TEMENOS™

TEMENOS T24

Security Management System

User Guide

Table of Contents

Introduction

Application Overview

The Security Management System (SMS.) controls who is allowed to use **T24** when they are allowed to use it and to what parts of the System they can have access. It will detect, stop and record any attempt at unauthorised use of the System. S.M.S. can also, if required, record all activities performed by selected Users.

Details of each authorised User are held in the *USER* file. These details include Sign On name, Password, permitted times of use and what parts of T24 may be accessed. Before being allowed to use **T24** each User must be identified to SMS by their Sign On name and Password, all subsequent activity is then checked against their User details before being permitted. For Internet Banking users, the *IB.USER* file uses the same functionality to provide an equal level of security.

Access to applications within **T24** can be controlled by the name of the application, the function they are allowed to perform, Input, Authorisation etc., and can even be controlled by the data itself. E.g., you can allow someone full functional access to the *ACCOUNT* application except, for staff accounts - where staff accounts are identified by the `CATEGORY` field in the *ACCOUNT* record.

SMS operates only within **T24** It is the responsibility of the installation security manager to ensure security of access to the network, the operating system and jBase. For example, ftp access to **T24** files should not be allowed.

Due to the manner in which with Windows NT is administered, a security issue has had to be overcome. Since on traditional UNIX systems the 'Sign-on' to **T24** is a two-stage process, first to the server followed by the **T24** system. Although SMS can only work within **T24,** a utility has been developed to allow the initial server login details to be shielded from the user, effectively automatically performing the server login and taking the user directly to the second **T24** sign on screen. This utility can be used by system administrators in conjunction with the **T24** SMS to provide a higher level of system security.

Important

For ease of implementation, **T24** is delivered with a LOGIN file that allows an exit to a jBase prompt. This should be altered before going live so that the exit is disabled. If the associated question and response is removed **(“START T24 Y/N=”)** then the wait and response will have to be removed from the Desktop or T24 Browser login scripts.

ARC-IB SMS and External users

The information on configuring SMS and External users for the ARC-IB (internet banking) applications is provided in in the ARC-IB User Guide as it is part of the inherent usage of ARC-IB.

USER

For every person who is allowed direct access to **T24** (i.e. by directly keying information into the system, rather than through the Internet Banking Interface), a record must be created on the *USER* file. This contains identification details and specifies precisely which information and facilities are available, the dates and times at which the system may be accessed and other information required for Security Management.

Although explained in more detail later it is important to note that it would be very unusual for every user to have a completely individual profile listing every application and function allowed/disallowed. The more normal practice would be to establish group records for business or departmental privileges and add or remove functionality on the individual user profiles.

But first the basic information and details are shown by usage type.

Identification

These fields specify for each user a unique Sign On Name, Name, Classification ('External' customer or 'Internal' employee of the bank) and Department. In addition, user preferences such as the language in which messages etc. should be displayed can be specified.

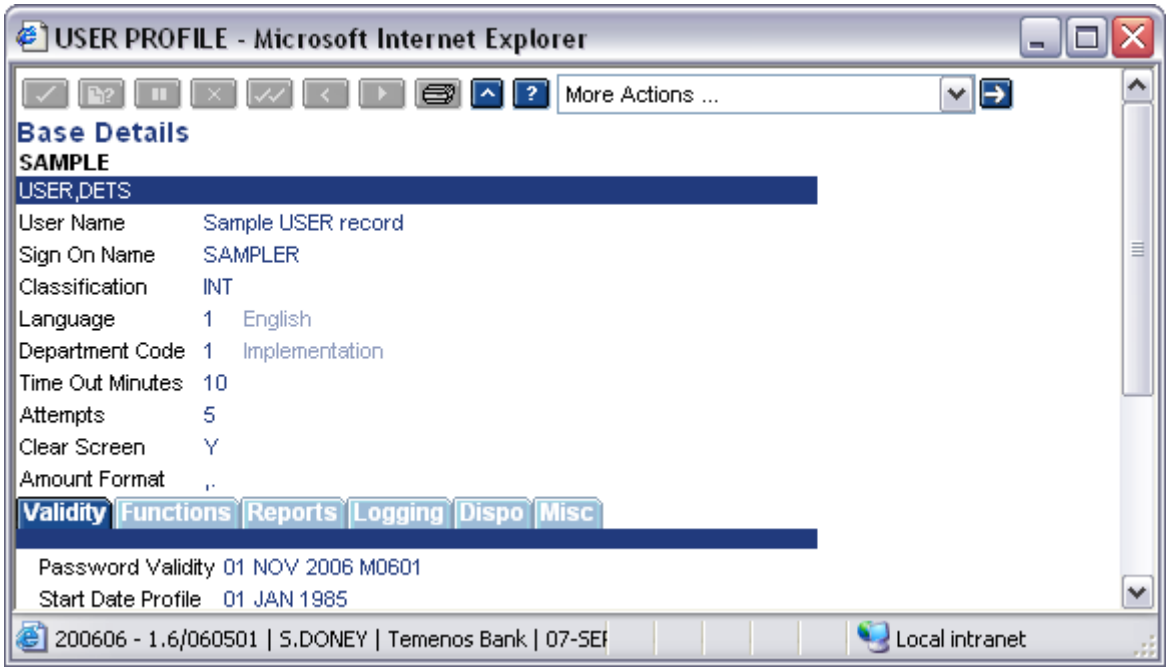


Figure 1 Identity details

Permitted Times Of Use

Details specifying when each User can access the System include Start Date, End Date, Start Time, End Time, the maximum time during which the User may be inactive without being Signed Off automatically. The number of unsuccessful attempts to Sign On allowed before the Password is disabled, the frequency with which the Password must be changed and dates between which the Password should be deactivated.

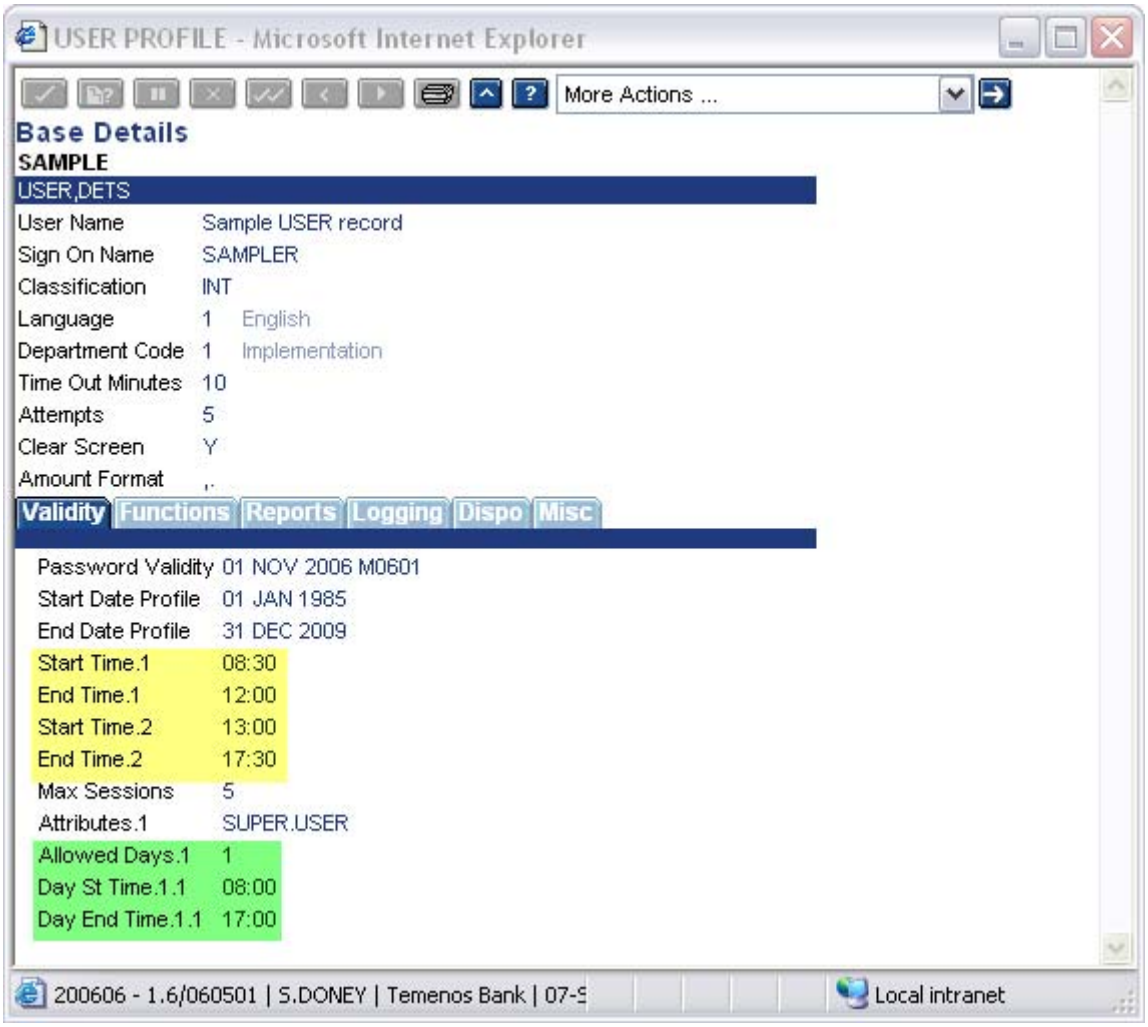


Figure 2 User profile: Permitted times of use

The time specified in the fields `START.TIME` & `END.TIME` applies for all days of the week. Different access times for each day of the week can also be set using the fields `ALLOWED.DAYS`, `ST.DAY.TIME` & `ST.END.TIME`

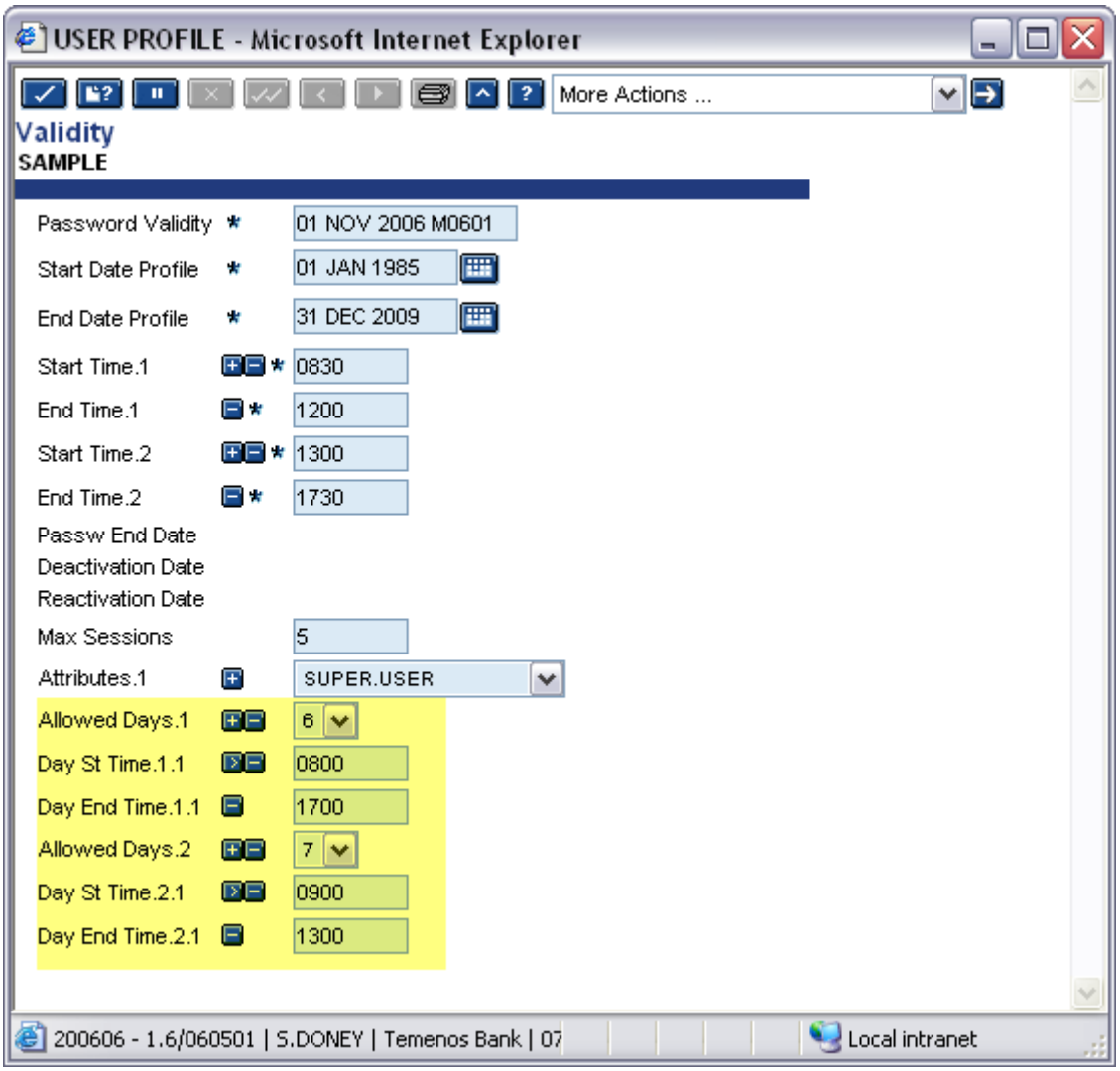


Figure 3 Allowed Days

It should be noted that the days of the week are numbered from 1-7 representing Monday (1) through to Sunday (7).

Permitted Customers and Accounts for external users

Where a user is classified as 'EXTERNAL' (through the CLASSIFICATION field), it is possible to restrict the *CUSTOMER* and *ACCOUNT* details to which they may be allowed access for specific applications. This may be done through the CUSTOMER and ACCOUNT fields.

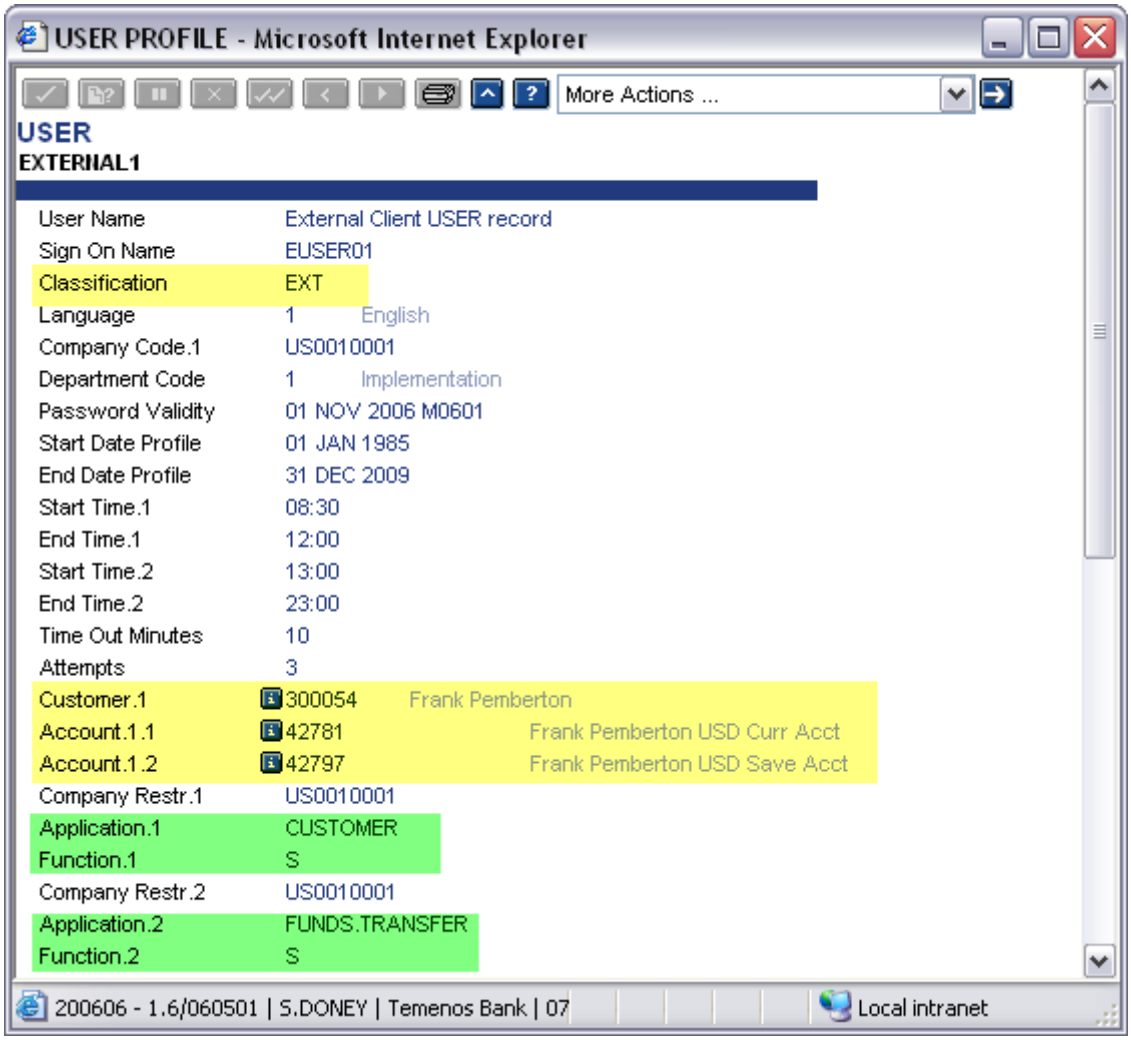


Figure 4 User profile for external user

In the above screenshot, the user is able to use the *FUNDS.TRANSFER* application to see transactions involving his *CUSTOMER* and/or *ACCOUNT* details.

In order to enable this functionality, it is important to note that the application *USER.EXTERNAL.FIELDS* also needs to be set up.

USER.EXTERNAL.FIELDS

This table will allow the user to specify for each **T24** application, which field number(s) identify a 'Customer' or 'Account' field. This must be set up in order to enable SMS settings established for EXTERNAL type users, to limit the user to only seeing the records where their *ACCOUNT/CUSTOMER* ids appear in the field number(s) defined here.

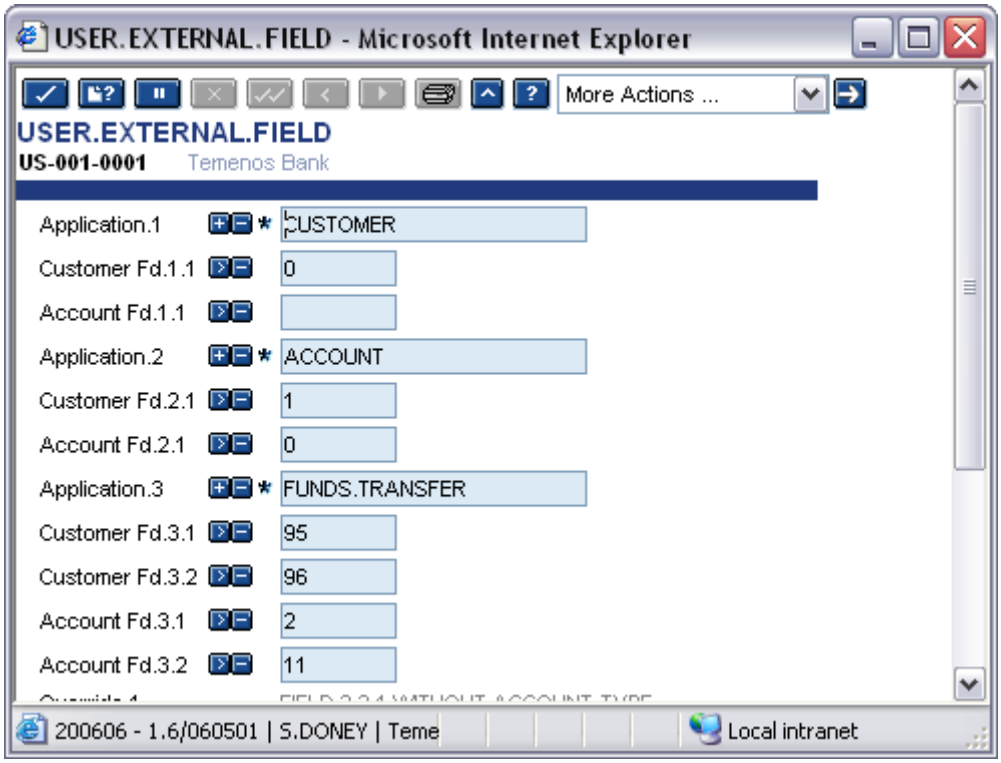


Figure 5 Setting up User external fields

So in the above screenshot an external user, when using the *FUNDS.TRANSFER* application, would only be able to view those contracts where their account number appeared in the `DEBIT.ACCT.NO` field or `CREDIT.ACCT.NO` field.

Permitted Activity

For all Users the information and facilities which may be accessed are specified at 4 levels: COMPANY; APPLICATION (and VERSION), FUNCTION and FIELD so that it is possible to specify precisely which records, belonging to which company, may be accessed and what may be done with them. Up to 999 combinations of Company, Application, Function and Field are allowed.

