

Customer notepad stamps in billing

Emin Gabrielyan

Switzernet

Created on [2008-07-27](#)

Abstract: Customers in the billing system have individual notepads designed for storing practically unlimited amount of text. The notepad values are accessible via the main billing database. In this document we introduce formatted tags, named stamps, designed for saving various types of individual data using customer notepads. The stamps and their attributes are formatted so as to be retrieved with database queries and easily classified.

Table of content:

Customer notepad stamps in billing	1
1. Tag format	3
2. Stamp format.....	4
2.1. Excel stamp builder	6
2.2. Pictograms.....	6
3. Stamps and samples.....	7
3.1. abuse.....	8
3.1.1. attributes👤.....	8
3.2. class	9
3.2.1. attributes✔.....	10
3.3. contract.....	10
3.3.1. attributes✔.....	11
3.3.2. triggered👤.....	11
3.3.3. searchable🔍.....	12
3.4. found.....	13
3.4.1. attributes✔.....	13
3.4.2. triggered👤.....	13
3.4.3. searchable🔍.....	13
3.5. identity	14
3.5.1. attributes✔.....	14
3.5.2. triggered👤.....	14

3.6.	learn.....	15
3.6.1.	attributes✓.....	17
3.6.2.	triggered👤.....	17
3.7.	limit.....	18
3.7.1.	attributes✓.....	20
3.7.2.	triggered.....	20
3.8.	lost.....	21
3.8.1.	attributes✓.....	22
3.8.2.	triggered👤.....	22
3.8.3.	searchable🔍.....	23
3.8.4.	Safe stamp building.....	24
3.9.	remark.....	25
3.9.1.	attributes👤.....	25
3.9.2.	Lost email address.....	25
3.10.	return.....	26
3.10.1.	attributes✓.....	26
3.10.2.	triggered👤.....	27
3.11.	ship.....	27
3.11.1.	attributes✓.....	28
3.11.2.	searchable🔍.....	28
3.12.	supply.....	29
3.12.1.	attributes✓.....	29
3.13.	tariff.....	30
3.13.1.	attributes✓.....	31
3.14.	undue.....	31
3.14.1.	attributes✓.....	31
3.14.2.	searchable🔍.....	31
3.15.	until.....	32
3.15.1.	attributes✓.....	32
3.15.2.	triggered👤.....	32
3.15.3.	searchable🔍.....	33
4.	Recurrent monthly queries.....	34
4.1.	Customer acquisition chart.....	35
5.	Text processing examples.....	37
5.1.	Notepad processing examples with Perl.....	37
5.2.	Notepad processing example with excel.....	38
5.2.1.	User defined excel function.....	40
6.	Conclusions.....	45
7.	References.....	45
7.1.	This document.....	45

7.2. XML	45
7.3. Perl and Excel	46
7.4. Excel VBA functions	46
7.5. Related to the “why” attribute of the “class” stamp	46
7.6. Choice of the names of stamps, attributes, and values	46
7.7. Office excel file	46
7.8. The original TODO task	46



1. Tag format

The stamp tags are defined for storing structured data using the customer notepad of the billing interface. The syntax of stamp is that of XML tags. A tag is a text, enclosed in angle brackets, which starts with a name of the element and can be followed by one or more attributes [xml]. The attributes provide full information on notepad stamps and the stamp tags do not have additional content. The tag of the stamp must contain a space and slash “ / ” just before the closing angle bracket of the tag:

```
<name attribute=value />
```

The exception is the `remark` tag reserved for comments. The free text must be enclosed between the start-tag `<remark >` and the end-tag `</remark>`:

```
<remark on=080601 >
Delivery failure to customer@bluewin.ch
</remark>
```

 All **new stamps are added on top** of the notepad. Therefore the stamps appear in the notepad in inverse chronological order :

```

<until on=090501 what=090831 />
<tariff on=080829 from=business to=private
why=contract />
<class on=080829 from=shop to=billable
why=contract />
<contract on=080825 what=flyer />
<supply on=080727 what=pap2t to=mediamarkt />

```

By relying on the inverse chronological order, short MySQL queries can be designed. If the order is not respected, certain queries cannot operate properly, without first ordering the tags according to the date of the `on` attribute. For examples where the order is important, see sections 3.4 and 3.8 for `lost` and `found` stamps.

2. Stamp format

So far, the following 15 stamps are defined:

<code>abuse</code>	<code>learn</code>	<code>ship</code>
<code>class</code>	<code>limit</code>	<code>supply</code>
<code>contract</code>	<code>lost</code>	<code>tariff</code>
<code>found</code>	<code>remark</code>	<code>undue</code>
<code>identity</code>	<code>return</code>	<code>until</code>

The 5 attributes below can store all parameters needed by notepad stamps. An attribute is usually applicable to several stamps.

`from` `on` `to` `what` `why`

The attribute `on` is applicable to all stamps and its value is either the date when the stamp is put into the notepad or the date of the

event (usually of an incoming letter) which triggered the stamp. The **on** attribute is obligatory for all stamps.

The table below shows the applicability of the 5 attributes for all 15 stamps. The last two columns of the table indicate on, whether the stamp will be recurrently used in database queries (indicated by sign: 🔍), and whether the stamp is directly triggered by an external event (indicated by sign: 🌐).

	from	on	to	what	why	🔍	🌐
abuse		🕒					
class	✓	🕒	✓		✓		
contract		📝		✓		🔍	🌐
found		✉️				🔍	🌐
identity		✉️		🌐			🌐
learn	✓	✉️		✓			🌐
limit	✓	🕒📝	✓		✓		🌐
lost		✉️🔄		📁✉️		🔍	🌐
remark		🕒					
return		✉️		☎️	✓		🌐
ship		✉️		☎️	✓	🔍	
supply		✉️	✓	☎️			
tariff	✓	🕒	✓		✓		
undue		🕒		📁		🔍	
until		✉️		📅		🔍	🌐

2.1. Excel stamp builder

Joined is an Excel file which permits to create a stamp by entering the input values of applicable attributes. The excel file contains one row per stamp and five columns for the input values of the attributes. If attributes are entered incorrectly an error message will appear. If the excel does not encounter errors, the tag of the stamp will be displayed:

	from	on	to	what	why	stamp
abuse		2008-08-01				<abuse on=080801 />
class	billable	2008-08-01	overdue			<class on=080801 from=billable to=overdue

[\[xls\]](#)

The excel file contains a sheet with the list of valid values of attributes. Whenever new values are introduced, the sheet of valid values must be updated.

2.2. Pictograms

Below are the significations of all pictograms used in this document:



Signifies the current date, in **yyymmdd** format, when the stamp is added into the notepad



Indicates that the attribute is applicable for the given stamp



Signifies that the attribute can be present or absent for the given stamp depending on specific conditions (see the attribute **why** for the stamp **class** in section 3.2 and for the stamp **limit** in section 3.7)



Signifies the date of the signature of the contract in **yyymmdd** format




Means that the stamp will be recurrently searched on monthly bases (or more frequently) by database queries for statistics or for mailing (see section 4)





Means that the stamp is triggered by an external event, such as an incoming mail, email, or a received shipment





Signifies the date of an incoming expedition (mail, email, or shipment) in **yyymmdd** format


- 


Signifies the country of the origin of the customer according to the identification document in [two letter ISO country code format](#) (the list of all countries is in the excel sheet of valid values of the stamp builder [\[xls\]](#), see section 2.1)
- 


Signifies the payment execution date on the wire transfer order (in `yyymmdd` format), which is sent us by the customer as a proof of the payment
- 


Signifies that the stamp is triggered by an external event only for specific cases (see section 3.7)
- 


Signifies the reception date (in `yyymmdd` format) of a letter, returned due to a bad postal address
- 


Signifies the date of an outgoing expedition
- 

Signifies the type of hardware such as “c450ip” or “pap2t” (all devices are listed in the excel sheet of valid values of the stamp builder [\[xls\]](#), see section 2.1)
- 

Signifies the issue date of an invoice (in `yyymm01` format), see sections 3.14
- 

Signifies the last date of the service in `yyymmdd` format
- 


Signifies that the following is a description of works to be explicitly carried out (usually updates of external how-to documents)
- 

Signifies that the stamp will be recurrently searched for monthly statistics
- 

Signifies that the stamp will be recurrently searched for the monthly mailing of reminders and of paper invoices

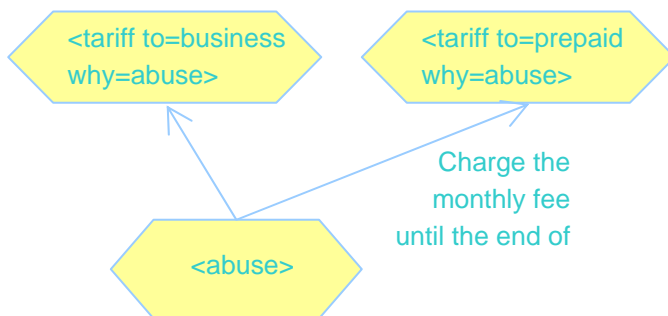
3. Stamps and samples

This section presents all defined stamps. For each stamp we have a subsection “[attributes](#)” showing all attributes applicable to the stamp (a line of the table of section 2). If the stamp is triggered by an external event, there is a subsection “[triggered](#)” where we explain the events causing the stamp and the accompanying processes to be carried out before putting the stamp. The subsection “[triggered](#)” often contains procedures to be documented or updated in corresponding how-to documents. When a process or a task must be documented (or

correspondingly updated) in how-to docs, the text is preceded by a pictogram . If the stamp is a subject of recurrent monthly queries, there is a subsection “[searchable](#)” describing the query and its purpose.

3.1. *abuse*

This stamp is added when we detect an abusing use of free minutes. The stamp can be followed by a change of rates:






Recall that the fresh stamps are always added on top.

```

<tariff on=080725 from=private to=business
why=abuse />
<abuse on=080722 />
  
```

This stamp is for information. No action must be taken obligatorily. User may have several [abuse](#) stamps put at different dates. More information about the type and the reason of each abuse can be added using [remark](#) (see section 3.9). The decision can be taken based on the history of [abuse](#) stamps. The blockage is one of the possible actions (no stamp is defined for blockage).

