has some experience with Delphi and the new EASY version is created with Delphi, that means adaptation are not too difficult.
   C) New Delphi versions are backward compatible.
   D) Suited for creation of GUIs and the additional logic.

   Disadvantages:

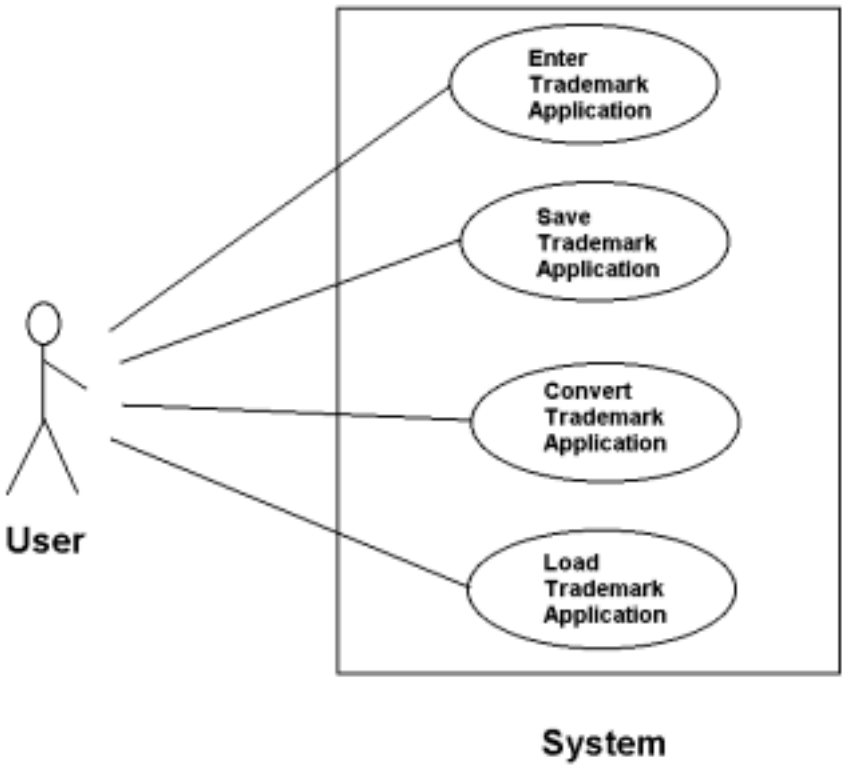
A) I have no experience with Delphi.

3.3 Decision

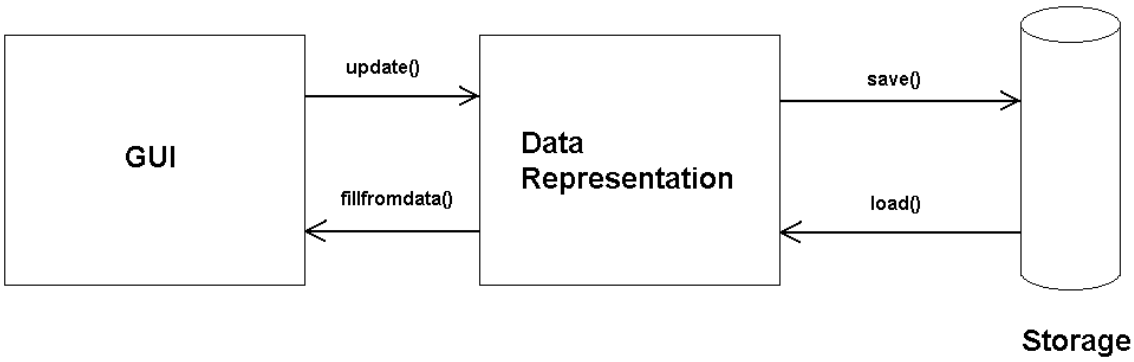
The representative of the Swiss Federal Institute of Intellectual Property Matthias Günter agreed with me that the second and the third solution are both adequate. I chose the second tool.

4 Design

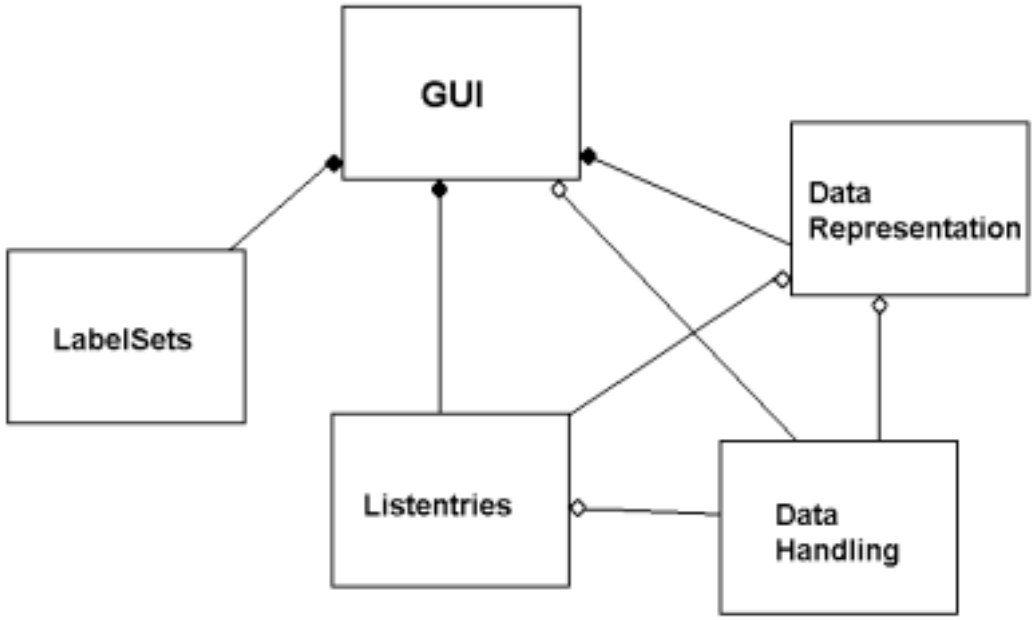
The usage of the system is visualised by the following use case diagram



The application must manage the necessary data for a Trademark Application. For this different instances of the data are managed as shown in the next image.



To achieve the functionality the system is divided in different groups of classes with different responsibilities as shown in the next graphic.



The filled diamonds stand for ‘have’ associations.
The hollow diamonds stand for ‘uses’ associations.

The responsibilities of the groups are explained in the next chapter.

5 Class Groups Responsibilities

5.1 GUI

Allows the user to enter a trademark in a way which is easy to understand.
Manage the load,printing and save of trademark applications.
Provide help functions to the user.
Maintain data integrity between DataRepresentation and itself.

5.2 Listentries

Provide all the entries for the comboboxes and lists of the GUI in the selected language.
Provide functions which translate the texts of the comboboxes and lists to the language independent code.
Contain the text for the helppages.
Represent the data for the entries in a way that can be adapted easily.

5.3 DataRepresentation

Can store all the data entered in the GUI.
Stores the data stored in itself into a String which when stored result in a SGML file of the correct format.
Read data out of String ,which was created out of a SGML file.

5.4 LabelSets

Contains all the labels for the GUI in the selected language.

5.5 DataHandling

Provide the functions to convert from different formats into other formats.
Provide useful parse functions.

6 Class Responsibilities

Classname	MainFrame
Superclass	Jframe
Subclasses	-
Class Group	GUI
Responsibilities	Allows the user to enter, save,export and load Trademark Applications . Hold all the GUI Panels.
Collaborations	Panels,Labelsets, Listentries,DataHandling, DataRepresentation, DescriptorTexts

Classsubgroupname	Panels
Classes	ApplicatorPanel,TrademarkPanel,GoodsServicesPanel, PriorityPanel, SpecialPanel,RemarksPanel,AttachementsPanel, IRApplicatorPanel, IRBasisPanel,IRGoodsServciesPanel,IRPaymentPanel,IRProtectionPanel, IRPrioLimitPanel
Superclass	Jpanel
Subclasses	-
Class Group	GUI
Responsibilities	Allows the user to enter data of a specific context. Maintain dataintegrity between itself and DataRepresentation.
Collaborations	LabelSets,Listentries,DataHandling, DataRepresentation, Base64Coder

Classname	HelpFrame
Superclass	Jframe
Subclasses	-
Class Group	GUI
Responsibilities	Displays the Helptexts
Collaborations	-

Classsubgroupname	Tables
Classes	TableApp,TableAtt,TableBasis,TableGS,TableIRApp,TableIRGS, TableIRLimit,TableLimitGsgr,TableIRP,TableIRPrio,TableIRPrio TableIRPrioGsgr,TablePrio,TableT
Superclass	DefaultTableModel
Subclasses	-
Class Group	GUI
Responsibilities	Store the data for a GUI table. Knows ist column and row size. Initialises a specific GUI table with the right column identifiers and The correct column layout(e.g. column width).
Collaborations	JTable, LabelSets

Classname	LDO
Superclass	
Subclasses	Listentries,LabelSet
Class Group	LabelSets and Listentries
Responsibilities	Knows its language.
Collaborations	-

Classname	LabelSet
Superclass	LDO
Subclasses	LabelSets
Class Group	LabelSets
Responsibilities	Knows its context.
Collaborations	-