Access to applications must be granted positively except for *ENQUIRY.SELECT*, which is used to run enquiries.

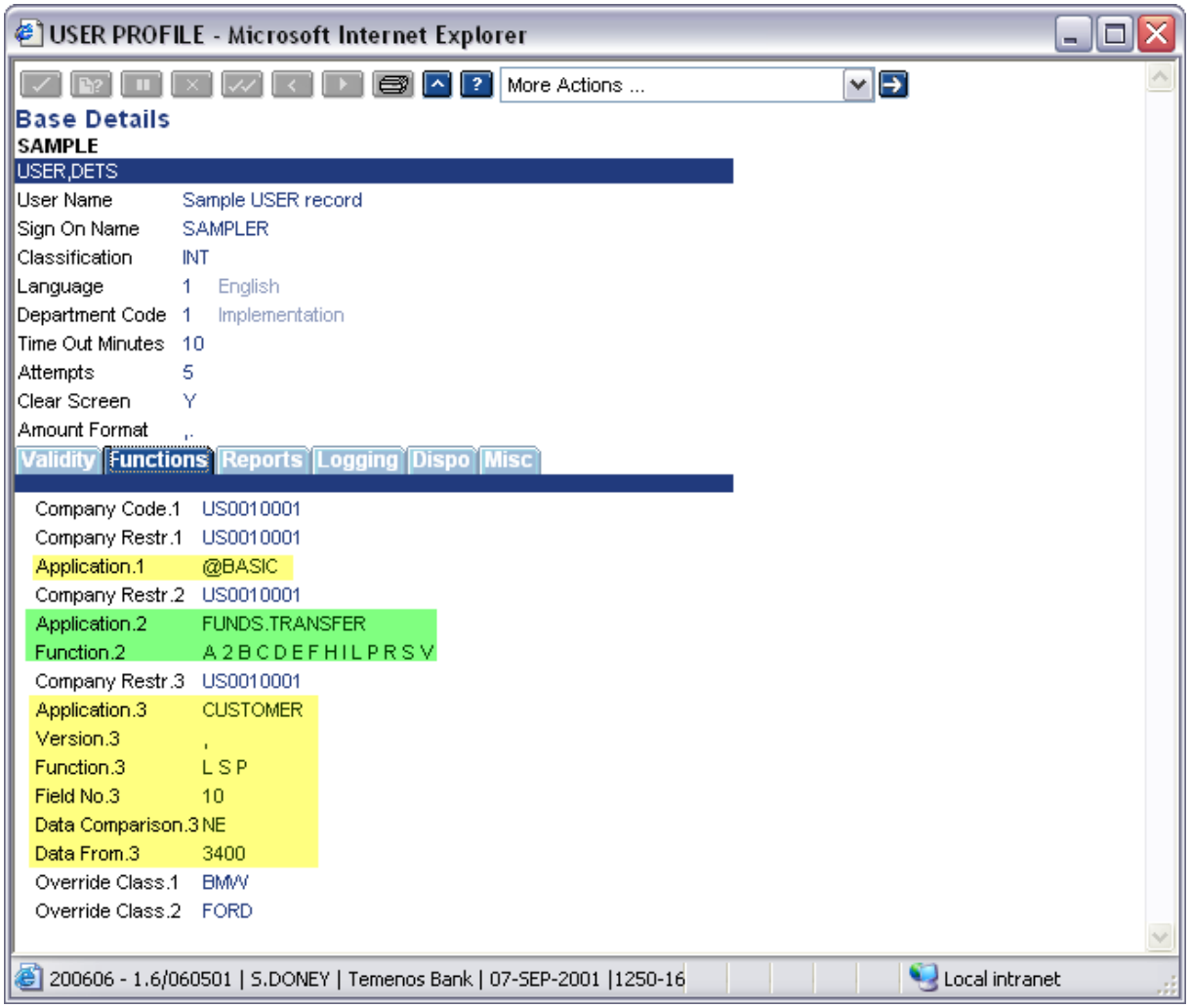


Figure 6 User permitted activity

In the above *USER* record the user is restricted to a few functions on *CUSTOMER* using a *VERSION*; the *FUNDS.TRANSFER* application and those specified in the *USER.SMS.GROUP* record '*BASIC*'.

To restrict the users access to particular data the following can be entered:

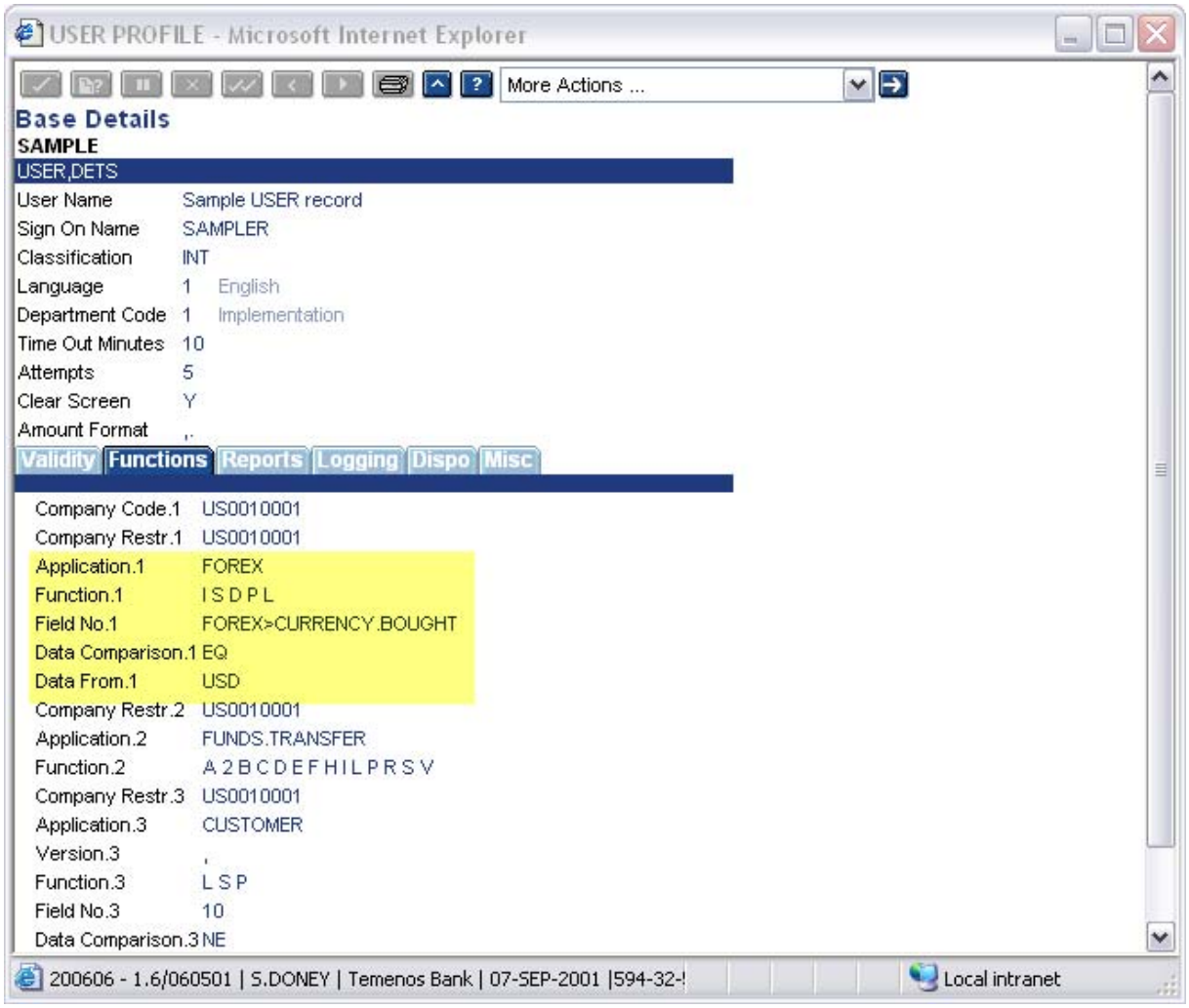


Figure 7 Restrict User's Access to data that can be entered

Note. Field numbers can be used but this would entail keeping them updated whenever a new release adds new fields to the application.

Here the user is further restricted to processing only USD contracts, where USD is the currency bought.

As previously mentioned it is more advantageous to set access rights & restrictions by using *USER.SMS.GROUP* and then linking these to the relevant *USER* records. For example a basic access group could be defined and set on each *USER* record.

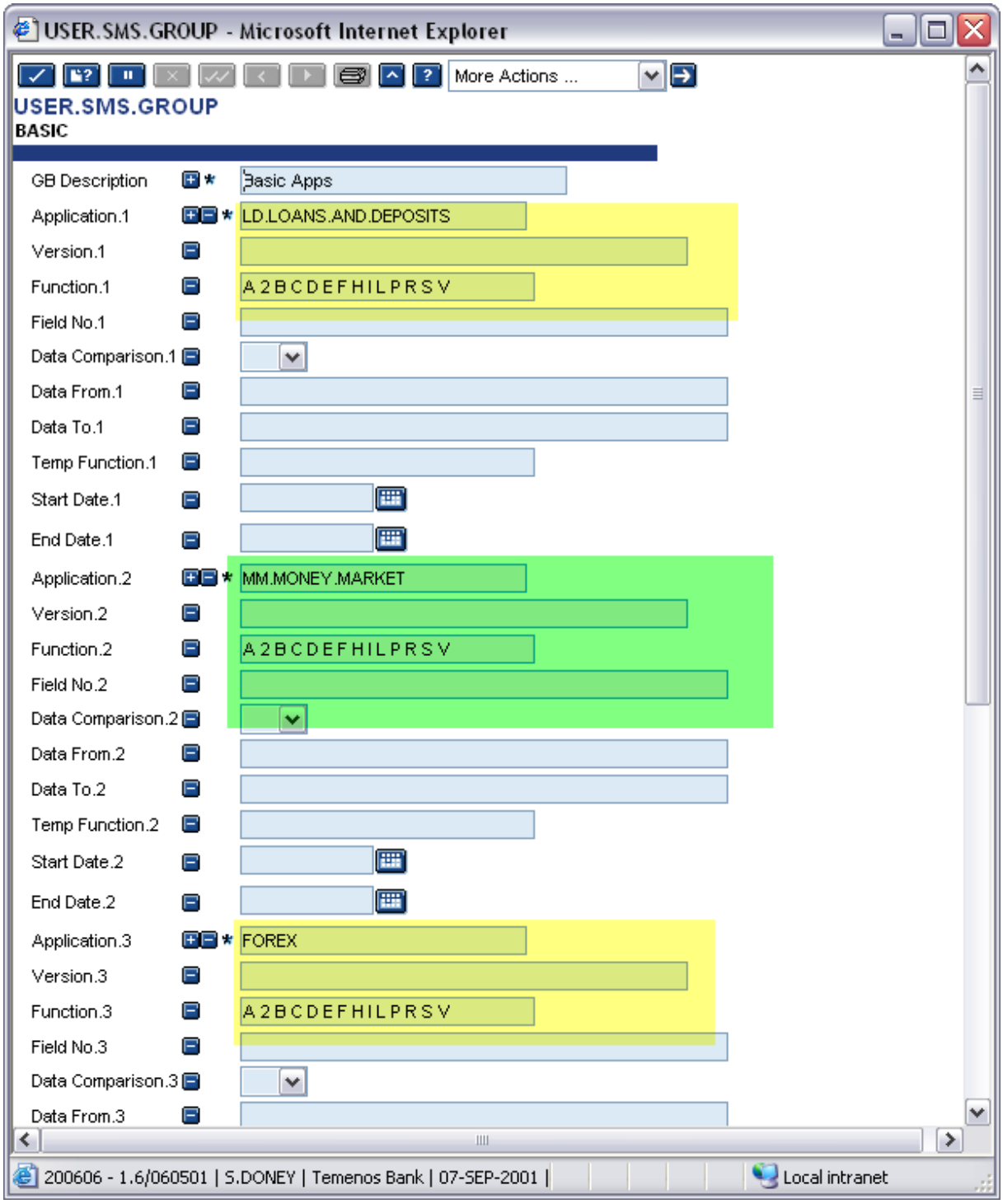


Figure 8 USER.SMS.GROUP profile

Here a group *BASIC* has been defined to enable access to a few basic applications. This can be linked to a *USER* profile by entering "@BASIC" in the `APPLICATION` field e.g.

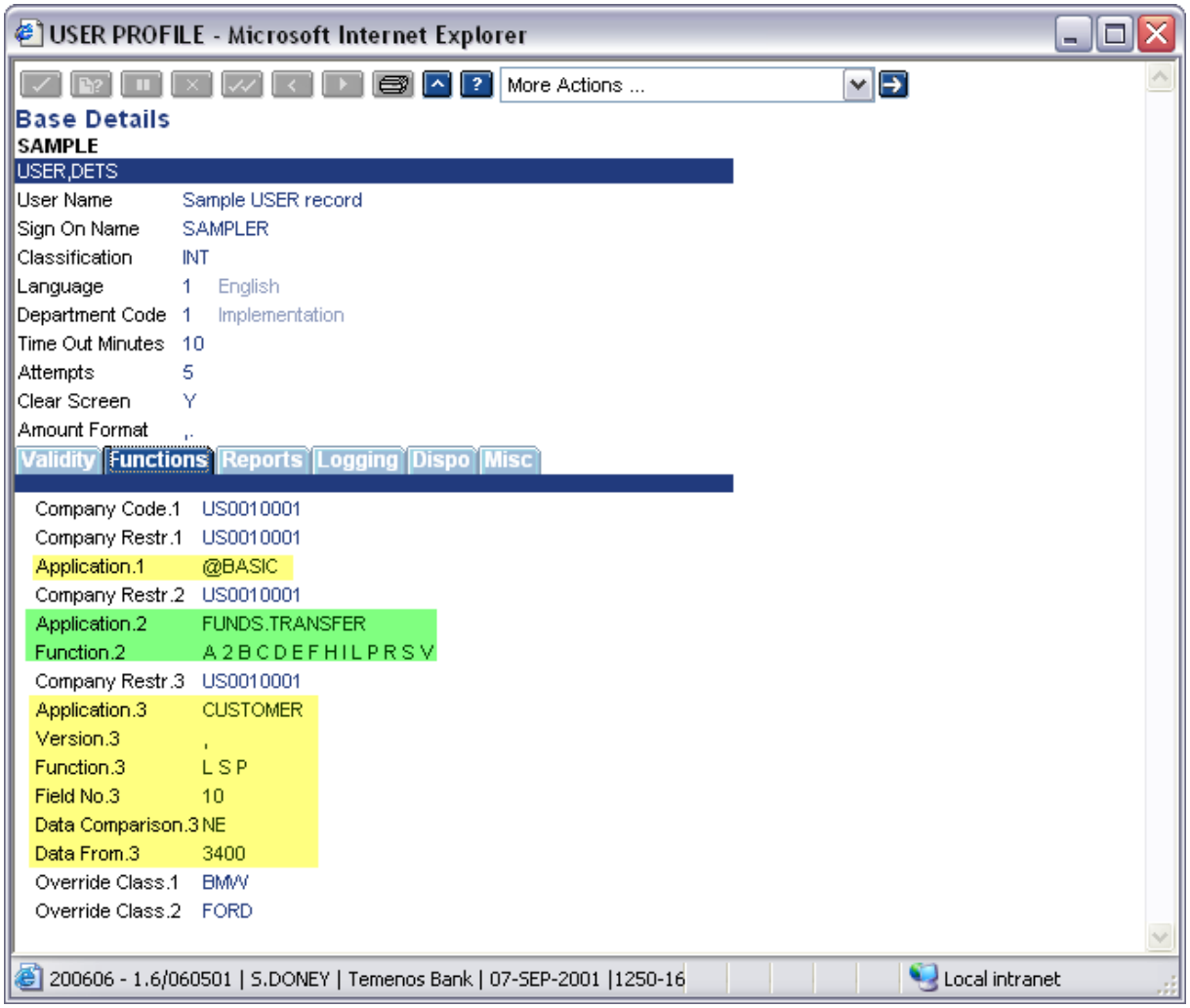


Figure 9 Adding USER.SMS.GROUP to USER record

In a site with a large number of users the use of *USER.SMS.GROUP* is strongly recommended.

All Programs – All Functions

There is an option to set a User record to have access to all the applications in the system (entering ALL.PG in the APPLICATION field) and to have all the allowed functions (by entering ALL in the FUNCTION field which will then expand into the standard functions).

Unusual functions are not included in the ALL category, for example the 'Q' function is reserved for auditors and would not normally be used by anyone else.

In standard mode (exclusive) **T24** will check the *USER* applications and functions and build a list of the permitted applications & functions in a logical manner. What this means is that if a *USER* is given the first option to use ALL.PG in the applications field and ALL in the functions field and then has another multi-value set listing *CUSTOMER* with just the 'L S P' functions then the restriction on *CUSTOMER* will be respected and only the functions listed will be allowed. This method can be used to give power users a more manageable profile by just restricting any sensitive applications.

There is another option that can be set (the ALL.PG.INC field on *SPF*) which is by default blank and allows the standard mode mentioned above to continue. However, if is set then it has a very specific change in the way the ALL.PG setting works in combination with other listed applications. This non-standard mode (inclusive) will add any functions listed under ALL.PG to any applications listed individually as well as any they have listed already. For example if ALL.PG was given passive functions such as 'L S P' and *CUSTOMER* was set as 'I D A' then the user would be allowed to use functions 'L S P I D A' on *CUSTOMER*

So the use of the ALL.PG, ALL & ALL.PG.INC settings should be used with due consideration to the effect they can have.

ENQUIRY.SELECT

Access to *ENQUIRY.SELECT* is granted without restriction if the application does **not** occur in the *USER* record or in an associated *USER.SMS.GROUP*. If you want to restrict a user to running particular enquiries, then you must grant access to the application *ENQUIRY.SELECT* and restrict it appropriately, perhaps by the enquiry name or part of the description field.

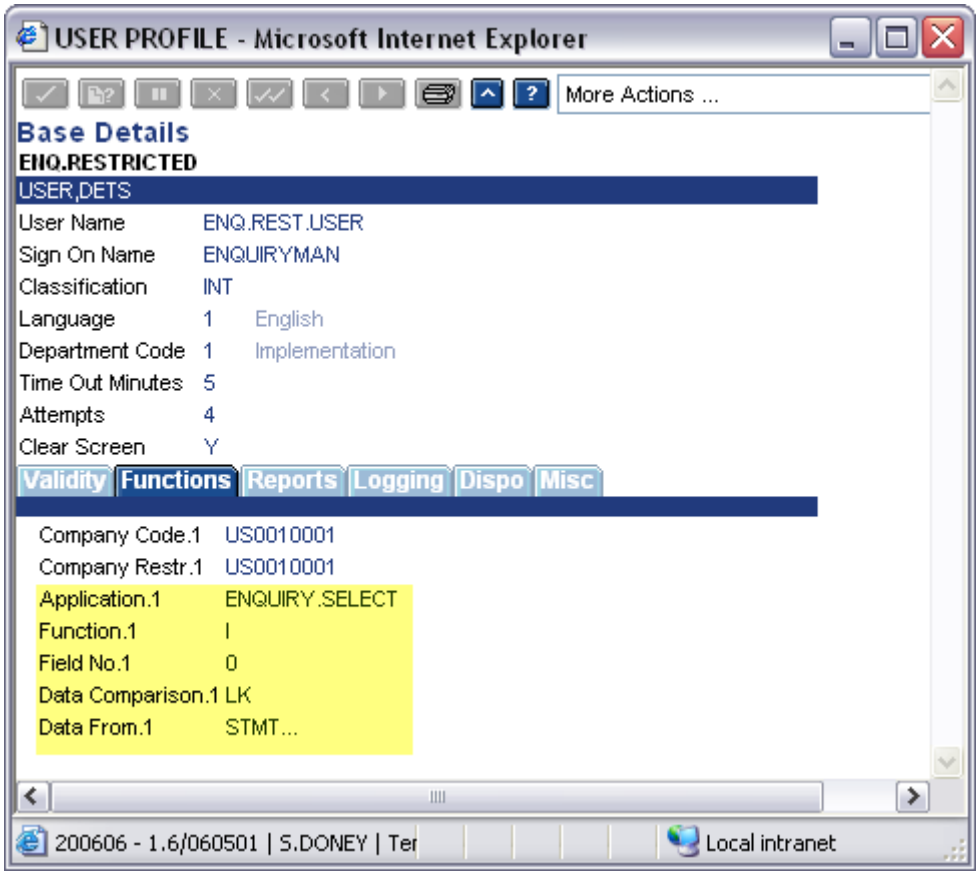


Figure 10 Restrict a User to certain Enquiries

In the above screenshot the user will only be able to use enquiries whose name starts with STMT.