3.1.1. attributes

	from	on	to	what	why		
abuse							

3.2. class

This stamp is added obligatorily when the customer changes the class.

```
<class on=080705 from=overdue to=collect
why=creditreform />
<class on=080705 from=overdue to=collect
why=poursuites />
<class on=080716 from=billable to=fraud
why=chargeback />
<class on=080625 from=billable to=overdue
why=lost />
<class on=080625 from=billable to=overdue />
<class on=080829 from=shop to=billable
why=contract />
```

All transfers to class “**collect**” must be accompanied by an attribute **why** indicating the collection agency where the case of the customer is submitted to. Note that a transfer to the class “**collect**” may take place only when the case is already submitted to an agency [[ch1](#)], [[ch2](#)], [[us1](#)] (The new meaning of “Cancelled Due”). The value “poursuites” is for the debt collection proceedings (office des poursuites et faites).








There are two entry doors to the “**overdue**” class: the first one is the 4th level of overdue reminder, in which case the **why** attribute of the **class** stamp shall miss, and the second one, when the user’s postal address is wrong, in which case the **class** stamp must be accompanied by a **why=lost** attribute.



The doc accompanying the excel file of processing of 4th reminders must be updated so as to add also the stamp **<class to=overdue />**.

3.2.1. attributes ✓

The attributes of the `class` stamp are used similarly to the attributes of two other stamps: `limit` (see section 3.7) and `tariff` (see section 3.13):

	from	on	to	what	why		
<code>class</code>	✓		✓		✓		
<code>limit</code>	✓	 	✓		✓		
<code>tariff</code>	✓		✓		✓		

3.3. contract

This stamp stores the date of the conclusion of the contract (i.e. the date of the signature of the contract).

```
<contract on=080205 what=paper />
<contract on=080205 what=fax />
<contract on=080205 what=scan />
<contract on=080625 what=esign />
<contract on=080205 what=prepaid />
```

The attribute `what` is equal to the form (or the media) of the signed contract which we have in our possession (often required for legal issues). Though in case of "`what=prepaid`" there is no juridical contract, the stamp must be put for the subscription date.

Below is an example of a notepad of a customer who bought an adapter in MediaMarkt, subscribed by returning the included flyer-contract, and cancelled the contract after a year.

```
<until on=090501 what=090831 />
<tariff on=080829 from=business to=private
why=contract />
```










```

<class on=080829 from=shop to=billable
why=contract />
<contract on=080825 what=flyer />
<supply on=080727 what=pap2t
to=mediamarkt_crissier />

```

3.3.1. attributes ✓

The value of the `on` attribute is the date of the signature of the contract. This value is important for determining a valid expiration date when a cancellation request is received. We set the value of the `what` attribute of the `until` stamp using the value of the `on` attribute of the `contract` stamp (see section 3.15).

	from	on	to	what	why		
<code>contract</code>				✓			
<code>until</code>							

3.3.2. triggered 🛠️🔧

The `contract` stamp is triggered by a reception of a signed contract. The `on` attribute is set to the date of the signature of the contract. In case of “`what=prepaid`” (a subscription without a contract), the `on` attribute is the date of the first payment without which the demand is not treated (as if the contract is not yet signed) and the customer is not created.

🛠️🔧 The how-to doc of customer registration rules must be updated so as to require the `contract` stamp upon the registration of the customer.

🛠️🔧 The `contract` stamp with the dates of contract signature must be added for all previously registered customers. All paper contracts must be processed for this purpose. Joined is an excel file, which can be used for manually entering the data of paper contracts [\[xls\]](#). The stamp will be calculated automatically for each entered line.

3.3.3. searchable 🔍

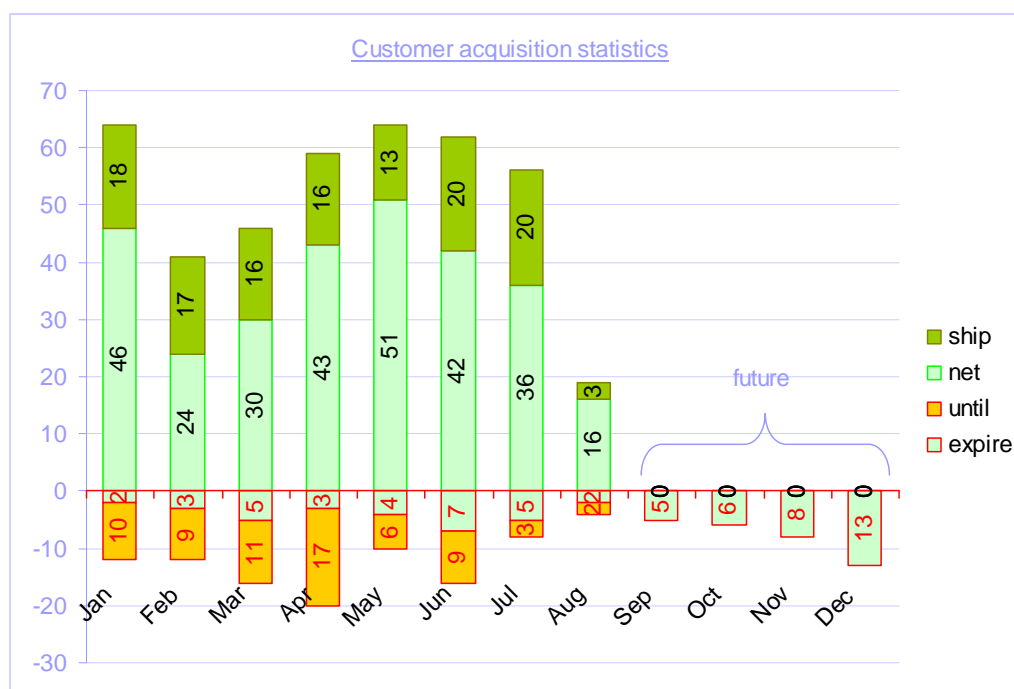
Recurrent monthly searches (section 3.15) of the **contract** and **until** stamps will be carried out for the customer acquisition statistics on the numbers of subscribing and leaving customers. These statistics will also show numbers of subscriptions with and without equipment (using the stamp **ship** presented in section 3.11). The attribute **on** of all stamps and the attribute **what** of the stamp **until** will be retrieved by database queries for computing the statistics for ranges of months.

	from	on	to	what	why	🔍	🌐
contract		📝		✓		🔍	🌐
ship		✉️		📞	✓	🔍	
until		✉️		📅		🔍	🌐



The web site with statistics must contain fully documented samples of the queries.

Section 4.1 shows a method of a treatment of an input list of stamps and presents the format of customer acquisition chart.












[xls]

3.4. found

This stamp means that the customer sent us a new postal address usually after receiving from us a request to send us a correct address due to a returned letter. The `found` stamp normally follows the stamp `lost` (put when an outgoing letter is returned undelivered as discussed in section 3.8).


```
<found on=080625 />
<lost on=080624 what=080606 />
```


3.4.1. attributes ✓

	from	on	to	what	why		
<code>found</code>							
<code>lost</code>							

3.4.2. triggered

The `found` stamp is triggered by the new postal address sent us via email (or via post). The `found` stamp means that the account (previously blocked due to `lost` stamp) is unblocked and the new address is entered into the billing. The `on` attribute is equal to the date when the new address is received (e.g. the date of the email of the customer containing the new postal address).

 The documentation of the treatment of this type of emails must be updated (or added) in how-to doc of support@ emails: (a) unblock the customer, (b) correct the address, and (c) put the `found` stamp

 The documentation of the treatment of this type of emails must be updated (or added) in how-to doc of billing@ emails

3.4.3. searchable 🔍

This stamp must be searched in combination with the `lost` stamp every time when carrying out recurring monthly mailing. When `lost` stamp is present and is not annihilated by a `found` stamp,

the customer must be excluded from the mailing. Since the stamps are added in inverse-chronological order (i.e. new stamps are added always on top of the notepad, see section 1), the locate MySQL function must check whether the position of the `found` tag in the notepad (if any) is not before the `lost` tag. If not, the customer with the `lost` stamp is excluded from the mailing.



A document presenting the procedure of mailing of reminder letters must have well documented samples of queries used in the processing. This can be a separate document or an annex in the working how-to doc reserved for an advanced user.

3.5. *identity*

This stamp is put when we receive the identity document of the customer. The `what` attribute is equal to the [two-letter ISO code](#) of the country of origin according to the received identification document.

```
<identity on=080701 what=CH />
<identity on=080701 what=FR />
<identity on=080701 what=DE />
```

3.5.1. attributes ✓

	from	on	to	what	why		
identity							

3.5.2. triggered

The stamp is triggered by a reception of the identity document. The `on` attribute must be equal to the date of the reception of the document. We increase the credit limit of the customer upon the reception of the identity document (see section 3.7). Below is an example of a customer notepad.

```
<limit on=080705 from=50 to=100 why=identity />
<identity on=080701 what=CH />
```

```
<contract on=080625 what=esign />
```



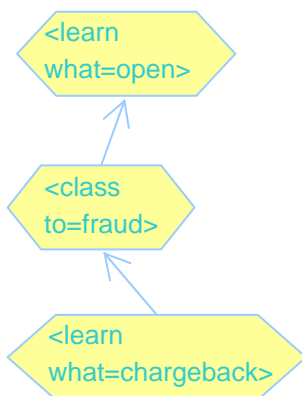
The how-to doc of treatment of identification documents must be updated. Updates may be needed for how-to docs of support@ and billing@ emails.

3.6. learn

This stamp is added when we receive updates on a customer from third parties such as from a collection agency. The following is the presently valid list of values of the `from` attribute: “`authority`”, “`creditreform`”, “`discovery`”, “`firstdata`”, “`linkpoint`”, “`police`”, “`poursuites`”, “`telekurs`”.

The example below shows a three-step case where (a) a chargeback notice is received from a credit card company, (b) the customer is transferred to fraud class, and (c) an investigation is opened by authorities.