Classsubgroupname	LabelSets
Classes	ApplicatorPanelLabelSet, TrademarkPanelLabelSet, PriorityPanelLabelSet, SpecialPanelLabelSet, RemarksPanelLabelSet, AttachementsPanelLabelSet, IRApplicatorPanelLabelSet, IRBasisPanelLabelSet, IRGoodsServciesPaneLabelSetl, IRPaymentPanelLabelSet, IRProtectionPanelLabelSet, IRPrioLimitPanelLabelSet, MainFrameLabelSet
Superclass	LabelSet
Subclasses	De/F/I/E LabelSets
Class Group	LabelSets
Responsibilities	Provide the interfaces for accessing the labelnames.
Collaborations	-

Classsubgroupname	DeLabelSets
Classes	DeApplicatorPanelLabelSet, DeTrademarkPanelLabelSet, DePriorityPanelLabelSet, DeSpecialPanelLabelSet, DeRemarksPanelLabelSet, DeAttachementsPanelLabelSet, DeIRApplicatorPanelLabelSet, DeIRBasisPanelLabelSet, DeIRGoodsServciesPaneLabelSetl, DeIRPaymentPanelLabelSet, DeIRProtectionPanelLabelSet, DeIRPrioLimitPanelLabelSet, DeMainFrameLabelSet
Superclasses	LabelSets
Subclasses	-
Class Group	LabelSets
Responsibilities	Implements the getter functions of the labelnames for German
Collaborations	-

Classname	Listentries
Superclass	LDO
Subclasses	De/F/I/E Listentries
Class Group	Listentries
Responsibilities	Provides all the codes of all options and possible entries in comboboxes, lists and some values for column of tables. Stores the codes in an easy to adapt manner.
Collaborations	HelpText, TranslationArray

Classname	DeListentries
Superclass	-
Subclasses	-
Class Group	Listentries
Responsibilities	Provides the translation of the codes defined in the superclass in German. Stores the translations in an easy to adapt manner.
Collaborations	HelpText, TranslationArray

Classname	DescriptorTexts
Superclass	-
Subclasses	De/F/I/E DescriptorTexts
Class Group	Listentries
Responsibilities	Defines all the decriptions which are used by the SGML to ISAF (in HTML) conversion. Stores the codes in an easy to adapt manner.
Collaborations	TranslationArray

Classname	DeDescriptorTexts
Superclass	DescriptorTexts
Subclasses	-
Class Group	Listentries
Responsibilities	Provide all the descriptions which are used by the SGML to ISAF (in HTML) conversion in German. Stores the codes in an easy to adapt manner.
Collaborations	HelpText, TranslationArray

Classname	HelpText
Superclass	-
Subclasses	De/F/I/E HelpText
Class Group	Listentries
Responsibilities	Provides access to the help texts.
Collaborations	Subclasses

Classname	DeHelpText
Superclass	-
Subclasses	De/F/I/E HelpText
Class Group	Listentries
Responsibilities	Provides access to the help texts in German.
Collaborations	-

Classname	TranslationArray
Superclass	-
Subclasses	-
Class Group	Listentries
Responsibilities	Manages a 2 dimensional Array with 2 columns (one is the ‚Code' and the other is the ‚Translation'). Provide several functions to get elements and set elements. Provide creation procedures which creates an Object out of a String Which is used to set all the codes. Provide function which set all the Translation out of a String.
Collaborations	-

Classname	DataHandler
Superclass	-
Subclasseses	-
Class Group	DataHandling
Responsibilities	Converts normal String to printable Strings with SGML char codes. Converts a SGML char coded String in a normal java String. Provides useful SGML parsing function. Provide useful HTML text production functions. Stores a Base64 Encoded String in a file with a random name.
Collaborations	SGMLCharCodes,Base64Coder

Classname	SGMLCharCodes
Superclass	TranslationArray
Subclasses	-
Class Group	DataHandling
Responsibilities	Stores SGML codes for special Characters. Provides access to these codes and the corresponding chars.
Collaborations	-

Classname	Base64Coder
Superclass	-
Subclasses	-
Class Group	DataHandling
Responsibilities	Converts bytes into printable String using Base64Encoding. Decodes a printable String using Base64Decoding . Decodes and saves a printable String in a File using Base64Decoding.
Collaborations	BASE64Encoder,BASE64Decoder

Classname	Conversion
Superclass	-
Subclasses	-
Class Group	DataHandling
Responsibilities	Converts a File which is in SGML File Format in a ISAF File.
Collaborations	DataHandler, DataRepresentation, Listentries

Classname	DataRepresentation
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Has the ability to store all the data of the GUI. Read data out of a String of a special SGML format. Write its data into a String in correct SGML format. Write its data into a String in HTML.
Collaborations	DataHandler,Chnapp,Chirec,Chatt.

Classname	Chnapp
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the National Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler,Nameadd,Pymg,Gsgr,Prigr

Classname	Nameadd
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the Name and Address Group of the National Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler

Classname	Pymg
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the Payment Group of the National Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler

Classname	Gsgr
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the Goods and Services Group of the National Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler

Classname	Prigr
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the Priority Group of the National Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler

Classname	Chirec
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the International Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	Nameadd_IR,Basgr, Gsgr_IR,Prigr_IR,Limgr.Pymg_IR

Classname	Nameadd_IR
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the Name and Address Group of the International Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler

Classname	Basgr
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the Basic Application and Registration Group of the International Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler

Classname	Gsgr_IR
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the Goods and Services Group of the International Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler

Classname	Prigr_IR
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the Priority Group of the International Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler,Gsgr_Prigr_IR

Classname	Gsgr_Prigr_IR
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the Priority Goods and Services Group of the International Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler

Classname	Limgr
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the Limitation Group of the International Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler,Gsgr_Limgr

Classname	Gsgr_Limgr
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the Limitation Goods and Services Group of the International Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler

Classname	Pymg_IR
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the Payment Group of the International Application. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler

Classname	Chatt
Superclass	-
Subclasses	-
Class Group	DataRepresentation
Responsibilities	Maintains the data of the Attachment Group. Reads data out of a String of a special SGML format. Writes its data into a String in correct SGML format. Writes its data into a String in HTML.
Collaborations	DataHandler

7 Descriptions of the formats

7.1 SGML File Format

Bytes which when interpreted as ASCII Characters result in a SGML Document. Only ASCII Characters from 00 to 7f are used. Special Characters are coded(see Appendix 1).Tags are used to structure the data(see Appendix 2).

Purpose of Format: Represent the data for the Trademark Application in a printable way which can be correctly interpreted by SGML parser.

7.2 SGML String Format

These Java Strings contain only Characters from 0000 to 007f. The internal java Character to Byte Converter will produce a SGML File(by converting a 00xy java character in a xy ASCII Characterbyte.) out of these Strings. Special Characters are coded the same way like SGML file characters. Tags are used to structure the data(see Appendix 2).

Purpose of Format: Allows the Internal Char to Byte Converter to produce bytes which when stored result in a file in the SGML File Format

7.3 Printable String Format

A String which contains only characters from 0000 to 007f.(without &,<,>,“). Special Characters are coded(see Appendix 1).

Purpose: of Format: Represent Text entered in textfields of the GUI in a way which can interpreted by SGML parser.

7.4 Base64Encoded Bytes

Bytes which when interpreted as ASCII contain only 64 printable Characters. 4 Bytes represent 3 Bytes of the original data(for Coding details see Appendix 3).

Purpose of Format: Allows the representation of bytes with printable characters so that the data can be displayed on paper and then scanned in an electronic format without losing data.

7.5 Base64Encoded String Format

A Java String containing only special printable characters (see Appendix 3).. 4 Characters represent 3 Bytes of the original data.

Purpose of Format: The internal java Character to Byte Converter will produce Base64Encoded Bytes out of these Strings.

7.6 ISAF File

When a file of this format is viewed with a HTML interpreter, the data of a Trademark Application in ISAF(International Standard Application Format) is displayed.

Purpose of Format: Allows the user to control the data of a Trademark Application with a HTML interpreter (e.g. webbrowser) and to print a ISAF document by using the print function of the HTML interpreter.

7.7 ISAF String

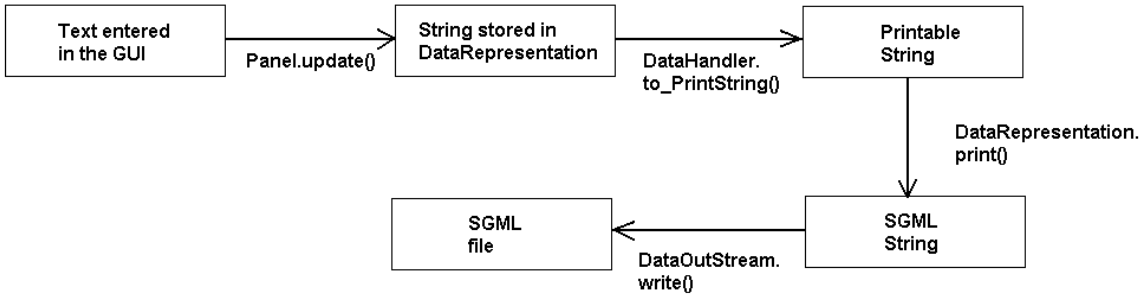
The internal java Character to Byte Converter will produce a HTML File(by converting a 00xy java character in a xy ASCII Characterbyte.) out of these Strings. Special Characters are coded. HTML Tags are used to structure the data according to ISAF.

