MGB Production manual
Version 2.0.21
Date: 2015-12-30

Revision log

Date	Document Release	Name	Description
02.02.2004		M. Boerner/J. Richelsen	First Release 1.0
24.03.2004		M. Boerner	Update installation stuff.
15.04.2004	1.2	M. Boerner	Update Backup / Archive chapter.
03.09.2004	1.2.1	M. Boerner	MGB Version 1.2 changes (New front-ends for user management and trader, Enhanced authorization concept, Implementation of single sign on) Separation of development and production (Responsibilities, Hardware) Expanded chapter "Regular Operation Instructions" Added user management process
22.10.2004	1.2.2	M. Boerner	Added Environment variables for unix and description of logging files.
03.11.2004	1.3.0	J. Richelsen	Added user role/department dependency.
15.01.2005	1.3.1	M. Boerner	Added user management to responsibility table and contact table.
26.04.2005	1.3.2	J. Richelsen	Added Java Web Start feature for auto check.
08.07.2005	1.3.3	J. Richelsen	Minor changes due to Hibernate3 and logging paths.
24.02.2006		M. Boerner	Added documentation for Sparkassen MGB.
05.07.2006	1.4.0	J. Richelsen	Luxemburg client
20.11.2006	1.4.3	J. Richelsen	Asia client
23.05.2007	1.4.5	J. Richelsen	Hong Kong client
18.01.2008	1.5.2	J. Richelsen	New server
11.03.2010	1.5.3	J. Richelsen	New FO LoanIQ
30.12.2010	1.6.0	J. Richelsen	Review
18.07.2011	1.8.0	J. Richelsen	Upgrade of tomcat/java
02.11.2011	1.9.0	J. Richelsen	Remove MGB Luxemburg
09.03.2012	1.9.3	J. Richelsen	Reprocess files
01.07.2012	2.0.0	J. Richelsen	Portigon
07.12.2012	2.0.3	J. Richelsen	File delivery of Paris and Prime
10.01.2013	2.0.4	J. Richelsen	New db-port
26.08.2013	2.0.8	R. Steger	Updated contact info for Bloomberg mapping
14.04.2014	2.0.11	R. Steger	Added note about Summit Amendment/Summit Derivative synchronization
14.08.2014	2.0.13	R. Steger	Fixed threshold parameter in section 8.3.8; fixed a few table headers in section 8.3
10.02.2015	2.0.18	R. Steger	Removed a number of obsolete sections (Sparkassen, certain source systems);

			updated troubleshooting information for support handover to RM_PST
02.10.2015	2.0.19	R. Steger	Updated C:D nodes; added documentation for new MGB 2.0.18 features; updated server information for Bluestar FMO environment
29.10.2015	2.0.20	R. Steger	Updated PARIS file support email
30.12.2015	2.0.21	R. Steger	Updated MM/FX file names

Contents

Figures

1 Introduction

This manual contains information pertaining to the production support of the Market Conformity Check tool (MCC). The German name of the tool is "Marktgerechte Bewertung" (MGB). The target groups of this document are operators and supporters who are responsible for the production environment.

1.1 The RM_PST Wiki

The RM_PST Wiki MGB page at http://itrisk.westlb.sko.de:9000/mediawiki/index.php/MGB contains additional information that is aimed at guiding first response to issues that may be identified by the production support team. It should be considered the authoritative source of information for such issues, taking precedence over this document.

2 Responsibilities

Due to the complexity of the tool for Market Conformity Checks (MCC, or MGB in German) several departments are sharing responsibilities.

Group	Organizational unit	Contact person	Telephone
technical responsibility and second level support	501-41920 IR Market Risk & Market Data Management	Uta Bondorf	1001-5563
First Level Support, application server operation(MGB Intranet) and database operation (contact to database provider)	501-41950 IT Risk Production	Jochen Pick	1001-3205
User management	501-45210 Security Management	Horand Staeck	1001-9457

The details of the scope of services and for each group can be found in the appendix.

The responsibilities/contacts for the various clients can be extracted most easily by querying for the controller who was last active in the database. This is possible with the following SQL:

```
SELECT t09_name, t01_firstname, t01_lastname, t01_email, t01_nt_id
  FROM t04_trade_historie
  JOIN t01_employee ON t01_id = fk_t04_t01_created_by
  JOIN t09_mandant ON t09_id = fk_t01_t02_mandant
 WHERE t04_id IN (SELECT MAX (t04_id)
                    FROM t04_trade_historie
                   WHERE t04_state_type = 'manual'
                   GROUP BY fk_t04_t10_man_mandant_code);
```

3 System overview

The MGB tool assists the Trade Controller in checking all trades as to whether they are traded at a price conforming to the market. The tool is based on a modern, scalable 3-tier architecture under the operating systems Windows and UNIX.

The business responsibility for these checks is distributed across different organizational units. The tool takes this into account by implementing support for multiple clients. Every data record in the system is assigned to a client using unambiguous client codes. The software contains both client specific as well as common modules.

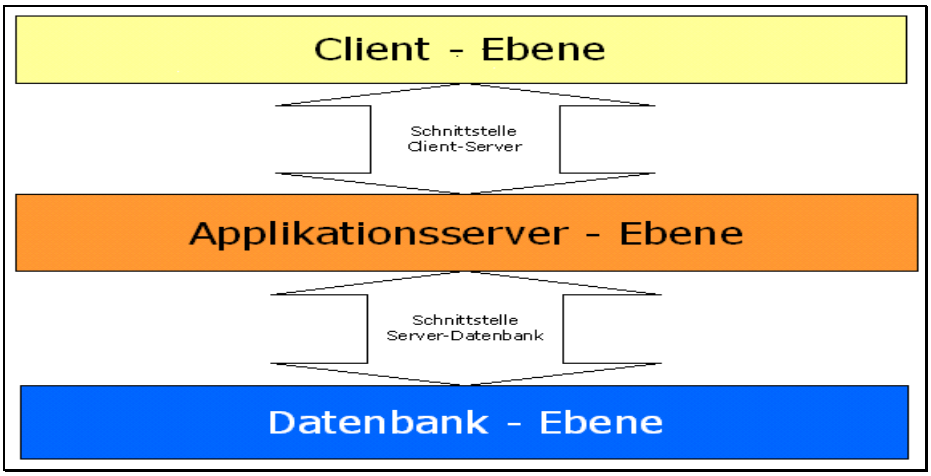


Figure 1: Overview of the 3 layers of the 3-tier architecture

3.1 Client layer

3.1.1 System requirements

A PC that meets the common Corporate Network Standard is expected. This includes the following software:

- Operating system Windows 7
- Microsoft Internet Explorer 8
- Sun Java Plugin 1.6.0_25 or greater (32-bit and 64-bit)
- Microsoft Office 2007 or greater

To retrieve prices from Bloomberg which is necessary to check certain trades, a Bloomberg client license is required as well. Due to the Bloomberg license policy, the request results remain on a local drive on the client.

At least 10 MB of free disc space are required to install the MGB client. The software uses the following directories:

%USERPROFILE%/mgb	Application specific files e.g. templates for printing.
%TEMP%	Temporary deployment location for JACOB DLL

<MARKET_DATA_PROXY_DIRECTORY>	Directory containing bloomberg market data. Can be configured in the application using the function Administration->Configuration.

Additionally, the Java runtime environment uses local directories to cache applet and Java WebStart data. This cache can be viewed in the Java Control Panel.

3.1.2 Description

The user client PC is the local element of the 3-tier architecture. The graphical user interface enables the user to enter all necessary data on the client PC and provides access to some default list views. The following figure illustrates all necessary components and interfaces of the client layer.

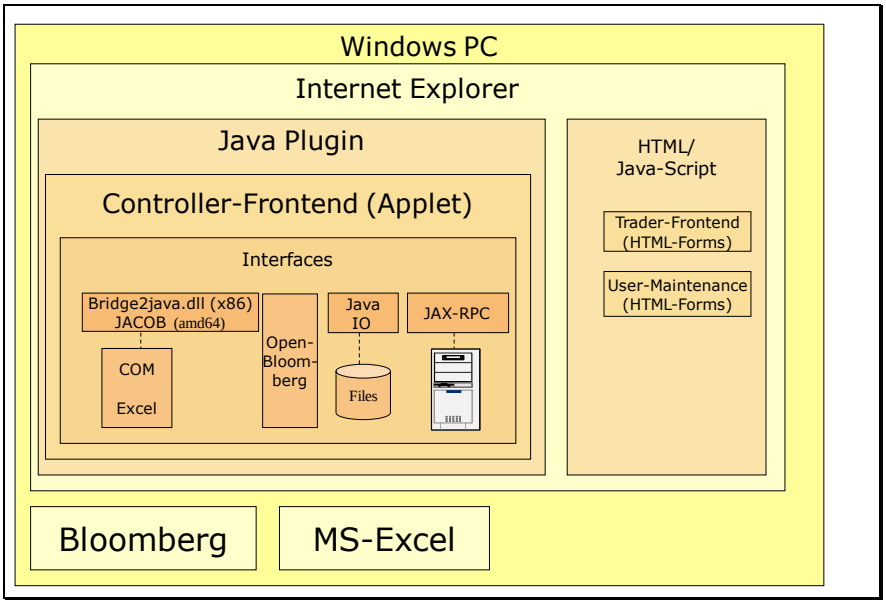


Figure 2: Software components of the client layer

The application client used by the trade controller is a signed Java applet which is executed by the Java Plug-In for the Internet-Explorer. At least the Internet-Explorer and the Java-Plug-In are required to run the rich client of the MGB controller application. Optimal use of all functions additionally requires Microsoft Excel and the Bloomberg client license. When the user selects the appropriate functions, the program calls native functions of these applications by either using a bridge from Java to COM (Excel) or, in the case of Bloomberg, the OpenBloomberg pure-Java API. (Up until MGB 2.0.8, the Bloomberg client was accessed using the older "bbapi", also utilizing the Java-COM bridge. However, this caused problems with Windows 7 clients, necessitating the move to the Java API.) The Java-COM bridge in use is either "Bridge2Java.dll" from IBM, which is copied by the application to the directory `%USERPROFILE%/mgb`, or, in the case of amd64 clients (64-bit IE and Java Plug-In), the JACOB client DLL which is copied into a temporary location and deleted again when the applet exits.

The Excel functionality allows exporting all table views in the tool.

The Bloomberg functionality is required to receive market data using the Bloomberg client license for a selected set of trades and check these trades automatically, i. e. without requiring the user to log on to the Bloomberg Terminal and look up the trades. The results of the Bloomberg queries are stored in comma-separated-value files on the client. Trades that do not seem to be market conform can be sent as reclamation to the trader with a few clicks. The function for generating emails makes use of the Java Mail API to communicate with a Lotus Notes server via the SMTP protocol.

All functions to access MGB server data are provided as Web Services by a JEE Servlet Container. The client uses the Open Source library `axis.jar` to access the Web Services. The communication protocol is HTTP.

The parts of the MGB application that are used by traders and User Management make use of HTML and Java Script only. These front-ends require Internet Explorer only.

3.2 Application server layer

3.2.1 System requirements

The server side of the MGB application requires a Servlet API 2.3 compliant JEE container and a JDBC driver which supports the currently installed Oracle database release, i. e. uses a class path that includes an Oracle JDBC driver as none is deployed as part of the MGB application. The remaining requirements are a full distribution of a Java 6 SDK environment, the Oracle SQL Loader, and SQL*Plus, the latter two of which are part of the Oracle client installation. It is still a requirement that the environment should support the OCI JDBC driver. MGB server-side shell scripts currently assume OCI connection strings in the Hibernate configuration file.

Only the following runtime environment components have been tested and are recommended for production:

- Oracle Solaris or RHEL
- Oracle JDK 1.6.0_26
- Tomcat 7.0.16
- Oracle Client 11.2.0

3.2.2 Overview

The below figure illustrates the structure of the application server layer. The Web application is packaged in one Web Archive (WAR) file that is deployed to a Tomcat application server. The figure shows some of the main modules of the application:

The `Applet` is embedded in a HTML Page and downloaded into the Java Plug-In at the client's first access.

The `AxisServlet` handles all data requests sent by the applet client. It translates the WebService requests into Java method calls. The mapping is defined in the file `server-config.wsdd`. For details read the Axis user guide [Jakarta-Axis]. The implementation of the services is located in a Java Bean named `MgbServiceImpl`.

The Struts `ActionServlet` is used as dispatcher for all thin client requests. It delegates the requests to action classes which make use of the `MgbServiceImpl` classes as well as the Axis servlet.

The `MgbDataHandlerServlet` polls on a local directory. Whenever the RMS&C Connect:Direct server, or (for some files) local copy operations copy a new file to this directory, it is loaded into the database.

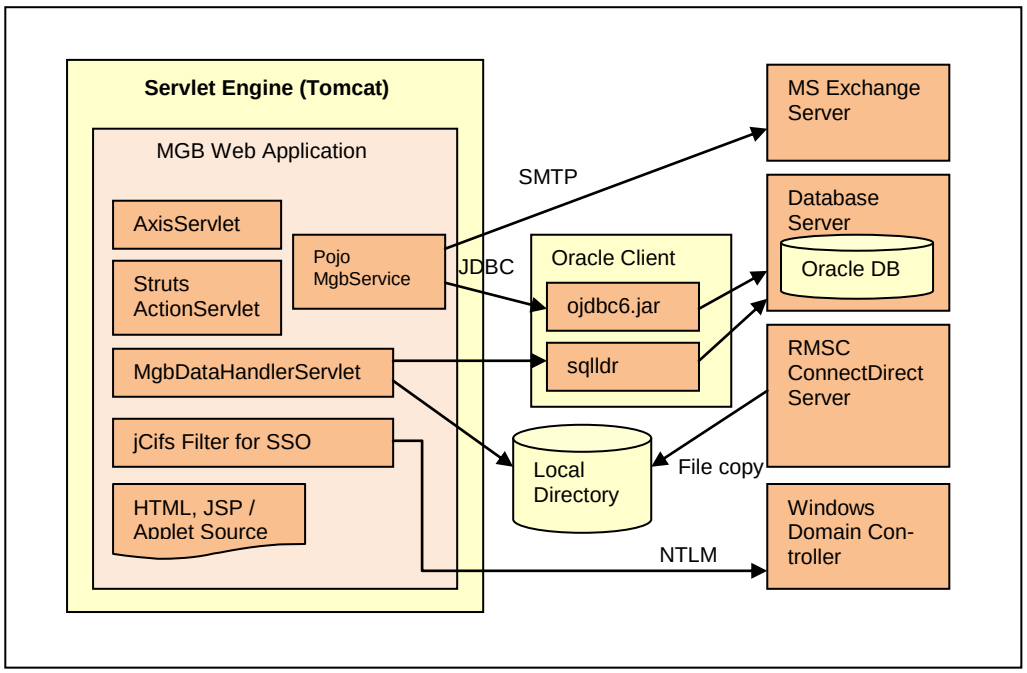


Figure 3: Software components of the application server

3.2.3 The database interface

The following figure illustrates the implementation of the database interface of the MGB tool. This aspect of the application might be helpful to know when you try to improve the application performance.