ENQUIRY

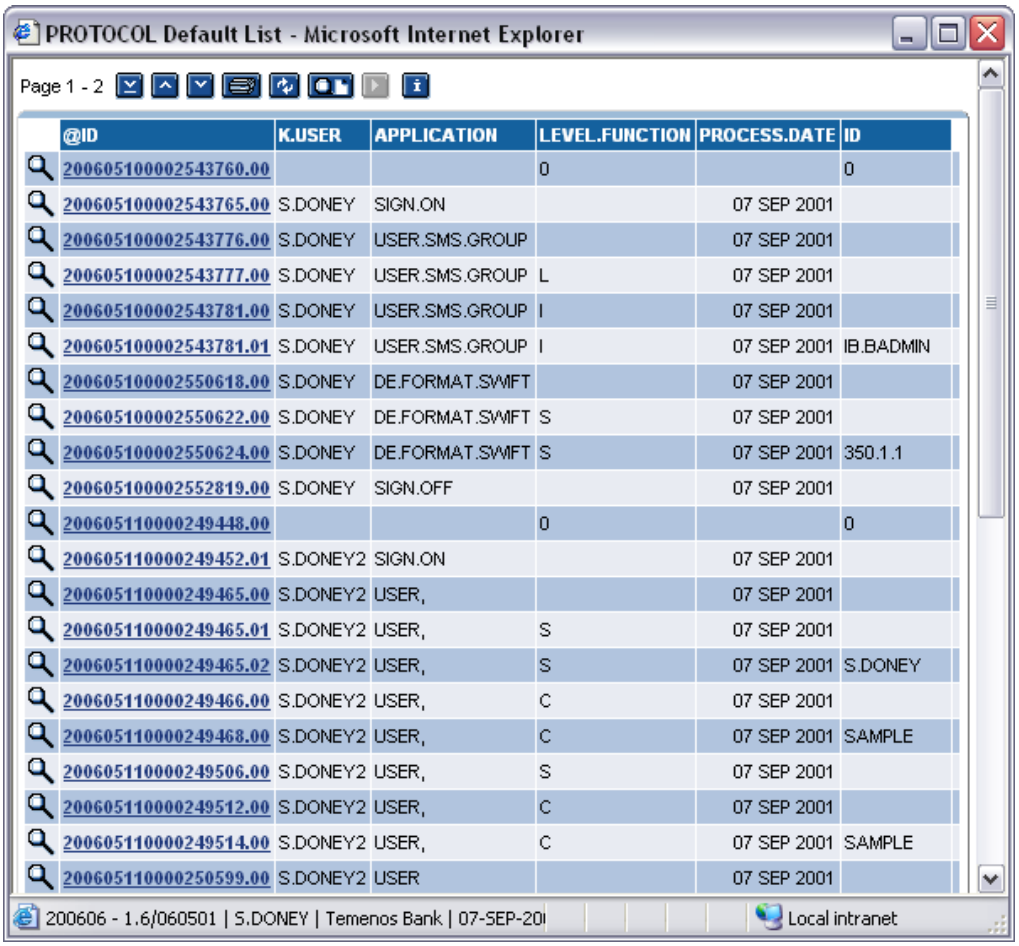
The SMS restrictions on a *USER* that refer to a file or application can be applied when an *ENQUIRY* is run. By default the enquiry returns data **without** reference to SMS but the fields `SMS.APPLICATION`, `SMS.ID` and `SMS.ABORT` can be set to use the restrictions on the data. This is covered in the ENQUIRY section of the User Guides.

Activity Logging

You can specify on the *USER* profile the level of user activity you wish to record. This can be at:

SIGN.ON.OFF	signing on and off
SECURITY.MGMT.L	running an SMS application
APPLICATION.LOG	running any application
FUNCTION.ID.LOG	entering a function and record id

Any combination of these can be chosen. Note: all security violations are recorded regardless of the logging specified here. The data is recorded in the *PROTOCOL* file, e.g.



@ID	K.USER	APPLICATION	LEVEL.FUNCTION	PROCESS.DATE	ID
200605100002543760.00			0		0
200605100002543765.00	S.DONEY	SIGN.ON		07 SEP 2001	
200605100002543776.00	S.DONEY	USER.SMS.GROUP		07 SEP 2001	
200605100002543777.00	S.DONEY	USER.SMS.GROUP	L	07 SEP 2001	
200605100002543781.00	S.DONEY	USER.SMS.GROUP	I	07 SEP 2001	
200605100002543781.01	S.DONEY	USER.SMS.GROUP	I	07 SEP 2001	IB.BADMIN
200605100002550618.00	S.DONEY	DE.FORMAT.SWIFT		07 SEP 2001	
200605100002550622.00	S.DONEY	DE.FORMAT.SWIFT	S	07 SEP 2001	
200605100002550624.00	S.DONEY	DE.FORMAT.SWIFT	S	07 SEP 2001	350.1.1
200605100002552819.00	S.DONEY	SIGN.OFF		07 SEP 2001	
200605110000249448.00			0		0
200605110000249452.01	S.DONEY2	SIGN.ON		07 SEP 2001	
200605110000249465.00	S.DONEY2	USER,		07 SEP 2001	
200605110000249465.01	S.DONEY2	USER,	S	07 SEP 2001	
200605110000249465.02	S.DONEY2	USER,	S	07 SEP 2001	S.DONEY
200605110000249466.00	S.DONEY2	USER,	C	07 SEP 2001	
200605110000249468.00	S.DONEY2	USER,	C	07 SEP 2001	SAMPLE
200605110000249506.00	S.DONEY2	USER,	S	07 SEP 2001	
200605110000249512.00	S.DONEY2	USER,	C	07 SEP 2001	
200605110000249514.00	S.DONEY2	USER,	C	07 SEP 2001	SAMPLE
200605110000250599.00	S.DONEY2	USER		07 SEP 2001	

Figure 11 Details of all activity record on the PROTOCOL file

The *PROTOCOL* record holds details of the activity, the user, the application, the time, the terminal etc.

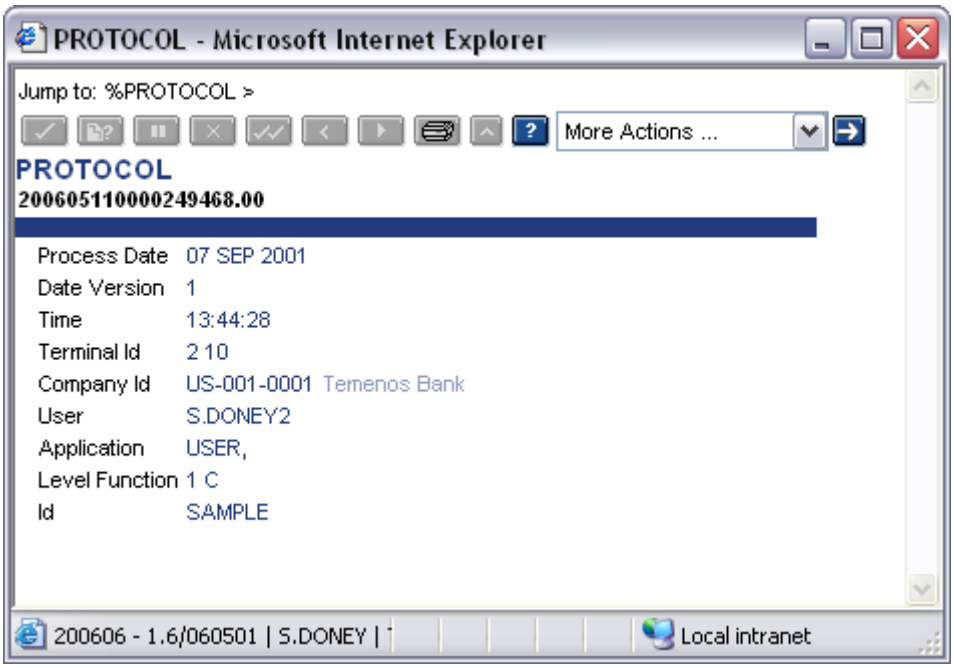


Figure 12 A PROTOCOL record.

In addition to on-line enquiries a protocol report is produced during the Close of Business using a REPGEN *PROTOCOL*.

PASSWORD VALIDATION

It is now possible to define the minimum length of all **T24** user passwords. **T24** can also check the integrity of the syntax of the **T24** user password. The password parameters are now defined in the *SPF*, *SYSTEM* record of the **T24** system. Passwords can now consist of a certain amount of characters from each group in the *SPF, SYSTEM* record i.e. uppercase alphabetic, lowercase alphabetic, numeric and other characters. The **T24** administrator will maintain these password parameters.

After the minimum length of the password has been increased in the *SPF*, *SYSTEM* record, then when existing **T24** users sign onto the system using passwords based on the previous minimum length, they will have to be informed that their passwords have been terminated and that they must capture new passwords that conform to the parameters defined in the *SPF*, *SYSTEM* record.

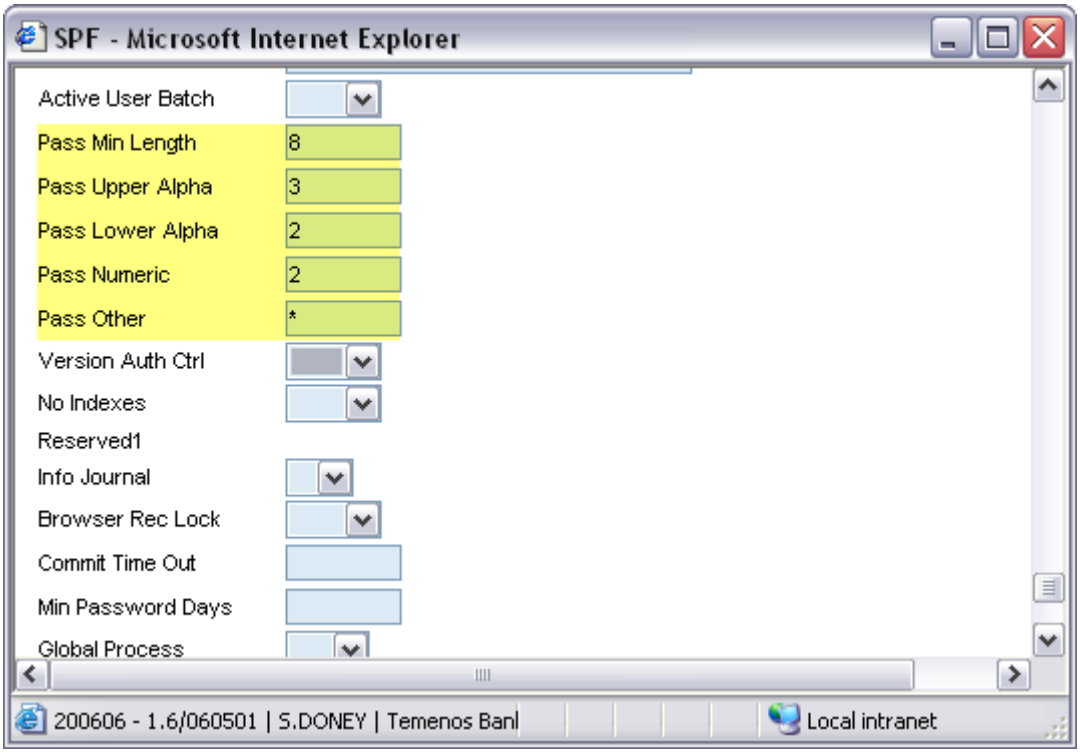


Figure 13 Password validation parameters

In this example, the administrator has defined the password parameters for all **T24** users on this system. So when users log in, the system lets the user know that their passwords must change to consist of 3 uppercase alphabetic, 2 lowercase alphabetic, 2 numeric characters and must consist of a "*". The password cannot be less than 8 characters long.

Re-Enabling User Profiles

The system administrator will be required to re-enable a users profile when the user has forgotten their password, wishes to sign-on during a deactivated period or when the operating system has logged him out.

To re-enable the user profile the application *PASSWORD.RESET* should be run.

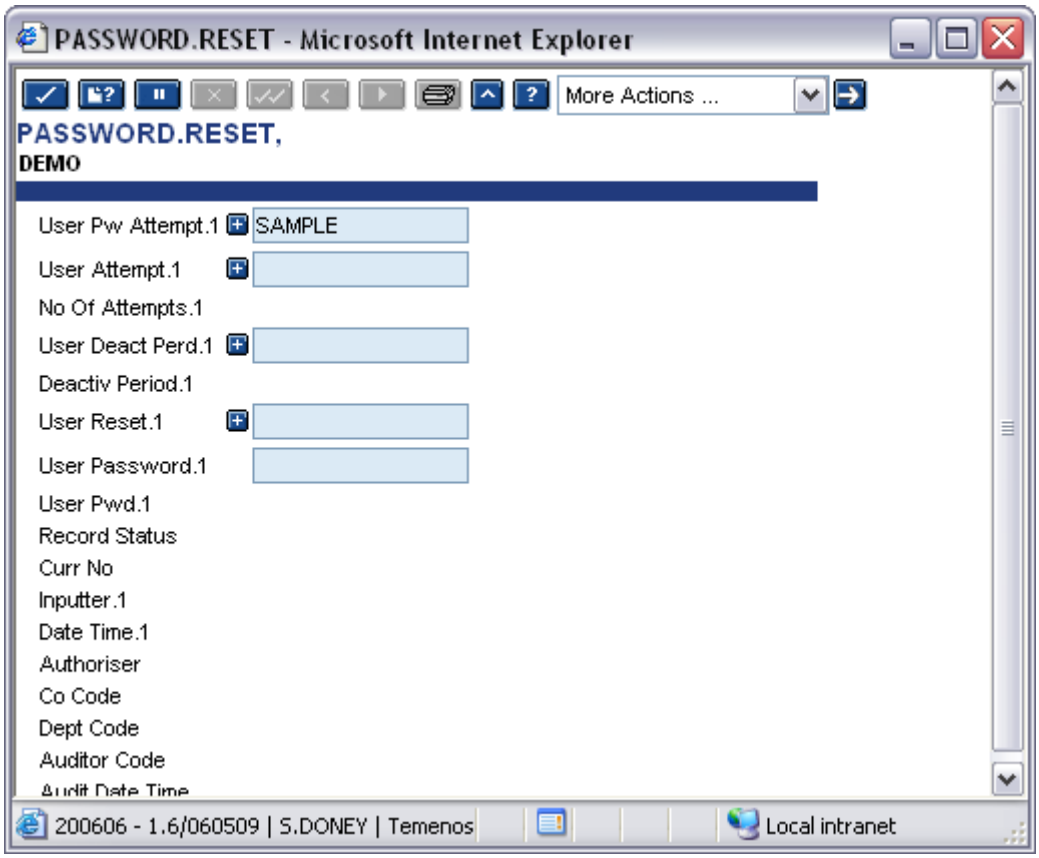


Figure 14 PASSWORD.RESET to re-enable a User profile

If the operating system logs out a user i.e. through a machine failure or UNIX kill, **T24** will still record them as signed on. Hence, if the user attempts to sign on again they will get the message:

To allow them to sign on again, the Administrator should run the application *SIGN.ON.RESET*.

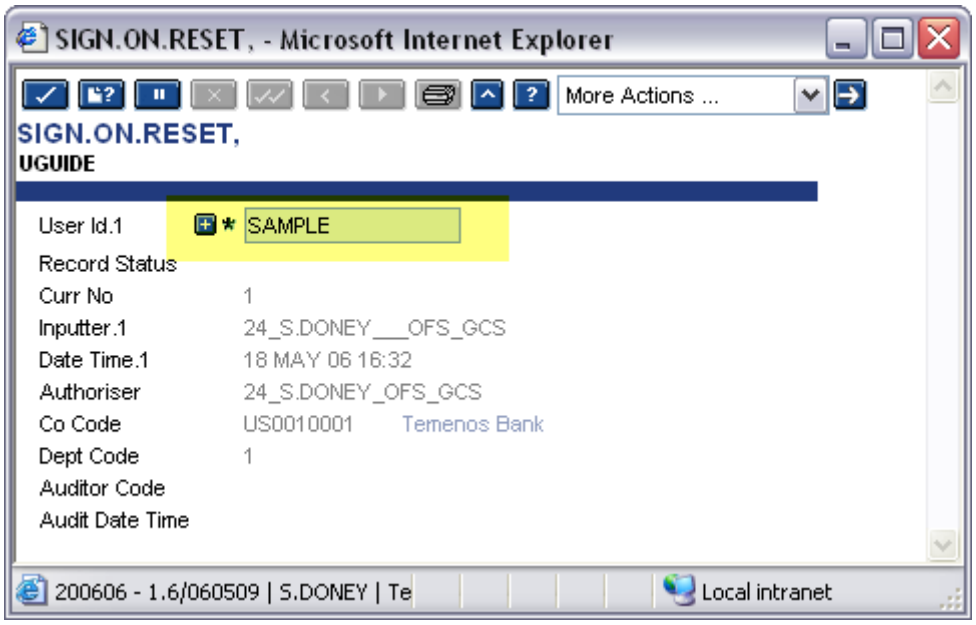


Figure 15 Sign On Reset to reset a User ID for use again

For both *PASSWORD.RESET* and *SIGN.ON.RESET*, multiple user profiles can be reset. Note: the Authorisation process resets the profiles; typically these would be used with a self-authorising *VERSION*; SMS regulations permitting.

Activating programs at SIGN In / SIGN Out

`SIGN.ON.ITEM` field.

The name of any Application, Enquiry, Script, etc. can be entered into the multi-valued field.

Each multi-valued field will be processed in the order in which it appears during the Sign On process, with the commands at the beginning of the field processed first.

`SIGN.OFF.ITEM` field.

This field allows a user defined DataBasic subroutine that accepts one parameter to be invoked during the Sign Off process. Enter the name of the subroutine that needs to be invoked during Sign Off into this field.

During the Sign Off process, the subroutine will be passed one parameter. If the parameter returns with the values 'N', or 'NO' the Sign Off process will be halted. Any other value returned will cause T24 to continue the Sign Off process.

A *PGM.FILE* definition for the subroutine is required for the subroutine to be run during the Sign Off process.

Setting a user with a new password

You can set a user password to a specific value for their first sign-on using the `USER.RESET` and `USER.PASSWORD` fields in the *PASSWORD.RESET* record.

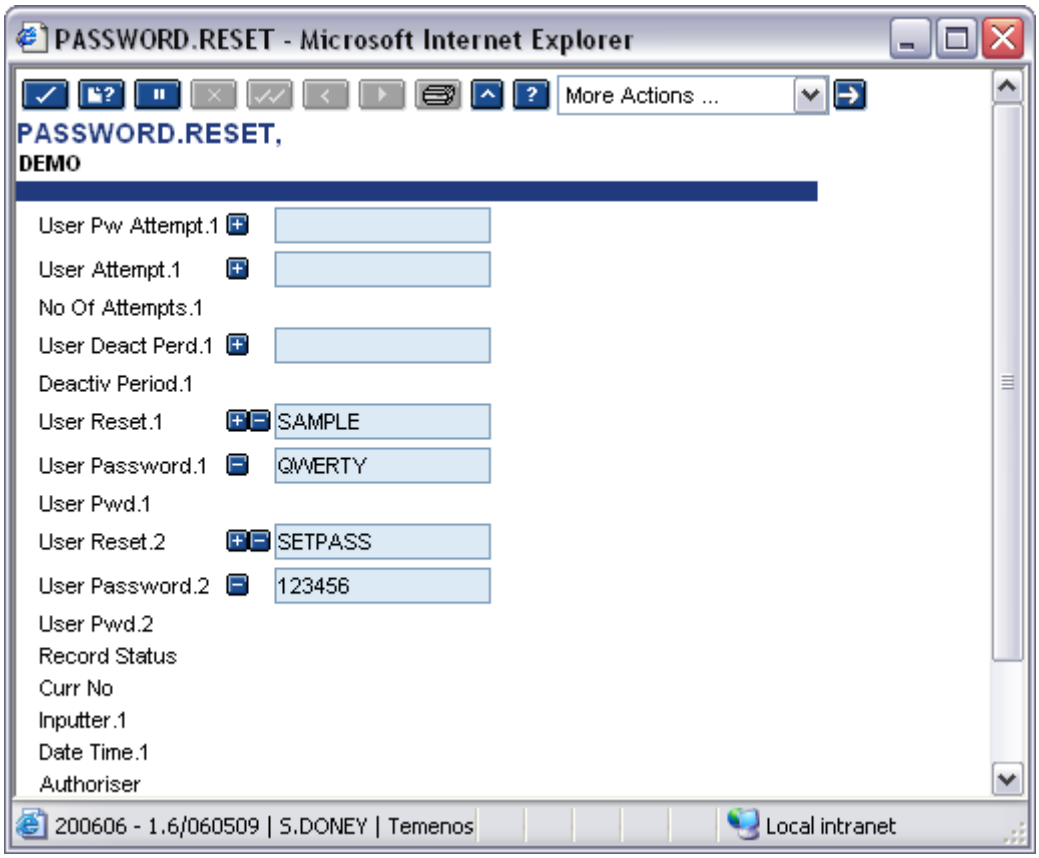


Figure 16 Resetting USER with one-shot passwords

The passwords are cleared from the record when it is committed, so you have to remember or note them to tell the users. The users will have to set a new password in the process of logging on. This process retains old passwords thus preventing re-use.