Purpose of Format: Allows the Internal Char to Byte Converter to produce bytes which when stored result in a ISAF File.

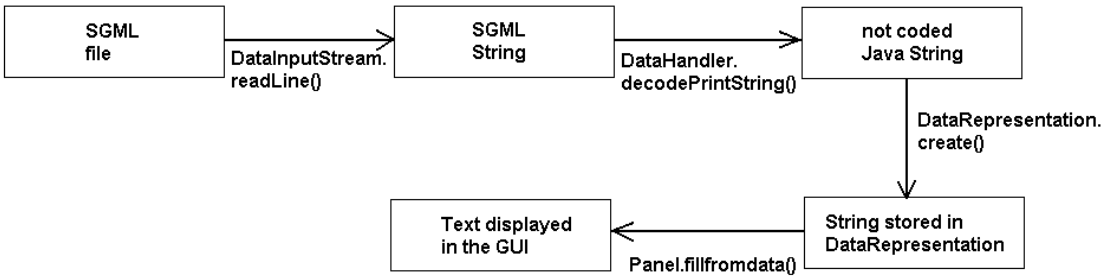
Remark: A Base64Encoded String is always a Printable String because the 64 used Characters are all ASCII of code 00 to 0f and none of them is &, <, > or “.

The following three graphics display how the data format is modified. The displayed functions are not all static, the first name in the expression only state the class which have this function. The graphics only display the changes of the formats and not the real sequence of function calls.

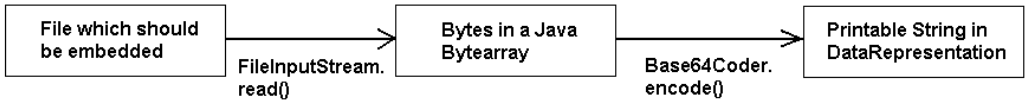
This graphic shows the data format flow, when you save a Trademark Application.
Here the data flow of a text field entry is shown



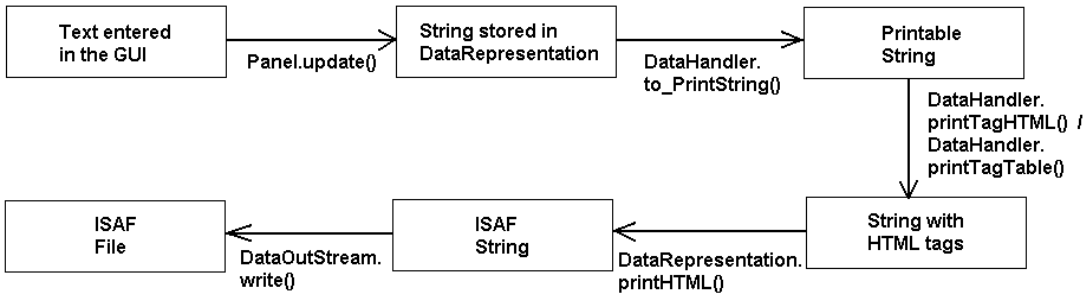
The next graphic shows the data format flow, when you load a Trademark Application, which have been saved by the Application. Here the data flow of a text field entry is shown.



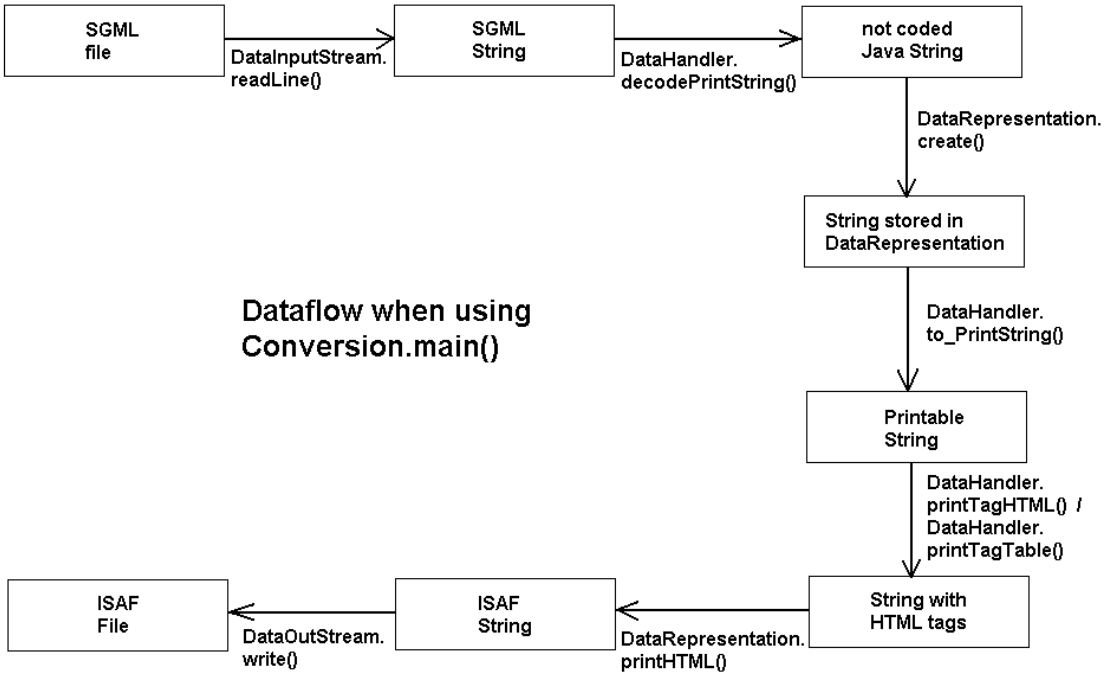
The next graphic shows the data format flow, when you embed a file using the Attachment Panel. The embedding allows to store the content of any file in a format which can be used for the SGML file and which allows to completly restore the file.



The next graphic shows the data format flow, when you convert the data of a Trademark Application into a ISAF conform HTML file using the export function of the MainFrame. Here the data flow of a text field entry is shown.



The last graphic shows the data format flow, when you convert Trademark Application stored in a SGML file into a ISAF conform HTML file using the main procedure of the Class Conversion (This Class is not part of the Application, it allows to convert a SGML file without starting the Application). Here the data flow of a text field entry is shown.



Dataflow when using Conversion.main()