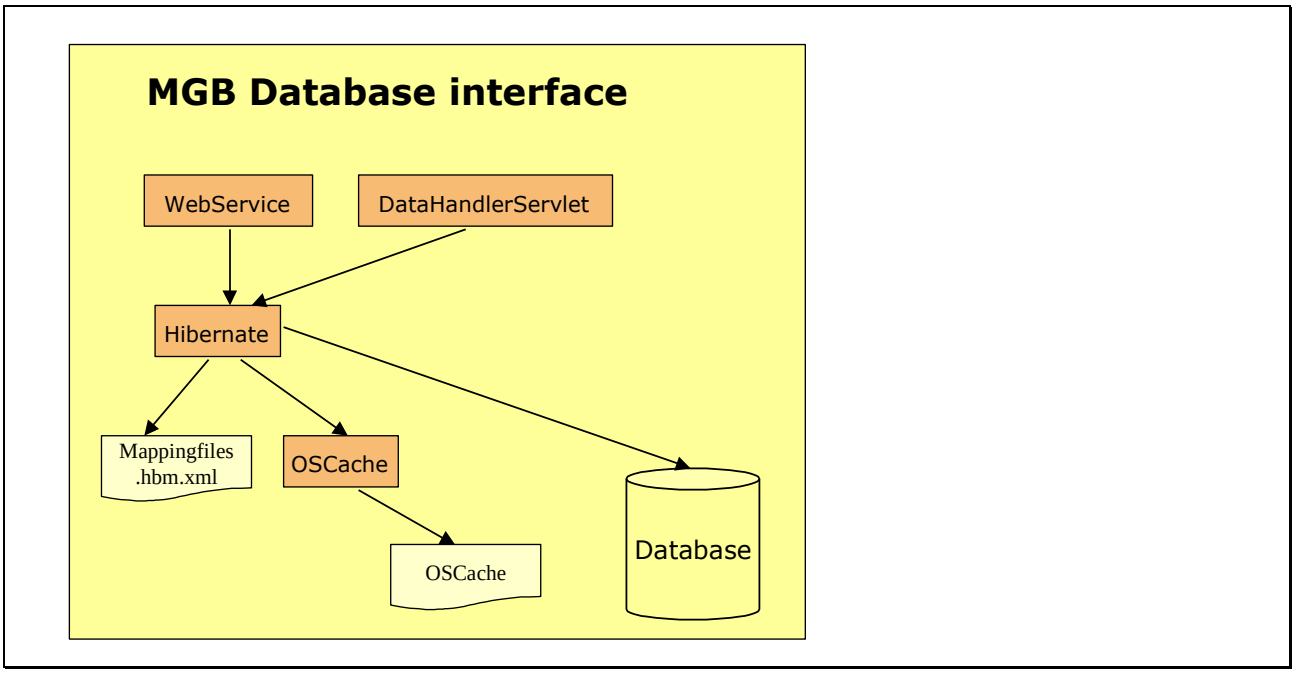


Figure 4: The database interface

All modules except the SQL loader scripts use the Hibernate library for loading Java Objects from the rela-

tional database into memory. Hibernate is a pure Java library under an open source license. The mappings between Java objects and tables as well as the mapping between associations and foreign key links in the relational database are defined in XML Files. The advantages of this approach are:

- Independence from database vendor
- No programming effort for type conversions or maintenance of sequence keys
- Database scheme and upgrades can be generated automatically
- Performance can be improved without programming effort by using standard caching functionality

When the application starts an object search (e.g. select all employees by name) the data will be loaded from the database. But when the application gets data by a primary or foreign key attribute (e. g., retrieve a job and then get all trades associated with this job) Hibernate will try to read the data from an object cache. The cache is in sync only when all write accesses are performed using the web application, i. e. through Hibernate, which is always the case for MGB (but see 7.7.6). For caching, Hibernate supports different libraries. MGB uses *OSCache*, a pure Java implementation under open source license, which is currently one of the most powerful products. E. g., the cache can be larger than the memory of the application server, because the cache would be swapped to disk.

3.3 Database layer

3.3.1 System requirements and description

The database layer is an Oracle database. MGB requires a technical user for the database access with the following roles assigned to it:

Role	Admin?	Default?	OS Granted?
CONNECT	No	Yes	No
RESOURCE	No	Yes	No

All database access is performed using the technical user. Authentication of users is implemented on the application server layer.

MGB has no specific requirements for the operating system of the database server. The following database releases have been tested:

- Oracle 11g

A new database schema can be created by using the Hibernate Tool "SchemaExport". This tool is able to create all tables or a script on the basis of the O/R mapping files. It creates statements for tables, constraints and sequences.

3.4 Environments and hardware

The following figure illustrates the hardware and environments of the MGB application.

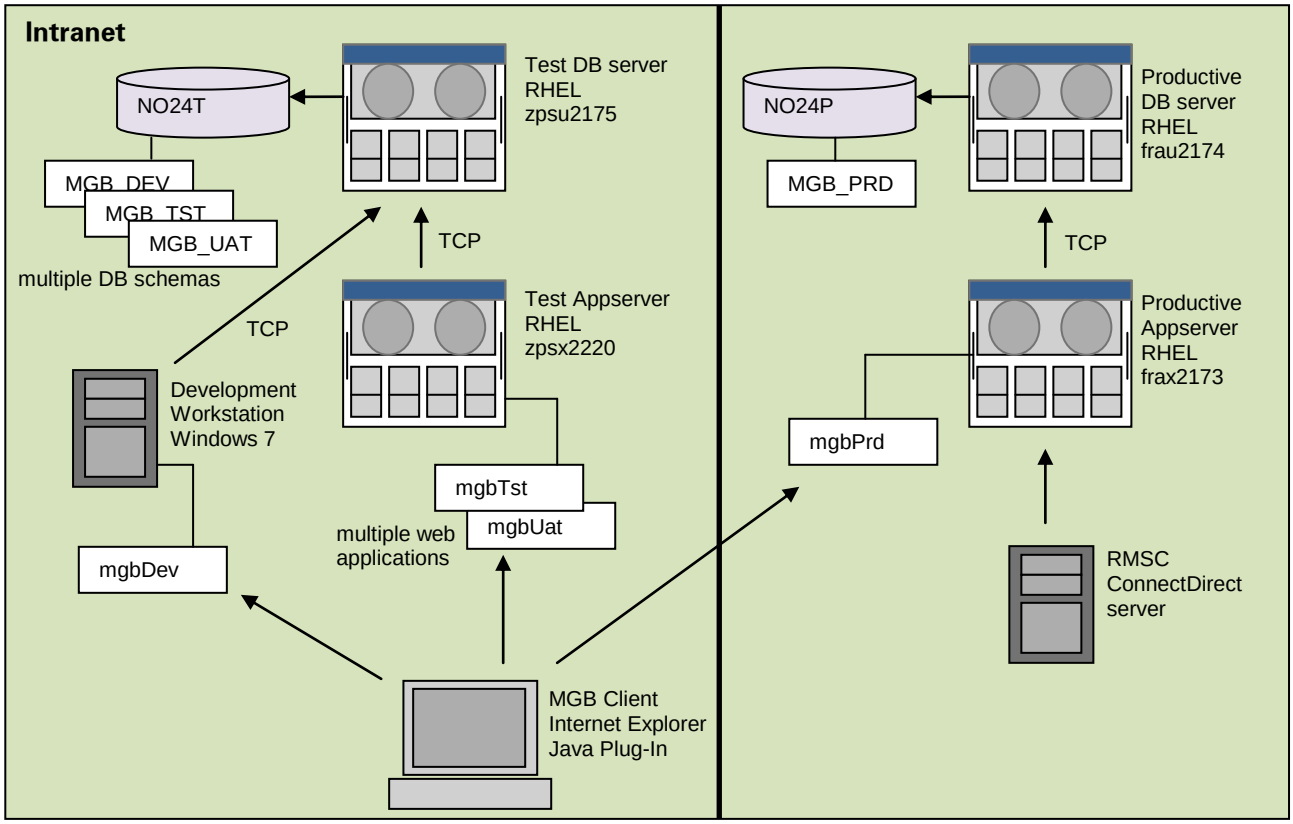


Figure 5: Hardware and environment structure

The following table describes the purpose of the different environments.

Environment	Purpose
mgbDev	This environment is used by the MGB developers for developing new releases and testing changes and migration scripts.
mgbTst	The test environment might be used by a tester of the development team to verify the quality of a new release before it is handed over to the user.
mgbUat	The user acceptance test database is used for the testing of a new release by the business units.
mgbPrd	The production environment.

An MGB environment consists of a Web Application, which is deployed to the application server in a WAR File. The WAR Files for the different environments have the same content, except for some properties contained in `.properties` Files. The most important properties are the database URL, the database user name, the password and the name of the environment displayed in the status bar of the application. Usually the environments will contain different releases of the MGB application.

The databases for the MGB application are split up across two database servers, each of which is shared by multiple applications. The test server contains a database with schemas for the different MGB development and test environments. The following schemas are installed: MGB_DEV, MGB_TST, and MGB_UAT. The production server runs a database instance which contains the schema MGB_PRD for the production environment.

MGB application servers run on three systems.

- The development instance (mgbDev) typically runs on a server instance controlled by the development team, e. g. on a development workstation.
- The test application server runs the user acceptance test environment (mgbUat).

- The mgbTst environment doesn't run continuously; if it is needed, it can be deployed to either a development workstation or the test application server.
- The production application server runs the production test environment (mgbPrd).

4 Security concept

4.1 Intranet Applicaton

4.1.1 Authentication

The MGB application uses a single-sign-on mechanism based on the NTLM protocol. NTLM is a proprietary protocol developed by Microsoft to authenticate users who access web servers or proxies on the Windows platform.

The NTLM protocol allows extracting user, domain name and encrypted (hashed) password from the HTTP headers. The user credentials are verified by the MGB application against an NT domain controller. NTLM is recommended in [Wsys2002] as one alternative for security in intranet applications and already successfully used in some other intranet applications.

The following figure illustrates the details for the single-sign-on mechanism used by the MGB application. Note that it depicts an additional Apache Web Server (Apache httpd) which may be used in a reverse proxy configuration to provide cleaner URLs; at the moment this is not the deployed configuration, meaning users log on to the Tomcat instance directly, starting the process with step 3 in the list below.

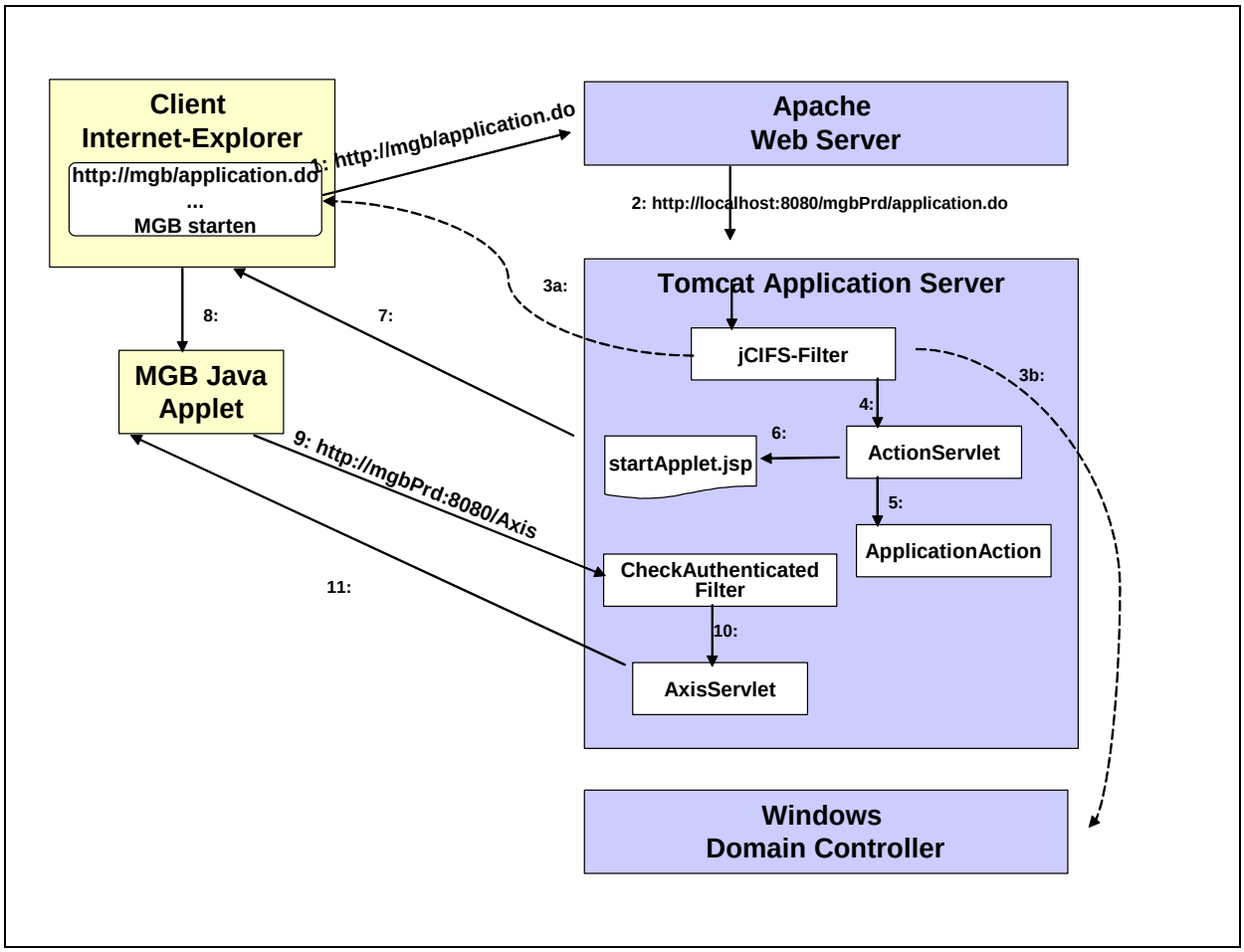


Figure 6: Login process

The following steps are executed when the user logs in to the MGB application:

1. The user enters the URL to start the controller applet e.g. http://mgb/application.do. An http request is being sent to the Apache Web Server.
2. The Apache Web Server that is configured as a reverse proxy routes the request to the Tomcat application server. The URL is converted to http://localhost:8080/mgbPrd/application.do.
3. The suffix .do is assigned to the Struts action servlet. jCIFS is configured to filter all requests of the Struts action servlet, so jCIFS is called first by the application server. jCIFS is an open source library from the Samba project which handles the server side of the NTLM protocol (see [Samba-Jcifs]). It communicates with Internet Explorer and the Windows Domain Controller and adds the NT user ID to the request. The IP address of the Windows Domain Controller is a parameter of the jCIFS-Filter that is located in the file `web.xml`.
4. The Struts action servlet calls the Java action "StartAppletAction" that is assigned to the request application.do. The mapping from the URL to this action is defined in the file `struts-config.xml`. The action checks if the user ID is contained in the HTTP header and checks if the Windows domain is trusted. The list of trusted domains is contained in the file `sso.properties`. If the check fails, an exception is thrown; otherwise the request is forwarded to the JSP startAppletPage.jsp.
6. startAppletPage adds the session ID as parameter to the object tag that will start the applet on the client side.
7. An HTTP response is sent back to the client. The HTTP response is an HTML page containing an object tag.
8. Internet-Explorer starts the Java-Plug-In.
9. The Applet sends SOAP messages to access data from the central database. The session ID is passed as HTTP GET parameter. Because CheckAuthenticatedFilter is configured to filter all Axis requests, it is called first by the application server. The filter uses the session ID to check if the user has been authenticated in steps 3 and 4.
10. If the check performed in step 9 is OK, the request is forwarded to the Web Service.
11. The Web Service response (SOAP message) is sent back to the client.