Setting a user with a new random password

A routine to generate random passwords (GEN.RANDOM.PSWD) is provided. To use it, set up a *VERSION* record with @GEN.RANDOM.PSWD in `VALIDATION.RTN` for *USER.PASSWORD*. For example:

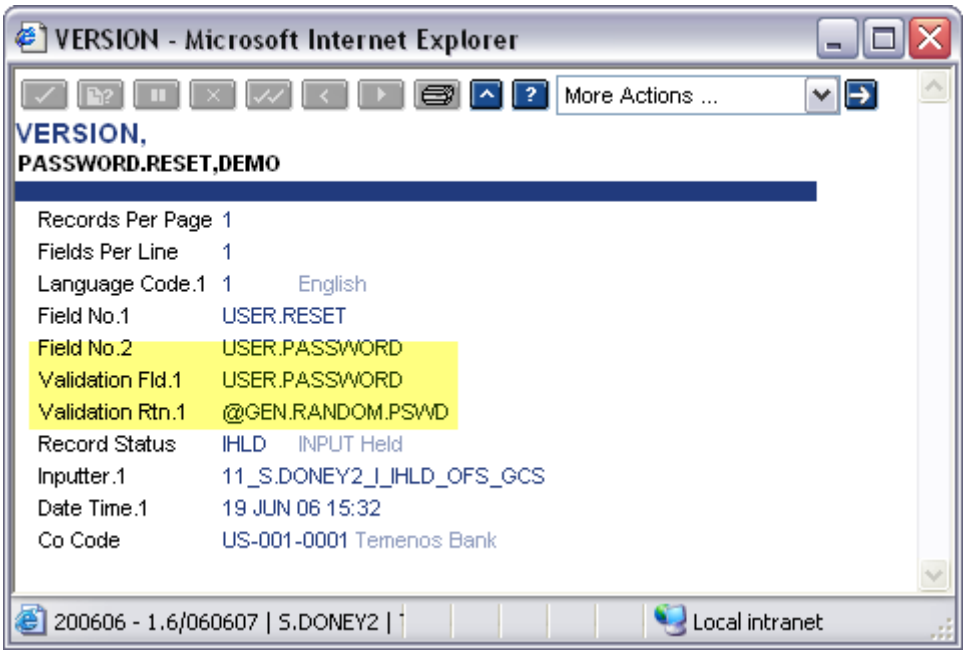


Figure 17 A VERSION, which resets a USER with a random password

This routine allows only one *USER* per record to be reset. The `USER.PASSWORD` field is cleared when the record is committed so it must be remembered or noted on input or it will have to be reset again. The routine also blocks further input in the password field and you must quit the application before you input another record.

If you want to ensure that sign-on without an initial password is not allowed then set up the following subroutine for use and apply it as a `BEFORE.AUTH.RTN` in the *VERSION* you apply to authorise a *USER* record.

```
File GLOBUS.BP , Record 'BLOCK.PASSWORD'
Command->
0001 ▌   SUBROUTINE BLOCK.PASSWORD
0002
0003 * This routine should be run from a VERSION record for the USER
0004 * application and will write '****' to the USER<PASSWORD> field
0005 * on a new USER record. This will prevent access until PASSWORD.RESET is
0006 * run to give the a password to login with.
0007
0008     $INSERT I_COMMON
0009     $INSERT I_EQUATE
0010     $INSERT I_F.USER
0011
0012     IF R.NEW(EB.USE.CURR.NO)  = '1' THEN
0013         R.NEW(EB.USE.PASSWORD) = '****'
0014     END
0015
0016     RETURN
0017
0018 END
```

Figure 18 Routine to prevent use of new USER until PASSWORD.RESET

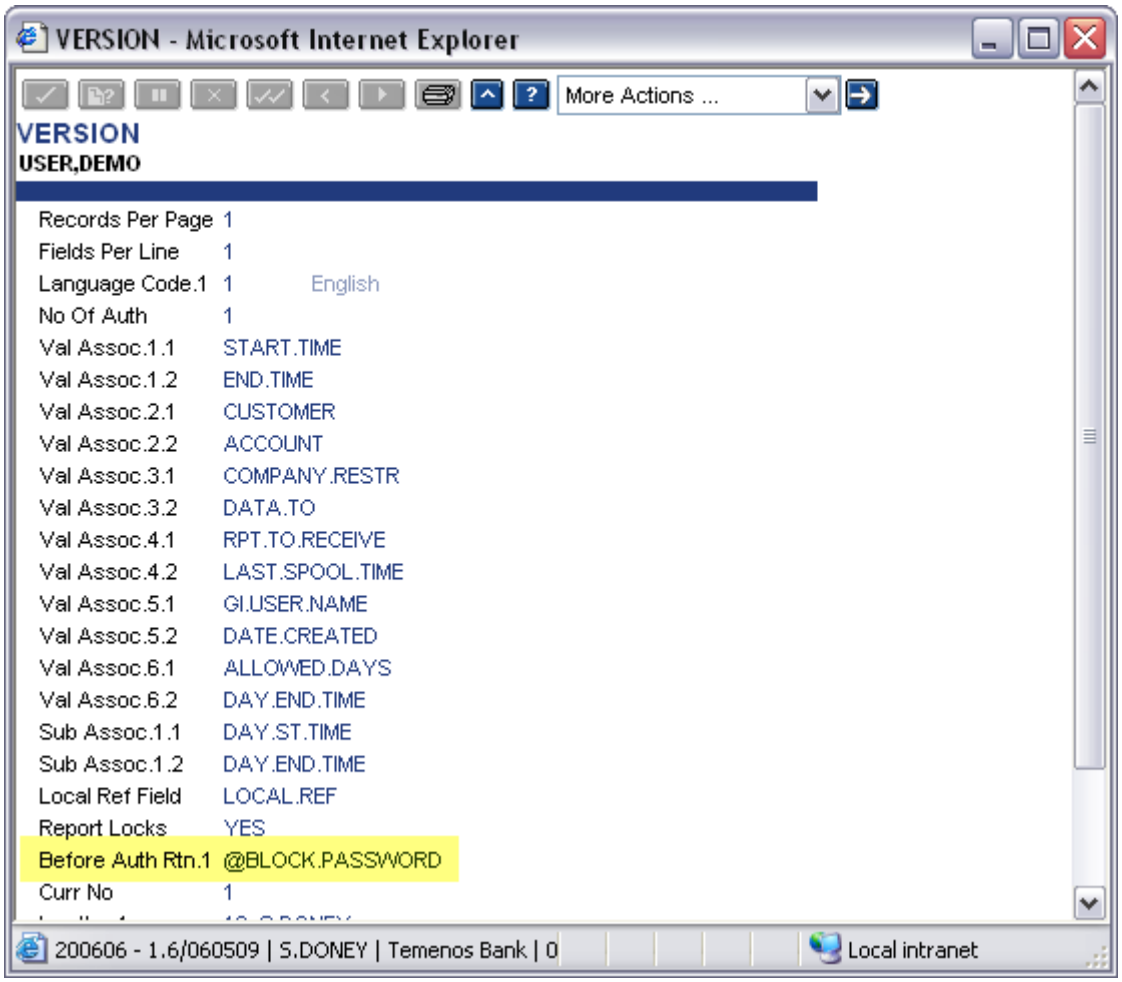


Figure 19 VERSION to prevent use of new USER until PASSWORD.RESET

Logging in based on days of a week

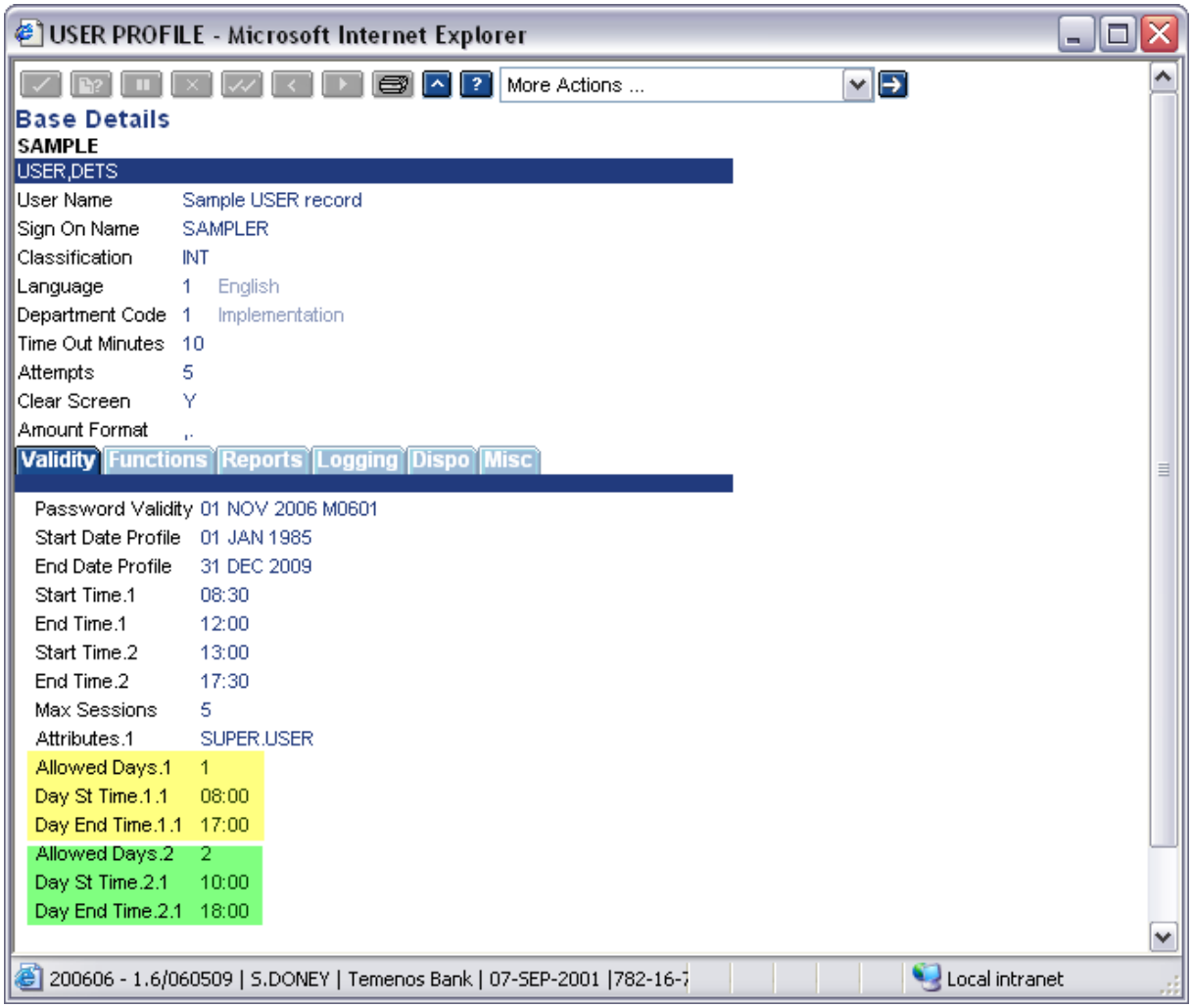


Figure 20 Day & Time checking

A special note: If in *USER* profile a different logging in time has been defined. It will override what has been defined in *USER.SMS.GROUP*

Defining USER.SMS.GROUP

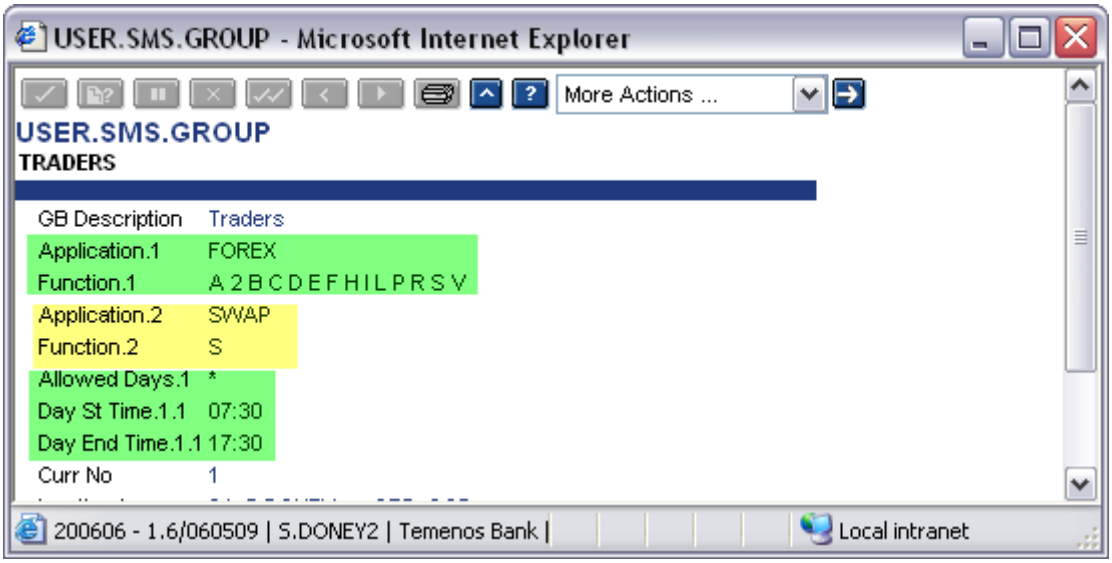


Figure 21 SMS Grouping

The above set-up has been done in such a way that the group of users who are defined as *TRADERS* will be able to have all functionality for *FOREX* application, while only “See” functionality for *SWAP* and no access to any other application.

They have access to the system between 07:30 A.M. and 17:30 P.M.

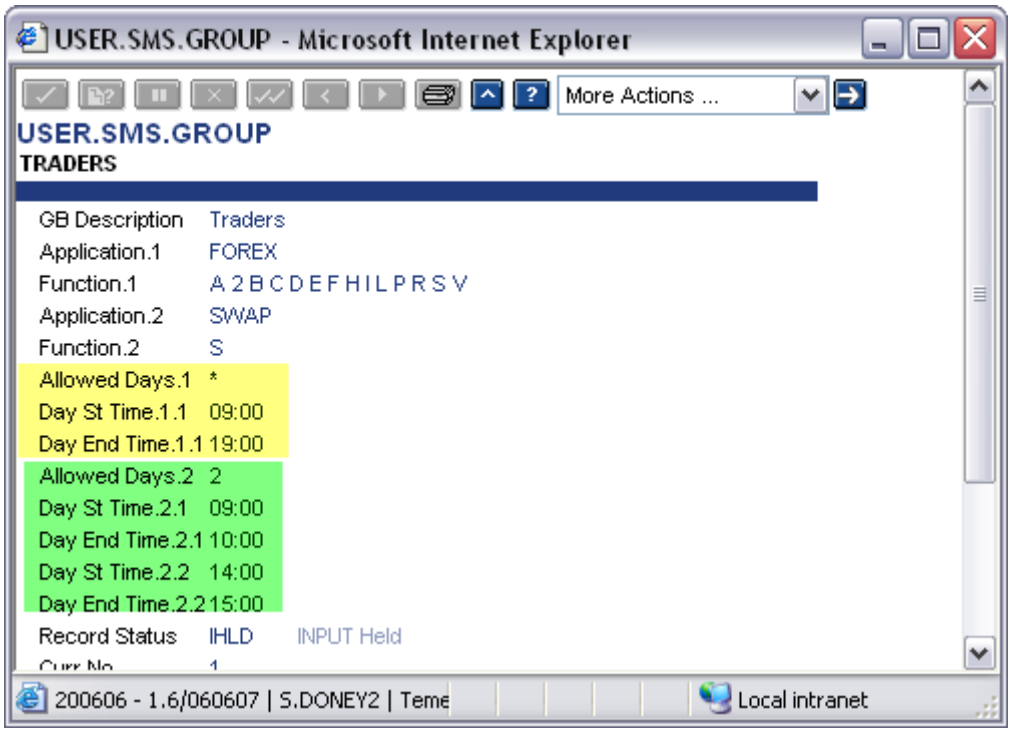


Figure 22 Group restrictions

Multiple timings within the same day can also be given. In the above example the user has rights to login on Tuesday between 09:00 AM and 10:00 AM. Also the user can login between 14:00 and 15:00 on the same day. The user is denied entry at any time other than the one set-up.

Linking USER.SMS.GROUP to USER

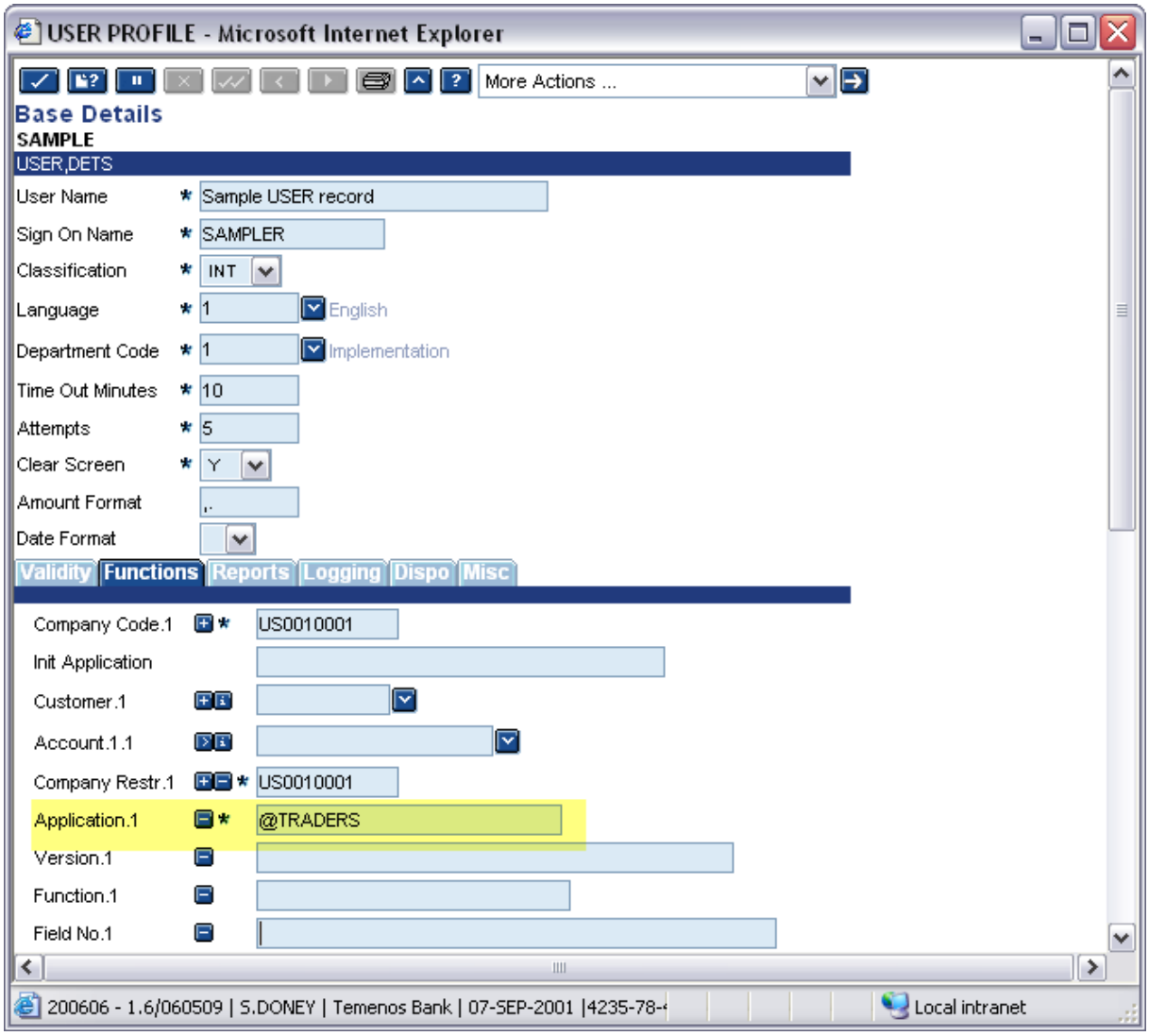


Figure 23 USER.SMS.GROUP

The *USER.SMS.GROUP* can be the sole set of rules/restrictions on the *USER* record or more usefully it can be combined with settings that are unique to that person.

These extra settings can add functionality; remove functionality or a combination of both on top of those in the specified *USER.SMS.GROUP* record.

Security Violations

Protecting **T24** from access by unauthorised users and preventing users from using parts of the system they are expressly forbidden to use requires an approach that is helpful enough to legitimate users when they make a mistake or have a problem but does not provide useful information for the unauthorised.

In the examples below some of the texts have been changed from the non-descript 'Security Violation' to something more explanatory for the purpose of illustrating that the messages are individual – the SMS logs should be checked for the actual nature of the violations.

Access to T24

The *USER* profile is set to restrict the actions of the staff member and depending on the way that each bank stipulates its security protocols the member of staff may encounter security warnings at various stages. The primary place of access is the sign-on screen where the user must enter both the user name and their individual password.