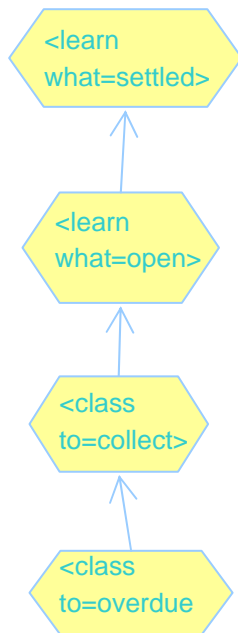
```
<learn on=080730 from=police what=open />
<class on=080716 from=billable to=fraud
why=chargeback />
<learn on=080711 from=firstdata what=chargeback
/>
```



Usually the stamp `class` with an attribute `to=collect` must be followed within a week by a stamp `<learn what=open />`, from a collection agency. The attribute `from` of the stamp `learn` must

match the attribute `why` of the earlier stamp `<class`
`to=collection />`:

```
<learn on=080715 from=creditreform what=open />
<class on=080705 from=overdue to=collect
why=creditreform />
<class on=080625 from=billable to=overdue />
```



Samples of stamps with all considered statuses received from a collection agency are shown below:





```
<learn on=080815 from=creditreform what=settled />
<learn on=080725 from=creditreform what=paying />
<learn on=080725 from=creditreform
what=unsolvable />
<learn on=080725 from=creditreform what=aboard />
<learn on=080725 from=creditreform what=found />
<learn on=080725 from=creditreform what=notfound />
```

The meanings of status keywords are straight: the `open` status indicates that the case is received and is opened by an agency (or another institution). The `notfound` and `aboard` statuses indicate

that the customer is not found or that the customer (according to the research with authorities) is moved to aboard. The **found** status signifies that the agency has found the new postal address of a lost customer, the old being not valid anymore (see the example of section 3.8). The **unsolvable** status is caused by a report of a state agent (Acte de défaut de biens). The **paying** and **settled** statuses indicate on a fact that the customer started partial payments, or have fully settled its debt. The **opposition** is a status received from the debt collection proceedings (Office des poursuites et faillites). For updates from the debt collection proceedings (Office des poursuites et faillites) the attribute **from** must be equal to "**poursuites**":


```
<learn on=080730 from=poursuites what=opposition />
```


3.6.1. attributes ✓

	from	on	to	what	why		
learn	✓			✓			

The attribute **from** specifies always the name of the organization and the attribute **what**, the updated status (refer to the list of valid values in the stamp builder excel file [\[xls\]](#))

3.6.2. triggered 📧

The **learn** stamp is always triggered by an update letter received from a third party. As indicated by the pictogram , the **on** attribute is set to the date (in **yymmdd** format) of the incoming letter.

 A how-to doc must be created for the treatment of all update letters received from collection agency. The how-to doc must contain the scans of all types of letter samples. The employee should be able to treat the incoming flow of update letters and update the notepads of corresponding customers following only the how-to doc.



Except adding the `learn` stamps, all accompanying tasks must be defined in the how-to docs. For example `what=paying` must cause an adjustment of the balance on our side. A `what=settled` must cause an adjustment of the balance and a transfer to a class “`Settled`” (presently named “`Cancelled Paid`”). In both cases the customer must receive from our side an update of status and the reason of the update.

3.7. *limit*

This tag is for keeping the trace of all changes of the credit limit. The default credit limit upon registration of the customer is of CHF50. The first change (increase) of the credit limit usually usually upon a reception of a copy of an identification document:

```
<limit on=080705 from=50 to=100 why=identity />
<identity on=080701 what=CH />
<contract on=080625 what=esign />
```

The credit limit is decreased when the user reaches its third reminder. There is no `why` attribute in such case, because the overdue invoice is the only reason of the credit limit decrease.



The doc accompanying the excel file of 3rd reminders submitted to cash@ must be updated to include the credit limit decreases and the corresponding `limit` stamps in notepads.



The excel file containing the customers reaching the 3rd reminder must contain additional columns containing: in particular the already constructed text of the stamp (with the old and new values of the credit limit and with the current date stamp computed with `=now()` excel function). Joined is an example of an excel file building the credit limit decrease stamp if the entered values are coherent [\[xls\]](#).

```
<limit on=080725 from=200 to=100 />
```

The limit can be increased for three other reasons distinguished by the attribute `why`. The attribute “`wire`” means a proof of the payment (a copy of a wire transfer order) supplied by the user

without the payment being yet received to our account. The value “[user](#)” is for credit limit increase requests approved on our side.

```
<limit on=080725 from=100 to=120 why=wire />
<limit on=080725 from=100 to=130 why=user />
```

The value “[history](#)” is for the credit limit increase suggestions made by periodic algorithm using the status of open invoices.

```
<limit on=080725 from=100 to=200 why=history />
```

[[Algorithm Version as of 2008-07-27](#)] When the customer reached the third overdue payment reminder his credit limit must be decreased to 100 CHF. On the other hand, a periodic task retrieves the customers reached their credit limit, and we increase the limit to the new level if the following three criteria are satisfied: (a) the last due invoice is paid in full; (b) the new credit limit does not exceed the double of the amount of the last invoice; and (c) the new credit limit does not exceed the double of the previous credit limit. The draft of credit-limit increase-decrease rules is submitted for the review, integration and validation

[[080706.overdue.nocredit](#)], [[ch1](#)], [[ch2](#)], [[us1](#)],
[[080708.1804.reminders](#)], [[ch1](#)], [[ch2](#)], [[us1](#)].

[[Algorithm upgrade as of 2008-08-01 suggested by Christian](#)] A single MySQL request retrieves all customers who (a) paid their last due invoice, (b) whose balance reached the 80% of their credit limit threshold, and (c) whose current credit limit is below the double of the last due invoice amount. The query retrieves the customers before expiration of their limit. The same MySQL query computes and suggests the new limit values (equal to the min of the double of the last due invoice, and the double of the current limit). The output is submitted to tasks@ for manual processing. Manual processing includes: change of the credit limit, adding the [limit](#) stamp in the notepad, and an update email to the customer.

[[Algorithm upgrade as of 2008-08-07 by Christian](#)] The default factor of 2 used for computation of the new credit limit (the min of

the double of the last due invoice, and the double of the current limit) shall vary depending on the contract and the presence of the identification document. The factor must be equal to 2 only if the `identity` stamp and one of the following `contract` stamps `<contract what=paper />`, `<contract what=flyer />`, `<contract what=fax />` are present. Otherwise, the factor must be smaller than 2.



The complete how-to doc (of spontaneous limit increases) must be designed. This how-to doc contains the MySQL request, the rules on how to create the excel file of tasks and submit it to tasks@, and the link to another (uploaded) document which must accompany each excel file. The excel file shall automatically generate the text of the `limit` stamp (to be copy-pasted) next to each user, containing the old limit, and the new limit values.



The MySQL request must be well documented in an advanced user section of the how-to doc (even if this section is not necessary for the processing of the tasks).

3.7.1. attributes ✓

	from	on	to	what	why		
<code>class</code>	✓		✓		✓		
<code>limit</code>	✓		✓		✓		
<code>tariff</code>	✓		✓		✓		

3.7.2. triggered

For a particular case, when the user supplies a proof of the payment (e.g. a copy of a wire transfer order) without the payment being yet received on our side, the stamp `limit` is considered as directly triggered by the payment proof. In such case, the value of the attribute `on` is equal to the date on the wire transfer order (and not to the current date).

```
<limit on=080725 from=100 to=120 why=wire />
```



The corresponding section of the how-to doc of treatment of billing@ emails must be modified for including the `<limit why=wire />` stamp, whenever the limit is increased due to a payment proof sent by the user.

With the exception of payment proofs, in all other cases the limit change occur due to internal decisions (related to the status of the open invoices) and are not triggered by external events. For the internal decisions, the value of the attribute `on` is the date when the change of the credit limit is carried out.

3.8. *lost*

This tag is used to indicate the lost postal address. We expect that the `lost` stamp will be often followed by a `found` stamp (see section 3.4) meaning that the user, in reply to our email, sends us a correct postal address.

```
<found on=080625 />
<lost on=080624 what=080606 />
```

The `lost` stamp can be followed by a transfer to a collection agency for the address discovery. The transfer can be carried out either immediately or after the second `lost` stamp. If the user does not have an email address (therefore is already switched to a forced paper invoice mode) the class transfer must occur immediately. The transfer to a collection agency must begin immediately upon the second `lost` stamp if the previous `lost` stamp is at least 30 day old. In such case the transfer must take place even if the user has a valid email address.

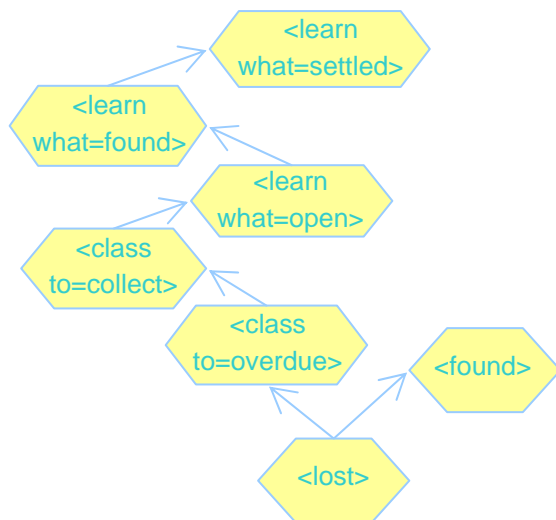
```
<learn on=080815 from=creditreform what=settled />
<learn on=080725 from=creditreform what=found />
<learn on=080715 from=creditreform what=open />
<class on=080705 from=overdue to=collect
why=creditreform />
```

```










<class on=080625 from=billable to=overdue
why=lost />
<lost on=080624 what=080606 />
<remark on=080601 >
Delivery failure to customer@bluewin.ch
</remark>

```

Below is a follow-up diagram of the **lost** stamp:



3.8.1. attributes ✓

	from	on	to	what	why		
<u>found</u>							
lost							

The **what** attribute is equal to the issue date of the letter which is returned by post, while the **on** attribute is the date of the reception of the returned letter (i.e. the date of the postal stamp if any for returned letters, or the date of our own “**Retourné le ...**” stamp which we put on each returned letter).

3.8.2. triggered

Each returned postal mail triggers this tag.