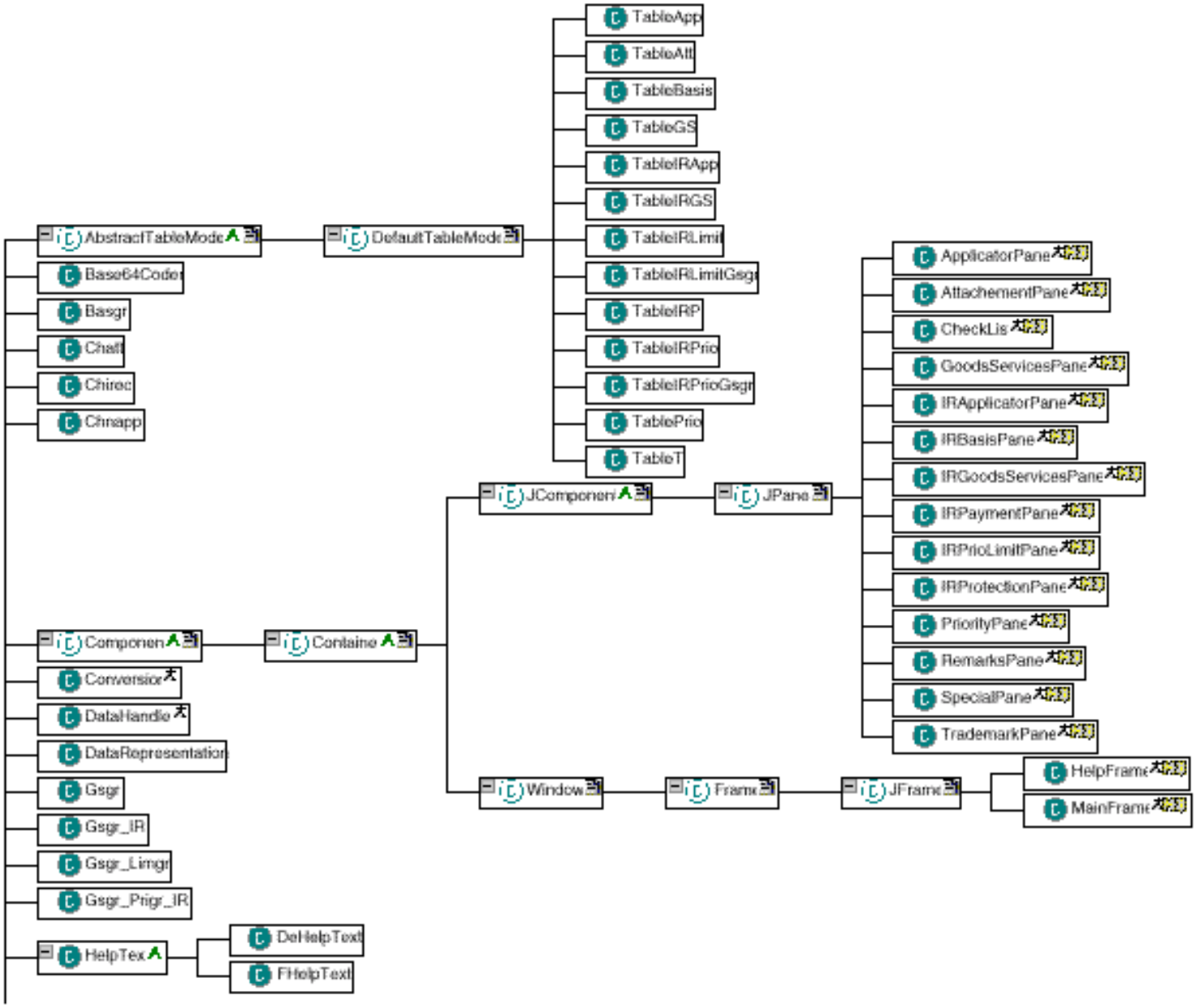
8 Class Diagrams

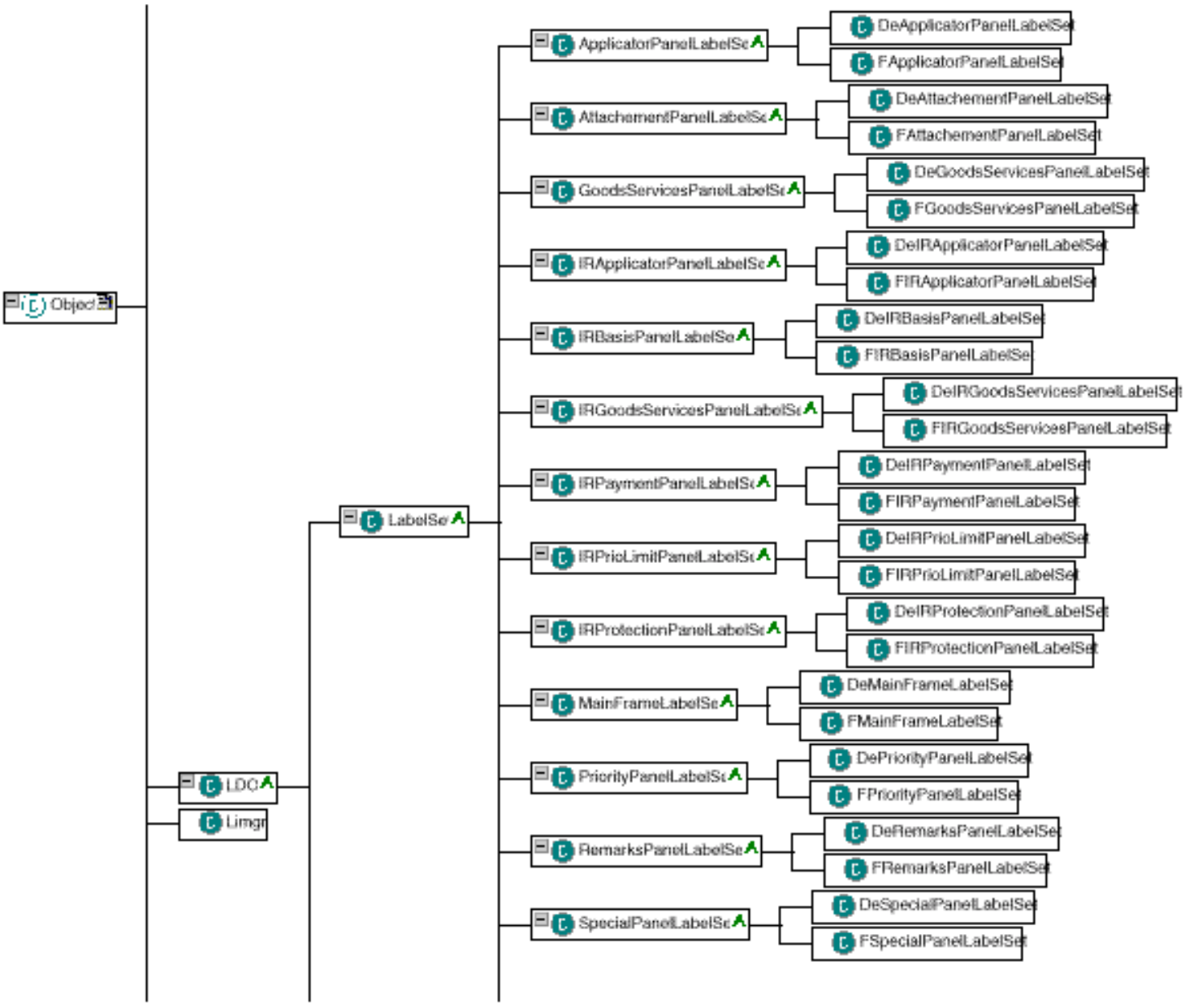
On the following three pages the class diagram of the program is shown.
The diagram was produced by Visual Age for Java. As you can see the ends of the class names are cut off, this would be another negative point to add to chapter nine.

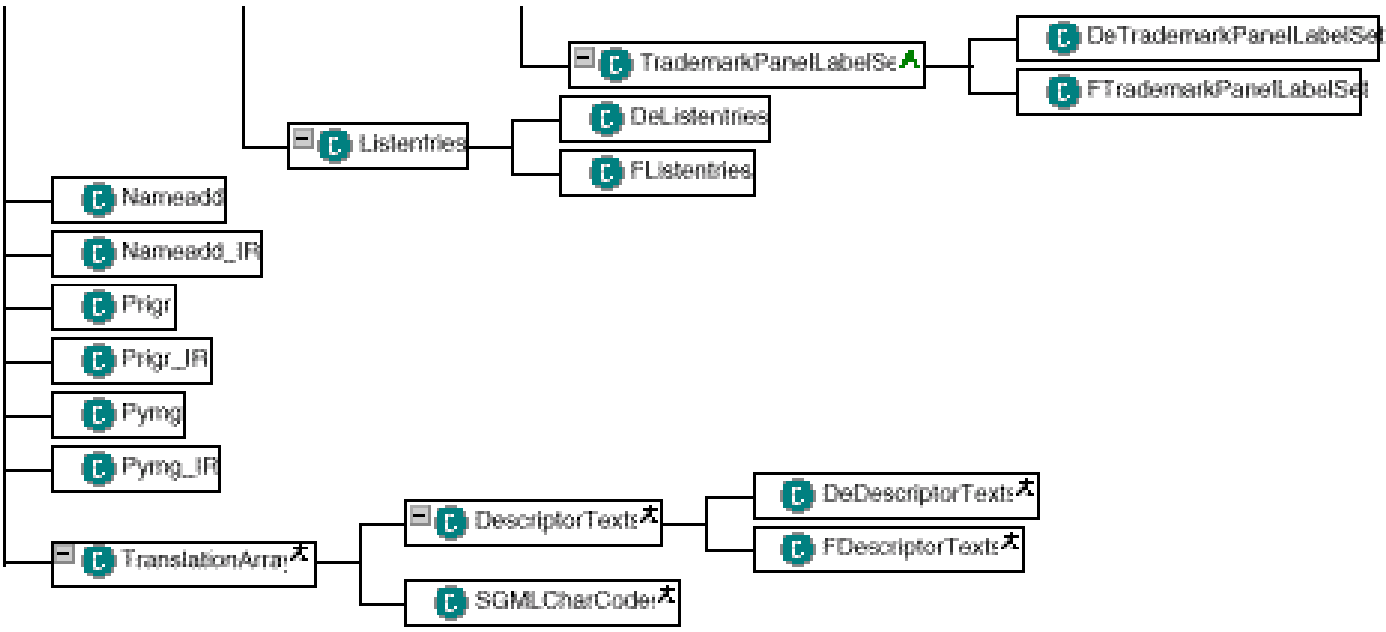
Description:

The filled Circle labeled with C indicates a Class which was designed for this program.
The hollow Circle labeled with C indicates a Standard Library Class.

The small text icon indicates, that this Class only has compiled class code(that means that there is not a readable java code)
The running man icon indicates, that there is a main procedure in the class which can be used as a startpoint for java.
The icon with two shapes(the last icon of MainFrame) indicates, that this class also has visible construction elements which can be viewed and modified using the Visual Editor of Visual Age for Java.
The small letter A indicates that the Class is abstract.
The Minus Icon states that there are Subclasses.