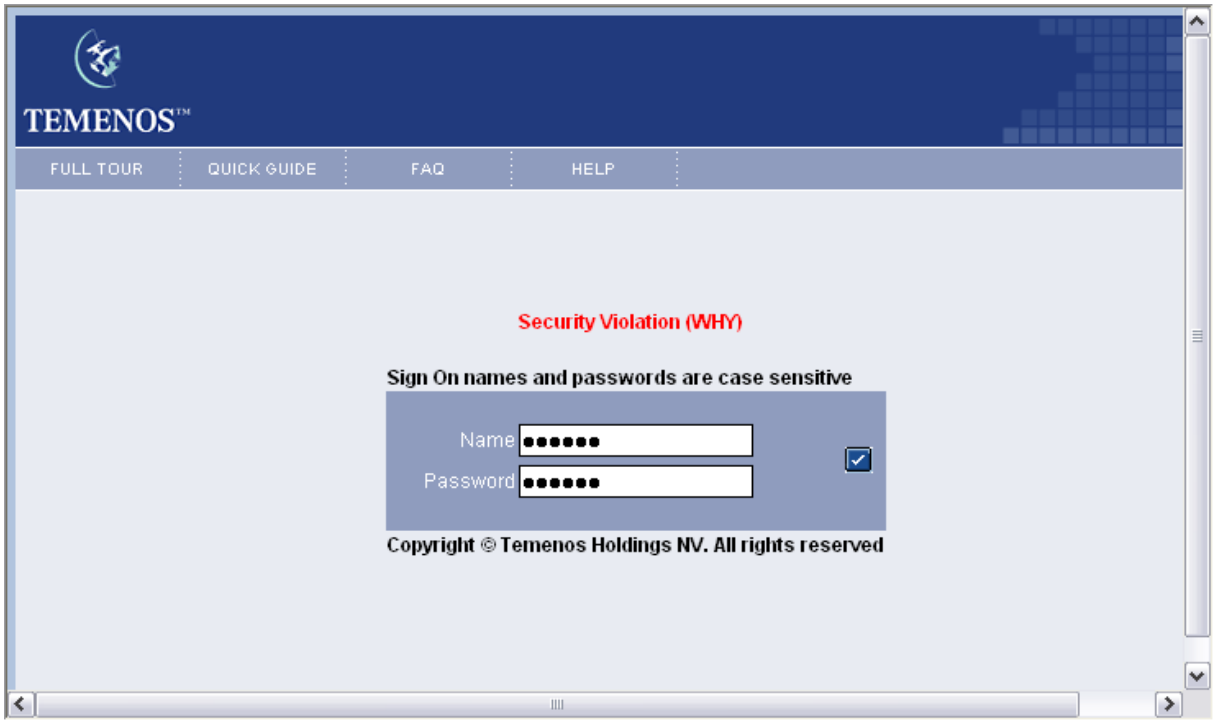


Figure 24 Sign on Invalid – in this case it's an invalid name

- An invalid name will be given a security violation warning
- An incorrect password will be given a security violation warning
- Too many attempts will give the same warning

Restrictions

The *USER* record will restrict the staff member (or external client) to certain parts of the database (by *COMPANY* code) and various applications and functions. Trying to use an application or function that is not allowed will trigger a security violation warning, which may be logged (if logging is activated).

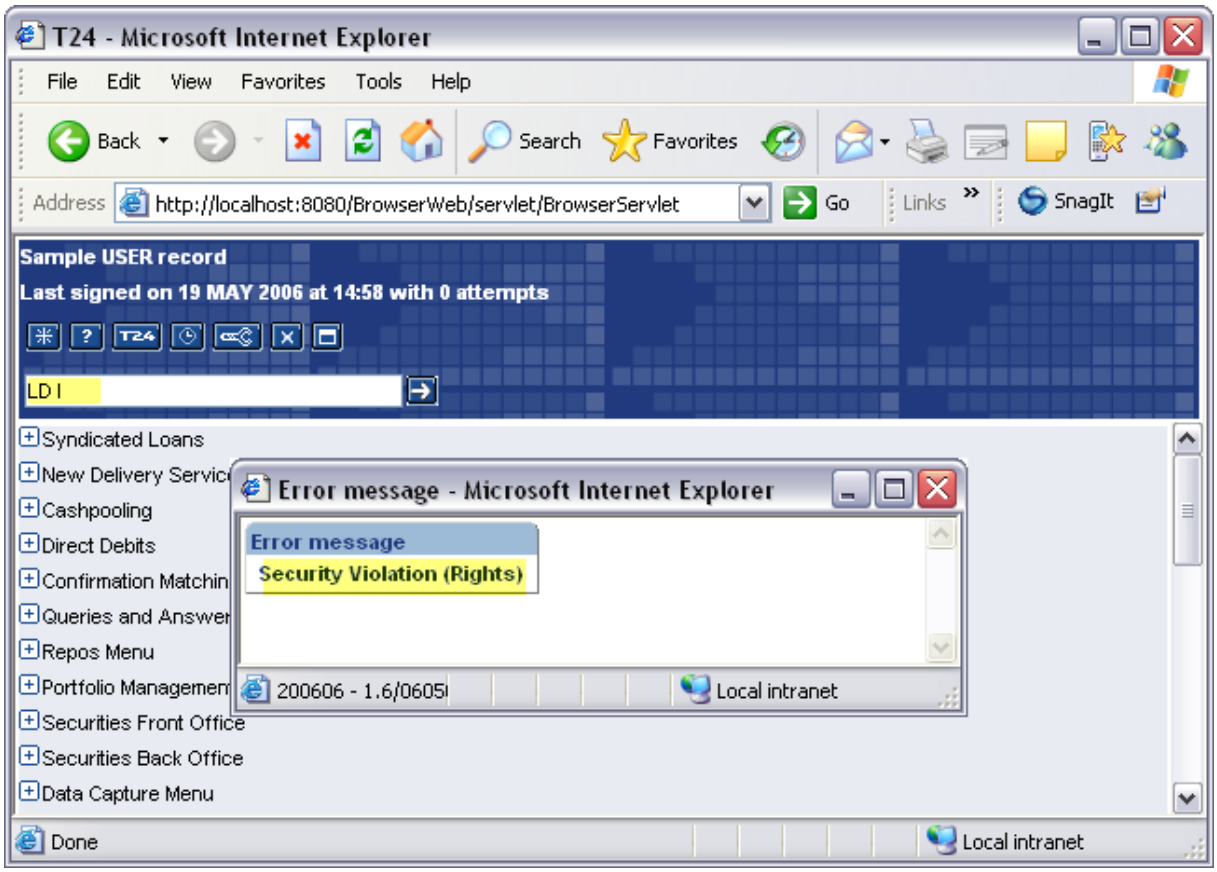


Figure 25 Trying to use an application not on the USER record

Maintenance

In instances where the *USER* profile is being modified and awaits authorisation **T24** will prevent access by that user on the basis of the change could be one removing access or privileges. Once the user record is authorised the person will be permitted to log in subject to the time & date restrictions imposed on them

Times

Should access be attempted outside of the times permitted, or outside of the days permitted a security violation will be given (and optionally logged).

USER records can be deactivated for periods where the person is absent for a period of time such as annual leave.

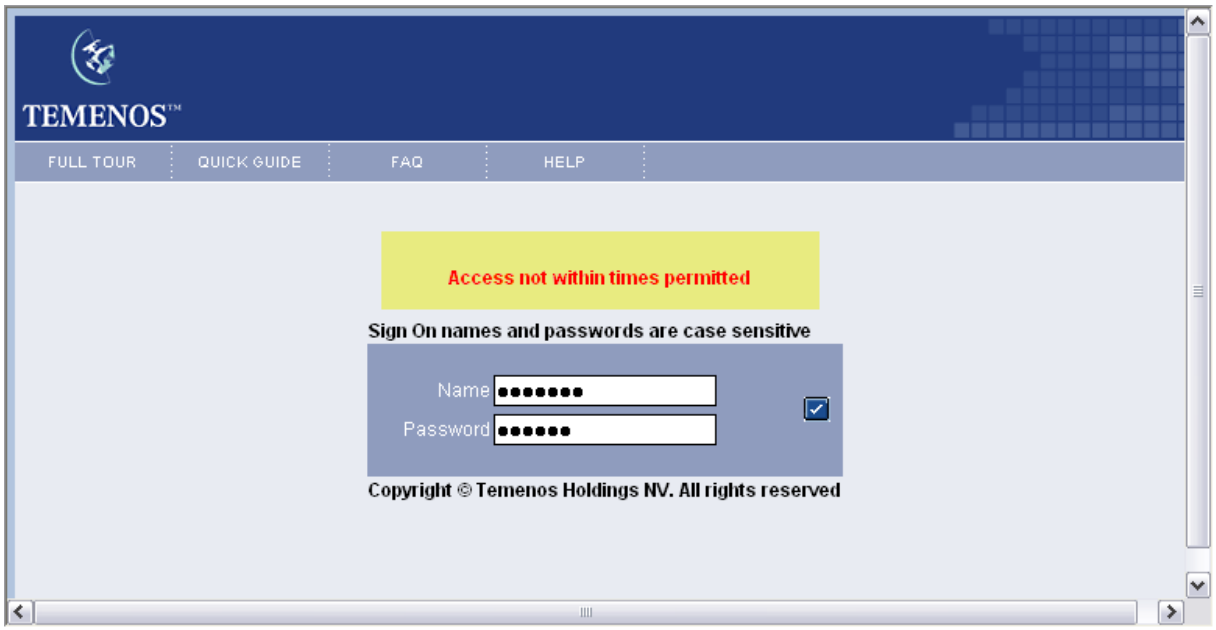


Figure 26 User trying to access the system outside of the permitted hours

OVERRIDE

In **T24** where circumstances occur that require the user to confirm an action which needs to be approved an override will be required. The concept of the override process spans from any user approving the override at input, up to a full system of tiered approval levels (using *DISPO.ITEMS* mentioned later in this document).

Overrides can be displayed as an Error, Message, Warning or Auto override. These are defined in the `TYPE` field on the *OVERRIDE* application.

In the following example, the first multi-value set (fields `GB MESSAGE.1` to `OVERRIDE ACTION.1`) defines the default override settings. The `CHANNEL.1` field must contain a Null value to ensure that a default setting is configured.

The *CALLCENTRE* channel in the following example is configured to display a Warning instead of the default override message.

Multiple channels can be defined in an *OVERRIDE* record. However, the same channel cannot be defined more than once in a single *OVERRIDE* record.

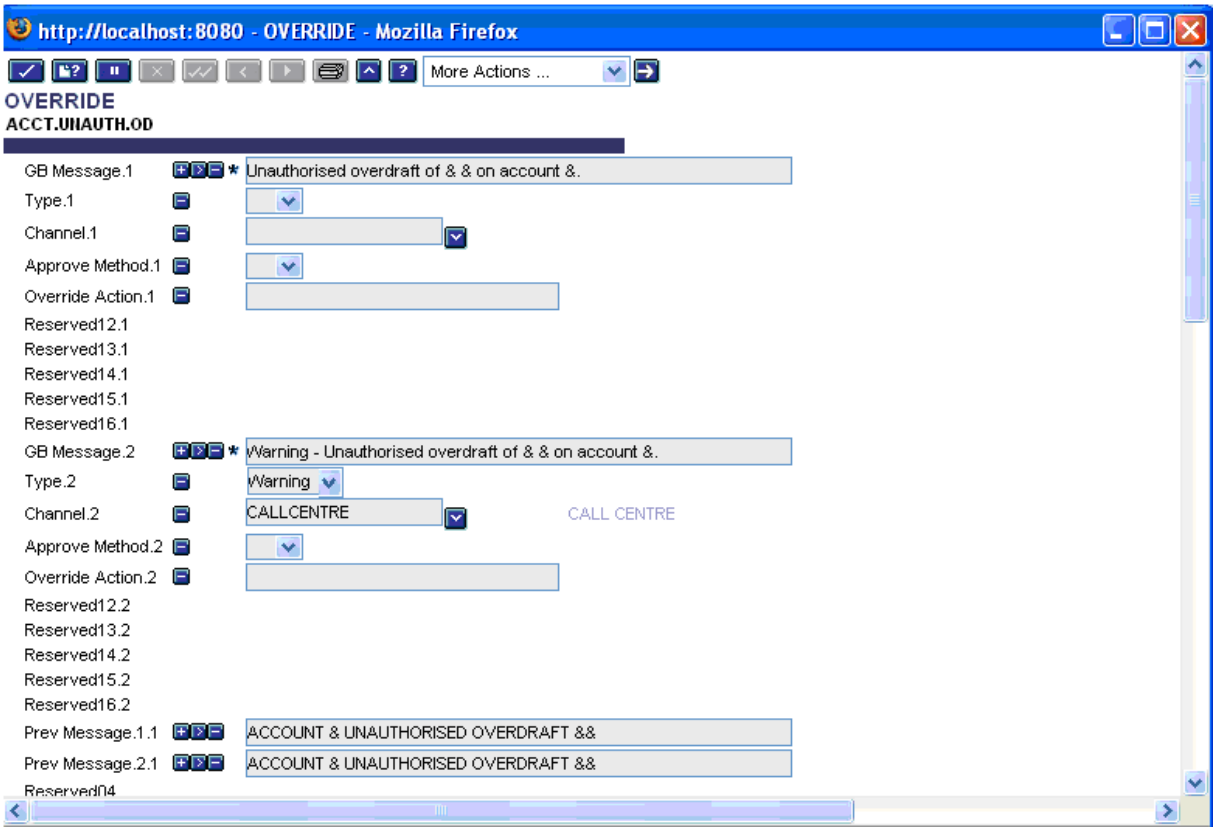


Figure 27 OVERRIDE

The following example illustrates the input of a transaction via the *CALLCENTRE* Channel. In this instance the override is displayed as a Warning – as per the above *OVERRIDE* record: -

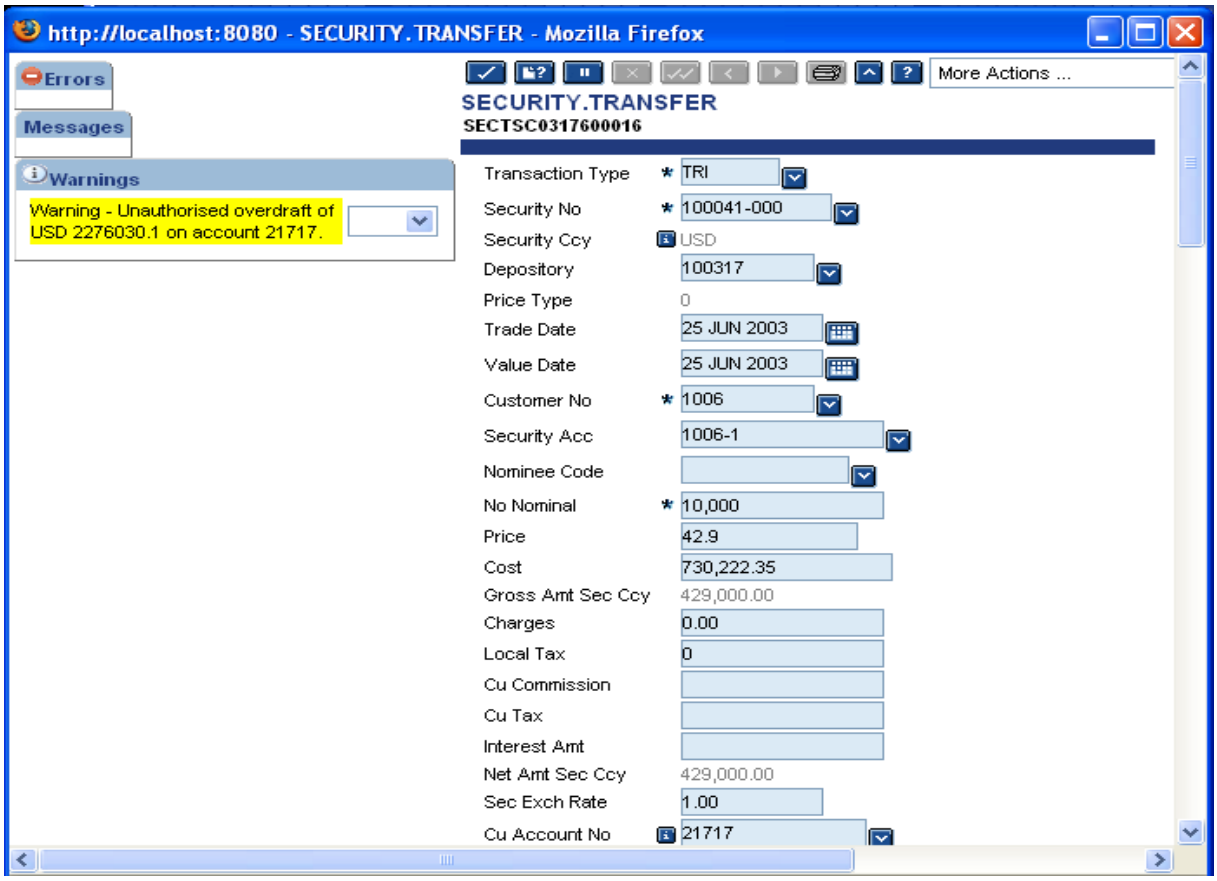


Figure 28 CALLCENTRE override

In the following example a Channel has not been defined on the *OVERRIDE* record. In this instance the `NUMERIC.ID` field value is displayed in the override message: -

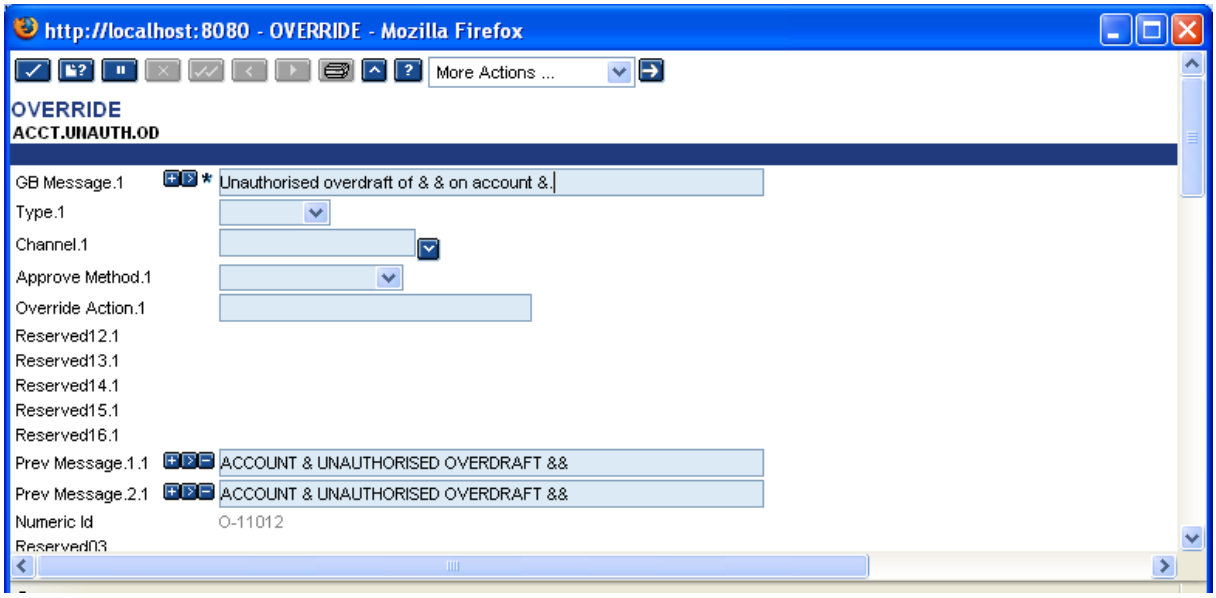


Figure 29 OVERRIDE

Figure 30 Numeric Id override

OVERRIDE CLASS

During the input of a transaction, **T24** can request the user to override a certain condition, e.g. an account is overdrawn or has a posting restriction. The overrides are recorded on the transaction so that the authoriser can view all overrides encountered during input.

The system administrator, normally in conjunction with the appropriate supervisor, can define certain conditions, which can only be overridden by a specific user. For example, you may wish to restrict the overrides of account overdrafts to supervisors only.

ACCOUNT-123456 UNAUTHORISED OVERDRAFT GBP 5000.00

To do this you define a class for the override, or overrides, and then allow that class for certain users. To define the class, use the application *OVERRIDE.CLASS*.