The following must be documented in the how-to doc of the treatment of returned mails: We must check (a) whether a new address is already communicated via email (go through all emails of the customer sent to `billing@`, to `support@`, and to `contracts@`), or (b) whether the address in the billing is already changed (possibly by the customer) and is not the same as the wrong expedition address on the returned envelop.



To be documented in the how-to doc of the treatment of returned mails: If upon the reception of a returned mail the address update is found, the `lost` stamp must be immediately followed by a `found` (section 3.4).



To be documented in the how-to doc of the treatment of returned mails: If upon the reception of a returned mail the address update is not found: The `customer must be blocked`. If the customer does not have an email address (and is therefore on the forced paper invoicing mode according to remarks of section 3.9), the customer must be immediately `moved to "overdue" class` (for being further transferred to collection class for the address discovery with the authorities).

```
<class on=080625 from=billable to=overdue
why=lost />
<lost on=080624 what=080606 />
```



To be documented in the how-to doc of the treatment of returned mails: If upon the reception of a returned mail the address update is not found and if customer has a valid email address, `we inform` that the account is blocked and `we require a new postal address` ASAP. But if the current `lost` stamp is already preceded by a 30-day old `lost` stamp, the customer is immediately `moved to "overdue" class` (for the transferred to collection). The customer receives an `update of status via email`.

As soon as the customer sends us its new address following the procedures of section 3.4, we enter the new address into the billing, we unblock the account, and we add the tag `found`.

3.8.3. searchable 🔍

At every recurrent mailing (for reminders and for paper invoices) we must exclude from the mailing all customers with bad postal addresses. For this purpose, a query retrieves all customers with `lost` tags except they have a `found` tag located on top of the

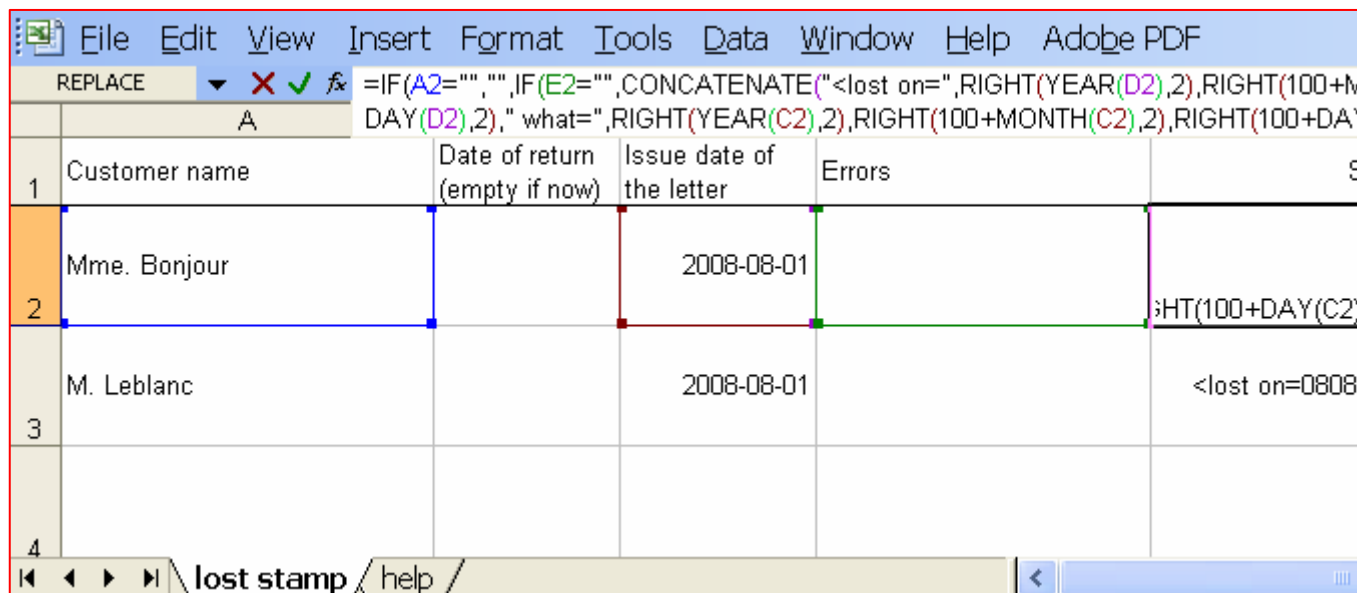
lost tag (i.e. chronologically after the **lost** tag as discussed in section 1).



The query must be documented in the advanced section of the mailing how-to doc.

3.8.4. Safe stamp building

Making sure that the stamps have no syntax errors is especially important for stamps used in database queries. Joined is an excel file for generating **lost** stamps out of the date values entered in excel sheet's input cells [xls]. Such file can be used by support employees for processing the piles of returned mails.



Below is the formula which creates the **lost** stamp:

```
=IF(A2=""," ",IF(E2=" ",CONCATENATE(" <lost on=" ,
RIGHT(YEAR(D2),2),RIGHT(100+MONTH(D2),2),RIGHT(10
0+DAY(D2),2), " what=" ,
RIGHT(YEAR(C2),2),RIGHT(100+MONTH(C2),2),RIGHT(10
0+DAY(C2),2), " />" )," ") )
```

The excel sheet controls also the correctness of the entered data. The stamp is created only when the entered data does not contain errors.



The excel file [xls] provided in this section must be integrated in the how-to doc of the treatment of returned mails.




3.9. remark

This tag is for entering any free text.

```
<remark on=080601 >
Delivery failure to customer@bluewin.ch
</remark>
```

The remark of the above example is entered due to an email error.

3.9.1. attributes

	from	on	to	what	why		
remark							

The following subsection describes the procedure carried out when the email address is lost.

3.9.2. Lost email address

When we receive a delivery failure message for the email address of the customer, we use a tag `remark` for noting the old email address (since the error could be not permanent and the email address may be still useful). The email address is removed from the customer interface. The `billing@` address is written instead. A subscription of 2 CHF/month for paper invoicing is enforced. The customer must receive the invoice (not delivered via email) in a paper form with an update: (a) the email was not delivered, (b) the address is deleted from the system, (c) your invoices will be sent via post, and (d) printing and mailing will cost you 2 CHF/month. If we encounter a `lost` tag, it means that the postal address is wrong as well. Then the customer must be immediately moved to the “`overdue`” class for a transfer to a collection agency for the postal address discovery.



The how-to doc of the treatment of email delivery failures must be designed.

3.10. return














This stamp is put when we receive back from a customer a telephone or an adapter.

```
<return on=080725 what=c450ip why=fault />
<return on=080725 what=pap2t why=trial />
<return on=080725 what=pap2t why=change />
<return on=080725 what=pap2t why=back />
<return on=080725 what=c470ip why=cancel />
```

The following is the list of presently valid values of the **what** attribute: “**asterisk**”, “**budgetone101**”, “**c450ip**”, “**c470ip**”, “**handytone286**”, “**handytone386**”, “**pap2t**”, “**spa3102**”, “**spa921**”, “**spa942**”.

The attribute **why** explains the reason of the return. The reason “**fault**” is used when the customer requires reparation or a replacement, “**trial**” is used when the customer cancels the engagement during the 15 day trial period, “**change**” is used when the customer wishes to change the model, “**back**” is used, when the customer keeps the subscription, but returns the device, “**cancel**” is used when the customer cancels the contract and returns a functional device without paying for it (even if the 2 month period of the right-of-return is expired we may accept to take back the device with a full or partial refund, depending on the condition of the device).

3.10.1. attributes ✓

	from	on	to	what	why		
return							
<u>ship</u>							
<u>supply</u>							

3.10.2. triggered

This stamp is triggered by a returned device. The `on` attribute is equal to the date of expedition according to the postal stamp (otherwise, if there is no postal stamp, the date of our own received-on stamp must be used). The account of the customer must be refunded by the value of the device.



The how-to doc of the treatment of returned devices must be designed or updated.

3.11. ship

This stamp must be put always when we send a device to the customer.

```
<ship on=080725 what=pap2t why=repair />
<ship on=080727 what=c450ip why=order />
```

The `what` attribute indicates on the model of the device. The following is the list of valid values of the `what` attribute: “`asterisk`”, “`budgetone101`”, “`c450ip`”, “`c470ip`”, “`handytone286`”, “`handytone386`”, “`pap2t`”, “`spa3102`”, “`spa921`”, “`spa942`”.

The `why` attribute indicates on whether the device is bought (and therefore the account is charged), or whether the device is repaired or is a replacement of a faulty one (and the customer account is not affected).



The how-to doc of treatment of orders must be updated such that the ship stamp is added as soon as we send a device to a new customer.

When there is a change of the model, the `why` attribute of the `ship` stamp must be still equal to “`order`”. The account is refunded when receiving the old functional model and is charged when shipping a new model.

```
<ship on=080727 what=c450ip why=order />
<return on=080725 what=pap2t why=change />
```



The how-to doc of device replacements for model changes (with a refund and charge) and for reparations (without refund or charge) must be designed and/or updated.

3.11.1. attributes ✓

	from	on	to	what	why		
return							
ship							
supply							

3.11.2. searchable 🔍

The stamp [ship](#) is used in recurrent database queries of monthly statistics on numbers of subscriptions with and without devices. These statistics relay also on stamp [contract](#) (see section 3.3) and stamp [until](#) (see section 3.15).

	from	on	to	what	why		
contract							
ship							
until							

Section 4.1 shows the format of the customer acquisition chart and an example of the treatment of the input data with an excel file [\[xls\]](#).

Less frequently, the stamp [ship](#), together with the stamp [return](#), is used for evaluating the priority of further processing of the customers in the class “[overdue](#)” (e.g. of transfer to “[collection](#)”). The case of a customer who never made payments but also never made calls may escalate slowly. But when such customer has the [ship](#) stamp without a [return](#) stamp the escalation must be fast.


3.12. supply

This stamp is put in the customer notepad, when we send a preconfigured phone to shops (see section 3.11 for the list of values of the **what** attribute). The following is the list of presently valid values for the attribute **to**: “agepoly”, “ams”, “darty”, “emobiles”, “mediamarkt_crissier”, “mediamarkt_meyrin”, “steg”.