4.1.2 Authorization

The single-sign-on mechanism described above restricts access to users which are logged in to the NT domain controller. Additionally, the application checks the authorization of the user. The following figure shows the database tables containing the information that is used for authentication and authorization.

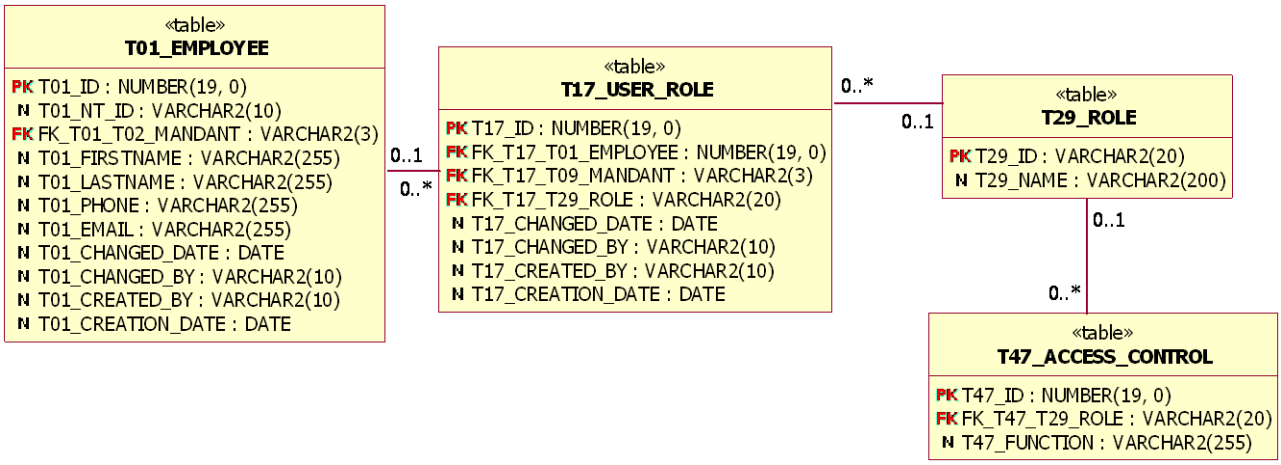


Figure 7: Database authorization tables

Here is a brief description of the tables:

Table	Description	Maintenance
T01_EMPLOYEES	Contains one record for every employee referenced in the database. It is not required that an employee has any roles.	Employees are created or changed by controllers, using the controller applet GUI, or by the user maintenance team. When a new employee is created by the controller application, the role "TRADER" is assigned automatically. (Other than assigning the "TRADER" role, no roles can be granted using the applet.)
T17_USER_ROLE	Contains one entry for every user role. A user role is associated to an employee and a role. It grants a subset of MGB functionality to the user.	The assignment of roles is maintained by the front-end for the user maintenance team.
T29_ROLE	Contains one entry for every role that can be assigned to an employee.	Static data that can't be changed by the application.
T47_ACCESS_CONTROL	Contains one entry for every function that is granted to a role.	Static data that can't be changed by the application.

The following roles have been defined to authorize an employee to use the MGB application:

Role	Privileges
TRADER	Traders are allowed to view data of a trade and to enter a comment, if the controller requested a comment.
CONTROLLER	Controllers have all privileges except the privileges for the maintenance of user roles and the specific admin privileges.
AUTO_CONTROLLER	Denotes a user with special privileges to set price information in trades. It is used by the non-interactive Java WebStart Bloomberg check program (see 7.2.1). This role should only be assigned to the special user *automgb* (once for every client).
ADMIN	The role is only required on test environments and should not be assigned to any employees in the production system! The administrator has the same privileges as a controller. Additionally they have access to the following functions: **Delete Jobs:** This function might be required if the loading step of a job ends in an error status or if the required storage time for the trades of a job expires. **Change user ID:** This function is useful to test the "dual control job" functionality (see 5.3) with a single user account. It bypasses single-sign-on.
SYSTEM_ADMIN	The role has all read-only privileges. Additionally, the system administrators has access to the following functions: **Delete Jobs:** This function might be used if the loading step of a job ends in error status or if the required storage time for the trades of a job expires. **Create jobs:** This function might be used when a file is loaded manually after an error was fixed.
USER_MAINTAIN_ADMIN	The role has privileges to find, view and maintain employee data and roles.
SPK_ADMIN	This role has privileges to create reports of trades with "Sparkassen" *(deprecated)*
READ_ONLY	The role has all privileges except for functions that modify any data.
REPORTER	This role has only access to a function that creates an Excel report of trades in a defined period.

4.1.2.1 Report location setting

To make use of the REPORTER role, as well as other roles as far as they allow – and are intended for – creating reports, it is necessary to set a report location in the T17_LOCATION column of the T17_USER_ROLE table. This report location is either one of the values in the T120_REPORT_ID column in the table T120_REPORT_CONFIGURATION, or the special value "ADMIN" to permit creating reports for all locations.

4.1.3 User Interface for the employee maintenance

4.1.3.1 Controller application

The controller front-end contains some basic functionality for employee maintenance that is required for the trade control workflow. The appearance of a new trader ID in the daily received data is an often occurring use case which has to be resolved directly and can best be resolved by the controllers themselves.

So a controller can create a new employee with role "TRADER" and change basic data like the name, email-address and the phone number. Also, they can revoke the privilege "TRADER", for example if the trader changes the department. Beyond that, the controller cannot grant any privileges for the application. They cannot change the assignment of other roles or change the Windows user ID that is used for single-sign-on.

The described basic functionality is required because the MGB application is connected to multiple front office systems.

The following screenshot shows the GUI mask for user maintenance in the controller application. The green icon indicates a trader, the red icon indicates a controller.

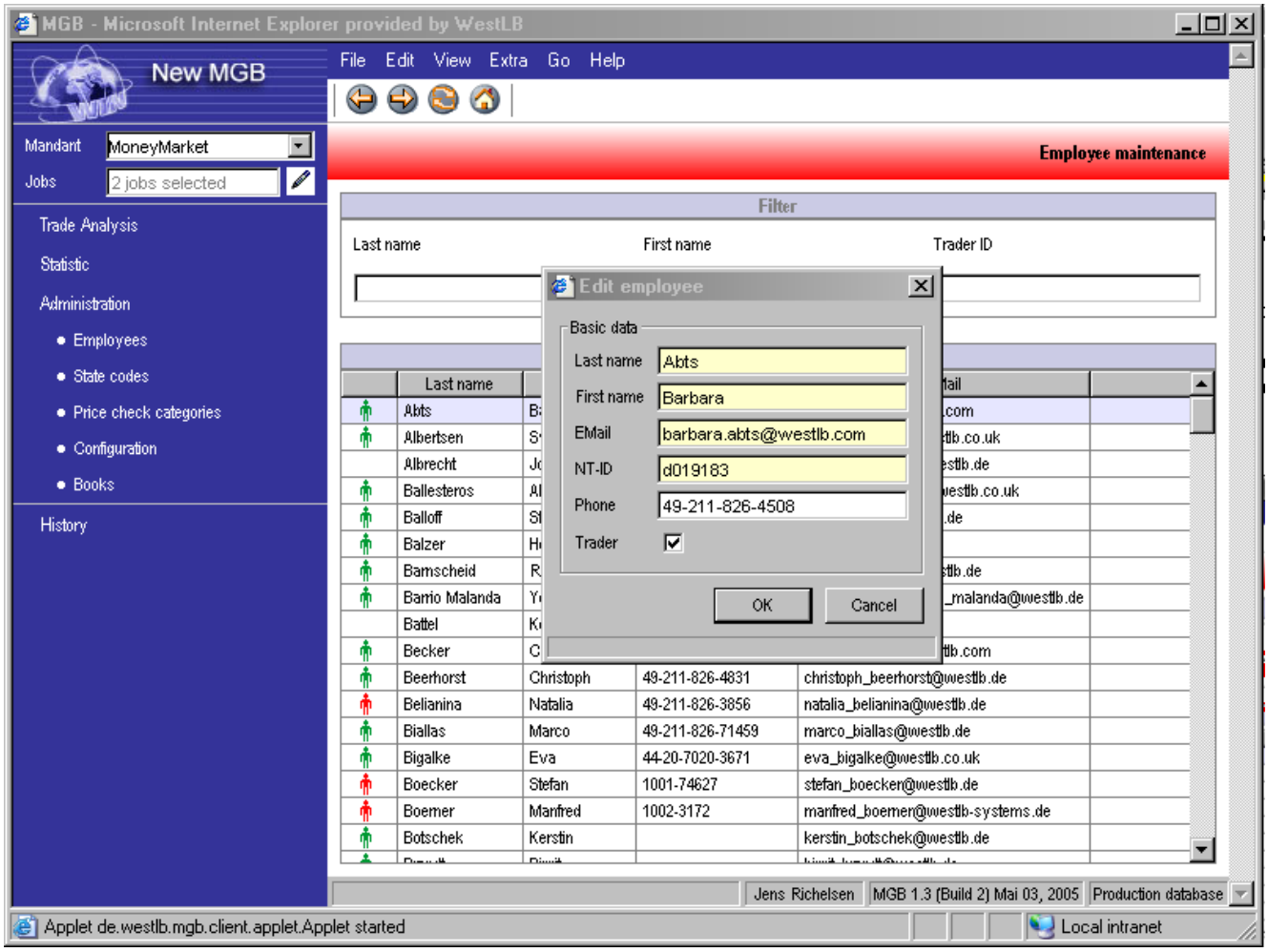


Figure 8: Employee maintenance in the controller front-end

4.1.3.2 User maintenance application

The following screenshots show the masks for the user maintenance. The user list can be entered by clicking on one of the "User maintenance" links in the left area of the MGB main mask.