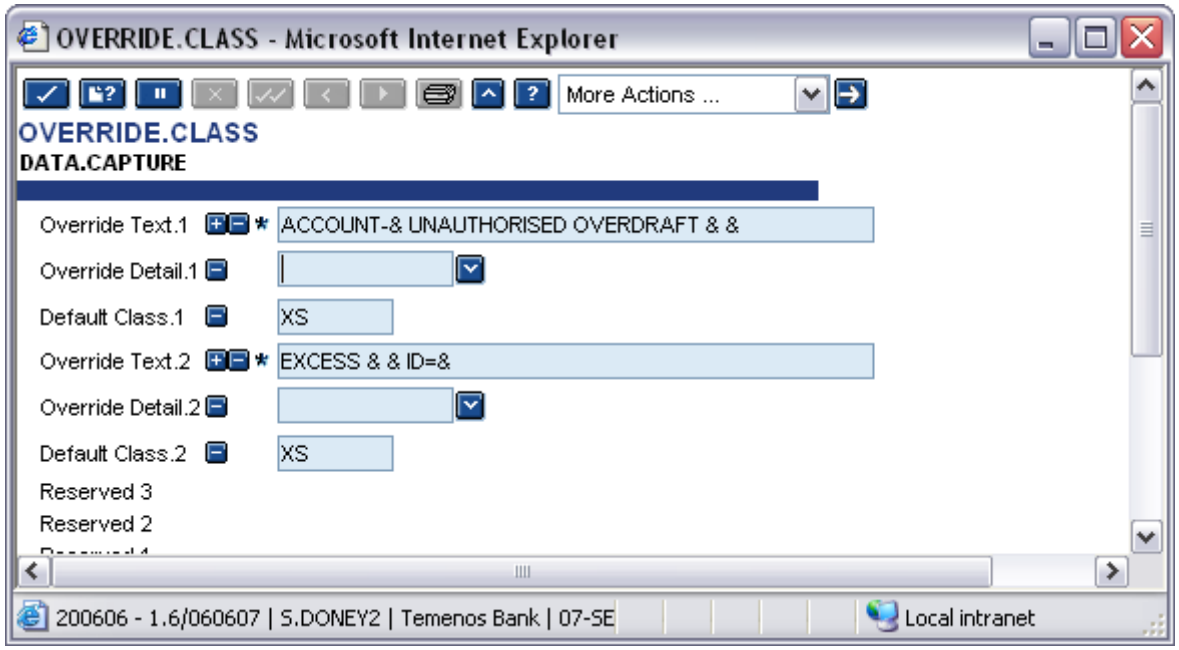


Figure 31 Overriding Conditions – defining class of 'XS'

Here a class of *"XS"* has been defined in *DATA.CAPTURE* for the overdraft and limit excess overrides. Note: The "&" embedded in the text defines where variable information, such as amounts, will be displayed.

To allow a user to override these messages *"XS"* must be entered on his *USER* profile.

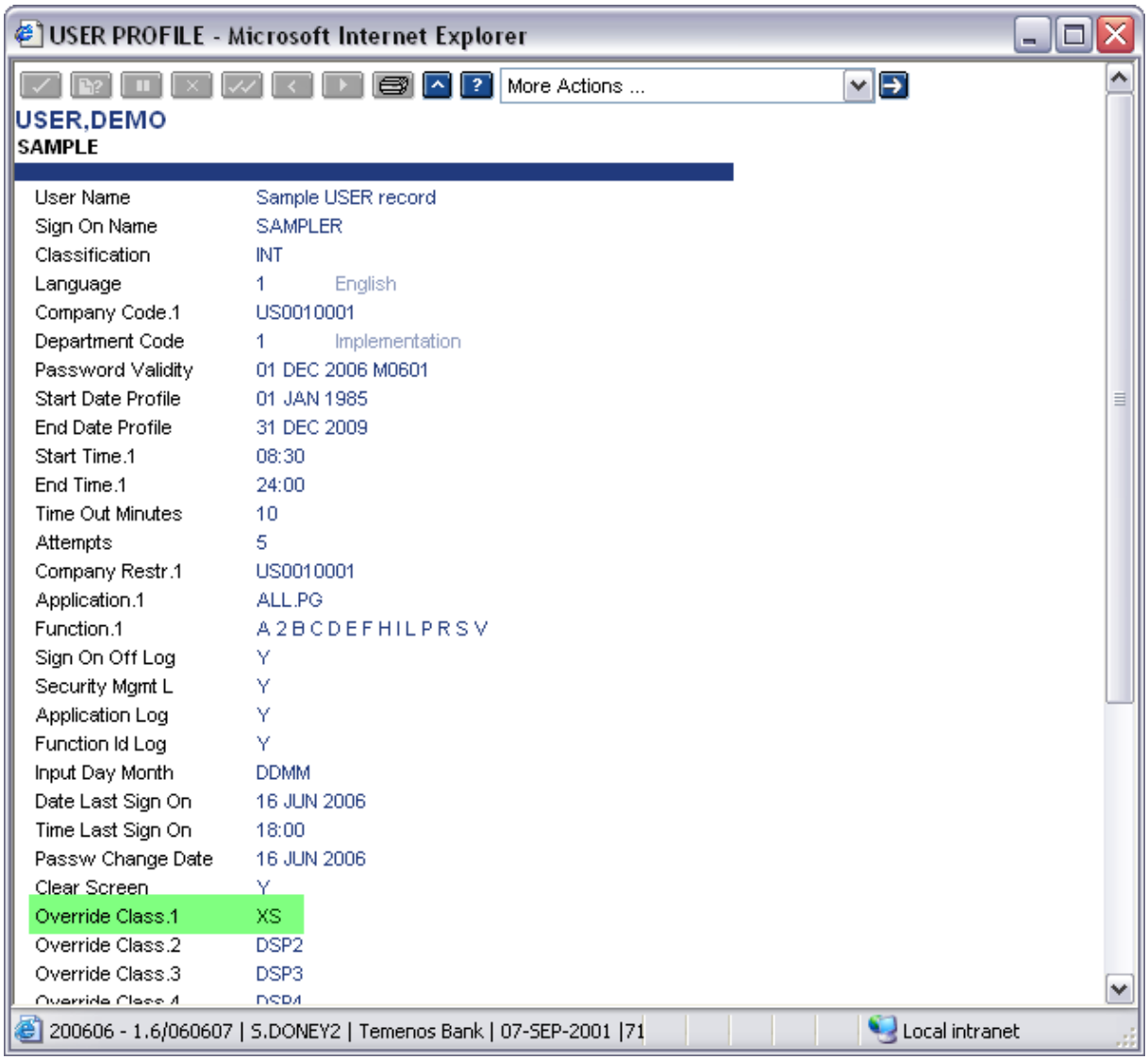


Figure 32 Define class 'XS' in User Profile

It should be noted that the authority to approve the override is based on the User, not the (A)uthorise function. If the Department Manager has a specific class of override that they alone can approve, and they input a deal, then the override is deemed to be approved. However, it is normally the Authoriser that does have the necessary privileges and usually where the overrides are approved. If an Authoriser cannot 'approve' all the overrides, the message "YOU CAN'T APPROVE ALL OVERRIDES" is displayed and the record remains unauthorised.

The status of the override can be seen in the audit fields.

Override	Status
No Line Allocated	Standard – No privileges required. Any authoriser can approve deal.
No Line Allocated*MGR	Has an *OVERRIDE.CLASS*. Authoriser needed with class MGR on their *USER* profile
No Line Allocated*MGR*I	Has an *OVERRIDE.CLASS*, but the inputter has the class MGR on their *USER* profile.
No Line Allocated*MGR*UserName	The *OVERRIDE.CLASS* MGR has been approved by the *USER* who appears in the position shown here as UserName

Figure 33 Override audit fields

Overrides can also be classified by the variable information shown in the message. Hence, you could restrict approval of overdrafts greater than USD 10,000.00 to specific users. To do this you enter the details in the application *OVERRIDE.CLASS.DETAILS* and link them to the *OVERRIDE.CLASS* record.

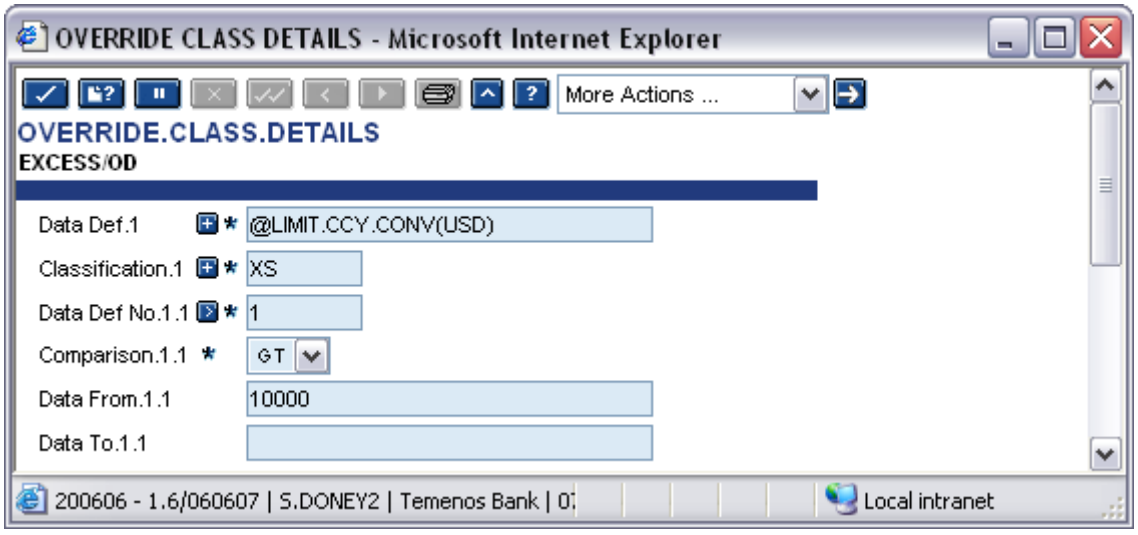


Figure 34 Specify overrides in the Override Class Details Record

In this example only users with the override class of *"XS"* can approve limit excess messages where the amount is greater than 10,000.00 US dollars. All other users can only approve amounts less than this.

See the helptext for the application *OVERRIDE.CLASS.DETAILS* for more information on the use of conversion routines and testing variable data.

Override Management – Auto Override Processing

Introduction

T24 has the functionality to automatically decline or accept generated overrides when using OFS (Open Financial Service). The STP (Straight Through Processing) functionality uses OFS to process orders and requires Auto Override Processing to automatically validate overrides - avoiding user intervention.

Installation

In order for Auto Override Processing to be enabled, certain steps must be taken to set up the validation process. This functionality enables the user to set up validations on overrides to apply to all applications or only to certain applications. It can also be set up to only perform certain validation depending on which OFS process is being performed.

For each override on the system, an *OVERRIDE* record exists which will contain the override text in the `MESSAGE` field of the *OVERRIDE* record. The `MESSAGE` text may contain one or more ampersands (&). These are known as variable placeholders. An example of a `MESSAGE` field within an *OVERRIDE* record can be seen in the screenshot below:

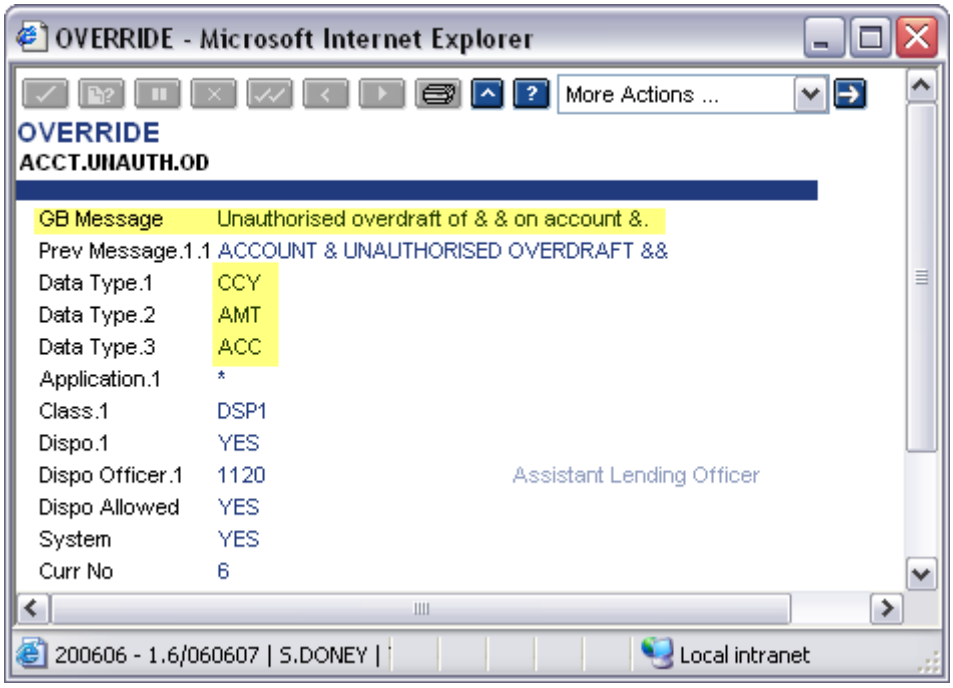


Figure 35 OVERRIDE Record example 1

The variable placeholders will be assigned data by the application that generated the override message, prior to the message being displayed. In the screenshot examples, the first variable placeholder will contain a Currency; the second variable placeholder will contain an Amount and the last the Account number.

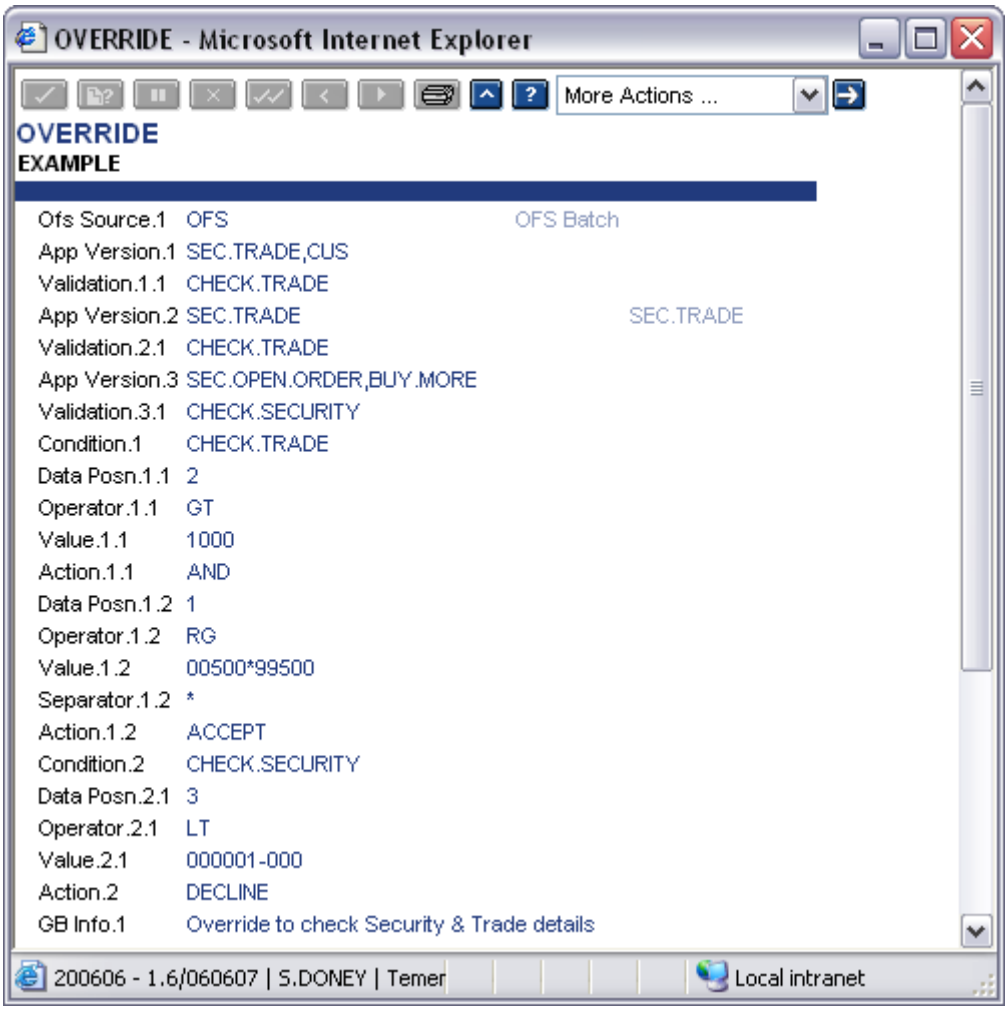


Figure 36 OVERRIDE record example 2

The following is a description of fields, which may have to be populated to enable the Auto Override functionality:

OVERRIDE

Field Name	Description
DATA.TYPE	This field is used to determine the data type of the data elements in the MESSAGE field which are represented by a (&).
DATA.DESCRIPTION	A text description of the above data being passed in.
OFS.SOURCE	Will contain the key to an *OFS.SOURCE* record, which contains the details of the OFS process to be used. This field will need to be input to perform Auto override processing.
APP.VERSION	This determines which Applications or VERSIONS of applications will perform automatic Override validation during an OFS process

SUBROUTINE	Enables the user to input a user defined subroutine to carry out the validation process. If this is the case, no further validation will be performed. The subroutine should return either a "Y" or "NO" to indicate whether the Override is automatically approved or not.
VALIDATION	This determines which validation routines to run from the CONDITION field for the associated Application or Version.
CONDITION	If no input has been made into the associated SUBROUTINE field, the system will use the validation parameters stored on the OVERRIDE record. This field is multi-valued & determines the Validations to be used. As soon as a TRUE condition is met with a Valid ACTION then no further validation will be performed.
DATA.POSN	This field refers to the position of the data passed in from the calling routine that is represented by the '&' in the MESSAGE.
OPERATOR	The operator (e.g. EQ, NE, GT etc) that will be used in conjunction with DATA.POSN & VALUE to form a selection sentence.
VALUE	The value is to be compared with the value in the field specified by the DATA.POSN.
SEPARATOR	This field is used to separator multiple entries of data in the VALUE field.
ACTION	This is the action that will be performed assuming that the data validation performed is *TRUE*. ***AND*** – This will connect one validation with the next one assuming that the result of this validation was *TRUE*. ***ACCEPT*** - Assuming that the data validation is *TRUE* the Override will be automatically accepted. ***DECLINE*** - Assuming that the data validation is *TRUE* the Override will be automatically declined.
INFO	User can input text regarding the override.
SYSTEM	If 'Yes' then the record is a ***TEMENOS*** generated record.

VERSION

Field Name	Description
AUTO.OVERRIDE	If this field is set to 'Yes', individual validation of overrides generated by this version will be performed.

Using Auto Override Processing

In order for the automatic validation of overrides to be performed, a *VERSION* must be used to process the transaction. The field AUTO.OVERRIDES in the *VERSION* should be set to 'Yes'.

The field OFS.SOURCE on the *OVERRIDE* record of each of the overrides generated by the transaction will then be read to obtain the key to the *OFS.SOURCE* record. This record will contain details of the OFS process, which is to be used for this transaction (more than one OFS process may be specified in an *OVERRIDE* record to allow different processing states for individual versions or applications). A check will then be made to ensure that the *VERSION* or application being used to process the transaction is present in the APP.VERSION field of the *OVERRIDE* record. For example, if the current transaction is being processed through the version *SEC.TRADE,BUY* , in order for auto validation to be performed, the APP.VERSION field must contain one of the following:-

1. **SEC.TRADE,BUY**

   The full version name.

2. **SEC.TRADE**

   The process will be performed for any *VERSION* of *SEC.TRADE*.

3. ***** (Asterisk)

   The process will be performed for any Application or *VERSION*.

If either the AUTO.OVERRIDES field in *VERSION* is not set to 'Yes', a valid *OFS.SOURCE* record key has not been entered into the OFS.SOURCE field of the *OVERRIDE* record or valid input can not be found in the APP.VERSION field of *OVERRIDE*, then the existing functionality will be performed whereby the override will be displayed and a user response will be requested.

If a SUBROUTINE has been specified in the *OVERRIDE* record relating to the APP.VERSION being processed, then this will then be called to perform the override validation. Otherwise, any processes specified in the VALIDATION field relating to this APP.VERSION will be performed.

For each VALIDATION field associated with the APP.VERSION, there will be a CONDITION field with the same name. Within this CONDITION, one or more condition sentences can be specified – to be performed on the data contained within the override variable placeholders (&) in the MESSAGE field. If the condition statement performed on the data is found to be 'True' then the action outlined in the ACTION field will be performed. Valid input into the ACTION field is either 'ACCEPT', DECLINE' or AND'. The ACTION 'AND' can be used to link one or more condition statements to construct a string of linked conditions before a final 'ACCEPT' or 'DECLINE' is specified.

If a condition is found to be true, then the ACTION field for the CONDITION will determine the next process.

If the ACTION field is ‘AND’, the condition statement is part of a string of condition statements. The ACTION field on the final condition statement will either be ‘ACCEPT’ or ‘DECLINE’

If the ACTION field for the condition is ‘ACCEPT’, then either the next VALIDATION will be performed for this APP.VERSION (if any) or the override will be automatically accepted (the equivalent of manually answering ‘YES’ to the override.

If the ACTION in a condition is ‘DECLINE’, then a message will appear on screen to notify the user that the override has been declined and an update will be made to the *EXCEPTION.LOG.FILE* with details of the declined override.

Multi-valued DATE.TIME audit information

If required, it is possible to record the date and time of every action on a record since its creation or last authorisation, along with the function used and the status of the record at that time. This can be activated by setting the `DATE.TIME.MV` field on the *SPF SYSTEM* record to YES.