TableApp
TableAtt
TableBasis
TableGS
TableIRApp
TableIRGS
TableIRLimit
TableIRLimitGsgr
TableIRP
TableIRPrio
TableIRPrioGsgr
TablePrio
TableT
AbstractTableMode
DefaultTableMode
Base64Coder
Basgr
Chatt
Chirec
Chnapp
JComponen
JPane
ApplicatorPane
AttachementPane
CheckLis
GoodsServicesPane
IRApplicatorPane
IRBasisPane
IRGoodsServicesPane
IRPaymentPane
IRPrioLimitPane
IRProtectionPane
PriorityPane
RemarksPane
SpecialPane
TrademarkPane
Componen
Containe
Conversior
DataHandle
DataRepresentation
Gsgr
Gsgr_IR
Gsgr_Limgr
Gsgr_Prigr_IR
Window
Frame
JFrame
HelpFrame
MainFrame
HelpTex
DeHelpText
FHelpText


Objec
LDC
Limgr
LabelSe
ApplicatorPanelLabelSe
DeApplicatorPanelLabelSet
FApplicatorPanelLabelSet
AttachementPanelLabelSe
DeAttachementPanelLabelSet
FAttachementPanelLabelSet
GoodsServicesPanelLabelSe
DeGoodsServicesPanelLabelSet
FGoodsServicesPanelLabelSet
IRApplicatorPanelLabelSe
DeIRApplicatorPanelLabelSet
FIRApplicatorPanelLabelSet
IRBasisPanelLabelSe
DeIRBasisPanelLabelSet
FIRBasisPanelLabelSet
IRGoodsServicesPanelLabelSe
DeIRGoodsServicesPanelLabelSet
FIRGoodsServicesPanelLabelSet
IRPaymentPanelLabelSe
DeIRPaymentPanelLabelSet
FIRPaymentPanelLabelSet
IRPrioLimitPanelLabelSe
DeIRPrioLimitPanelLabelSet
FIRPrioLimitPanelLabelSet
IRProtectionPanelLabelSe
DeIRProtectionPanelLabelSet
FIRProtectionPanelLabelSet
MainFrameLabelSe
DeMainFrameLabelSet
FMainFrameLabelSet
PriorityPanelLabelSe
DePriorityPanelLabelSet
FPriorityPanelLabelSet
RemarksPanelLabelSe
DeRemarksPanelLabelSet
FRemarksPanelLabelSet
SpecialPanelLabelSe
DeSpecialPanelLabelSet
FSpecialPanelLabelSet


TrademarkPanelLabelSe
DeTrademarkPanelLabelSet
FTrademarkPanelLabelSet
Listentries
DeListentries
FListentries
Nameadd
Nameadd_IR
Prigr
Prigr_IR
Pymg
Pymg_IR
TranslationArra
DescriptorTexts
DeDescriptorTexts
FDescriptorTexts
SGMLCharCode

9 Visual Age for Java of IBM

Using IBM's Visual Age for Java, I encountered some severe bugs:

1. Visual Age offers the possibilities to save information in the class which would allow to recreate the Visual View of the Class when reimported out of Java Classes (it does it by adding a new function which contains all the necessary data in a special code. The body of this function, which contains the data, is then commented out so that this function does nothing when called). This is a very nice feature, but it is not working. Instead of recreating the whole Visual Data it only recreates the Main Panel or Frame , other Frames or Panels and most connections are not restored.
2. TableModels Classes should manage data for a GUI table. The idea of it is that the visible and the internal representation should be separated (That means that GUI Table manages the visible part like column width or highlighting the selected row and the TableModel manages the data events like changed values or deleted row. If the GUI table is not created out of the TableModel (which means that e.g. the Column width is set automatically, ignoring the column width of the GUI table) the TableModel only manages one row of the GUI and is therefor useless. I avoided that problem by creating the GUI tables out of the Model and then set the column identifiers and width of the GUI table out of the creation procedure of the TableModel.
3. There is an option for exporting classes which allows to export all referenced (not standard) classes also. This would be of course very useful, if it would work. It only exports some of the referenced swing classes and omit some of them so that the swing class library is still needed (and therefor the whole procedure is for nothing).

Despite these bugs and the fact that the program crashes sometimes (which is not that bad, because the program is almost always capable of restoring all data) Visual Age for Java is a good tool for creation GUI with additional code (it has good code support tools and debugger) if you are able to avoid the few bugs mention above.

10 User Informations

10.1 Production of a Jar File with Visual Age for Java

1. Select the packet TA.
2. Press Alt+D and choose „Exportieren“ in the Menu which has poped up.
3. Choose „Jar-Datei“, then click on „Weiter“.
4. Choose the file name and directory.
5. Select the „.class“ checkbox and the „Den Inhalt der Jar Datei komprimieren“ checkbox, all other checkbox should not be selected..
6. Click on the details button right of the „.class“ checkbox.
7. Select the project which contains the TA packet and also select the JFC class libraries (after this over 1600 classes should be selected)
8. Press „Ok“ to return to the previous menu.
9. .Press „Fertigstellen“ to create the Jar-File.

10.2 What do you need to run the Application

1. JDK 1.1.2 or higher.
2. For good Performance Petium Processor or equivalent suggested.
3. At last 10 MB Main Memory.
4. Windows 95 or Windows NT suggested, also runs on UNIX or LINUX, but the metrics do not match perfectly, so that textes are difficult to read.

10.3 How to run the Application

For Windows 95/Windows NT:

1. Open a Dos Window
2. Change to the directory where your Java Run Time Environement is situated (e,.g. c:\jdk1.2\)
3. Change to bin directory
4. Type in the following commando:java –Xms 10000000 –cp *jar-file* TA.MainFrame , where *jar-file* is the full path(with the filename) of the jar file which contains the classes for the Application..

For most other systems:

1. Use a commando interpreter.
2. If the interpreter reconizes the java command go to step 4.
3. Change to the directory where your Java Run Time Time Environement is situated or load the correct module.
4. Type in the following commando:java –Xms 10000000 –cp *jar-file* TA.MainFrame , where *jar-file* is the full path(with the filename) of the jar file which contains the classes for the Application..

10.4 How to run the Convert Function(Converts SGML Data File in HTML)

For Windows 95/Windows NT:

1. Open a Dos Window.
2. Change to the directory where your Java Run Time Environement is situated (e,.g. c:\jdk1.2\).
3. Change to bin directory .
4. Type in the following commando:java –cp *jar-file* TA.Conversion *sgml-file destination Language* , where *jar-file* is the full path(with the filename) of the jar file, which contains the class Conversion., *sgml-file* is the full path of the sgml file you want to convert, *destination* is the full path of the HTML file you want to create (should have a htm or html postfix) and *Language* is the Language Code of the Language you want to use for the HTML file (e.g.De for Geman).

For most other systems:

1 Use a commando interpreter.
2 If the interpreter reconizes the java command go to step 4
3 Change to the directory where your Java Run Time Time Environement is situated or load the the correct module
4 Type in the following commando:java –cp *jar-file* TA.Conversion *sgml-file destination Language* , where *jar-file* is the full path(with the filename) of the jar file, which contains the class Conversion., *sgml-file* is the full path of the sgml file you want to convert, *destination* is the full path of the HTML file you want to create (should have a htm or html postfix) and *Language* is the Language Code of the Language you want to use for the HTML file (e.g.De for Geman).

10.5 Other Important User Information

1. Long Initilization Time: The starting of the application takes up to two minutes.
2. Long Language Change: The changing of the language takes up to one and a half minutes.
3. No help information and only German supported(Language change to Frensh possible, but the textes are only test textes) in this version.
4. Long Embedding Time: Embedding files over 300 kB size takes up to 30 seconds.
5. Conversion of SGML to HTML produces new files for embedded files and linkes for the electronic attached files(the indicated file name is used)

11 Overall experience

The work I done for the IPI was very different to what I imagined before:

In contrast to the exercises at the University I was able to work quite independently which had good and negative effects:

- Because I worked alone and not in a team of programmers I thought I was not supposed to do a lot of design work. This was a quite sever error and I had to do a lot of redesign to correct the bad structure of the program.
- I could realise my own ideas, some of them led to a dead end other I think were a success and so I learned a little bit to distinguish the two kind of ideas.
- Working at home and not having a fixed deadline, the project was an exercise of self discipline.
- When I had problems, I first hesitated to consult the IPI, but I learned that the competent (and always prompt) answer of the people of the IPI resolved in most of the cases my problems.
- I was able to work uninterrupted and with a flexible hours schedule.

Working for a “real” project there were some new aspects:

- The customers wanted the application the way they have imagined and of course I tried to design the application that way, but in some cases I had to tell them that some features are technical very difficult to implement and I offered some alternatives
- The requirements were not given the way I was used of the exercises at the University, so the people of the IPI and I had meetings to clarify some issues.
- The people of IPI all were very friendly and helpful and answered all my questions to my satisfaction.
- The project was running in parallel to other related projects, so the requirements changed during the project (e.g. the SGML tag definitions changed a lot) and a lot of redesign had to been done.
- I did a lot of testing while programming, so I was surprised that the persons who tested my application found some capital errors and I realised that it is not possible to always find the own errors and that an outstanding person has a better chance to detect them.
- The iterative system development is not myth but it really takes place in real projects.

Experience with Java

- Java allows to create good object oriented programs but also bad ones, so it is still important to have a good class design.
- Some of the classes I created were not necessary, because there was already a library class which could do almost the same, so sometimes it is time saving to look at the library classes to find an appropriate class. But if you found one you still have to test if this do the thing you wand it to do, this especially the case with classes of external libraries (e.g. one of the classes I used for Base64Coding had even a bug).
- Automatic created code (by tools like Visual Age) can be very large(all the class files of my application need more then 500kB).

- Java is slow, especially graphic dependent actions.
- There are many ways to tune a java application(e.g. use of StringBuffer instead of String), but it is almost impossible to tune automatically created code(in the case of my application the graphical interface)

12 Thank notes

I want to thank all the persons of the IPI which have helped and consulted me during my projects especially I want to thank Matthias Günter who has agreed to supervise the project for the IPI.
I also want to thank Prof. Nierstasz for agreeing to this project and all the administrational support.
Last but not least I want to thank Isabelle Huber for the extra work she had to do because of my project.