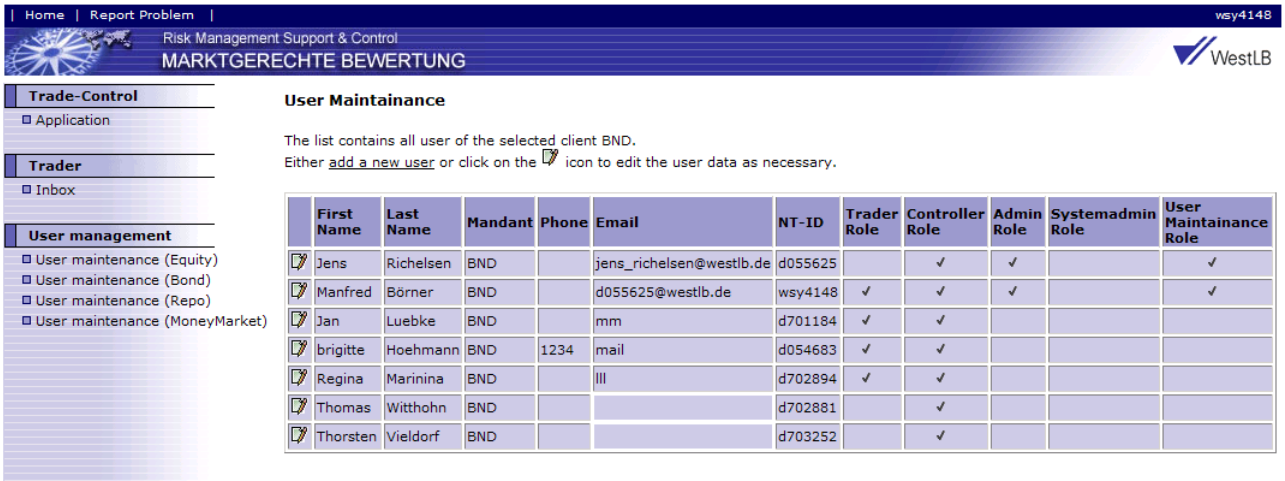


Figure 9: Mask "User list" in user maintenance front end

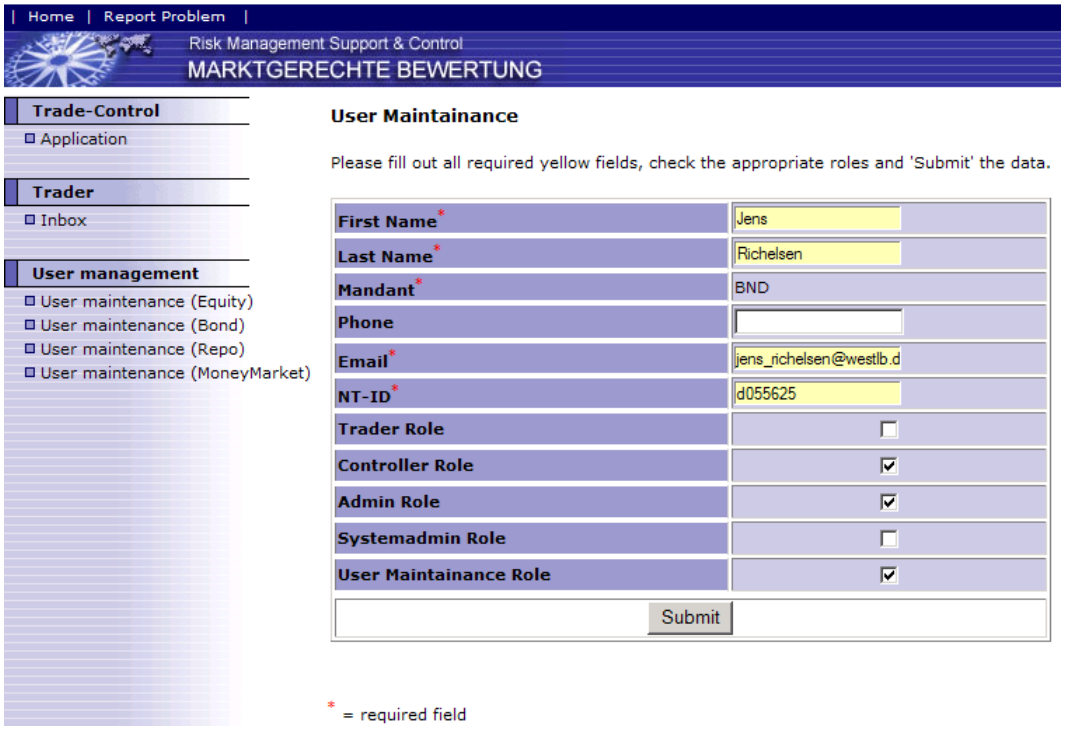


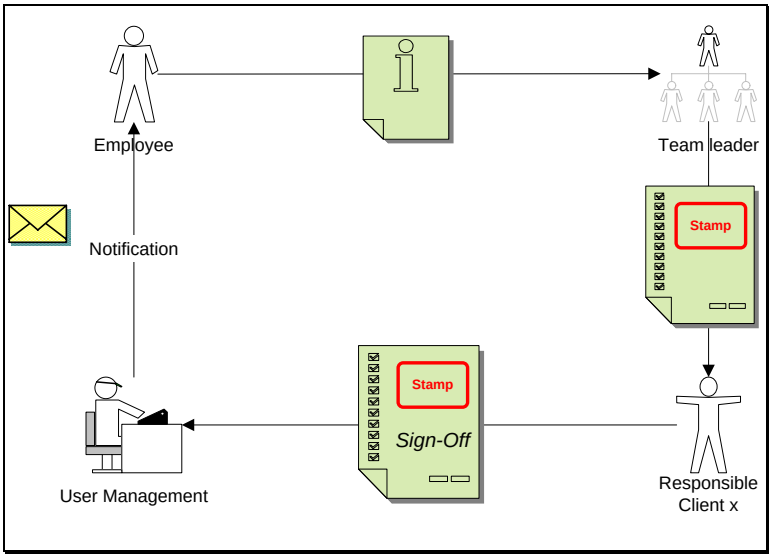
Figure 10: Mask "User detail" in user maintenance front end

5 Processes

5.1 Employee and role maintenance

The following figure illustrates the workflow for authorization changes. The application form can be found in the Intranet at https://westgate.westlb.net/

The employee fills in the online form. The team leader signs the required changes and sends the form to the responsible person of the respective MGB client, who checks and re-signs the requested change and forwards the form to user management. If required, the employee is notified by user management as soon as the change has been implemented.



The various roles are to be given to users in certain departments only, as described in the following table:

Role	Client	Department
TRADER	any	This role is automatically given to employees who are mapped to a front office traderID by a CONTROLLER.
CONTROLLER	any	MRM, separated by location and product group. Details are documented in the "MGB_Usermanagement_Rahmenplan" document
AUTO_CONTROLLER		Technical role which is only granted to the special user *automgb* once per client.
SPK_ADMIN	REP, BND	Department trading with Sparkasse *(deprecated)*
ADMIN	any	The ADMIN role is only required in test environments and should not be assigned to any employees in the production system!
SYSTEM_ADMIN	any	BCP / IT Second Level Support / Production Support
USER_MAINTAIN_ADMIN	any	User Management
REPORTER	any	MRM
READ_ONLY	any	Audit

5.2 Employee and role review

On request of the "User review" team, the current users and roles are reviewed and resigned by the CONTROLLER department above.

5.3 Parameter maintenance

Parameters can only be changed in a process that involves two distinct users (this is called "dual control" in the MGB application). Currently the following data tables are maintained using the dual control approach. Because the tool is using a generic approach for these features, the feature could be easily adopted to cover further data tables.

Table	Description
T13_ADDON	Holds the add-on values that are used when checking the price deviation for special trade types
T05_INSTRUMENT	Holds the mapping between an ISIN, or FX/MM commodity and a priceCheckCategory
T03_EXCHANGE	Contains the Bloomberg (Reuters) exchange codes
T21_EXCHANGE_MAPPPING	Maps the source system exchange codes to the Bloomberg exchange codes
T42_MGB_CONFIGURATION	Contains various parameters like a bagatelle threshold.
T11_PRICE_CHECK_CATEGORY	Contains the parameters (min, max, time) used to check price deviation, grouped in categories

The change of a data item is requested by using the edit masks in the usual way, but instead of changing the data, the software stores a change request job. The existence of a change request job is reflected at the GUI level by an appropriate check box and the name of the requester.

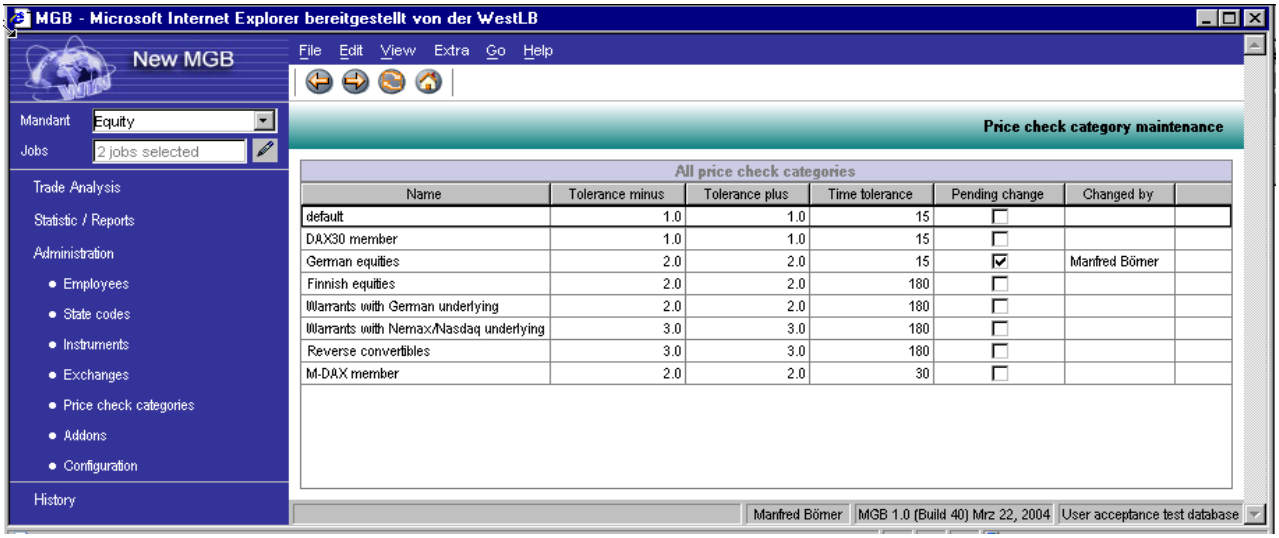


Figure 11: Reflecting change requests at the GUI level

Additionally, all users can use the mask "Dual control jobs" to view all existing change requests in the client. They can see the requestor of the intended operation which might be an insert, update or delete and they can view details to the requested change. All controllers are allowed to confirm all change requests except their own.

The following screenshot illustrates how requested changes are reported and how the differences between old and new record are shown:

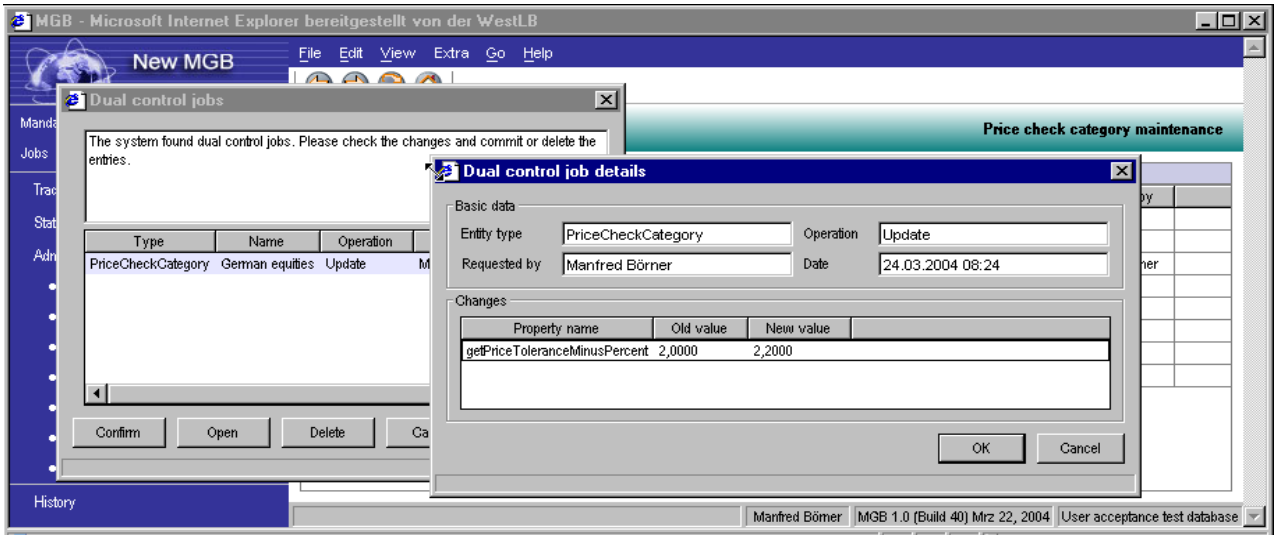


Figure 12: Dual control job mask

5.4 Problem and change management

Any problems and change requests are stored and managed in Mantis, a web-based bugtracking system which is maintained by T-Systems.

Bugtracking tool
http://crms.static.westlb.net/mantis

New problems and change requests can be entered with "Report Bug". All requests should be assigned to a category, which is either the MGB client in which the problem occurred, or one of several predefined generic categories. The developer analyses new bugs and sets the status to "resolved" after fixing the bug. The requester finally closes the entry. If developers change the software, they enter a change request as well. The responsible persons are informed by the Mantis system automatically. Additionally, all changes are part of the release notes generated for the User Acceptance Test.

5.5 Configuration management

The complete history of the source code, as well as any third-party tools and libraries required to build it, are stored in the configuration management system CVS. CVS is a popular open source tool for version management. The CVS Information Page http://www.cvshome.org contains detailed information on CVS.

The MGB project makes uses of a shared instance of CVS used by multiple risk management projects. The backup of the CVS data is included in the backup concept of the department 001-23810 "BCP/Integration Support/Production". The production of the CVS server is not part of the MGB project.

CVS parameter	**Value**
Host	cvs-pfs
Port	default
Repository path	/app/cvs/cvs_repository/riscman
Module	WestLB_RMSC_MGB

To access CVS, one needs a CVS client like Eclipse. The following description assumes that Eclipse is used as CVS client. Of course it is possible to use alternative clients like WinCVS, cvsnt, etc.

5.5.1 Eclipse setup

Select the CVS Repository Exploring perspective in eclipse. Choose "New->Repository Location…" in the context menu of the CVS Repository view.

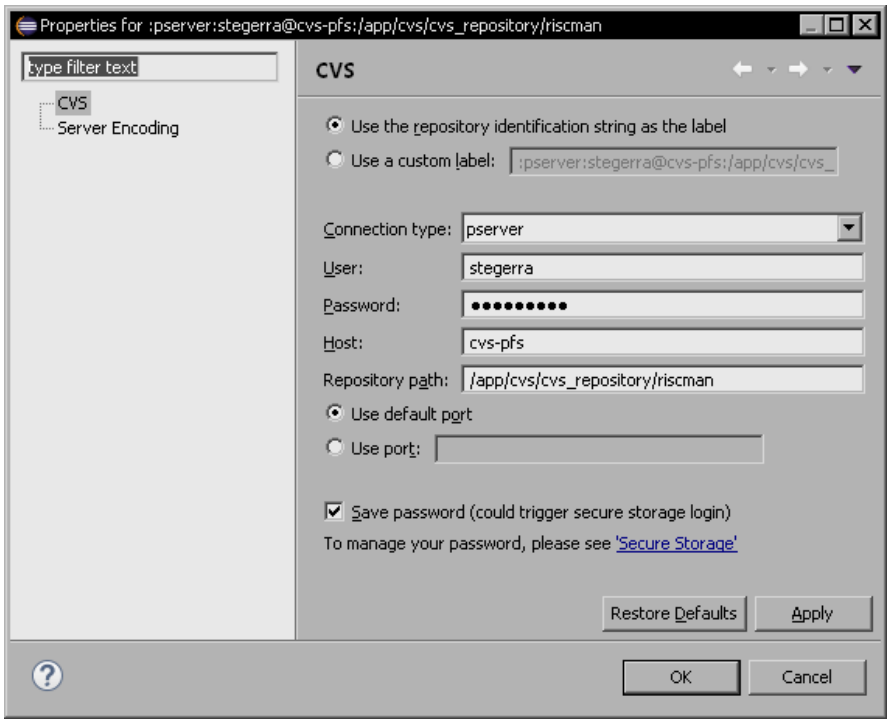


Figure 13: Overview CVS repository setup in eclipse

5.5.2 Freeze a version

To freeze a new release select the project in Eclipse and choose "Team->Tag as Version..." in the context menu of the Java project WestLB_RMSC_MGB. A version has to be frozen whenever source code, parameter files or a prerequisite library are changed from the productive state. Otherwise the build is not reproducible when a rollback to this version is required.

The following convention for the release tag is to be used: {initials}_{major_number}_{minor_number}_{update_number}_{build_number},
e.g. JR_1_2_4_70.

- `{Initials}` are the a two number digit for the initials of the users.
- `{major_number}` is the major release number
- `{minor_number}` is the minor release number
- `{service_release_number}` is the number of the service release
- `{build_number}` is an automatically generated build number stored in the file `build.number`.

5.5.3 Check out an existing version

Select the project and choose "Replace With -> Branch or Version..." in the context menu to access a specific version. Note that changes to the database schema need to be rolled back manually if the old version has conflicts with the recent database schema.