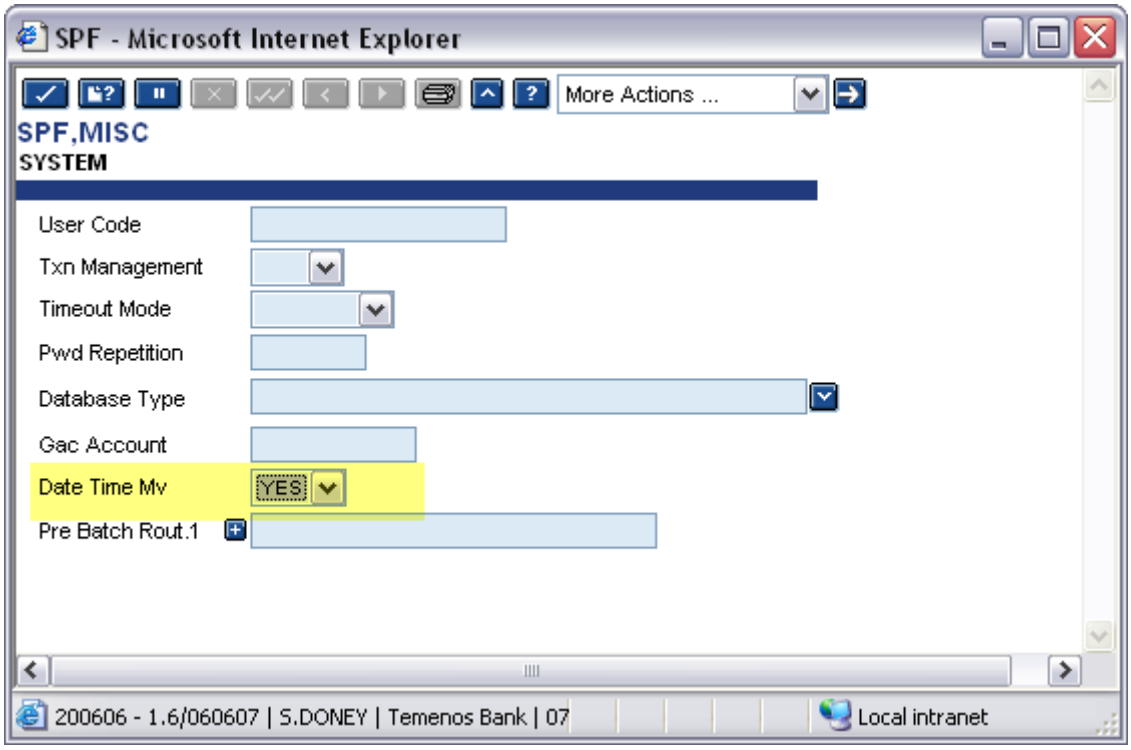


Figure 37 Flag to record the data & time of all record actions

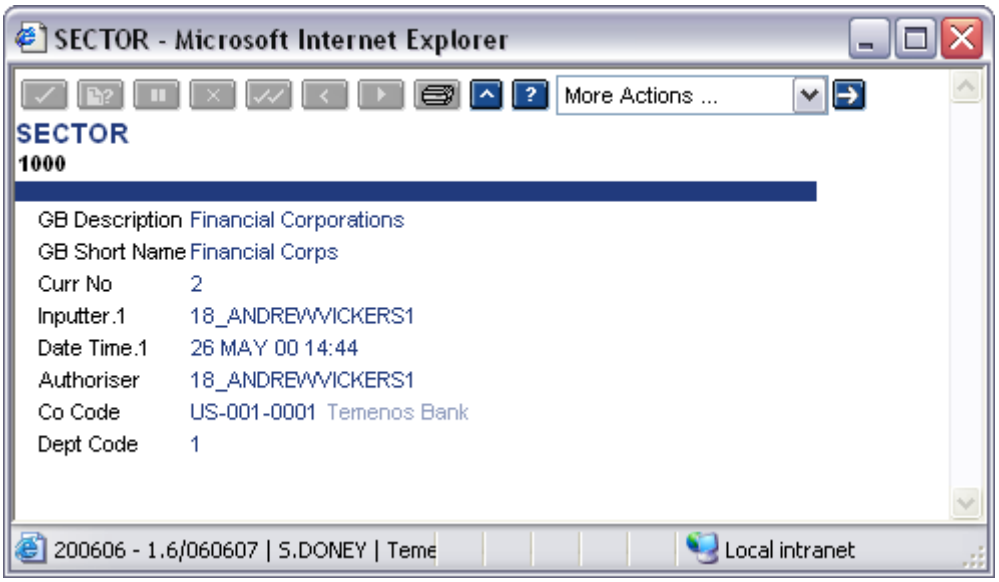


Figure 38 Record of Sector-Default action

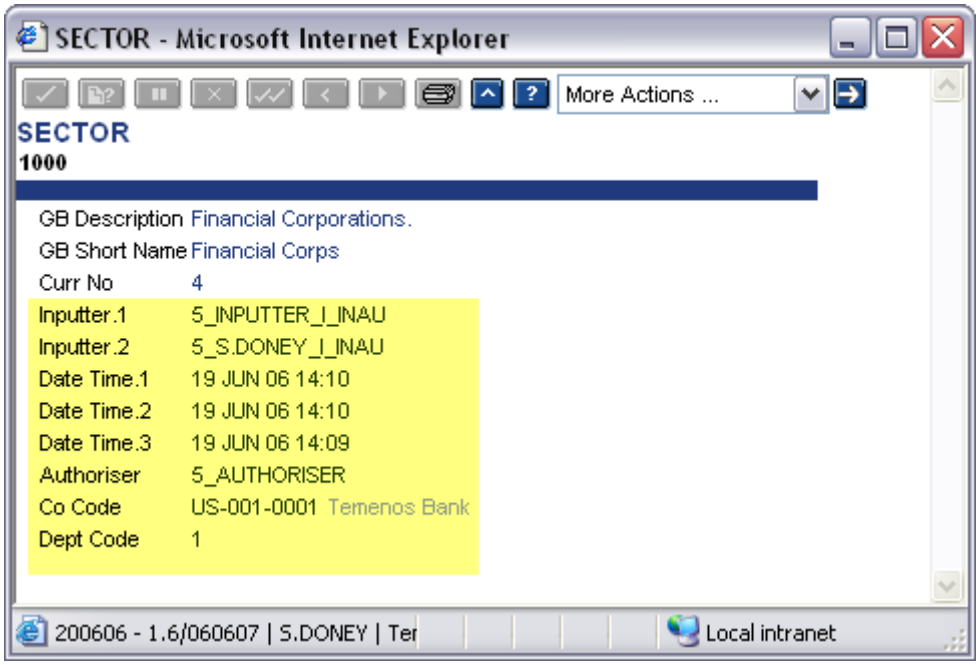


Figure 39 Multiple time and date record updates

User Attributes

The field ATTRIBUTES on the *USER* profile can be used to control specific **T24** functionality. The following options are available and have the following affect:

COMMAND.LINE	The user is allowed the use of the command line in T24 Browser.
DEV.STUDIO	Reserved for future use.
EXPLORER	Allows the user to use the Application explorers
LOCK.DEACTIVATION	Prevents USER access to the User Deactivation listed in Tools dropdown list.
LOCK.DESIGNERS	Prevents USER access to the listed Designer Tools dropdown list.
LOCK.MISC.ITEMS	Will bring up a Security Violation when the User Abbreviations Toolbar, Enquiry and Report lists are used.
NO.ENQUIRY.EXPORT	Prevents USER Exporting Enquiry data from an Enquiry screen, the icon will be dimmed and non reactive.
ENQUIRY.INDEX	Allows access to the enquiry index
REALTIMEENQUIRY	Allows the use of real time enquiries for this user. When signing onto T24, Browser will create another session for use by the real time enquiries. This does use an additional database license, but not an additional T24 license.
LOCK.PREFERENCES	If the user is given this option then the 'User Preferences' option under the 'Tools' menu on the Desktop toolbar will be disabled. This will prevent the user from gaining access to various Desktop settings including file locations and some system administrative functions.
SUPER.USER	The user has access to all of the features detailed above, and for all future functionality with the exception of REALTIMEENQUIRY.

Some of these fields are only used by the Desktop interface and are not used by BROWSER and as such may become obsolete when Desktop support ceases.

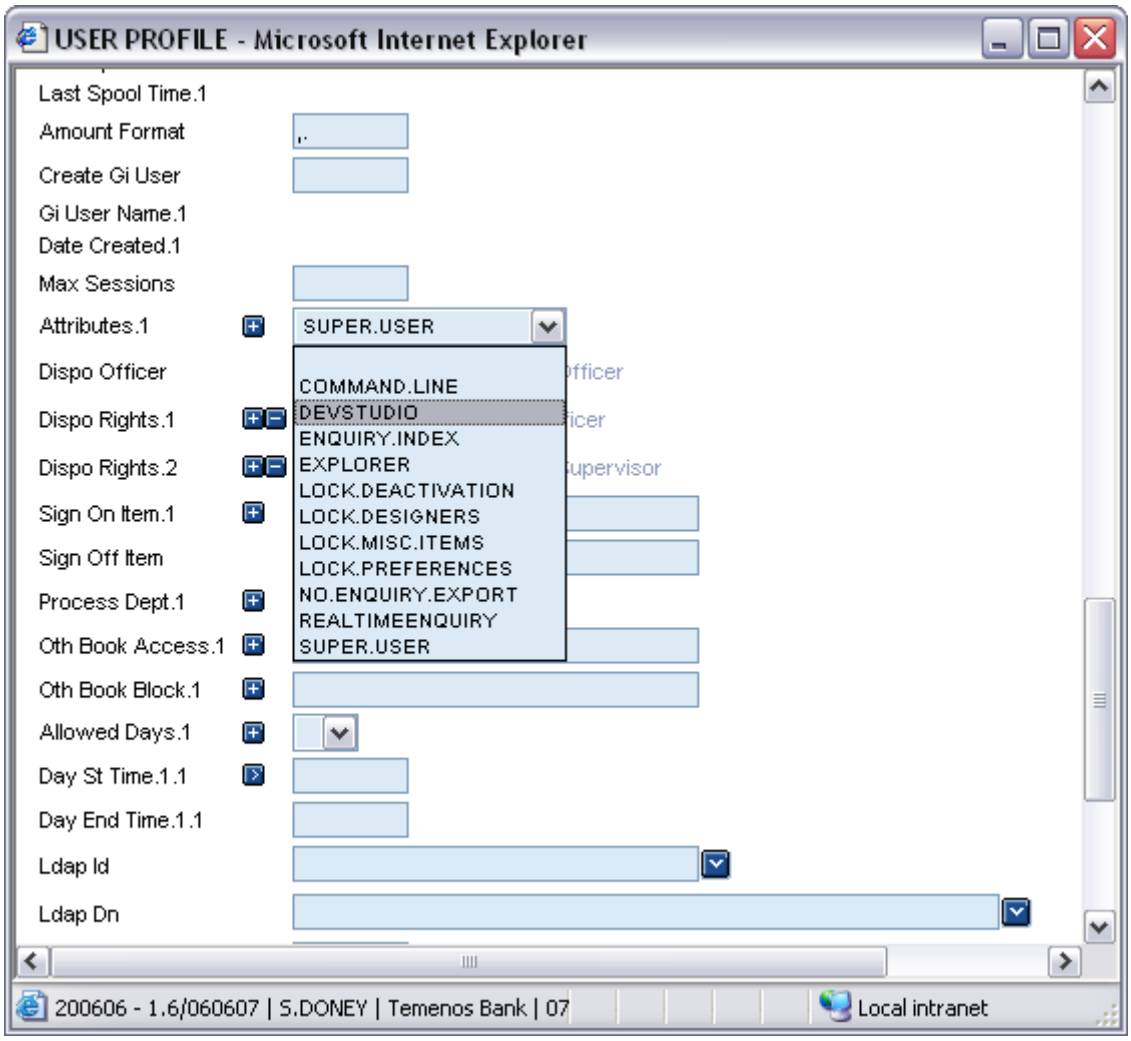


Figure 40 Field ATTRIBUTES can be multi-valued to allocate specific User Attributes

Intelligent Override Processing - The Dispo System

Introduction

Dispo processing is the ability to direct overrides to the people whose attention the override requires, and is in addition to the functionality offered by *OVERRIDE.CLASS* and *OVERRIDE.CLASS.DETAILS*.

OVERRIDE

The *OVERRIDE* table contains a list of the overrides that **T24** uses. The table contains overrides that are used by all modules of **T24**.

The *OVERRIDE* application will eventually supersede the *OVERRIDE.CLASS* application. A utility is provided to convert existing *OVERRIDE.CLASS* records into the appropriate *OVERRIDE* records. The utility is called *CONV.OVERRIDE.CLASS.G10.2*, and is run by verifying the *CONVERSION.DETAILS* record. The utility can be re-run multiple times.

For each *OVERRIDE*, the associated *OVERRIDE.CLASS* and *OVERRIDE.CLASS.DETAILS* information is entered for each application. The wildcard character (*) is used to define all other applications that are not specifically defined.

The `DISPO.ALLOWED` field on the *OVERRIDE* records specifies if a given *OVERRIDE* is available for use with the DISPO system. Where set, the `DISPO` field accepts input and when an *OVERRIDE* is raised from the application a *DISPO.ITEM* record will be created.

Conditional Processing – `CONDITIONAL.OVR`

Some Dispo Overrides are dependent on other Dispo Overrides. For example, if a Funds Transfer transaction causes an account to go into overdraft and there is also a limit override, then both overrides are for Dispo processing. But if the account is not overdrawn, then the limit override is not for Dispo processing. The limit override is dependent/conditional on the overdraft override being present.

Transaction Indicator –`TRANSACTION.IND`

This field specifies if the *TRANSACTION* file needs to be checked to see if the transaction is exempt from Dispo processing. If YES is found in this field then the `DISPO.EXEMPT` field on the *TRANSACTION* file is checked. If the `DISPO.EXEMPT` field contains YES then the transaction is exempt from Dispo processing, and no further Dispo processing takes place. If `DISPO.EXEMPT` field contains NO then normal Dispo Override processing takes place.

Precedence Processing – `PRECEDENCE`

This signifies the order in which applications need to be checked and the field used to determine which *DISPO.OFFICER* the override must be routed to.

The syntax for each of the values in this field is as follows: *APPLICATION*`>FIELDNAME`

Where:

APPLICATION	The name of the APPLICATION to check
>	Separator
FIELDNAME	The name of the field containing DISPO Officer within the APPLICATION

APPLICATION is limited to the application specified in the APPLICATION field on the override table, *ACCOUNT*, *CUSTOMER*, *LIMIT* and *POSTING.RESTRICT* applications.

When the search fails to locate a *DISPO.OFFICER* in the PRECEDENCE list the *DISPO.OFFICER* on the Override record is used as the default *DISPO.OFFICER*.

DISPO.PARAMETER

This application stores settings for Dispo processing.

Only two fields are required in the *SYSTEM* record. The fields are:

`OFS.SOURCE.ID` – **OFS Source Id**.

When a contract is created in a multi-company environment in one company, and then Dispo items APPROVED in another company, an OFS message is created which authorises the original contract.

This field specifies the *ID* of a record in the *OFS.SOURCE* table that has been set up for BATCH processing. The Dispo System uses this data to locate the directory where the OFS message needs to be created.

Should this field not be defined at an instance of an OFS message needing to be created, a warning message is displayed requesting that the user changes company in order for the contract to be authorised.

`DEFLT.CUR.DISP.OFF` - **Default Current Dispo Officer**

Default current Dispo officer that **T24** uses to replace the Dispo officer, if all items on a Dispo Items record have been APPROVED.

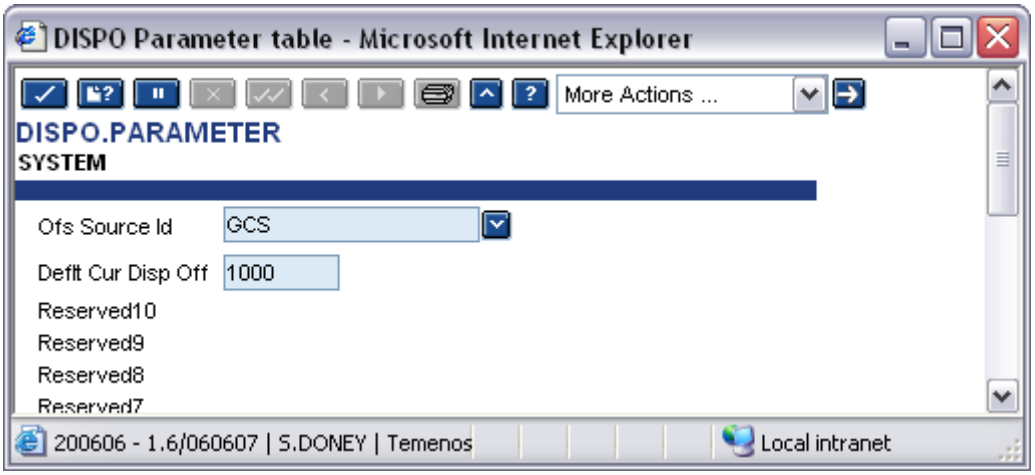


Figure 41 DISPO.PARAMETER record

DISPO.OFFICER

The *DISPO.OFFICER* file allows a hierarchy of officers to be defined that can approve overrides and the amounts that the officer can approve (the OVERDRAFT.AMT). The DISPO.AMOUNT field specifies the amount up to which the *DISPO.OFFICER* is able to make a comment, before routing the item for the attention of another officer.

In the event that a *DISPO.OFFICER* is unavailable for a period of time, messages can be routed to alternative officers for the duration of the period. The field ROUTE.TO is used to specify which officer messages are to be routed to. The DATE.FROM, DATE.TO, TIME.FROM, TIME.TO fields specify the period that the *DISPO.OFFICER* is unavailable for.

Automatic and manual routing is covered later in the guide.

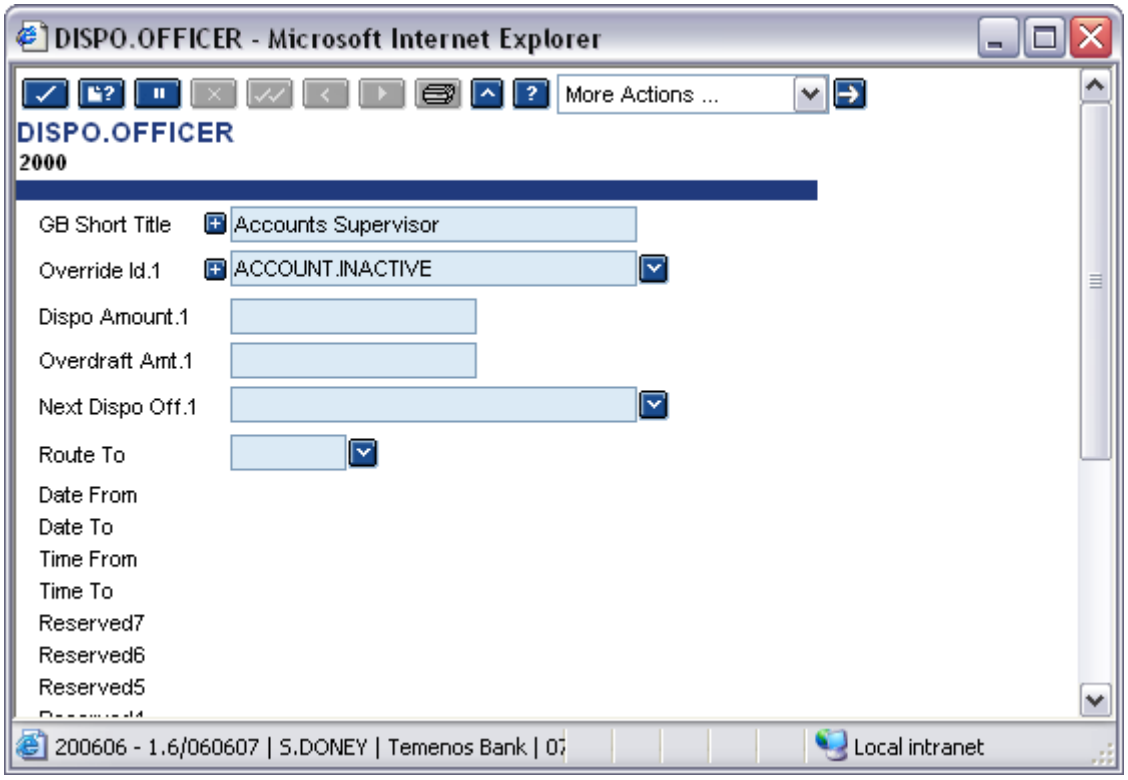


Figure 42 DISPO.OFFICER record

ACCOUNT & CUSTOMER

Both *ACCOUNT* and *CUSTOMER* may be linked to the *DISPO.OFFICER* table. The fields that are used and their meanings are detailed in this table:

Field	Table	Usage
DISPO.OFFICER	*CUSTOMER* and ACCOUNT	The officer who has the overall responsibility for the *ACCOUNT* or *CUSTOMER*. Where there is no *DISPO.OFFICER* specified at the *ACCOUNT* level, then the officer at the *CUSTOMER* level will be used instead.
STOP.OFFICER	*ACCOUNT*	The DISPO.OFFICER who is the responsible party for handling stopped cheques. When a STOP.OFFICER has not been defined then the one used on *ACCOUNT* takes precedence.
XX<POST.RESTRICT	*ACCOUNT*	The *POSTING.RESTRICTION* that the officer in the field RESTRICT.OFFICER is responsible for.
XX>RESTRICT.OFFICER	*ACCOUNT*	The *DISPO.OFFICER* who is the responsible party for the restrictions defined in the field POST.RESTRICT. When a RESTRICT.OFFICER has not been defined then the one used on *ACCOUNT* takes precedence.