Appendix 1: SGML Codes for special Characters

Description	Code	Character	Entity name
quotation mark	&#34;	"	&quot;
ampersand	&#38;	&	&amp;
less-than sign	&#60;	<	&lt;
greater-than sign	&#62;	>	&gt;
non-breaking space			
inverted exclamation	&#161;	¡	&iexcl;
cent sign	&#162;	¢	&cent;
pound sterling	&#163;	£	&pound;
general currency sign	&#164;	¤	&curren;
yen sign	&#165;	¥	&yen;
broken vertical bar	&#166;	¦	¦ &brkbar;
section sign	&#167;	§	&sect;
umlaut (dieresis)	&#168;	¨	¨ ¨
copyright	&#169;	©	&copy;
feminine ordinal	&#170;	ª	&ordf;
left angle quote, guillemotleft	&#171;	«	&laquo;
not sign	&#172;	¬	&not;
soft hyphen	&#173;	-	&shy;
registered trademark	&#174;	®	&reg;
macron accent	&#175;	¯	¯ &hibar;
degree sign	&#176;	°	&deg;
plus or minus	&#177;	±	&plusmn;
superscript two	&#178;	²	&sup2;
superscript three	&#179;	³	&sup3;
acute accent	&#180;	´	&acute;
micro sign	&#181;	µ	&micro;
paragraph sign	&#182;	¶	&para;
middle dot	&#183;	·	&middot;
cedilla	&#184;	¸	&cedil;
superscript one	&#185;	¹	&sup1;
masculine ordinal	&#186;	º	&ordm;
right angle quote, guillemotright	&#187;	»	&raquo;
fraction one-fourth	&#188;	¼	&frac14;
fraction one-	&#189;	½	&frac12;

half			
fraction three-fourths	¾	¾	¾
inverted question mark	¿	¿	¿
capital A, grave accent	À	À	À
capital A, acute accent	Á	Á	Á
capital A, circumflex accent	Â	Â	Â
capital A, tilde	Ã	Ã	Ã
capital A, dieresis or umlaut mark	Ä	Ä	Ä
capital A, ring	Å	Å	Å
capital AE diphthong (ligature)	Æ	Æ	Æ
capital C, cedilla	Ç	Ç	Ç
capital E, grave accent	È	È	È
capital E, acute accent	É	É	É
capital E, circumflex accent	Ê	Ê	Ê
capital E, dieresis or umlaut mark	Ë	Ë	Ë
capital I, grave accent	Ì	Ì	Ì
capital I, acute accent	Í	Í	Í
capital I, circumflex accent	Î	Î	Î
capital I, dieresis or umlaut mark	Ï	Ï	Ï
capital Eth, Icelandic	Ð	Ð	Ð Đ
capital N, tilde	Ñ	Ñ	Ñ
capital O, grave accent	Ò	Ò	Ò
capital O, acute accent	Ó	Ó	Ó
capital O, circumflex accent	Ô	Ô	Ô
capital O, tilde	Õ	Õ	Õ
capital O, dieresis or umlaut mark	Ö	Ö	Ö
multiply sign	×	×	×
capital O, slash	Ø	Ø	Ø
capital U,	Ù	Ù	Ù

grave accent			
capital U, acute accent	Ú	Ú	Ú
capital U, circumflex accent	Û	Û	Û
capital U, dieresis or umlaut mark	Ü	Ü	Ü
capital Y, acute accent	Ý	Ý	Ý
capital THORN, Icelandic	Þ	Þ	Þ
small sharp s, German (sz ligature)	ß	ß	ß
small a, grave accent	à	à	à
small a, acute accent	á	á	á
small a, circumflex accent	â	â	â
small a, tilde	ã	ã	ã
small a, dieresis or umlaut mark	ä	ä	ä
small a, ring	å	å	å
small ae diphthong (ligature)	æ	æ	æ
small c, cedilla	ç	ç	ç
small e, grave accent	è	è	è
small e, acute accent	é	é	é
small e, circumflex accent	ê	ê	ê
small e, dieresis or umlaut mark	ë	ë	ë
small i, grave accent	ì	ì	ì
small i, acute accent	í	í	í
small i, circumflex accent	î	î	î
small i, dieresis or umlaut mark	ï	ï	ï
small eth, Icelandic	ð	ð	ð
small n, tilde	ñ	ñ	ñ
small o, grave accent	ò	ò	ò
small o, acute	ó	ó	ó

accent			
small o, circumflex accent	&#244;	ô	&ocirc;
small o, tilde	&#245;	õ	&otilde;
small o, dieresis or umlaut mark	&#246;	ö	&ouml;
division sign	&#247;	÷	&divide;
small o, slash	&#248;	ø	&oslash;
small u, grave accent	&#249;	ù	&ugrave;
small u, acute accent	&#250;	ú	&uacute;
small u, circumflex accent	&#251;	û	&ucirc;
small u, dieresis or umlaut mark	&#252;	ü	&uuml;
small y, acute accent	&#253;	ý	&yacute;
small thorn, Icelandic	&#254;	þ	&thorn;
small y, dieresis or umlaut mark	&#255;	ÿ	&yuml;

Appendix 2: SGML Tags Electronic form

The end tags are not shown, but are always mandatory. The type column contains the following information:

- O– Mandatory field
- M – The tag is part of MECA
- A – Automatically filled by form (validator)
- T – Group tag
- X – a part of the transactions which is NOT implemented by the forms.

One of the problems is that a form tag must build a group tag and a first data tag in one step. This is not indicated, but will be done by the converter automatically. The triggering tag is NOT mentioned. It will be automatically assigned by the converter.

Content of the electronic form for national trademarks and the tags in the form

The type and values do in many cases directly reflect BAGIS. In some cases a minimal value translation must occur.

Master structure

The same master structure is used for national application and international registration.

Content	Form Tag	SGML-TAG	Type	Type and Values	Occurance
Header	<M0>	<TMTRANS>	OTMA	<TMTRANS> ***Description:*** This tag is compulsory on all transmissions.	1
Tag to identify tm communication relation to the Madrid system	<M1>	<TMMADRI>	OMA	Identifies the beginning of Madrid data This tag is compulsory on all transmissions. This tag is used in MECA, but has no relevante for the time being in the Swiss tagging. However, it is set for conformance reasons. No value us set.	1
Official date of notification of transactions	<M2>	<TMTDATE>	OMA	N8 (YYYYMMDD) ***Description:*** Contains the date of the sending of the SGML, can be set, but will be ignored	1
Sender ID	<M4>	<TMSEID>	OM	A10 (upper case, no entities) ***Description:*** This tag is compulsory on all transmissions and identifies the sender (company) of the transmission. It might be that this indentification is not useful in any case. But at least for the distributed EASY version we can set it.	1

Content	Form Tag	SGML-TAG	Type	Type and Values	Occurance
Receiver ID	<M5>	<TMREID>	OM	A10 (upper case, no entities) ***Description:*** This tag is compulsory on all transmissions and identifies the intended receiver of the transmission. Is IGE-CH usually	1
Transaction code	<M6>	<TMTCH>	OA	Type of Transaction . Defines a trademark application as the content Allows extensions to <PATTCH> and so on.	1
Effective date of a transaction version	<M7>	<TRAEDAT>	OM	N8 ***Description:*** This tag shows the date on which a particular version of a transaction is effective from. This tag is used to define the version of the SGML stream. Will be automatically set by EASY.	1
Transaction version number	<M8>	<TRANSVE>	OMA	N7.1 ***Description:*** Is the version number of a transaction and is compulsory on all transactions. Used for SGML only	1
Language of the original request code	<M9>	<ORIGLAN>	OMA	A2 (taken from form) selects the standard language of the transactions	1
Transaction Generator Identification	<M10>	<TRANSGI>	OMA	A20 – Identifies the transaction generator. Current values: * CH-EASY 2. * JavaApp * DB2DB * other	1
Auxiliary database update date Identification	<M11>	<TRANSADD>	OMA	N8	1
Start of mark transaction	<M12>	<CHTMT>	OT		1:N
Referencenumber of the transaction	<M12a>	<CHTMTRNR>	T		1
Start of national application	<M13>	<CHNAPP>	T		0:1
**** in a table of its own		**** in a table of its own			

Content	Form Tag	SGML-TAG	Type	Type and Values	Occurance
End of national application	</M13>	</CHNAPP>			
Begin of international registration	<M14>	<CHIREG>	T		0:1
**** in a table of its own		**** in a table of its own			
End of international registration		</CHIREG>			
Begin of attachment group	<M15>	<CHATT>	T		0:N
Attachment Type	<M15.1>	<CHATTTYP>	O	A1 0 Sonstiges 1 Markenabbildung 2 Einzelvollmacht 3 Prio-Beleg 4 Reglement Garantiemarke Reglement Kollektivmarke 5 Passkopie 6 Generalvollmacht	1
Description of Attachment	<M15.2>	<CHATTDES>		A50 Description in case of <CHATTYP>=0. Ignored in all the other cases.	1
Transmission method	<M15.3>	<CHATTMET>		A1 0 bereits am Institut vorhanden 1 Durch Person ausgeliefert (Briefpost, Kurier, Persönlich) 2 Befindet sich als separates Attachment im Email3 Ist in den Datenstrom eingebettet 4 Telefax	1
Filetype	<M15.4>	<CHATTFTY>		A3 for 2 and 3: type of file The Institute will use an internal image format (TIFF and JPEG) and PDF. All other formats are NOT archieved, but are converted to the archievable ones.Currently accepted: • JPG • BMP • GIF • PNG • TIF • PDF • DOC	1
Version number	<M15.5>	<CHATTVNR>		A20 Version number of the	1