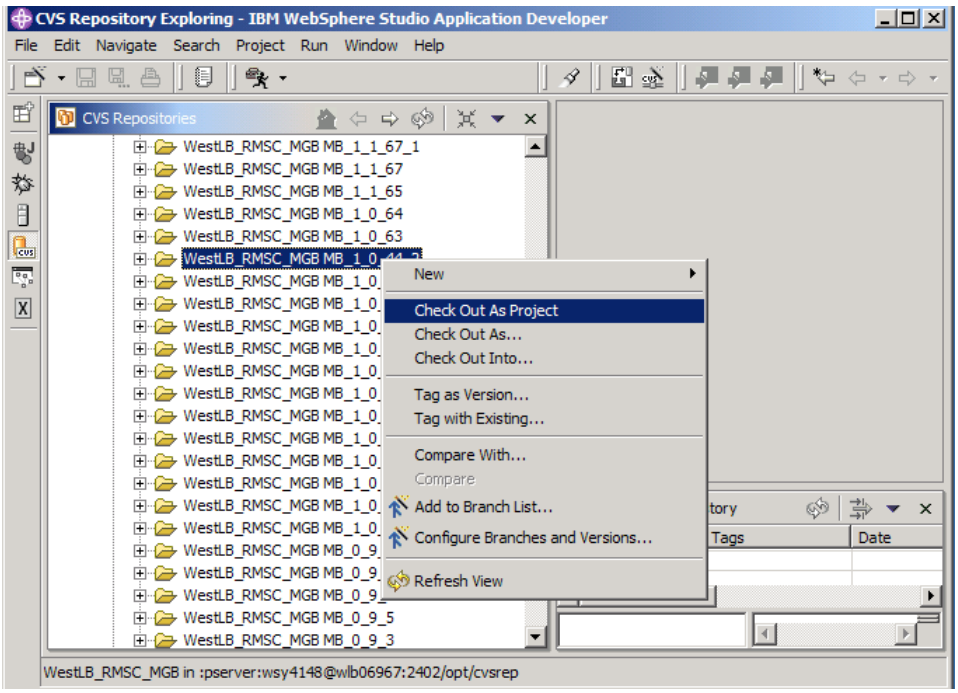


Figure 14: CVS Repository exploring in Eclipse

Binary distributions of the MGB tool are placed in the directory `build/server/`.

5.5.4 Building the application

To build the application, the tool *ant* is used.

5.5.4.1 Intranet application

The configuration of *ant* can be found in the file `build.xml` that is located in the project directory `WESTLB_RMSC_MGB`. To build all environments for development, test/UAT and production just call `ant build.all`. The script then creates deployment files for all environments in the directory `build/server`.

5.6 E-Mail reports

MGB uses cronjobs (running as the MGB user account) to send periodic e-mail reports.

The jobs are implemented as a pair of Korn Shell scripts (`run_all.ksh`, `run_statistic.ksh`) that use SQL*Plus to look up entries in the database table T121_STATISTIC_REPORT. These entries contain descriptive information about when to start the individual reports, and what report script to call with what parameters. The report script is an SQL*Plus script which executes a parameterized SQL statement. After the SQL is executed, `run_statistic.ksh` calls `sendmail` to dispatch the result as e-mail. The list of recipients is also read from T121_STATISTIC_REPORT.

SQL results as well as the generated HTML Code are archived in `$TOMCAT_HOME/../../mgbdata/mgbPrd/report/archive`; if a report is later re-requested by the business team, it can be retrieved from this directory.

The T121_STATISTIC_REPORT table cannot be edited in the MGB controller applet and isn't populated by any server process. It is exclusively maintained through manual SQL access. Below is a description of the columns of that table.

Column	Description
T121_ID	A manually assigned, unique numeric ID
T121_NAME	The name of the report as it will appear in e-mails
T121_MAIL_LIST	A comma-separated list of e-mail addresses to send the report to. Group mailboxes are only supported as far as the Lotus Notes SMTP server accepts them as Internet e-mail addresses, i. e. only if they can also be addressed from outside senders. HP support can normally assign Internet e-mail addresses to group mailboxes as needed.
T121_CRON	A string that begins with a comma, ends with a comma, and in between contains a comma-separated list (without spaces) of day numbers (in the sense of Oracle's `TO_DATE(sysdate,'DDD')` conversion). These specify the days of the year when the report should be sent.
T121_SQL_FILE_NAME	The name of the report script to call; it must be located in `$TOMCAT_HOME/../../mgbdata/mgbPrd/report`
T121_SQL_TEXT	*(unused)*
T121_SQL_START_DATE	An SQL expression of type DATE that should be used by the report script to specify the lower bound of the date period to select data for. The expression will be passed as `&1` to the report script.
T121_SQL_STOP_DATE	An SQL expression of type DATE that should be used by the report script to specify the upper bound of the date period to select data for. The expression will be passed as `&2` to the report script.
T121_SQL_PARAM	A custom parameter to pass to the report script. This is usually used to indicate a client or location selection, which the report script then looks up in T120_REPORT_CONFIGURATION. The expression will be passed as `&3` to the report script.
T121_OUTPUT_FILE	*(unused)*

6 Test

The test process for the MGB tool includes functional tests by the development team and user acceptance tests by all business teams. The test process is managed by the person who is technically responsible for the MGB tools. A detailed description of the test process is contained in the document `MGB Test Concept.doc`. The installation of a new release on the production environment requires the acceptance of the person owning the technical responsibility for the MGB tool who manages the test process.

7 Operating instructions

This chapter describes issues and solutions for the production support team to run the MGB application in a productive environment.

7.1 Application start-up

The application can simply be started by accessing the appropriate URL in Internet Explorer (e. g. http://mgb.westlb.sko.de:8092/mgbPrd).

7.2 Running the automatic check in batch mode on the client

7.2.1 Download and start the client batch

The batch can simply be started by typing the appropriate address in Internet Explorer (e. g. http://mgb.westlb.sko.de:8092/mgbPrd/autoCheck.jnlp). This resource is a launch descriptor for the Java WebStart facility (`javaws.exe`). It downloads the client batch program and runs it from the WebStart cache. The program then starts a Java cron daemon (http://www.opensymphony.com/quartz/tutorial.html) and writes heartbeat information to the database every 5 minutes. In the database, the hostname of a client PC and a cron-like expression define scheduling of the autoCheck process (this is stored in the table `T42_MGB_CONFIGURATION`).

Only if the specified client name is equal to the one from which the JNLP URL is invoked, the process continues by selecting the most recent MGB job and dispatching its unprocessed Bloomberg requests.

To automatically start the process whenever logging on to the client PC, put a batch file in the Windows "Startup" folder that executes the following line:

```
"javaws.exe" http://mgb.westlb.sko.de:8092/mgbPrd/autoCheck.jnlp
```

The automatic Bloomberg request requires that the last valid Bloomberg logon was less than 7 day ago. The user does not need to stay logged on. Do not connect to Bloomberg from another client while you are logged on, because the session gets invalid and the Bloomberg download does not work.

To ensure that the directory structure that MGB needs is present on the client, start the MGB tool once to populate the `%USERPROFILE%\mgb` folder before invoking the client batch for the first time.

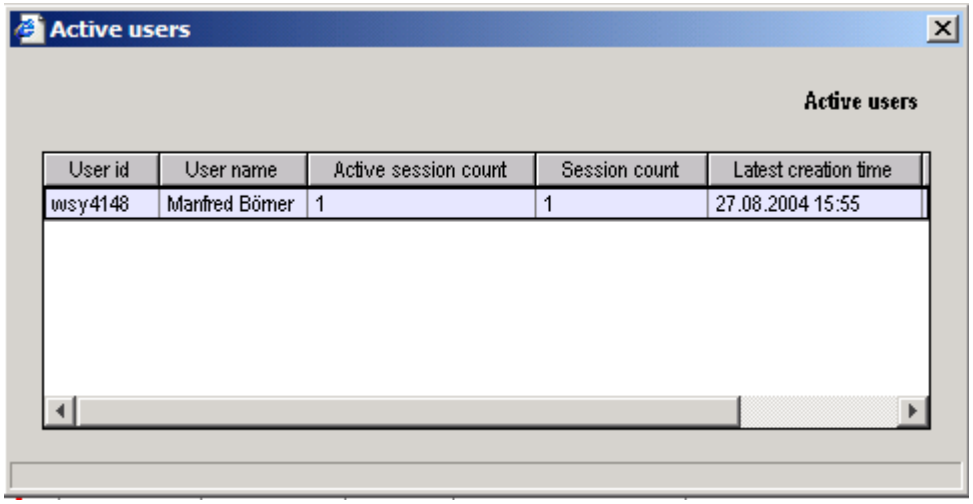
7.2.2 Stop the client batch

Either a change of the name of the "batch enabled client" or a change of the state of the heartbeat task from 'RUNNING' to 'STOP' will shut down the scheduler after the next heartbeat is written to the database. ALternatively, the Java process (`javaw.exe`) may be killed using the Microsoft Task Manager, which is not the recommended method.

7.3 Application server restart

The application server (Tomcat) is running on an RHEL server. Tomcat configured to start automatically at start-up time. In some situations it might be necessary to restart the application server, e. g. if the database password changed or the database server has been restarted. When you need to restart the application server you should check where required if there are users still working or batch jobs are running. To check if

users are working, select the function "Extras->User statistic" in the controller application. This function displays the MGB sessions for all clients.



Since MGB 2.0.18, sessions are serializable and users will normally not lose data during a restart. However, if the expected downtime is unclear, users should be informed. Note that after a restart, the "Active users" view will not list sessions that were carried over.

To check if jobs are running you can check the database table T08_JOB for jobs in state "RUNNING". Running jobs should not be interrupted if in any way possible.

You have two alternatives to restart the application:

7.3.1.1 Restart using the Tomcat GUI

The advantage of this alternative is that it does not disturb any other web application using the same Tomcat instance. However, sometimes this approach does not work, and it may carry a slight risk of memory leaks (this often depends on the *PermGen* – Java heap permanent generation – usage profile of Tomcat or the application).

Start the Tomcat home page **Apache Tomcat/7.0.16** of the application server in the web browser, click Administration->Tomcat manager and login as `admin`. A list of all installed web applications is displayed.

Applications				
Path	**Display Name**	**Running**	**Sessions**	**Commands**
/	Welcome to Tomcat	true	1	Start Stop Reload Remove
/Wiki	JSPWiki	true	0	Start Stop Reload Remove
/admin	Tomcat Administration Application	true	0	Start Stop Reload Remove
/examples		false	0	Start Stop Reload Remove
/manager	Tomcat Manager Application	true	0	Start Stop Reload Remove
/mgbDev	WestLB_RMSC_MGB	true	0	Start Stop Reload Remove
/mgbPrd	WestLB_RMSC_MGB	true	2	Start Stop Reload Remove
/mgbTst	WestLB_RMSC_MGB	true	0	Start Stop Reload Remove
/mgbUat	WestLB_RMSC_MGB	true	0	Start Stop Reload Remove
/tomcat-docs	Tomcat Documentation	true	0	Start Stop Reload Remove

Figure 15: Tomcat Manager

Select the command "Stop" and "Start" for the appropriate web application.

7.3.1.2 Restart using shell scripts

On RHEL, stop Tomcat from the `$TOMCAT_HOME/bin` directory by using the `shutdown.sh` script and ensure that all Tomcat processes are killed. Then start Tomcat by running the script `startup.sh`.

There are also shell scripts in ~a223prod/bin that are designed for operation by RM_PST. These provide a wrapper around Tomcat's scripts with some error checking.

Look at the application log directory of Tomcat (`$TOMCAT_HOME/logs`; but see 11.1.2) to examine any errors.

7.4 Check log files

This chapter describes the location of the application log files. The log files might be checked if a problem is reported by a user.

7.4.1 Tomcat

Tomcat writes log information to files in the `./logs` directory in its installation directory.

7.4.2 Application server

The logging of the server can be configured in the file `mgbServerLog4j.properties` file. A brief description of the most important logging switches is contained in the file. The log files can be found in the `$TOMCAT_HOME/logs` directory.

7.4.3 Data import

`sqlldr` generates a log file for each invocation. It is created in the import working directory and archived in the archive directory. The locations are defined in the `import.properties` file.

The archived files have the following naming pattern:

`<import.system>.<yyyyMMdd_hhmmss>_<job_id>.<ext>`

The `job_id` is a database internal id that indicates the import run. It is shown in the GUI in the job selection lists. `<ext>` is either `log` for the log-file, `dat` for the data file and `bad` for the file with the rejected records.

The `MgbDataHandlerServlet` is also generating output. It can be configured by setting the

`log4j.logger.de.westlb.mgb.converter=warn`

line in the `$TOMCAT_HOME/webapps/mgbPrd/mgbServerLog4j.properties` file from `warn` to `debug`.

7.4.4 Database interface

The database is accessed via Hibernate. To show the SQL statements that are executed against the database, uncomment the

`#hibernate.show_sql=true`

line in the file `$TOMCAT_HOME/webapps/mgbPrd/WEB-INF/classes/hibernate.properties.`

(This file must only allow `u+rw`, i. e. 600 access in a production environment as it contains the database password.)

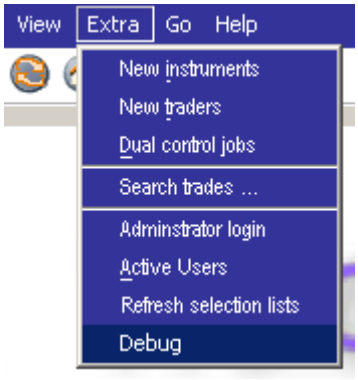
To show even more information, change the `log4j.logger.org.hibernate=warn` line in the `mgbServerLog4j.properties` file from `warn` to `debug`.

7.4.5 Client (Applet)

Client log messages are logged to the Java console file in the directory `%APPDATA%\Sun\Java\Deployment\log` if logging has been enabled in the Java Control Panel (`javacpl.exe`). Another way to display the file content is to select "Display Java Console", also in the Java Control Panel.

Since MGB 2.0.18, all log statements are also written to the `%USERPROFILE%\mgb\mgb.log` file.

To change the log level of the client the following menu item can be selected:



7.4.6 Client (Bloomberg batch)

The files are stored at `%USERPROFILE%\mgb\javaws.log*`. The file extension number indicates older files. The output level is generally set to `debug`. The file is only located on the client where the Bloomberg batch is running.

7.5 Backup

The backup of the data ensures that the system can be recovered in case of hardware defects, faulty operation or migration failures.

Database

The facility to backup the Oracle database is provided by T-Systems. The service is based on standard service level agreements between Portigon and T-Systems.

Market data files

The responsibility for the backup of the locally stored Bloomberg market data is assumed by the responsible business departments. The departments are going to use a network drive as destination for market data files or copy the files regularly to a network drive.