USER

The final stage of the installation is to link *USER* records to *DISPO.OFFICER* records, which is done in the *USER* application:

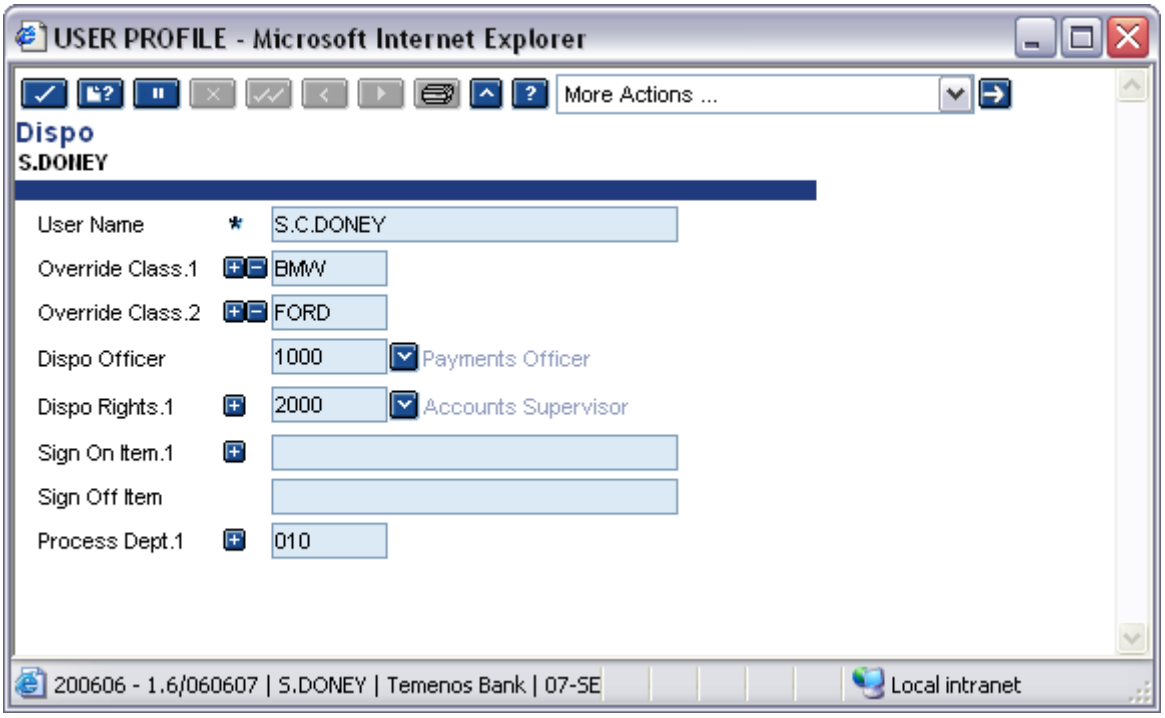


Figure 43 Linking USER records to DISPO.OFFICER records

Using Intelligent Overrides

When an *OVERRIDE* is encountered which is flagged for Dispo processing for the application that raised the *OVERRIDE* (as defined in the *OVERRIDE* table), a *DISPO.ITEM* record will be produced for the contract. Whilst this item is in effect with a status of NEW, the contract may only be authorised by a *USER* whose `DISPO.OFFICER` (set on the *USER* profile) matches that of the item.

The *DISPO.SUMMARY ENQUIRY* displays, for each *DISPO.OFFICER*, the number of items pending both for today, and for previous days.

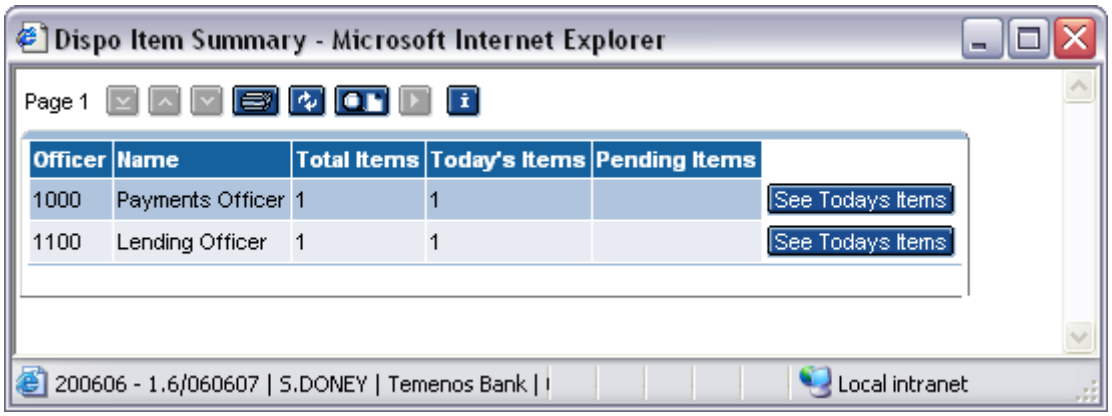


Figure 44 Dispo Summary Enquiry

From this *ENQUIRY*, the *USER* may choose to show the details of today's items for a particular officer, or for the previous days. This invokes the *DISPO.DETAILS ENQUIRY* with the appropriate selection criteria.

The *ENQUIRY DISPO.DETAILS* provides a *USER* with information regarding overrides that require their attention.

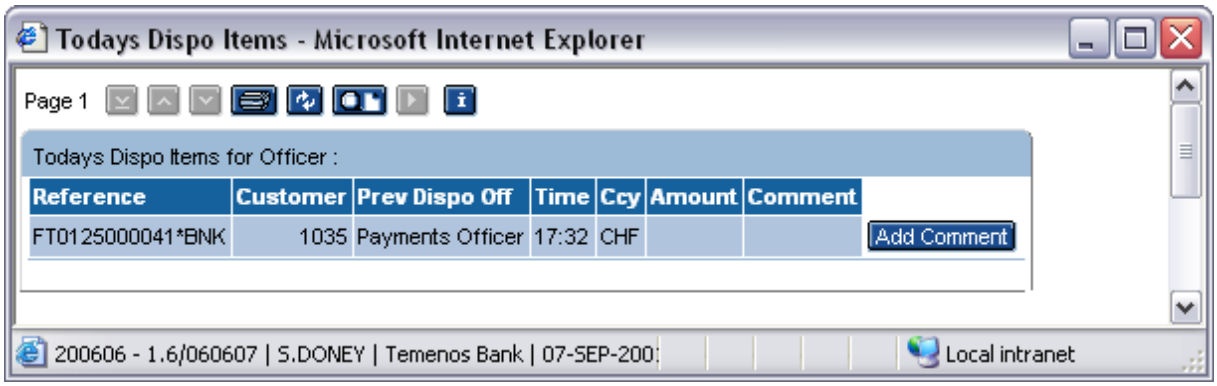


Figure 45 Dispo details enquiry

This is a real time enquiry. Refer to the *ENQUIRY* chapter of the System Administration User Guide for full details on installing and using real time enquiries.

To the user, this means that when an item becomes marked for their attention, the enquiry updates automatically.

From this *ENQUIRY*, the user may:

- Add a comment to the item and forward the item for the attention of another officer.
- Approve the item.
- View the item in full.

Adding Comments to Dispo Items

Comments on a *DISPO.ITEMS* record can be used in two ways. The obvious one of making special notes on the record to record something but the more important usage in *DISPO.ITEMS* is for giving information to the next approver.

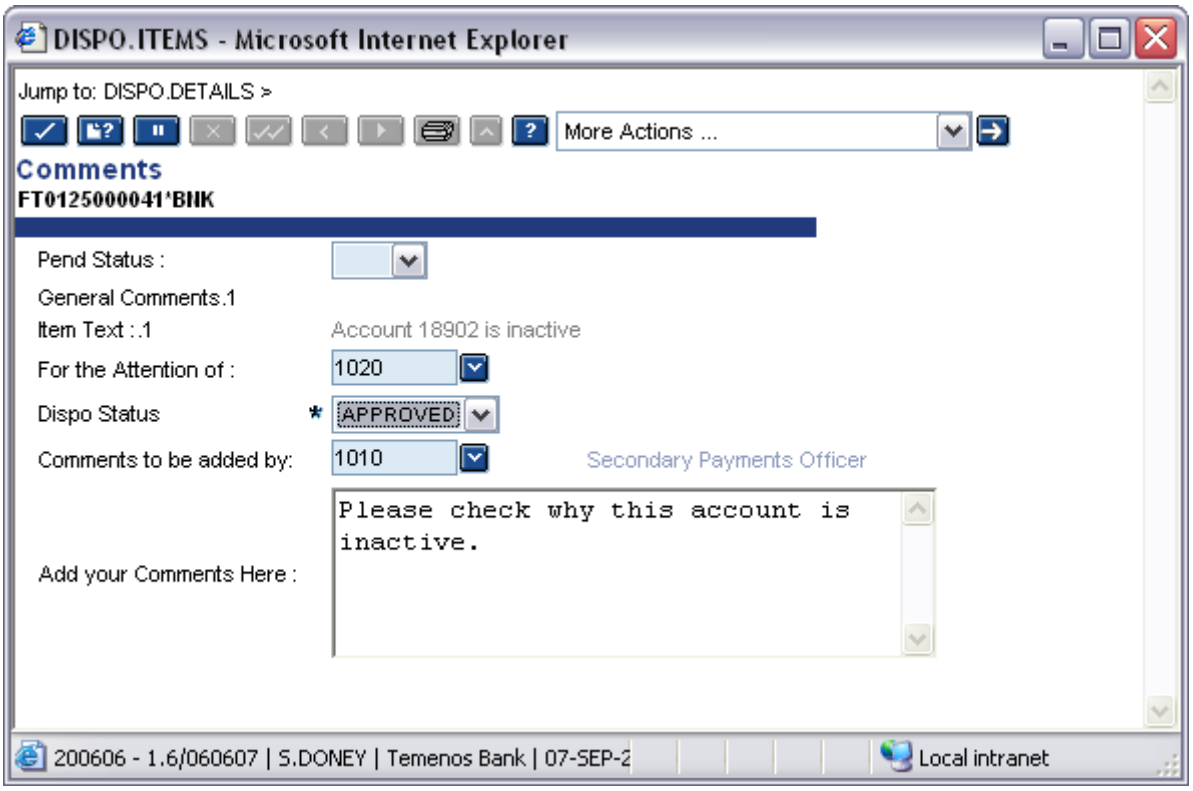


Figure 46 Add a comment to a Dispo Item use DISPO.ITEMS,COMMENTS .

The `COMMENT.OFFICER` field is cleared, and the item is routed to the Supervisor.

It is quite normal in banking to seek an approval from a higher-ranking officer, for something like an overdraft, where a supervisor gathers any necessary information and confirms to the senior officer that the overdraft should be approved in this specific case as any necessary checks or controls have been made.

When a comment is processed in *DISPO.ITEMS* the comments are stored in the `GENERAL.COMMENTS` field and are appended with information on who created the comments.

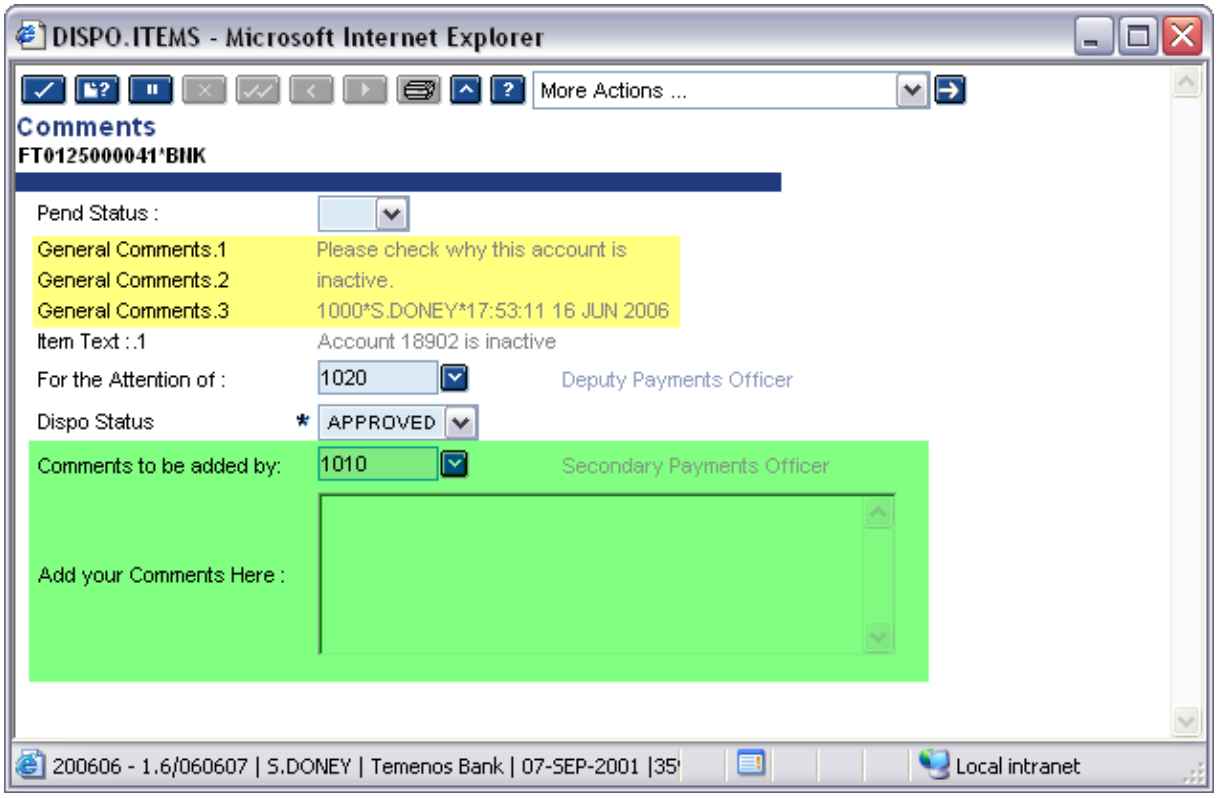


Figure 47 Adding more comments to DISPO.ITEMS record

Below the actual comments an audit type line is added which is comprised as follows:

DispoOfficer*Operator*DateTimeStamp

Dispo Officer The Officer defined in the `DISPO.OFFICER` field on *USER* profile record.

Operator The user that performed the update.

DateTimeStamp The date and time the update was made.

Manual Routing

There are two types of manual routing:

- Comment Routing
- Approval Routing

Comment routing allows multiple *DISPO.OFFICER* to comment on an item before it is routed to the *DISPO.OFFICER* to carry out the approval. It is the `DISPO.OFFICER` of the *DISPO.ITEMS* and not the `COMMENT.OFFICER` that controls which *USER* may approve an item.

Comment routing is achieved by entering a new *DISPO.OFFICER* into the `COMMENT.OFFICER` field on the *DISPO.ITEMS*

Approval routing modifies the *USER* who may authorise the contract.

This is accomplished by modifying the `DISPO.OFFICER` field on the *DISPO.ITEMS*.

NB. The *DISPO.ITEMS* records may be updated only via a zero authoriser *VERSION*.

Automatic DISPO Routing

The examples are based on USD as the local currency, and the following information:

Officer Code	Officer Description	Disposition Amount	Overdraft Competence	Next Level of Disposition
1100	Account Office A	15,000	10,000	1101
1101	Account Officer B	20,000	20,000	1105
1105	Account Officer C	999,999,999,999	24,000	
2400	Top Management	0	999,999,999,999	

Figure 48 Examples of Automatic Dispo Routing

An overdraft excess of **USD 12,000** on a limit assigned to Loan Officer 1100 will be automatically routed to that Officer. The excess is lower than the Dispo amount of that Officer but higher than the Officer's competence amount. This officer can only route manually the transaction after having added their comments to the higher level of competence.

An overdraft excess of **USD 17,000** on a limit assigned to Loan Officer 1100 due to the fact that the excess is higher than the Dispo amount of this Officer will be automatically re-routed to the next disposition level => Officer 1101. This Officer has the competence to authorise the transaction.

An overdraft excess of **USD 27,000** on a limit assigned to Loan Officer 1100 due to the fact that the excess is higher than the Dispo amount of this Officer will be automatically re-routed to the next disposition level till the system finds the correct disposition officer => Officer 1105. This officer cannot approve the transaction because his competence level is lower than the overdraft excess. However, because of his Dispo limit, he receives this transaction to allow him to indicate his comments and forward the transaction to the Top Management for authorisation.

The overdraft is **USD 120,000**, but the limit of **USD 100,000** is not exceeded because the credit balance of **USD 50,000** from another account is being used to offset the debit balance. Should the credit balance be reduced below **USD 20,000** or removed from offsetting the facility, then overrides will be produced to alert the user that the limit will be exceeded.

Pending Status

Each *DISPO.ITEMS* record has a Pending Status flag, which is stored in the field PEND.STATUS. This is used by T24 to determine which items are to be re-routed back to the officer that last had the item following overnight processing. If an item is held overnight, with this flag being set to NO or left empty, then the *DISPO.ITEMS* record will be reset to the first *DISPO.OFFICER* as per the parameters for Precedence processing, routing for unavailable *DISPO.OFFICER*, etc. If this flag is set to YES, then the *DISPO.ITEMS* record will not be altered.

Tracking updates made to Dispo Items

Each time a *DISPO.ITEMS* record is updated the GEN.COMMENT field is updated with the contents of the COMMENTS field and the following details are appended:

The Dispo Officer that made the update, the User ID of the User that made the update, followed by the time and date that the update was made.

Approving Dispo Overrides

When the input of a record causes an *OVERRIDE* and that *OVERRIDE* is marked for DISPO then a *DISPO.ITEMS* record is created. The status of each DISPO item on the record must be marked "APPROVED" before the originating record can be fully authorised. Only a *USER* who has a DISPO.OFFICER record with the application and appropriate competence will be able to approve a disco item.

Only after all the items within a *DISPO.ITEMS* record are approved, can the contract record be finally **A**uthorised. The normal rules for a USER's applications and *OVERRIDE.CLASS* apply.

The *DISPO.ITEMS* record may be updated only after all related non-dispo authorisations have taken place on the originating record.

In a multi-company environment it is possible for one of two paths to be followed during authorisation of a Dispo

Dispo can be configured to utilise OFS to perform final authorisation for all transactions that are created in remote Company accounts. The *SYSTEM* record in the *DISPO.PARAMETER* application contains a reference to an *OFS.SOURCE* record that has been set-up to run in BATCH mode. On approval of all Dispo Items within a *DISPO.ITEMS* record, an OFS message with an instruction to authorise the original contract is created.

The second path taken is when *DISPO.PARAMETER* application has not been configured and the original contract requiring authorisation is in a remote company. **T24** displays a message requesting that the user changes company to approve the Dispo.

DISPO.ITEMS,COMMENTS – AUTH.ROUTINE field

The AUTH.ROUTINE field within the DISPO.ITEMS,COMMENTS version has been updated to action a BASIC subroutine called DISPO.NEXT.VERSION when a record has been committed using the DISPO.ITEMS,COMMENTS version.

The BASIC subroutine called **DISPO.NEXT.VERSION** has been supplied to:

For Multi company environments:

- Check if original contract needs to be authorised in a different company.
- If so, is it possible to generate an OFS message to complete the Authorisation?
- If it is not possible to generate an OFS message – advise the user to change company to the company where the contract was originally created.

For Single company environments:

- Calculate the command required to complete authorisation of the original contract.
- Store the command in R.VERSION(EB.VER.NEXT.VERSION).
- T24 performs the command stored in: R.VERSION(EB.VER.NEXT.VERSION).

If it is not possible to create the OFS message, or the T24 environment is set up for Single Company only, calculate the command required to complete authorisation of the original contract.

The subroutine DISPO.NEXT.VERSION performs the following calculation when calculating the name of the version that is required to complete the authorisation of the original contract.

Can "**Application,DISPO**" version be found in VERSION table?

No? Can "**Application,**" version be found in VERSION table?

No? Can "**Application**" version be found in VERSION table?

Once the version name is known, a command is constructed. This command is comprised of:

VersionName* A *OriginalContractId

VersionName	The version that performs final authorisation of original contract.
A	The Authorise function
OriginalContractId	The Contract ID of the Original Contract.

A COMMON variable R.VERSION(EB.VER.NEXT.VERSION) is updated with the command to be performed. At this point no updates are made to the *VERSION* table.

Finally, T24 performs any command found in the COMMON variable R.VERSION (EB.VER.NEXT.VERSION).

Final authorisation can then be performed by viewing the original contract displayed, and committing the contract.

If the Application,DISPO VERSION record has the NEXT.VERSION field set to DISPO.ITEMS,COMMENTS, then once the contract has been authorised, the DISPO.ITEMS,COMMENTS window will be displayed ready for the next item.