Content	Form Tag	SGML-TAG	Type	Type and Values	Occurance
				document format. E.g. WORD97, PDF 3.0, PDF 4.0	
Filename	<M15.6>	<CHATTFNA>		In case of transmission method 2: compulsary; in the case of transmission method 0:refers to the unique document number in our local directories.	1
Embedded file	<M15.7>	<CHATTEMB>		In the case of transmission method 3: the embedded file.	1
End of attachment group	</M15>	</CHATT>			
End of trademark transaction group		</CHTMT>	A		1
Begin of Audit group	<M16>	<TMAUDIT>	MT	The audit group is used mainly for multiple transaction (with multiple types) SGML-streams. We will not encourage this. However, the Audit group is for conformance with MECA built into the system. In the current system <TRANTYP> is always TMTCH and <TRANNO> is 1.	1
Transaction group	<M17>	<TRANGR>	MT		1:N
Transaction Type	<M17.1>	<TRANTYP>	M	A8	1
Number of transactions in that group	<M17.2>	<TRANNO>	M	N7	1
		</TRANGR>	M		
		</TMAUDIT>	M		
End of transmission		</TMTRANS>	M		

Tagging for the national part of the form

Content	Form Tag	SGML-TAG	Type	Type and Values	Occurance
Update identifier		<CHUPDATE>		Refnr	0:1
Start of name and address group		<NAMEADD>	MT		1:M
Adresstype	<N1.0> <N2.0>	<NAMEADTY>	OM	A2 X Main Applicant H Applicant V Representative N Internal representative temporary during the Gesuchsstadium. Removed at registration and not published	1

Content	Form Tag	SGML-TAG	Type	Type and Values	Occurance
		***Begin of registry address			
Customer number for register address (if known)	<N1.1>< N2.1>	<CHADDNRR>		N10	1
Name (Company)	<N1.2> <N2.2>	<CHNAME>	OM	A126	1
Address	<N1.3> <N2.3>	<CHADDR>	M	A168	1
Postalcode	<N1.4> <N2.4>	<POSTCDE>		A8 (is separated for REGADR and for postal code comparison)	1
Place	<N1.4a> <N2.4a>	<CHPLACE>		A75	1
Country	<N1.5> <N2.5>	<COUNTRY>	M	A2	1
Nationality		<NAT>	OM	A2 (according to ST.3)	1
		*** Begin of Versandaddress			
Customer Address Group		<CHADDGRP>	G	12.1.1.1 Customer Address Group	0:1
Customer number for contact address (if known)	<N1.6> <N2.6>	<CHADDNR>		N10	1
Contact name	<N1.7> <N2.7>	<NAML1>	M	A30	1
Contact address line 1	<N1.8> <N2.8>	<ADDRL1>	M	A30	1
Contact address line 2	<N1.9> <N2.9>	<ADDRL2>	M	A30	1
Contact address line 3	<N1.10> <N2.10>	<ADDRL3>	M	A30	1
Contact address line 4	<N1.11> <N2.11>	<ADDRL4>	M	A30	1
Postalcode	<N1.12> <N2.12>	< POSTCDE>	O	A8 (is separated for REGADR and for PLZ comparison)	1
Place	<N1.13> <N2.13>	<CHPLACE>	O	A23	1
Country	<N1.14> <N2.14>	<COUNTRY>	M	A2	1
Language Code	<N1.15> <N2.15>	< LANID >	M	A2	1
Personal contact first name	<N1.16> <N2.16>	<NAML1C>	M	A50	1
Personal contact name	<N1.17> <N2.17>	<NAML2C>	M	A50	1
Phone number	<N1.18> <N2.18>	<PHONNUM>	M	A20	1
Fax number	<N1.19> <N2.19>	<FAXNUM>	M	A20	1
Email-Address	<N1.20> <N2.20>	<CHEMADDR>	O	A50	1

Content	Form Tag	SGML-TAG	Type	Type and Values	Occurance
Customer Address Group		</CHADDGRP>	G	Customer Address GRoup	0:1
End of name and address group		</NAMEADD>	M		1:M
Reference number of the customer	<N3.1>	<CHREF>		A25	
Dossier name of the customer	<N3.2>	<CHDOSSIER>		A25	
Trademark	<N4.1>	<MARKVE>	M	A340 The mark in case of mark type indicator = W. The description in case of O	0:1
Mark type Indicator	<N4.2>	<CHT1MTYPI>		A1 W Wortmarke B Bildmarke K Kombinierte Marke A Akustische Marke 3 3-D-Marke C Farbmarke G Geruchsmarke O Andere Markenart	
Mark type Description	<N4.3>	<CHTMTYPO>		A20 The mark type in case of mark type indicator = O	
Express-Marke	<N4.4>	<CHTMEXP>	O	A1 (Y, N)	1
Payment group		<PYMG>	MT	A1	1
Fee Type indicator	<N5.1>	<PYMTIND>	OA	A1 G Basic fee K Class fee E Express fee	1
Payment Mode indicator	<N5.2> <N5.5>	<PYMMTYP>	O	A1 7 bill requested 8 to account of the Institute	1
Date of Payment	<N5.2> <N5.6>	<PYMDAT>	M	N8 Not used für CH-trademark.	1
Deposit account number	<N5.3> <N5.7>	<PYMACC>	M	A10	1
Deposit account ID	<N5.4> <N5.8>	<PYMACCI>	M	A40 Bei CH-Marke: Prüfstring zu Deposit account number	1
End of Payment group		</PYMG>	M	A1	1
Start of goods and services group		<GSGR>	T		1:M
Class of good and services	<N6.1>	<NICCLAI>	OM	N2	1
Statement of the class	<N6.2>	<GSTERMO>	OM	A2000	1
End of goods and services group		</GSGR>			0:M
Statement of the classes	<N6.2>	<GSTERMO>	OM	A6000	1
Type of TM (individual, guarantee, collective)	<N7>	<TYPMARI>	O	A2 K Kollektivmarke	1

Content	Form Tag	SGML-TAG	Type	Type and Values	Occurance
				G Garantiemarke I Individualmarke	
Start of priority group		<PRIGR>	MT		1:M
Priority date	<N8.1>	<PRIAPPD>	M	N8	1
Priority country code	<N8.2>	<DCPCD>	M	A2	1
Priority number	<N8.3>	<PRIAPPN>	M	A20	1
End of priority group		</PRIGR>			0:M
Color claim	<N9>	<COLCLAI>	M	A134	1
Remark	<N10>	<CHREMARK>		A500	1

International Registration

Content of the electronic form for the international registration and the tags in the form. The base is the EN transaction. However, new tags had been added and the order of tags was changed.

Description	Form Tag	Transaction Tag	Mand/ Meca	Systematics	Occurance
Update identifier		<CHUPDATE>		Refnr	0:1
Start of name and address group		<NAMEADD>	MT		1:M
Name and address type code	<I1.1><I2.1>	<NAMADTY>	M	A2 V Representative (as sent to WIPO)I C Correspondence address (of X if different from CH basic registration or application; will be sent to WIPO)	1
Customer number for register address (if known)	<I1.2><I2.2>	<CHADDNRR>		N10	1
Name	<I1.3> <I2.3>	<CHNAME>	OM	A126	1
Address	<I1.4> <I2.4>	<CHADDR>	M	A168	1
Postal code	<I1.5> <I2.5>	<POSTCDE>	O	A8	
Place	<I1.6> <I2.6>	<CHPLACE>	O	A84	
Country code	<I1.7> <I2.7>	<COUNTRY>	OM	A2	1
Nationality	<I1.8> <I2.8>	<NAT>	OM	A2 (according to ST.3)	1
Customer Address Group		<CHADDGRP>	G	Customer Address Group	0:1
Customer number for contact address	<I1.9> <I2.9>	<CHADDNR>		N20	1
Contact name	<I1.10> <I2.10>	<NAML1>	M	A50	1
Contact Address line 1	<I1.11>	<ADDRL1>	M	A50	1