Archiving

The archiving ensures that the trade and check data can be accessed for a time period of at least 10 years. To fulfill it is intended to keep all historical trade data online in the Oracle database. The increase of the database disk space has been estimated as 1 GByte per year.

7.6 Re-running e-mail reports

If an e-mail report is requested by the business team or by auditors some time after it has been sent, it can

normally be retrieved from the archive location (see 5.6) without having to run it again.

Re-running the e-mail reports faces two challenges:

- If the original Korn Shell scripts are used, all parameterization has to happen through the T121_STATISTIC_REPORT database table (see the description in 5.6). This means that the entry for the report has to be altered – or created as modified copy – to
    - contain only certain e-mail addresses
    - contain the desired start/stop dates
    - contain the correct day number to run on (i. e., today's date)
- Regardless of whether the original Korn Shell scripts are used or not, the report cronjob normally is executed at 19:00 Düsseldorf time, meaning it already takes the new Hong Kong COB into account, but only the old ones for other source systems as the first non-Hong Kong COB arrives at 23:00. This means that the stop date for re-runs has to be set to 19:00 proper, which is different from what the reports normally do. Otherwise jobs between 19:00 and 23:59 that day would be added to the original run.

To make it easier to ensure the correct cut-off time for the re-run and at the same time, avoid side-effects for later reports, it is recommendable to

- not use the Korn Shell scripts, and
- leave T121_STATISTIC_REPORT untouched.

Instead, open the SQL script referred to by T121_STATISTIC_REPORT in an SQL client, such as Oracle SQLDeveloper. Then copy the three parameter expressions from T121_STATISTIC_REPORT into the same worksheet and assign them to `&1`, `&2`, and `&3`, respectively. Take extra care not to leave a `TRUNC()` function around the stop date, neither for parameter assignment nor in the actual SQL, but to use the cronjob's cut-off time instead.

The SQL can then be executed interactively in order to export the result from the SQL client as XLS, tab-separated text, etc.

7.7 Troubleshooting

This chapter describes some known problems that might appear during operation.

7.7.1 Error message "Applet finished" during application startup

When a user reports that the message "Applet finished" occurs at application startup time you should do the following steps to solve the problem.

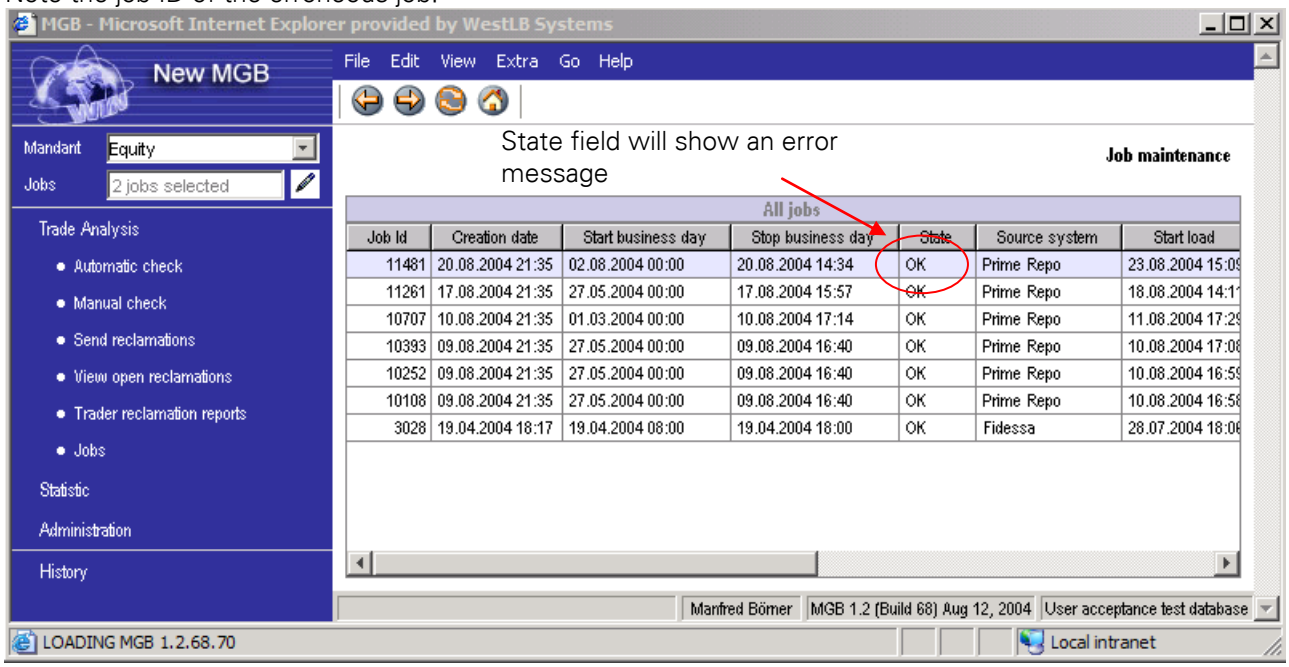
1. Try to login yourself.
2. **If you get the same message:** This means that there is a problem with the application server. In most cases the database system has been restarted and the cached database connections are stale. Restart the Tomcat process and try to log in again. Check the application log files.
3. **If the login succeeds:** Check the Java console output of the client to get an idea about the problem.

7.7.2 Controller reports a job could not be loaded

When a user reports that the loading phase of a job failed, perform the following steps to solve the problem.

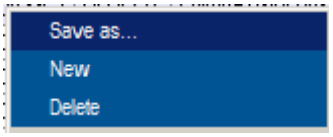
1. **Check the job status in the application**: Log in to the controller application and select the "Mandant"(client) where the problem occurred. Then click on the folder bar item Trade Analysis->Jobs to display the job list in the application. The job list will contain a line that shows an error message in the

state column and some basic information about the job like number of load errors, convert errors etc. Note the job ID of the erroneous job.



2. **„Error while converting…" message:** The reasons for this problem can vary. The log files on the server need to be analyzed, searching for the job ID noted above.
3. **„Error while loading data…" message:** There might be several reasons for the problem, e. g.
   - the trade count in the last line does not match the number of lines in the file,
   - a line contains a value that is too long for a string field
   - a string field contains an invalid character
   - an erroneous delimiter shifts the fields of a row

   To get more details, open the "Log view" window via the "Save as…" context menu entry:



   The window shows the `.log` and the `.bad` files left behind by the SQL Loader. It also offers to save the data file locally for further analysis.

   Repeat load: If the file is incomplete or contains an invalid format the contact person for the interface has to be informed and the loading process has to be repeated. It can be started manually by using the function "New" in the context menu of the job list.

7.7.3 User reports problem "You are not authorized to use the MGB application!"

This problem occurs when the user is indeed not authorized to use the MGB application or the user is logged in with a Windows account ID that differs from the ID maintained in the MGB application. Ask the user for the name and Windows ID and check employee data in the MGB tool (see chapter 4.1.3, User Interface for the employee maintenance).

7.7.4 User reports missing file

The files from the various front office systems are transferred via Connect:Direct.

- Details regarding the transfer can be found in chapter 8.3 (Source systems).
- A list of contact persons can be found in chapter 9.2 (Service and contacts).

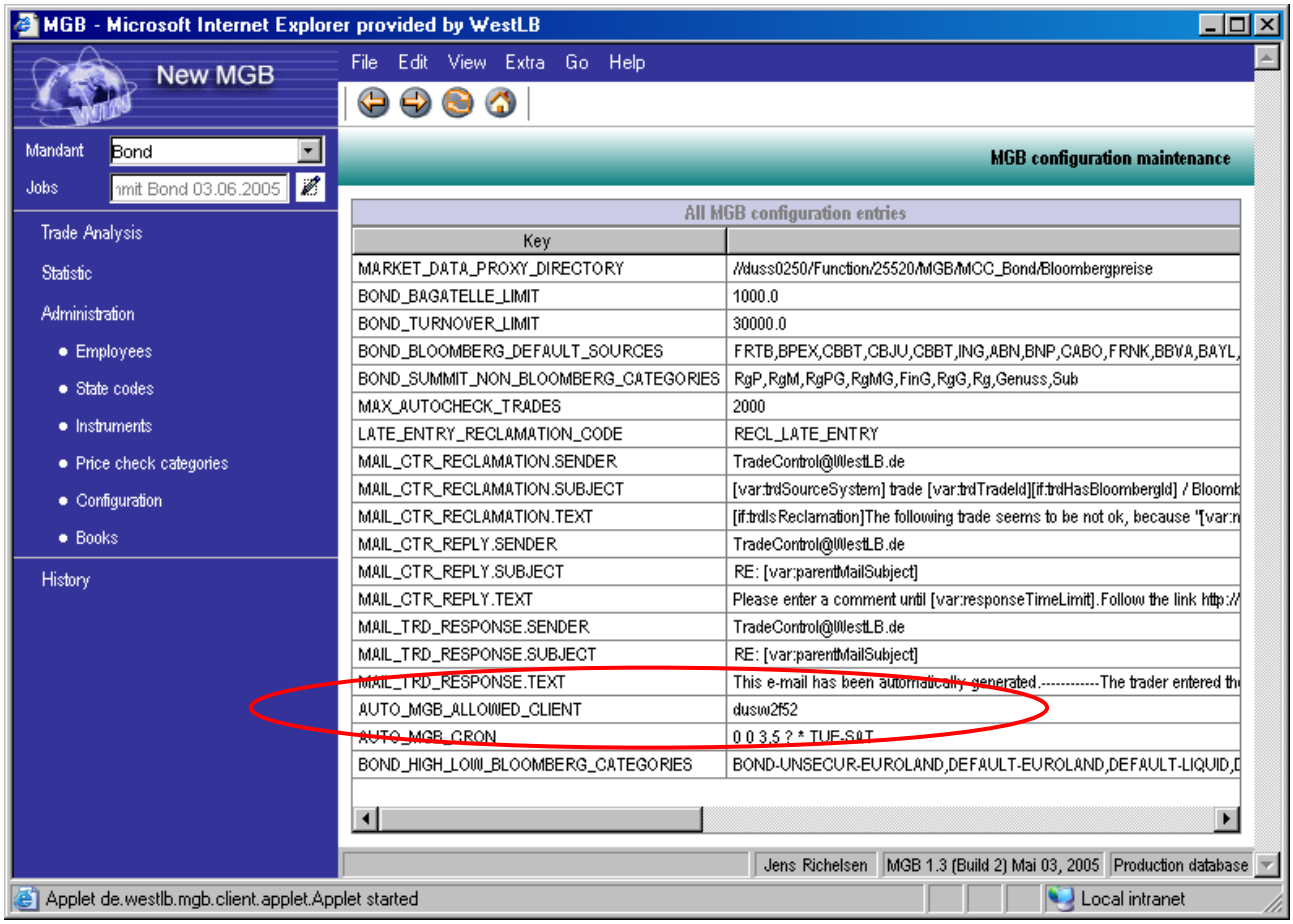
To check the status of the import processes, call the following URL:

http://mgb.westlb.sko.de:8092/mgbPrd/CheckImport

Using links on this page, the processes may be stopped and restarted.

7.7.5 User reports a problem with automatic importing of Bloomberg data for Bonds/Repos

For Bond and Repo clients, a special feature is set up that imports Bloomberg data automatically on the client (see 7.2). Only one client at any time is allowed to run this job. The configuration is stored in the database as shown below. The time where the download is attempted is configured in cron-like syntax (e.g. from Tuesday to Saturday at 3am and at 5am). Both can be changed using the usual 2-controller process.



Check if new Bond data was available when the automatic Bloomberg download started. (Look at the "Stop convert" time of the first entry in the list "Trade Analysis"->"Jobs" in the MGB tool and compare it to the start times of the Bloomberg download process)

Check if the heartbeat process is running. Either in the database (MGB_PRD_READ@RMMGBPRD) by running

```
SELECT * FROM (
SELECT      t63_start_date, t63_stop_date, t63_changed_date,
            t63_message, t63_state, t63_client, t63_user
FROM        t63_mgb_task
```

```
WHERE       t63_task_name = 'Mgb HeartBeat task'
AND         fk_t63_t09_mandant = 'BND'
ORDER BY    t63_id DESC
) WHERE     rownum < 2;
```

or by looking into the log files on the client located in `%USERPROFILE%\mgb\javaws.log*`.

This process is running on the same client as the download process and is connecting to the database every few minutes (5 min).

Ensure that someone has successfully logged on to Bloomberg on the client PC within the last 7 days. It is recommended not to stay logged on, since the Bloomberg session gets invalid if the user connects to Bloomberg from another client.

Ensure that the logged in user has already used the MGB Tool (applet) on that client (check if the directory `%USERPROFILE%\mgb` exists).

Fatal exceptions are stored in the database (MGB_PRD_READ@RMMGBPRD). Run the query:

```
SELECT * FROM (
SELECT      t63_start_date, t63_stop_date,
            t63_message, t63_state, t63_client, t63_user
FROM        t63_mgb_task
WHERE       t63_task_name LIKE 'Automatic Check for %'
AND         fk_t63_t09_mandant = 'BND'
ORDER BY    t63_id DESC
) WHERE     rownum < 3;
```

Or look into the log files on the client located in `%USERPROFILE%\mgb\javaws.log*` to see more details.

If necessary:

- restart the background process that schedules the automatic Bloomberg download (see 7.2).
- start the Bloomberg download manually on the client.

7.7.6 User reports inconsistent data in different views

Inconsistent data in different views might be the result of data changes that are differently cached. Especially after MGB installation and direct manipulation in the database, the cache should be deleted by running http://mgb:8092/mgbPrd/clearCache.do. This step might take some time (seconds to minutes) and finishes with a success message.

7.7.7 An import file is not processed

For each configured import file (see `$TOMCAT_HOME/webapps/mgbPrd/WEB-INF/classes/import.properties`) a separate thread is polling for new files. If a new file is found, it waits for the completion of the transfer and copies the file to a working directory. After the file is processed it starts looking for a new one.

Note: Files that are configured with a `cobdepends=<SourceSystemID>` property in `import.properties` are only processed if a job for the specified other source system has already been imported for the same COB.

All import threads are started after startup of the Tomcat server.

If for some reason a thread crashed, all import threads can be controlled by the servlet MgbDataHandlerServlet. The threads can be stopped and restarted this way:

Stop threads: http://mgb.westlb.sko.de:8092/mgbPrd/MgbDataHandlerServlet?action=stop

Restart threads: http://mgb.westlb.sko.de:8092/mgbPrd/MgbDataHandlerServlet?action=restart

7.7.8 E-mail reports are not sent

E-mail reports use `sendmail`, which is provided by the Postfix mailer since September 2015. The most likely cause for missing e-mails is a problem in the server's email configuration. Unfortunately, on RHEL a normal user cannot check the mail log for errors. This means that the `mailq` command is the only way to check whether emails are queuing up on the machine or not.