If the phone is sold and the customer sends us a signed flyer-contract, the stamp **supply** shall be followed by two stamps **contract** and **class**, and possibly also by the third stamp **tariff**.

```
<tariff on=080829 from=business to=private
why=contract />
<class on=080829 from=shop to=billable
why=contract />
<contract on=080825 what=flyer />
<supply on=080727 what=pap2t to=mediamarkt />
```

The accounts of telephones supplied to shops use the **business** tariff (following the billing terminology, the business product does not have a 9 CHF maintenance fee and the monthly fee is charged via a separate subscription of 9 CHF).

 The supply tag must be added in the notepads of all accounts before the shipment to shops.

3.12.1. attributes ✓

	from	on	to	what	why		
<u>return</u>					✓		
<u>ship</u>					✓		
supply			✓				

3.13. tariff

This stamp must be added each time the billing product of the customer (containing the tariff) is changed. The value of the **why** attribute can be “**user**”, if the tariff change is requested by the customer, can be “**contract**”, if the tariff is changed upon the reception of a flyer-contract accompanying a phone bought in a shop, and can be “**abuse**”, if the tariff change is enforced upon an abuse detection.

```
<tariff on=080829 from=business to=private
why=contract />
<tariff on=080725 from=private to=business
why=abuse />
<tariff on=080725 from=business to=private
why=user />
<tariff on=080725 from=private to=prepaid
why=user />
```

Tariffs are changed also in scope of the overdue reminders. As soon as user reaches the second level of overdue reminders, the free destinations become chargeable by applying the business rates. When all due invoices are paid, the tariff is changed back to “**private**”.


```
<tariff on=080725 from=business to=private
why=nooverdue />
<tariff on=080725 from=private to=business
why=overdue />
```




The doc accompanying the excel file of customers who reached the 2nd level of reminders must be updated such that the tariff is changed.










The how-to doc of treatment of support@ and billing@ emails must be updated, whenever a tariff change is involved.

 Treatment of the flyer-contracts must be updated for requiring the **tariff** stamp.

 Periodic task must be defined for moving customers from **business** tariff back to **private** if all due invoices are paid.

3.13.1. attributes ✓






	from	on	to	what	why		
class	✓		✓		✓		
limit	✓	 	✓		✓		
tariff	✓		✓		✓		

3.14. undue

This tag cancels all reminders and all fees to be issued due to non-payment of an invoice of a specified date. The issue date of the invoice in question is given in the attribute **what**. The effect of the stamp applies to the specified invoice and to all its predecessors. **The balance is not cancelled**. If the invoice is not paid, its balance is transferred to the next month, and the reminders and corresponding fees will apply to the invoice of the next month in case its payment is also delayed. The attribute **on** indicates the date, when the **undue** stamp is put.

```
<undue on=080725 what=080601 />
```


3.14.1. attributes ✓

	from	on	to	what	why		
undue							

3.14.2. searchable 🔍

This tag is considered when generating the overdue reminders. By a proper choice of the date of the attribute **what** applied to a given

customer, one can reset the level of overdue reminders from 4th to 1st and, if justified, restart the reminding cycle.

 The procedure of overdue reminders must be updated. The MySQL queries retrieving the undue stamp must be well documented in an advanced user section.











3.15. *until*

This tag is added to indicate the expiration of the contract. The **on** attribute is equal to the date of the demand, and the **what** attribute is equal to the date, until which the service will be continued.

```
<until on=090501 what=090831 />
<tariff on=080829 from=business to=private
why=contract />
<class on=080829 from=shop to=billable
why=contract />
<contract on=080825 what=flyer />
<supply on=080727 what=pap2t to=mediamarkt />
```

3.15.1. attributes ✓


For properly computing the value of the **what** attribute of the **until** stamp, the presence of the **contract** stamp (see section 3.3) is very helpful.


	from	on	to	what	why		
contract							
until							


3.15.2. triggered

This tag is triggered by the cancellation letter of the customer. The **on** attribute corresponds to the date of the cancellation letter of the customer. The **what** attribute corresponds to a valid date of

expiration of the contract (the closest valid date following the date specified by the customer).

 The account of the customer must be programmed to expire on the expiration date of the **what** attribute. If the **contract** stamp is missing, the **contract** stamp must be added retroactively with a correct value of the **on** attribute, corresponding to the date of the signature of the contract. The monthly fee of 9 CHF is computed until the end of the service and is charged at once, manually. The subscription which cannot be programmed to discontinue in the future, is stopped. The user is informed that the monthly fees are charged in one go.












 All currently pending expirations must be tagged with the **until** stamp.

 All cancellations which already took place starting from 2008-01-01 or starting from 2008Q2 must be also tagged with **until** stamps.

 The how-to doc of cancellation requests must be updated.

3.15.3. searchable



The **until** stamp is used for recurrent customer acquisitions and movements statistics. Customer movements and acquisitions are represented by the following four key numbers: (a) the monthly numbers of cancellation demands (retrieved from the **on** attribute of the **until** stamps), (b) the number of already expired contracts (retrieved from the **what** attribute of the **until** stamps), (c) the number of new contracts (from the **on** attribute of the **contract** stamps), and (d) of the number of purchased shipments (from the **on** attribute of the **ship** stamps with **why=order**).













	from	on	to	what	why		
contract							
ship							
until							

Section 4.1 shows the format of the customer acquisition chart and an example of the treatment of the input data with an excel file [xls].

We do not need to search the **until** stamp during the mailing of reminder letter. The mailing process of reminder letters, for a proper choice of the text, needs only to rely on the current status of the line (up or down). The four reasons why the line can be down are queried from the billing database (without relying on stamps): (a) the account is blocked (usually must never occur), (b) the customer is blocked, (c) the account is expired, and (d) the credit limit is reached.

4. Recurrent monthly queries

Below is the list of stamps, which will be (a) recurrently searched for monthly statistics (i.e. the stamps **contract**, **ship**, and **until**) marked by a pictogram , (b) for monthly mailing (i.e. the stamps **found** and **lost**) marked by a pictogram , and (c) for computing the level of the overdue reminder (the stamp **undue**).

	from	on	to	what	why		
contract							
found							
lost							
ship							
undue							
until							



The samples and the documentation of the MySQL, Excel, or Perl queries are pending. The how-to documents for mailing the reminders and for creating the statistics must contain fully documented samples of all request.

In the following subsection we discuss monthly statistics. We show methods of a treatment of the input data (containing an unordered set of stamps retrieved from the database) using an excel file. A visual layout is suggested for the customer acquisition statistics chart [[xls](#)].

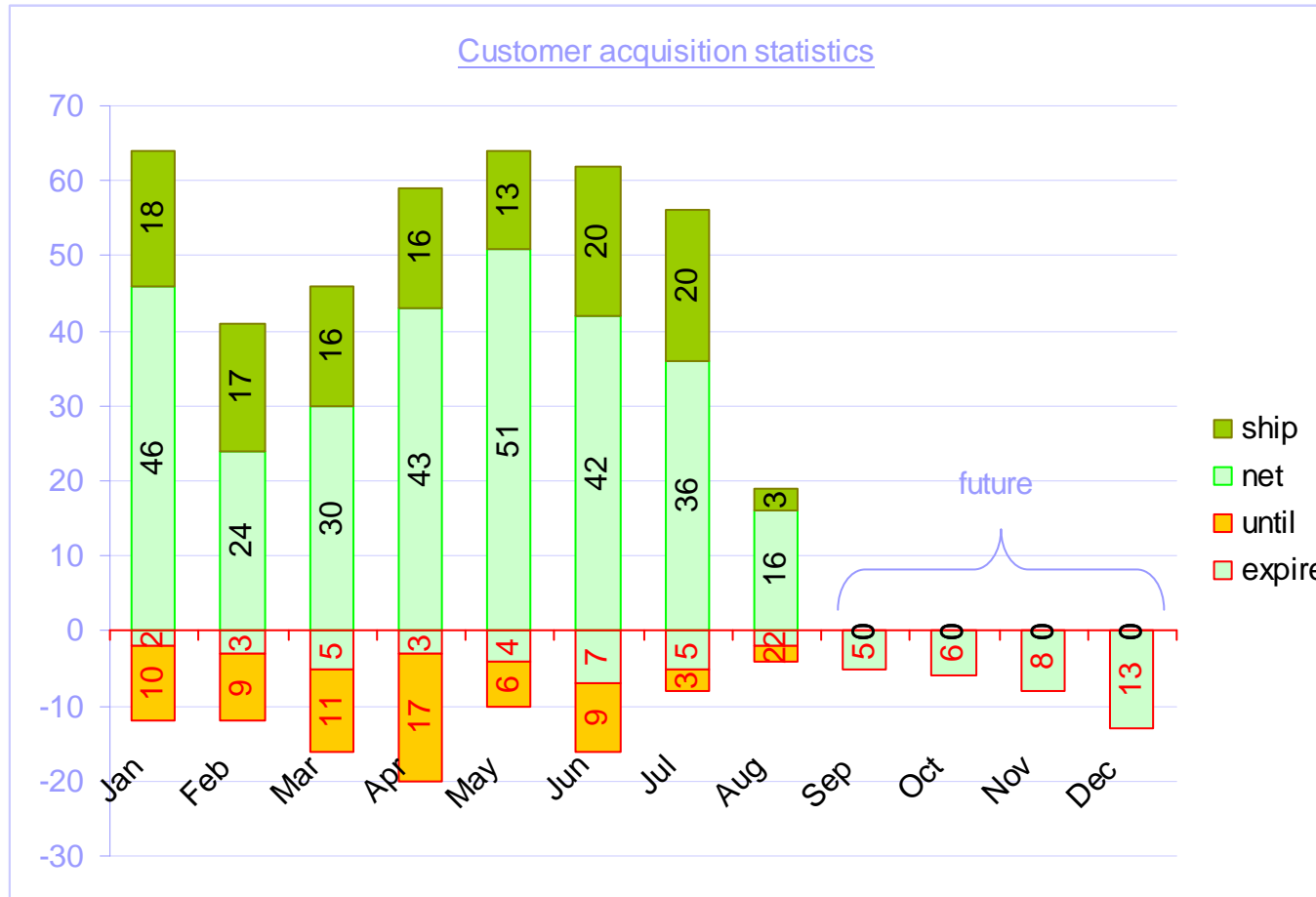
4.1. Customer acquisition chart

The monthly statistics can be computed by analyzing all [contract](#), [ship](#), and [until](#) stamps currently entered into all notepads of the billing system. A MySQL query shall retrieve all concerned stamps of all customers and create an unordered list, as shown below.