Description	Form Tag	Transaction Tag	Mand/ Meca	Systematics	Occurance
	<I2.11>				
Contact address line 2	<I1.12> <I2.12>	<ADDRL2>	M	A50	1
Contact address line 3	<I1.13> <I2.13>	<ADDRL3>	M	A50	1
Contact address line 4	<I1.14> <I2.14>	<ADDRL4>	M	A50	1
Postalcode	<I1.15> <I2.15>	<POSTCDE>	O	A8 (is separated for REGADR and for PLZ comparison)	1
Place	<I1.16> <I2.16>	<CHPLACE>	O	A76	1
Country	<I1.17> <I2.17>	<COUNTRY>	M	A2	1
Language Code	<I1.18> <I2.18>	< LANID >	M	A2	1
Personal contact first name	<I1.19> <I2.19>	<NAML1C>	M	A50	1
Personal contact name	<I1.20> <I2.20>	<NAML2C>	M	A50	1
Phone number	<I1.21> <I2.21>	<PHONNUM>	M	A16	1
Fax number	<I1.22> <I2.22>	<FAXNUM>	M	A16	1
Email-Address	<I1.23> <I2.23>	<CHEMADDR>	O	A50	1
End of Customer Address Group		</CHADDGRP>	G		
End of name and address group		</NAMEADD>			1:M
Reference number of the customer	<I3.1>	<CHREF>	O	A50	
Dossier name of the customer	<I3.2>	<CHDOSSIER>	O	A50	
Start of basic application and registration group		<BASGR>		Logically only one of the following three items can be used.	1:M
Basic application number	<I4.1>	<BASAPPN>	M	A20	0:1
Basic registration number	<I4.2>	<BASREGN>	M	A20	0:1
Basic application reference number	<I4.3>	<BASREFN>		Used to refer to a previous electronic CH application as long as the application number is not available. A20	0:1
End of basic application and registration group		</BASGR>			1:M
International registration based on application	<I4.4>	<CHRGMETH>	O	A1 A – application, R-Registration Default: R	0:1
Start of designated		<DESG>			1:M

Description	Form Tag	Transaction Tag	Mand/ Meca	Systematics	Occurance
Contracting Party group					
Designated Contracting Party code	<I5.1>	<DCPCD>	M	A2	1
End of designated Contracting Party group		</DESG>			1:M
Start of goods and services group		<GSGR>		Contains all goods and services claimed for the international registration.	1:M
Nice class number	<I6.1>	<NICCLAI>	M	N2 Im CH-Teil werden andere Tag-Bezeichnungen verwendet (class of good and services bzw. Statement of the classe)	1
Goods and service	<I6.2>	<GSTERMT>	X	A2000 or greater In French.	1
End of goods and services group		</GSGR>			1:M
Goods and service	<I6.3>	<GSTERMT>	X	A6000 or greater In French.	0:1
Start of priority group		<PRIGR>			0:M
Priority date	<I7.1>	<PRIAPPD>	M	N8	1
Priority Contracting Party code	<I7.2>	<DCPCD>	M	A2	1
Priority number	<I7.3>	<PRIAPPN>	M	A20	0:1
Start of goods and services group		<GSGR>	MX		0:M
Nice class number	<I7.4>	<NICCLAI>	MX	N2	1
Goods and service terms	<I7.5>	<GSTERMT>	MX	A2000 or greater	0:1
End of goods and services group		</GSGR>	MX		0:M
Goods and service	<I7.6>	<GSTERMT>	X	A2000 or greater In French.	0:1
End of priority group		</PRIGR>			0:M
Colors claimed description	<I8.1>	<COLCLAI>	X	A300 in French.	0:1
Start of limitation group		<LIMGR>	TX	The goods and services claimed for the international registration are specified in the <GSGR> groups. Exceptions (i.e. limitations) for some countries are indicated by this <LIMGR> group. The rules are: 1. If no limitation group is present for a specific designated contracting party then it means that all goods and services specified in all <GSGR> groups are valid for that country. 2. Only one limitation group per country is allowed.	0:M

Description	Form Tag	Transaction Tag	Mand/ Meca	Systematics	Occurance
				3. If not all classes specified in the <GSGR> group are listed in a limitation group for a specific country then the not listed classes shall be treated as not limited. (The listed classes are treated as specified by <LSLIMT>.)	
Designated Contracting Party code	<I9.1>	<DCPCD>	X	A2	1
Start of goods and services group		<GSGR>	TX		1:M
Nice class number	<I9.2>	<NICCLAI>	X	N2	1
Limited list code	<I9.3>	<LISLIMT>	X	A2 1 delete from basic list the content of GSTERMT; 2 list for this country and class limited to GSTERMT. Default: 2	1
Goods and service terms in French	<I9.4>	<GSTERMT>	X	A2000 or grater	1
End of goods and services group		</GSGR>	X		1:M
Goods and service	<I9.5>	<GSTERMT>	X	A2000 or greater In French. Dieses Tag verursacht ein Problem mit den Interpretationsregeln. Pro Staat kann man in dieser LIMGR ein einziges solches Tag haben. Es stellt sich die Frage, ob man dann die GSGR dafür ausschlissen soll (dh Benutzer kann nur entweder oder verwenden; die Klassenangaben allein machen mE keinen grossen Sinn). Verwendet er dieses GSTERMT nicht, so soll keine Einschränkung erfolgen. Verwendet er es, so gilt für diesen Staat diese Liste gesamthaft positiv oder negativ; es braucht aber noch ein Tag, wo der Benutzer dies so festlegen kann (siehe anschliessend)	0:1
Limited list code	<I9.3>	<LISLIMT>	X	A2 1 delete from list 2 list limited to Default: 2	0:1

Description	Form Tag	Transaction Tag	Mand/ Meca	Systematics	Occurance
End of limitation group		</LIMGR>	X		0:M
Payment group		<PYMG>	T		1:M
Fee Type indicator		<PYMTIND>	OA	A1 B Basic fee (Grundgebühr) N National fee I Individual fee X Extension fee (Ergänzungsgebühr) A Additional fee (Zusatzgebühr)	1
Payment Mode indicator	<I9.1> <I9.5> <I9.9> <I9.13> <I9.17>	< PYMMTYP >		A1 1 Current account at WIPO 2 Payment to WIPO 3 Check for WIPO sent to the Institute 4 Check for WIPO sent to WIPO 5 To bank account of WIPO 6 To postal account of WIPO 7 bill of the Institute requested 8 to an account at the Institute National fee (fee type N) must be paid by payment mode 7 or 8. Payment mode 3 is not possible at all.	1
Date of Payment	<I9.2> <I9.6> <I9.10> <I9.14> <I9.18>	<PYMDAT>	M	N8	1
Account number	<I9.3> <I9.7> <I9.11> <I9.15> <I9.19>	<PYMACC>	M	A10	1
Deposit account ID or other relevant payment information depending on the mode indicator	<I9.4> <I9.8> <I9.12> <I9.16> <I9.20>	<PYMACCI>	M	A40	1
Payment group		</PYMG>	M	A1	1
Remark	<I10>	<CHREMARK>		A500	1

Appendix 3: Base64Coding

The following text is an extract of RFC 1421 (Privacy Enhancement for Electronic Mail):

Proceeding from left to right, the bit string resulting from step 3 is encoded into characters which are universally representable at all sites, though not necessarily with the same bit patterns (e.g., although the character "E" is represented in an ASCII-based system as hexadecimal 45 and as hexadecimal C5 in an EBCDIC-based system, the local significance of the two representations is equivalent).

A 64-character subset of International Alphabet IA5 is used, enabling 6 bits to be represented per printable character. (The proposed subset of characters is represented identically in IA5 and ASCII.) The character "=" signifies a special processing function used for padding within the printable encoding procedure.

To represent the encapsulated text of a PEM message, the encoding function's output is delimited into text lines (using local conventions), with each line except the last containing exactly 64 printable characters and the final line containing 64 or fewer printable characters. (This line length is easily printable and is guaranteed to satisfy SMTP's 1000-character transmitted line length limit.) This folding requirement does not apply when the encoding procedure is used to represent PEM header field quantities; Section 4.6 discusses folding of PEM encapsulated header fields.

The encoding process represents 24-bit groups of input bits as output strings of 4 encoded characters. Proceeding from left to right across a 24-bit input group extracted from the output of step 3, each 6-bit group is used as an index into an array of 64 printable characters. The character referenced by the index is placed in the output string. These characters, identified in Table 1, are selected so as to be universally representable, and the set excludes characters wit particular significance to SMTP (e.g., ".", "<CR>", "<LF>").

Special processing is performed if fewer than 24 bits are available in an input group at the end of a message. A full encoding quantum is always completed at the end of a message. When fewer than 24 input bits are available in an input group, zero bits are added (on the right) to form an integral number of 6-bit groups. Output character positions which are not required to represent actual input data are set to the character "=". Since all canonically encoded output is an integral number of octets, only the following cases can arise: (1) the final quantum of encoding input is an integral multiple of 24 bits; here, the final unit of encoded output will be an integral multiple of 4 characters with no "=" padding, (2) the final quantum of encoding input is exactly 8 bits; here, the final unit of encoded output will be two characters followed by two "=" padding characters, or (3) the final quantum of encoding input is exactly 16 bits; here, the final unit of encoded output will be three characters followed by one "=" padding character.

Printable Encoding Characters

Value	Encoding	Value	Encoding	Value	Encoding	Value	Encoding
0	A	17	R	34	i	51	Z
1	B	18	S	35	j	52	0
2	C	19	T	36	k	53	1
3	D	20	U	37	l	54	2
4	E	21	V	38	m	55	3
5	F	22	W	39	n	56	4
6	G	23	X	40	o	57	5
7	H	24	Y	41	p	58	6
8	I	25	Z	42	q	59	7
9	J	26	a	43	r	60	8
10	K	27	b	44	s	61	9
11	L	28	c	45	t	62	+
12	M	29	d	46	u	64	/
13	N	30	e	47	v		
14	O	31	f	48	w		
15	P	32	g	49	x	(pad)	=
16	Q	33	h	50	y		