1. If yes: Address the issue to HP.
2. If no:
    a. Check with `crontab -l` whether the MGB report cronjob is active (`run_all.sh`). Also check the ~a223prod/log/mgb_management_report.log file for error messages related to the cronjob.
    b. If no errors could be identified, try sending a test mail to the affected mailbox with `sendmail -t` to confirm the issue (more information on how to do this is available in the RM_PST Wiki, see 1.1). Use this information to address the issue to HP.

8 Interfaces to front-office systems

8.1 General dataflow

The trade data from the various front office systems is transferred via files. The data is extracted and transferred to the MGB import directory with Connect:Direct, where it is picked up by MGB and imported into a flat database table using the SQL-loader. The process checks if any records were rejected and if any records are missing compared to the number of records announced in the meta-data information of the data file. Then the corresponding converter is invoked and the data is stored in the MGB data model.

Summarized information about missing, rejected or inconvertible records, along with some additional meta data, is stored in the job table in the database.

8.2 General file format

The files are supposed to have the following format.

The files contain semicolon delimited ASCII format. Fields that might contain the delimiter can be enclosed by double quotes.

A header line contains the column names. A trailer line gives some additional meta information. The format is:

`#;<number_of_trades>;<start_time>;<stop_time>;<extraction_time>`

The field `number_of_trades` show the number of trades the file should contain. This number should be calculated in an independent way, taking the front-office-specific extraction process into account. The start and stop time indicates the business time interval of the extracted trades. The date format is "`dd.MM.yyyy hh:mm`".

8.3 Source systems

The files described below are automatically copied to the directory (`$TOMCAT_HOME/../../mgbdata/mgbPrd/import`) that is monitored for new import data.

8.3.1 Paris (PRS)

PARIS (Portfolio Administration Reconciliation & Information System) is not a front office system but the global repository that holds Trading and Banking book static data. Since mid-2015, this functionality is technically part of SAP. A (non-incremental) report that defines all necessary data used by the MGB Tool is stored in the public reporting area of Paris. The report name is 'MCC-Export'.
There are two files that contain the book information of Portigon and EAA books.

Paris (PRS)	
Report name	MCC-Export
Source file names	\\DUSS0198\287cd$\OUT\Mandant001\Paris\PARIS2_MCC_M001.CSV \\DUSS0198\287cd$\OUT\Mandant201\Paris\PARIS2_MCC_M201.CSV
Extracted at	06:15
Transferred at	07:00

Transferred from	DUSS0198
Target file names	/p/a223/sp1e/mgb/mgbdata/mgbPrd/import/prs.dat /p/a223/sp1e/mgb/mgbdata/mgbPrd/import/prs_eaa.dat
Target server name	frax2173
Target user name	a223prod
Comment	Transfer is triggered from mgb-server a223prod@ frax2173 using $TOMCAT_HOME/../../mgbdata/mgbPrd/scripts/download_all_mgb_paris_files.ksh from a cronjob: `0 7 * * 1-5 /p/a223/sp1e/mgb/mgbdata/mgbPrd/scripts/download_all_mgb_paris_files.ksh >$HOME/log/download_all_mgb_paris_files_`date +\%Y\%m\%d_\%H\%M\%S`.log 2>&1`

8.3.2 Prime (PRM)

Prime contains equity trades.

Prime (PRM)	
Source file name	/app/prd/tev/prod/data/tev.YYYY-MM-DD/TEV_ExtractMGB.out
Extracted at	04:00:00
Transferred from	dusv1005
Transferred by	primeprd (using the snodeid a223prod)
Target file names	/p/a223/sp1e/mgb/mgbdata/mgbPrd/import/prm.dat
Target server name	frax2173
Target user name	a223prod
Archive file name (on file server)	\\DUSS0485\\produktion\Applications\MGB\Import\PRIME\mgb_Prime_yyyy-mm-dd.txt

8.3.3 Summit Bond (SMB, SMG)

Summit Bond (SMB, SMG)	
Source file name	/tmp/mgb_bond_yyyymmdd.txt
Extracted at	22:30 - 00:30
Extraction filter	DUTC_MCC_BO
Extracted by	mccReport -F DUTC_MCC_BO -ASOFDATE ${YESDATE3} -PATH ${SUMMITSPOOLDIR}/mgb_bond_${YESDATE112}.txt
Job name	PASUMDUSMSCST2C0022_mccreport
Transferred from	CD_LN_P01_X0105
Transferred by	CDTransfer -T ${CD_TRANSFER_TYPE} -F ${SUMMITSPOOLDIR}/mgb_bond_${YESDATE112}.txt -I ${CD_ITERATIONS} -W ${CD_WAIT} -D /p/a223/sp1e/mgb/mgbdata/mgbPrd/import/smb.dat -R CDPFSP08 -o unix
Target file name	/p/a223/sp1e/mgb/mgbdata/mgbPrd/import/smb.dat
Target server name	frax2173
Target user name	a223prod
Archive file name (on file server)	\\DUSS0485\Function\\25520\Daten_Archiv\MGB Daten\Fix Income\mgb_bond_yyyymmdd.txt

The file includes trades for different clients, which are separated in the subsequent import process. The de-

tails are described in the IT-Concept for Bonds. Each client will get its own job ID, but the archive file will be named after the job ID of the first client that is processed by the importer.

8.3.4 Summit Repo (SMR, SRL, SRG)

Summit Repo (SMR, SRL, SRG)	
Source file name	/tmp/repoMccRep_mmdd.txt
Extracted at	23:00
Extraction filter	DUTC_MCC_REP
Extracted by	repoMccRep -ASOF ${YESDATE3} -F DUTC_MCC_REP -REPOTERM 3M -O ${SUMMITSPOOLDIR}/repoMccRep_${YESDATE13}.csv
Job name	PASUMDUSMSCST1C0059_repo_MccRep
Transferred from	CD_LN_P01_X0105
Transferred by	CDTransfer -T ${CD_TRANSFER_TYPE} -F ${SUMMITSPOOLDIR}/repoMccRep_${YESDATE13}.csv -I ${CD_ITERATIONS} -W ${CD_WAIT} -D /p/a223/sp1e/mgb/mgbdata/mgbPrd/import/smr.dat -R CDPFSP08 -o unix
Target file name	/p/a223/sp1e/mgb/mgbdata/mgbPrd/import/smr.dat
Target server name	frax2173
Target user name	a223prod
Archive file name (on file server)	

The file includes trades for different clients, which are separated in the subsequent import process.

The details are described in the IT-Concept for Repos. Each client will get its own job ID, but the archive file will be named after the job ID of the first client that is processed by the importer.

8.3.5 Summit Foreign Exchange (TFD, TFL, TFN, TFG)

Summit Foreign Exchange (TFD, TFL, TFN, TFG)	
Source file name	SMT_FX_DUS.dat
Extracted at	06:20
Extraction filter	DUTC_MCC_FXN
Extracted by	mcc -F DUTC_MCC_FXN -AMEND ${YESDATE3} -O ${SUMMITSPOOLDIR}/SMT_FX_DUS.dat
Job name	PASUMDUSSGDMCCC0007_MCC_FX
Transferred from	CD_LN_P01_X0105
Transferred by	CDTransfer -T ${CD_TRANSFER_TYPE} -F ${SUMMITSPOOLDIR}/SMT_FX_DUS.dat -I ${CD_ITERATIONS} -W ${CD_WAIT} -D ${CD_DEST_MCC}/SMT_FX_DUS.dat -R CDPFSP08 -o unix
Target file name	/p/a223/sp1e/mgb/mgbdata/mgbPrd/import/SMT_FX_DUS.dat
Target server name	frax2173
Target user name	a223prod

The file includes trades for different clients, which are separated in the subsequent import process.

The details are described in the IT-Concept for FX Summit. Each client will get its own job ID, but the archive file will be named after the job ID of the first client that is processed by the importer.

Summit Foreign Exchange (TFD, TFL, TFN, TFG)	
Source file name	SMT_FX_HKG.dat

Extracted at	16:10
Extraction filter	DUTC_MCC_AFX
Extracted by	mcc -F DUTC_MCC_AFX -AMEND ${YESDATE3} -O ${SUMMITSPOOLDIR}/SMT_FX_HKG.dat
Job name	PASUMTYOSGDREPC0014_mcc_FX_Apac
Transferred from	CD_LN_P01_X0105
Transferred by	CDTransfer -T TRANSFER -F ${SUMMITSPOOLDIR}/SMT_FX_HKG.dat -I ${CD_ITERATIONS} -W ${CD_WAIT} -D /p/a223/sp1e/mgb/mgbdata/mgbPrd/import/SMT_FX_HKG.dat -R CDPFSP08 -o unix
Target file name	/p/a223/sp1e/mgb/mgbdata/mgbPrd/import/SMT_FX_HKG.dat
Target server name	frax2173
Target user name	a223prod

The file may include trades for different clients, which are separated in the subsequent import process.

The details are described in the IT-Concept for FX Summit. Each client will get its own job ID, but the archive file will be named after the job ID of the first client that is processed by the importer.

Summit Foreign Exchange (TFD, TFL, TFN, TFG)	
Source file name	SMT_FX_EAA.dat
Extracted at	16:10
Extraction filter	DUTC_MCC_EFX
Extracted by	mcc -F DUTC_MCC_EFX -AMEND ${YESDATE3} -O ${SUMMITSPOOLDIR}/SMT_FX_EAA.dat
Job name	PASUMEAASGDMCCC0003_MCC_FX
Transferred from	CD_LN_P01_X0105
Transferred by	CDTransfer -T TRANSFER -F ${SUMMITSPOOLDIR}/SMT_FX_EAA.dat -I ${CD_ITERATIONS} -W ${CD_WAIT} -D /p/a223/sp1e/mgb/mgbdata/mgbPrd/import/SMT_FX_EAA.dat -R CDPFSP08 -o unix
Target file name	/p/a223/sp1e/mgb/mgbdata/mgbPrd/import/SMT_FX_EAA.dat
Target server name	frax2173
Target user name	a223prod

The file may include trades for different clients, which are separated in the subsequent import process.

The details are described in the IT-Concept for FX Summit. Each client will get its own job ID, but the archive file will be named after the job ID of the first client that is processed by the importer.

8.3.6 Summit Money Market (TMD, TML, TMN, TMG)

Summit Money Market (TMD, TML, TMN, TMG)	
Source file name	SMT_MM_DUS.dat
Extraction filter	DUTC_MCC_MMN
Extracted at	06:20
Extracted by	mcc -F DUTC_MCC_MMN -AMEND ${YESDATE3} -O ${SUMMITSPOOLDIR}/SMT_MM_DUS.dat
Job name	PASUMDUSSGDMCCC0005_MCC_MM
Transferred from	CD_LN_P01_X0105

Transferred by	CDTransfer -T ${CD_TRANSFER_TYPE} -F ${SUMMITSPOOLDIR}/SMT_MM_DUS.dat -I ${CD_ITERATIONS} -W ${CD_WAIT} -D ${CD_DEST_MCC}/SMT_MM_DUS.dat -R CDPFSP08 -o unix
Target file name	/p/a223/sp1e/mgb/mgbdata/mgbPrd/import/SMT_MM_DUS.dat
Target server name	frax2173
Target user name	a223prod

The file includes trades for different clients, which are separated in the subsequent import process.

The details are described in the IT-Concept for MM Summit. Each client will get its own job ID, but the archive file will be named after the job ID of the first client that is processed by the importer.

Summit Money Market (TMD, TML, TMN, TMG)	
Source file name	SMT_MM_HKG.dat
Extraction filter	DUTC_MCC_AMM
Extracted at	06:20
Extracted by	mcc -F DUTC_MCC_AMM -AMEND ${YESDATE3} -O ${SUMMITSPOOLDIR}/SMT_MM_HKG.dat
Job name	PASUMTYOSGDREPC0013_mcc_MM_Apac
Transferred from	CD_LN_P01_X0105
Transferred by	CDTransfer -T TRANSFER -F ${SUMMITSPOOLDIR}/SMT_MM_HKG.dat -I ${CD_ITERATIONS} -W ${CD_WAIT} -D /p/a223/sp1e/mgb/mgbdata/mgbPrd/import/SMT_MM_HKG.dat -R CDPFSP08 -o unix
Target file name	/p/a223/sp1e/mgb/mgbdata/mgbPrd/import/SMT_MM_HKG.dat
Target server name	frax2173
Target user name	a223prod

The file may include trades for different clients, which are separated in the subsequent import process.

The details are described in the IT-Concept for MM Summit. Each client will get its own job ID, but the archive file will be named after the job ID of the first client that is processed by the importer.

Summit Money Market (TMD, TML, TMN, TMG)	
Source file name	SMT_MM_EAA.dat
Extraction filter	DUTC_MCC_EMM
Extracted at	06:20
Extracted by	mcc -F DUTC_MCC_EMM -AMEND ${YESDATE3} -O ${SUMMITSPOOLDIR}/SMT_MM_EAA.dat
Job name	PASUMEAASGDMCCC0001_MCC_MM
Transferred from	CD_LN_P01_X0105
Transferred by	CDTransfer -T TRANSFER -F ${SUMMITSPOOLDIR}/SMT_MM_EAA.dat -I ${CD_ITERATIONS} -W ${CD_WAIT} -D /p/a223/sp1e/mgb/mgbdata/mgbPrd/import/SMT_MM_EAA.dat -R CDPFSP08 -o unix
Target file name	/p/a223/sp1e/mgb/mgbdata/mgbPrd/import/SMT_MM_EAA.dat
Target server name	frax2173
Target user name	a223prod

The file may include trades for different clients, which are separated in the subsequent import process.

The details are described in the IT-Concept for MM Summit. Each client will get its own job ID, but the archive file will be named after the job ID of the first client that is processed by the importer.

8.3.7 Summit Derivative (SDD, SDE, SDG)

Summit Derivative (SDD, SDE, SDG)	
Source file name	/tmp/mcc_trade_yyyymmdd.csv
Extracted by	mcc_trade -F DUTC_MCC_DER -ASOF ${YESDATE3} -O ${SUMMITSPOOLDIR}/mcc_trade_${YESDATE112}.csv
Extracted at	
Extraction filter	DUTC_MCC_DER
Job name	PASUMDUSMKTMCCC0002_mcctrade
Transferred from	CD_LN_P01_X0105
Transferred by	CDTransfer -T ${CD_TRANSFER_TYPE} -F ${SUMMITSPOOLDIR}/mcc_trade_${YESDATE112}.csv -I ${CD_ITERATIONS} -W ${CD_WAIT} -D /p/a223/sp1e/mgb/mgbdata/mgbPrd/import/smt_drv_dus.dat -R CDPFSP08 -o unix
Target file name	/p/a223/sp1e/mgb/mgbdata/mgbPrd/import/smt_drv_dus.dat
Target server name	frax2173
Target user name	a223prod
Archive file name (on file server)	

The file includes trades for different clients, which are separated in the subsequent import process.