Input list of all billing stamps

```
<until on=080216 what=090430 />
<ship on=080710 what=asterisk why=order />
<until on=070829 what=081031 />
<contract on=080602 what=paper />
<ship on=070913 what=spa942 why=order />
<contract on=070916 what=flyer />
<contract on=071016 what=prepaid />
<contract on=080503 what=paper />
<contract on=080722 what=fax />
<contract on=070907 what=esign />
<ship on=071221 what=budgetone101 why=order />
<contract on=080510 what=fax />
<contract on=071116 what=flyer />
<ship on=070928 what=budgetone101 why=order />
```

The joined excel file [\[xls\]](#) builds the monthly statistics based on a randomly generated input list. The randomly generated input list must be replaced by a true set of stamps retrieved from the database.



[\[xls\]](#)

Histogram “[expire](#)” represents the customers whose contract is expired and the service is discontinued during the given month (due to a prior cancellation letter). In the above chart values of this histogram are available in future, due to currently received [until](#) stamps with expiration dates in the future (specified by the [what](#) attributes of the stamps).

Histogram “[until](#)” represents the cancellation requests, received during the given month (the [on](#) attribute of [until](#) stamp), but entering into force in the future.

Histogram “net” is the net number of customers acquired during the given month without devices. The value is equal to the number of subscriptions (signed contracts) minus the number of customers who expired (i.e. definitively disconnected) during the same month, and minus the customers subscribed with devices. The brut number of new customers is the sum of all green areas, i.e.: (a) the number of expired customers (the light gray histogram below the zero line), (b) the net number of acquisitions without devices (the light gray histogram above the zero line), and (c) the number of subscriptions with shipments (the dark gray histogram).

Histogram “ship” is the number of acquired customers with devices.



Web page of statistics is pending. The advanced user section of the page must contain the well documented database queries.

5. Text processing examples

Joined is a text file, containing several samples of each stamp [txt].

5.1. Notepad processing examples with Perl

The following example creates a Perl array of all samples encountered in the notepad. The command prints all elements of the array corresponding to (for example) stamp `class`:

```
$ cat a21.txt | perl -e '@a=<>; $="" ; $_="@a";  
s/[\r\n]//g; @a=split/>/; foreach $_  
(@a){if(/<class/) {s/^.*</</; s/$/>\n/; print;}}'  
<class on=080705 from=overdue to=collect  
why=creditreform />  
...
```

```
<class on=080625 from=billable to=overdue  
why=lost />
```

The following example retrieves the complete list of distinct stamp names from all encountered stamp samples.

```
$ cat a21.txt | perl -e '@a=<>; $="" ; $_="@a";  
s/[\r\n]//g; @a=split/>/; foreach $_  
(@a){if(/<[a-z]/) {/^.*<([a-z]+)/; $_=$1."\n";  
print; }}' | sort | uniq  
abuse  
...  
until
```

The following example retrieves all distinct attribute names from all encountered samples:

```
$ cat a21.txt | perl -e '@a=<>; $="" ; $_="@a";  
s/[\r\n]//g; s/ ([a-z]+)=/ [$1]=/g; @a=split/\]/;  
foreach $_ (@a){if(/\[/) {s/^.*\[/; s/$/\n/;  
print;}}' | sort | uniq  
from  
on  
to  
what  
why
```

The joined log file contains full printouts [\[log\]](#).

5.2. Notepad processing example with excel

Joined is an example of processing the notepad text with Excel formulas. The excel file [\[xls\]](#) has one cell containing the full text of the notepad with all samples [\[txt\]](#). For any given stamp name, the

text processing formulas find, a sample in the text file, and retrieve the values of all its attributes.

Open the excel file [xls] and enter stamp names in the first column. A cell of the same line will display a full sample found in the notepad matching the entered stamp name. The individual values of the attributes of the sample will be also computed and displayed in separate cells of the line.

For each entered stamp name, the line will capture the notepad's first matching sample following the position of the previously captured stamp of the same name (if any).

Below is an excel formula for retrieving the values of individual attributes. The name of the attribute is stored in cell I\$1, and the sample of the stamp is stored in cell \$E2. The formula computes and shows the value of the attribute in question (if it is present).

```
=IF(ISERROR(SEARCH(CONCATENATE(" ",I$1,"=* "),$E2)),"",
MID($E2,SEARCH(CONCATENATE(" ",I$1,"=* "),$E2)+LEN(I$1)+2,
SEARCH(" ",$E2,SEARCH(CONCATENATE(" ",I$1,"=* "),$E2)+1)-
SEARCH(CONCATENATE(" ",I$1,"=* "),$E2)-2-LEN(I$1)))
```

	Full tag found in the notepad	length	The values of all parameters
1			
2	<abuse on=080722 />	19 ok	:080722:::
3	<class on=080705 from=overdue to=collect why=creditreform />	60 Long	overdue:080705:collect::creditreform
4	<contract on=080205 what=paper />	33 ok	:080205::paper:
5	<found on=080625 />	19 ok	:080625:::
6	<identity on=080701 what=CH />	30 ok	:080701::CH:

[xls]

5.2.1. User defined excel function

Below is a body of a user defined macro function "attrval", which replaces a long Excel formula (find references for user defined macro functions in section 7.4).

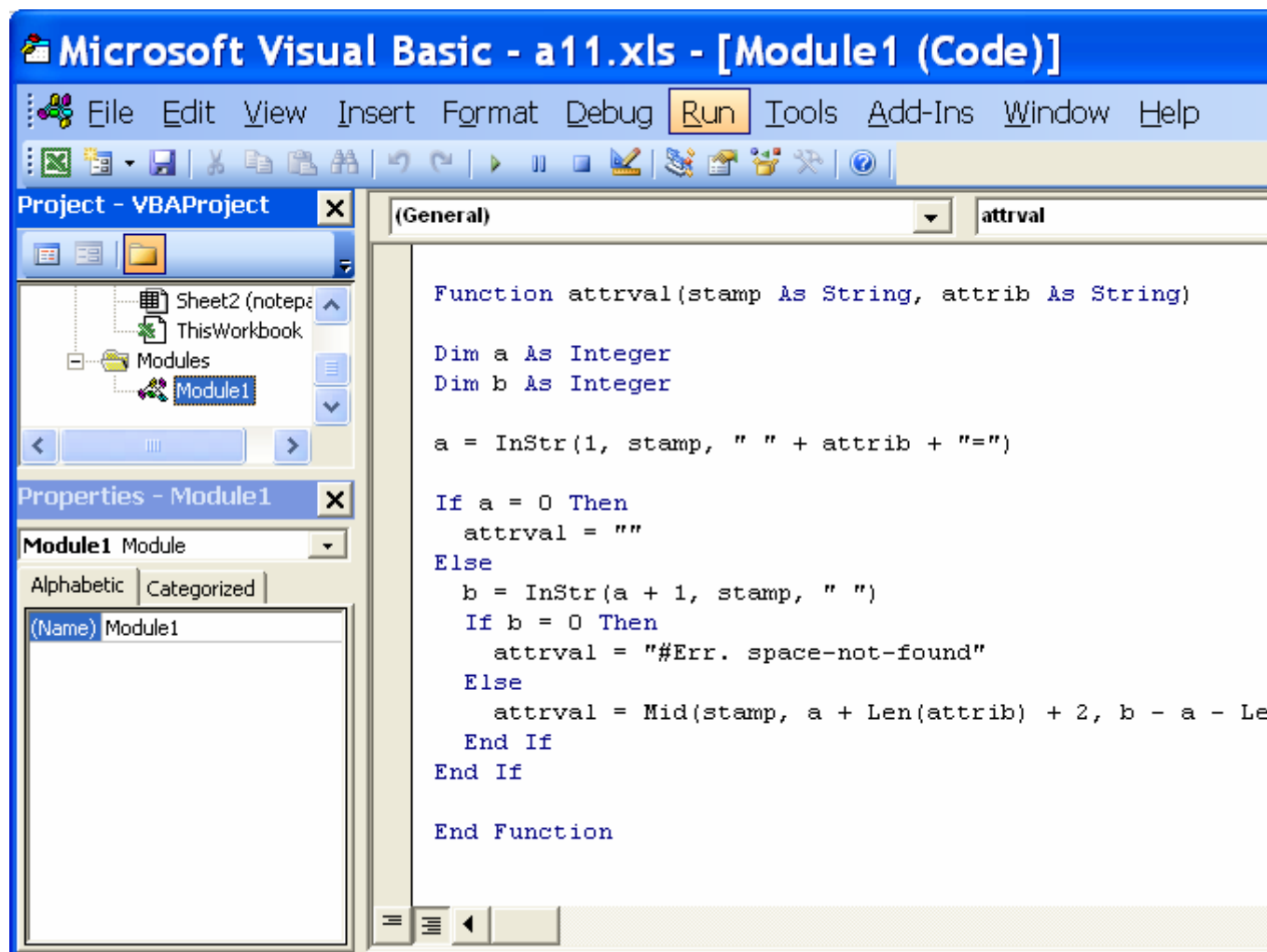
```
Function attrval(stamp As String, attrib As String)

Dim a As Integer
Dim b As Integer

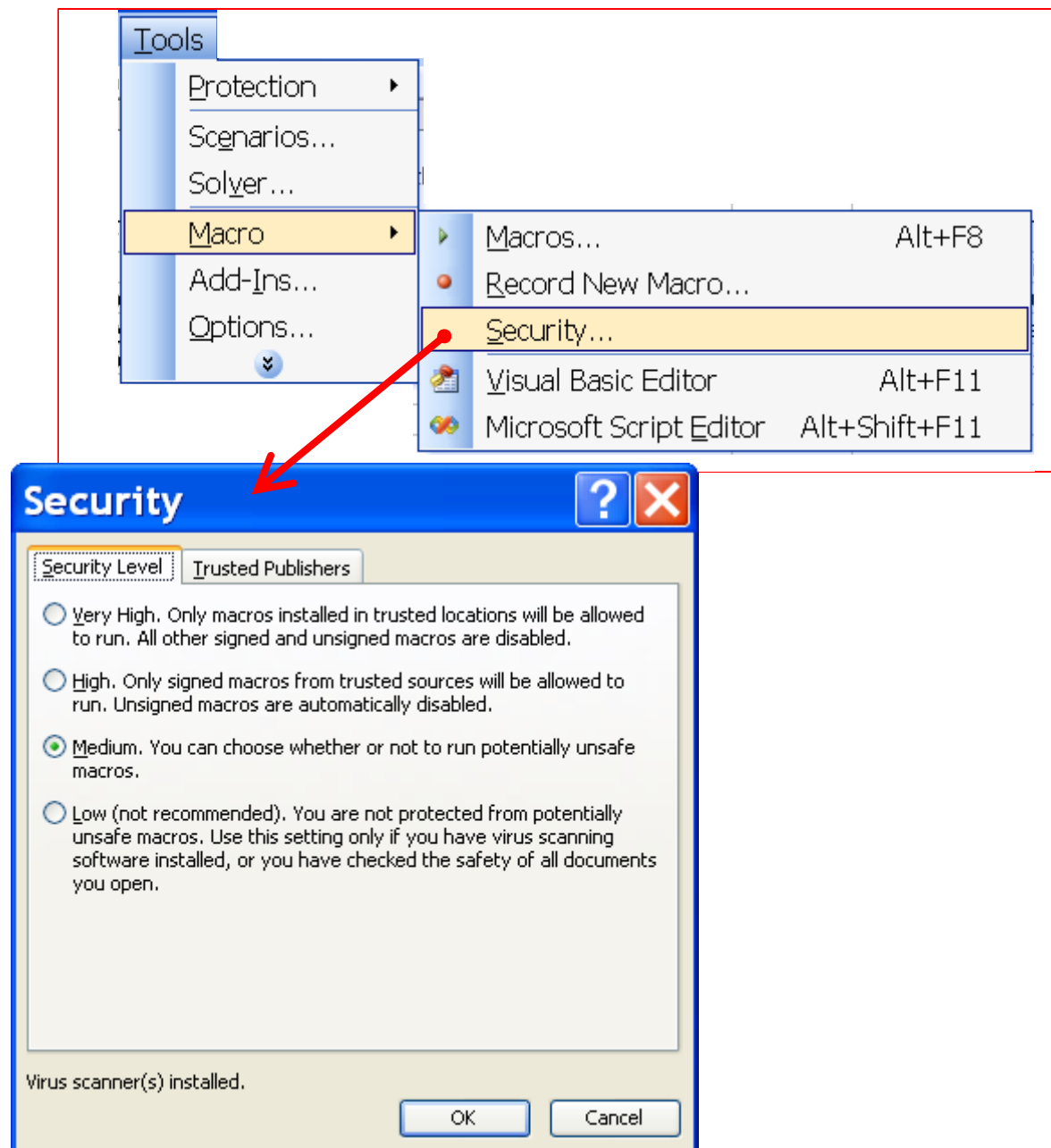
a = InStr(1, stamp, " " + attrib + "=")

If a = 0 Then
    attrval = ""
Else
    b = InStr(a + 1, stamp, " ")
    If b = 0 Then
        attrval = "#Err. space-not-found"
    Else
        attrval = Mid(stamp, a + Len(attrib) + 2, b - a - Len(attrib) - 2)
    End If
End If

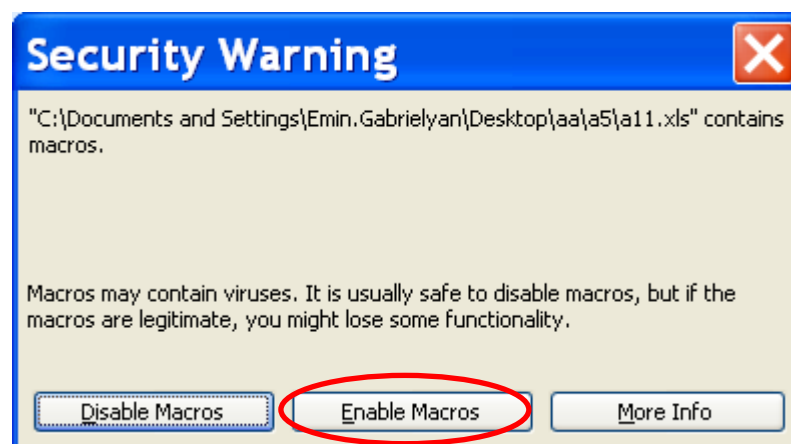
End Function
```