The details are described in the IT-Concept for Derivatives. Each client will get its own job ID, but the archive file will be named after the job ID of the first client that is processed by the importer.

8.3.8 Summit amendment report (SAM)

Summit Amendment (SAM)	
Source file name	/tmp/TAR_yyyymmdd.txt
Extracted by	tradeamendmentreport -F LNDETRAMRPT -ASOF ${YESDATE112} -CSV -THRESH 0 -O ${SUMMITSPOOLDIR}/TAR_${YESDATE112}.txt
Extracted at	
Extraction filter	LNDETRAMRPT
Transferred from	CD_LN_P01_X0105
Transferred by	a352ft1
Target file name	/p/a223/sp1e/mgb/mgbdata/mgbPrd/import/smt_amend.dat
Target server name	frax2173
Target user name	a223prod
Archive file name (on file server)	

The file includes amendments of trades. The information is used by the other clients to compare and analyse changes to prior versions of trades.

Note: Summit Application Support promised that this file always arrives **before** the Summit Derivative file (this agreement was reached via email in April 2014). This makes sense because MGB requires that amendments are known before the corresponding trades are imported. Starting with MGB 2.0.18, a dependency check prevents the import of a Summit Derivative file before an Amendment file for the same COB is available. This check was introduced because the agreed order never worked for re-runs, which was a frequent source of trouble.

Note about re-runs: If this file cannot be delivered in time because of technical issues and a re-run is at-

tempted, note that the date parameter to the `tradeamendmentreport` executable expects the date in a special format (example: … -ASOF "30/07/14" …). The parameter given in the daily run might *not* serve as an example, because the current COB is also the default and wrongly formatted date parameters simply make the executable fall back to it.

8.3.9 Bloomberg Request Mapping (BBM)

BBM is not a front-office system but a set of (non-incremental) data files that specify Bloomberg request sizes and request strings. They are made available by the PSS application, which transfers them from a business-maintained network share. The transfer is triggered on the PSS side, using the FTL (File Transfer Light) tool.

Bloomberg Request Mapping (BBM)	
Source file name	
Extracted by	FTL (File Transfer Light)
Extracted at	daily, 21:00 Duesseldorf time
Extraction filter	
Transferred from	
Transferred by	
Target file name	/p/a223/sp1e/mgb/mgbdata/mgbPrd/import/BBG_*.dat
Target server name	frax2173

9 Production services

9.1 Requirements on support and availability

The following requirements on support availability are reconciled with the business teams:

Support time:	8 pm – 5 pm on workdays
Maximum time for disaster recovery:	3 days

9.2 Service and contacts

contact persons	
role	**contact**
MGB First Level Support/ MGB Production service	RM_PST hotline: 0211-826-74413 Mail: RM_PST
MGB Development / Second Level Support	Raimund Steger: 0211/826-5188 Jens Arbeiter-Kaum: 0211/826-3895
Prime data delivery	Christian Keutel, 0211/826- 8597 Mail: AIS_Prime_Support_DUS@portigon.com
User Management	501-45210 Security Management Mail: user_management_westlbdus@portigon.com
Summit Repo data delivery	Production mail: WLB_SUMMIT_PRODUCTION_SUPPORT@portigon.com Development mail: DUSS_SummitApplicationSupport@portigon.com
Summit Bond data delivery	Production mail: WLB_SUMMIT_PRODUCTION_SUPPORT@portigon.com Development mail: DUSS_SummitApplicationSupport@portigon.com
Summit Derivative data delivery	Production mail: WLB_SUMMIT_PRODUCTION_SUPPORT@portigon.com Development mail: DUSS_SummitApplicationSupport@portigon.com
Summit MM/FX data delivery	Production mail: WLB_SUMMIT_PRODUCTION_SUPPORT@portigon.com Development mail: DUSS_SummitApplicationSupport@portigon.com
Summit Amendment data delivery	Production mail: WLB_SUMMIT_PRODUCTION_SUPPORT@portigon.com Development mail: DUSS_SummitApplicationSupport@portigon.com
PARIS data delivery	Production mail: WLB_SAP_FI_SUPPORT@portigon.com Business mail: WLB_SAP_FI_SUPPORT@portigon.com
Bloomberg Request Mapping	Production mail: Patrick Moehler, mrm_equities@portigon.com Development mail: Soenke Frenzen (Risk IT)

9.3 Service level description

9.3.1 Database

The databases are running on shared servers provided by HP. For a detailed description of the service level, contact IT Risk Production.

The service includes system administration and hardware support for the RHEL operating system. The service for the production database includes Oracle database (incl. processor standard edition license) and database administration. The database is running on an RHEL operating system with 50 GB disk space reclaimed for the MGB application (numbers may vary over time, for exact figures contact IT Risk Production).

9.3.2 Application server

The application servers for user acceptance test and production are located in computing centers run by HP. The service includes the support and maintenance of hardware and the RHEL operating system. The servers are used by IT Risk Production as shared application servers for IR Market Risk & Market Data Management.

10 Project documentation

10.1 Document locations

The project documentation can be found on the server \\duss0501\501-4xxxx\501-41000_Projects\Tradecontrol_it\TC-IT-Support\MGB Tools\MGB ab 04

The detailed structure is explained in the following table:

\bonds\concepts\	General concept documents of the bond client
\bonds*	Various documents of the bond client
\concepts\SystemParameter	Description of the system parameter
\concepts\gui_concept	Gui concept
\concepts\Marktdaten_webservice	Documents handling the data request to Bloomberg
\concepts\Datensicherung	Backup concept
\concepts\StatisticConcept	Concept of the statistic gui
\concepts\ReclamationProcess	Concept of the reclamation process
\concepts\Architektur	General architecture
\concepts\businessModel	Java business model
\concepts\Anwendungsfallbeschreibungen	Use case descriptions
\Documents\ProductionManual\	Production manual
\Documents\UserManual\	User manual
\equity\concepts\	General concept documents of the equity clients
\equity\fidessa\	Special documents of the fidessa client
\equity\prime\	Special documents of the prime client
\equity\SystemManual\	System manual of the equity clients
\equity\SystemParameter\	Description of the system parameter of the equity clients
\equity*	Various documents of the equity clients
\MM_FX\summit*	IT concept for the MoneyMarket/FX clients
\repo\concepts\	General concept documents of the repo client
\repo\SystemManual\	System manual of the repo client
\repo\SystemParameter\	Description of the system parameter of the repo client
\repo*	Various documents of the repo client
\Infrastruktur\	Various documents related to the technical infrastructure
\Projektmanagement\	Various documents of the project management like meeting minutes, project plans, open issue list,
\Test_Documents\	Testing documentation (see MGBTestConcept.doc)

10.2 Archiving

Versions of the documents are supposed to carry the version number in the file name.

The files on the server DUSS0250 are backed up daily and archived for 1 year.

To archive the data for 10 years, it is necessary to copy the data to the archive server of RMS&C (NTSAPPLI0010394). The backup of that server is performed once a month.

Once a year, last day of the year, the files and subdirectories of the directory \\duss0164\236_sit-cc\Projects\Tradecontrol_it\TC-IT-Support\MGB Tools\MGB ab 04\ must be copied to the RMS&C archive server and additionally a DVD must be created, to archive the data for 10 years. The DVDs are archived at the project managers office.

Details are described in Datensicherung_RMC&S_11_20030812c.doc. (Contact person: G.Baur)

11 Application server installation

All required software components are located in the CVS Repository (see configuration management).

11.1 Initial installation

The following components have to be installed and configured concerning the guidelines:

11.1.1 Operating system

The following installation guide assumes a server with RHEL 6.

11.1.2 Directory structure

The following directory structure has to be created, where `$MGB_HOME` is an appropriate directory (e. g., `$TOMCAT_HOME/../../mgbdata`) and `MGB_ENV` is one of the MGB environments mgbDev, mgbTst, mgbUat or mgbPrd:

$MGB_HOME/MGB_ENV/archiv
$MGB_HOME/MGB_ENV/cache
$MGB_HOME/MGB_ENV/import
$MGB_HOME/MGB_ENV/work
$MGB_HOME/MGB_ENV/scripts
$MGB_HOME/MGB_ENV/logs
$MGB_HOME/MGB_ENV/jobarchiv

`$MGB_HOME/MGB_ENV/logs` and `$MGB_HOME/MGB_ENV/jobarchiv` are configurable locations in `env_{dev|tst|uat|prd}.properties`, so deviating paths might be used in some environments.

The configuration files for the SQL loader (`import_*.ctl`) have to be copied to `$MGB_HOME/MGB_ENV/scripts`.

11.1.3 Java Development Kit (JDK 1.6.0_26)

The JDK is simply unzipped from the vendor distribution file into `$TOMCAT_HOME/../jdk1.6.0_26/`.

11.1.4 Oracle Client

From the Oracle client installation, the JDBC driver, SQL Loader, and SQL*Plus are required.

11.1.5 Environment Variables on a UNIX operating system

On a UNIX operating system, a number of environment variables have to be defined for the application server process. Currently, all Oracle related variables as well as `JAVA_HOME` are set in `~/.profile`. A copy of that file is documented in the RM_PST Wiki (see 1.1).

Java options used by the Tomcat process are set in `$TOMCAT_HOME/bin/catalina.sh` (see 11.1.6). (Note that this deviates from Apache's recommendation, in which site-specific settings are maintained in `$TOMCAT_HOME/bin/setenv.sh`.)

11.1.6 Installation of Tomcat (tomcat-7.0.16)

Step 1:

The software can be installed with default setup.

Step 2:

Change the port number in the file `server.xml` to appropriate values. This is done in the following section:

```
<!-- A "Connector" represents an endpoint by which requests are received
     and responses are returned. Documentation at :
     Java HTTP Connector: /docs/config/http.html (blocking & non-blocking)
     Java AJP  Connector: /docs/config/ajp.html
     APR (HTTP/AJP) Connector: /docs/apr.html
     Define a non-SSL HTTP/1.1 Connector on port 8092
-->
<Connector port="8092" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
```

Step 3:

Add the following lines to the script `catalina.sh` start script if it is run on UNIX:

```
JAVA_OPTS="$JAVA_OPTS -Xms2048m -Xmx4096m -Djava.awt.headless=true -d64"
export JAVA_OPTS
```

Tomcat will run with an initial heap size of 2048 MB and a maximum heap size of 4096 MB. It also starts up with the 64-bit JRE.

Step 4:

Add the following lines to the script catalina.sh start script if it is run on UNIX:

```
CLASSPATH="$CLASSPATH":"$ORACLE_HOME"/jdbc/lib/ojdbc6.jar
export "$CLASSPATH"
```

That way, the current JDBC driver is added to the Tomcat class path.

11.2 Update installation (provided script)

When a new release is made available for installation in production, an installation script called `install_mgb.ksh` will be placed in `/tmp/` by the MGB development team. This script is available in CVS and should be revised by MGB developers before every release. By executing this script, all necessary steps will be performed without user interaction.

11.3 Configuration files

The MGB application reads application parameters from property files. The following sections contain the listing of the most important parameter files. Comment lines starts with the "#' character.

11.3.1 hibernate.properties

This file is located in the Tomcat directory `.\webapps\<environment>\WEB-INF\classes`. It contains

all the properties used by the Hibernate library.

11.3.2 import.properties

This file is located in the Tomcat directory `.\webapps\<environment>\WEB-INF\classes`. It contains all properties used by the MgbDataHandlerServlet.

11.3.3 native.properties

This file is located in the Tomcat directory `.\webapps\<environment>\client`. The client is reading this file at startup time. It compares the native files cached at the client side with the information about these files in the properties file. If one of the native files does not exist or is older than the date in the file, it copies the file from the server.

11.3.4 version.properties

This file is located in the Tomcat directory `.\webapps\<environment>\client`. The client is reading this file at startup time. It is used to display information about the version in the status bar and the about box.

```
product=#PRODUCT_NAME#
version.major=#VERSION_MAJOR#
version.minor=#VERSION_MINOR#
version.update=#VERSION_UPDATE#
build=#BUILD_NUMBER#
version.date=#BUILD_TIME#
```

11.3.5 sso.properties

This file is located in the Tomcat directory `.\webapps\<environment>\WEB-INF\classes`. It contains all properties used by the CheckAuthenticatedFilter.

```
checkAuthentication=no
dbDriverClass=oracle.jdbc.driver.OracleDriver
dbUrl=jdbc:oracle:thin:@wlb06407:1521:o20t
startupView=page/frames.jsp
trustedDomains=MDWESTLB,GSA-WLB,WESTLB-SYSTEMS,EMEA-WLB,APAC-WLB,DIRECT_LOGIN
```

11.3.6 mgb_log4j.properties

This file is located in the Tomcat directory `.\webapps\<environment>\WEB-INF\classes`. It is used by the log library.

11.3.7 oscache.properties

This file is located in the Tomcat directory `.\webapps\<environment>\WEB-INF\classes`. It is used by the OSCache library.

11.3.8 server.properties

This file is located in the Tomcat directory `.\webapps\<environment>\WEB-INF\classes`. It is used by the all mgb server components.

11.3.9 hibernate_mappings.properties

This file is located in the Tomcat directory `.\webapps\<environment>\WEB-INF\classes`. It contains

all persistent MGB java classes. This file is referenced by the file server.properties.

11.3.10 client.properties

This file is located in the Tomcat directory `.\webapps\<environment>\client`. The client is reading this file at startup time. It is used from different client packages and classes.

11.3.11 tnsnames.ora

To connect to the database the following entry must exist in the `tnsnames.ora` file:

```
RMMGBTST.WORLD =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = RMMGBTST.westlb.sko.de)(PORT = 1522))
    )
    (CONNECT_DATA =
      (SID = RMMGBTST)
         (GLOBAL_NAME = RMMGBTST.world)
    )
  )

RMMGBPRD.world =
   (DESCRIPTION =
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = TCP)(HOST = rmmgbprd.westlb.sko.de)(PORT = 15133)
        )
      )
      (CONNECT_DATA =
       (SID = RMMGBPRD)
          (GLOBAL_NAME = RMMGBPRD.world)
      )
    )
```

12 Open Issues

Any open issues are reported in the bug tracking tool described in chapter 5.4.