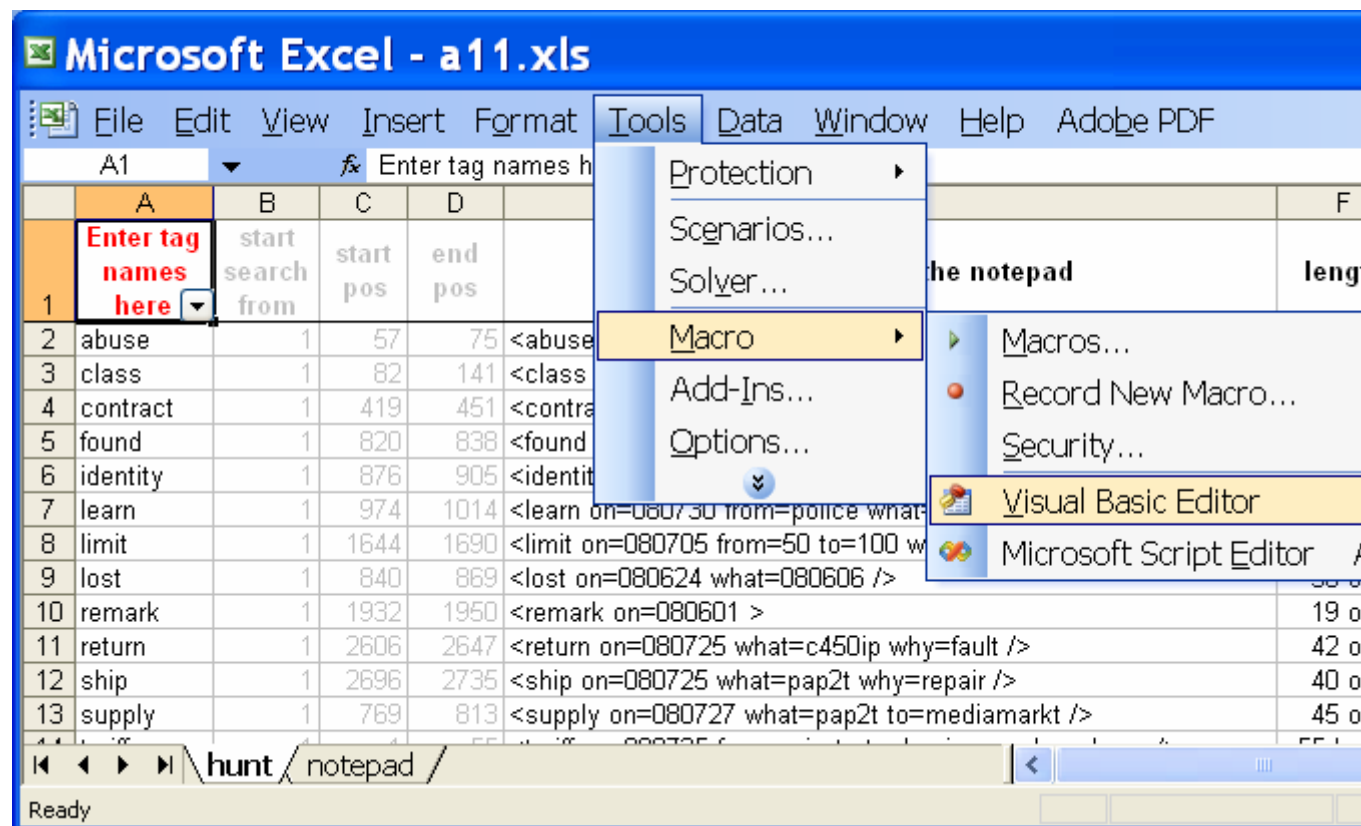
To use the example, change the security of macros to a medium level via **Tools** → **Macro** → **Security** menu:



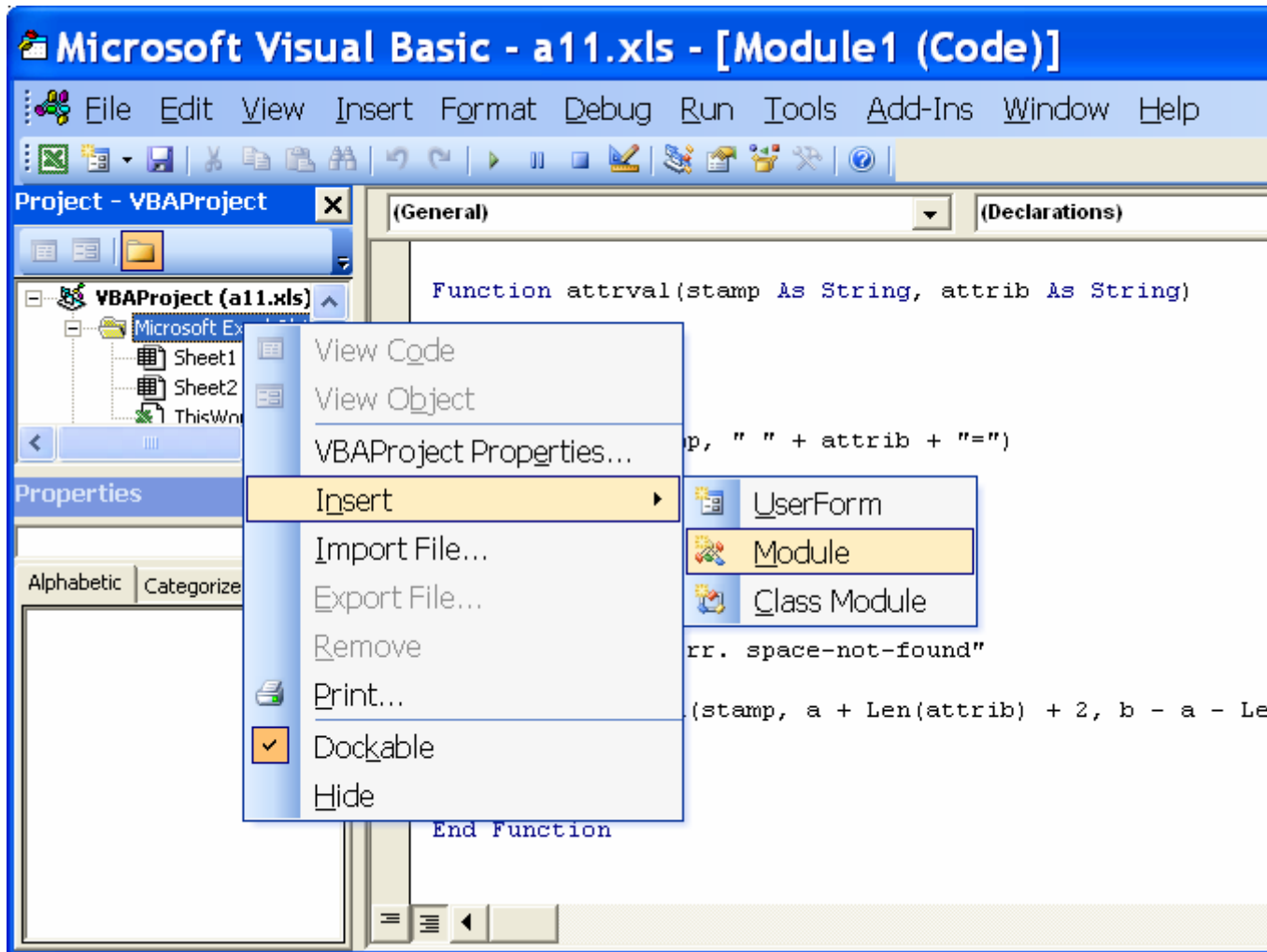
Download the excel file containing the macro [xls]. When opening the excel file, choose enable macros:



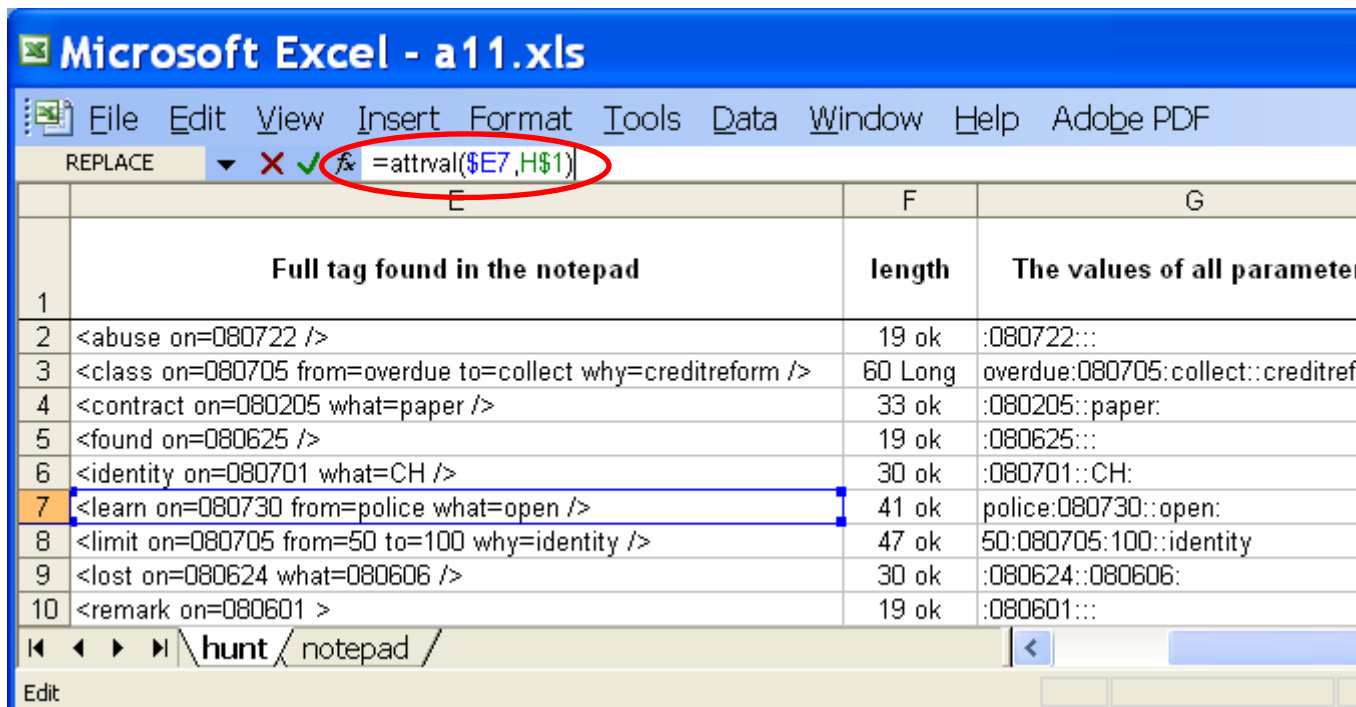
Alternatively, you can download the previous excel file [xls] with long formulas and create the macro yourself by copy-pasting the body of the macro function shown at the beginning of this subsection (5.2.1) into the macro window. Press **Alt-F11** or use menu **Tools** → **Macro** → **Visual Basic Editor**:



Insert a module (using the right-click pop-up menu) and copy-paste the function body into the module:



As soon as you save the module, the macro function can be used from the excel sheet (instead of the long formula):



[xls]

6. Conclusions

We will use customer notepads of billing interface for logging important events and status changes related to individual customers. Such data shall be well formatted and structured so as they can be used in database search queries. For this purpose we defined the following 15 stamps: `abuse`, `class`, `contract`, `found`, `identity`, `learn`, `limit`, `lost`, `remark`, `return`, `ship`, `supply`, `tariff`, `undue`, and `until`. Format of stamps follows the XML specifications. The parameters of each stamp are provided via one or more of the following 5 attributes: `from`, `on`, `to`, `what`, and `why`.

New stamps may be required in future. Stamps `subscribe` and `unsubscribe` may be needed for indicating the beginning and the end of various subscriptions (9 CHF/month for the line, or 2 CHF/month for paper invoices). Stamp `refer` may be needed for keeping a trace on references made by the customer.

7. References

7.1. *This document*

Customer stamps for the billing notepad [[doc](#)], [[pdf](#)], [[htm](#)]:

<http://switzernet.com/public/080803-customer-stamps/>

<http://unappel.ch/public/080803-customer-stamps/>

<http://4z.com/public/080803-customer-stamps/>

7.2. *XML*

<http://4z.com/public/080302-postfinance-xml2doc/>

<http://switzernet.com/public/080302-postfinance-xml2doc/>

<http://unappel.ch/public/080302-postfinance-xml2doc/>

<http://4z.com/public/080302-postfinance-xml2doc/sg4-credit.txt>

<http://en.wikipedia.org/wiki/XML>

7.3. Perl and Excel

<http://www.comp.leeds.ac.uk/Perl/start.html>

<http://www.cpearson.com/excel/ArrayFormulas.aspx>

<http://www.spreadsheetsconverter.com/websamples.htm>

7.4. Excel VBA functions

[http://msdn.microsoft.com/en-us/library/aa297628\(office.10\).aspx](http://msdn.microsoft.com/en-us/library/aa297628(office.10).aspx)

[http://msdn.microsoft.com/en-us/library/aa221602\(office.11\).aspx](http://msdn.microsoft.com/en-us/library/aa221602(office.11).aspx)

http://www.exceltip.com/st/Writing_Your_First_VBA_Function_in_Excel/631.html

7.5. Related to the “why” attribute of the “class” stamp

http://switzernet.com/company/080712-cancelled-due-class/#_Toc203646861

http://unappel.ch/company/080712-cancelled-due-class/#_Toc203646861

http://4z.com/company/080712-cancelled-due-class/#_Toc203646861

7.6. Choice of the names of stamps, attributes, and values

http://en.wikipedia.org/wiki/Grace_period

<http://en.wikipedia.org/wiki/Chargeback>

<http://www.merriam-webster.com/dictionary/undue>

7.7. Office excel file

The stamps will possibly help us to get rid of the local copies of the office files [[email](#)]

7.8. The original TODO task

[080703 Christian, Sona] Create a doc defining the rules of customer tagging. We decided to tag customers using the notepad of the billing interface. The tags are short keywords in square parenthesis. If the customer has tags they must be in the very beginning of the notepad. The following tags are discussed

[invoice post] for customers receiving invoices via post, and [creditreform pending], [creditreform abandon], [creditreform unsolvable], [collection office pending], [collection office opposition], [collection office unsolvable] for debt collection via the Credit Reform company or the Debt Collection Proceedings (office des poursuites). Tasks must be created and submitted to tasks@ for tagging. Information must be retrieved from the data received from Credit Reform. Make tests with MySQL to make sure that we can search in the database according to the tags.

[080704] The tags should be preceded by the date in YYMMDD format, example: [080704 creditreform pending]

[080704] The transfer from one representative to another must be tagged such as: [080704 class cancelled-due]. The MySQL request samples must be provided in the doc.

[080704] Tag business signifies that the customer is switched from private subscription to business, for example due to abuse. Such tag can look like this [080704 tariff business abuse]

[080706] The following tag stamps to be considered as well: [contract] for the date when the contract is signed, it will be needed for calculation of the minimal contractual period; [credit] for keeping a trace on the credit limit increase or decrease; [fraud], [abuse], [identity], [080706 expire on 2011-08-29].

[080707] Another stamp to add in the notepad is [<yymmdd> ship <hardware>] and [<yymmdd> return <hardware>]

[080708] Stamps to add: [blocked], [address error], [email error], and [expire] for scheduled cancellations

[Reference] To-Do page [[ch1](#)], [[ch2](#)], [[us1](#)]

* * *

Copyright © 2008 Switzernet

