Druck vom 24. 01.2001 17:00.14 umschlag

SIEMENS

GINA V4.0

General Interface for Network Applications
System Administrator Guide

Comments
Suggestions
Corrections

The User Documentation Department would like to know your opinion on this manual. Your feedback helps us to optimize our documentation to suit your individual needs.

Fax forms for sending us your comments are included at the back of the manual.

Order number of this manual:
GINA V4.0 System Administrator Guide – September 2000

Druck:vorm 24. 01.2001 17:50:15 titel

GINA V4.0

General Interface for Network Applications
System Administrator Guide

Edition September 2000

Copyright and Trademarks

GINA is a registered trademark of Siemens Business Services GmbH & Co OHG.

SINIX® Copyright © Siemens Nixdorf Informationssysteme AG 1990.
SINIX is the UNIX® System derivative of Siemens Nixdorf Informationssysteme AG.
Reliant® is a registered trademark of Pyramid Technology Corporation.
UNIX is a registered trademark in the United States and other countries, licensed exlusively through X/Open Company Limited.

Base: OSF/Motif™, Copyright © Open Software Foundation, Inc.
X Window System™, Copyright © Massachusetts Institute of Technology.

OSF/Motif is a registered trademark of Open Software Foundation, Inc.
X Window System is a registered trademark of Massachusetts Institute of Technology.

Copyright © Siemens Business Services GmbH & Co OHG 2000.

All rights reserved.
Delivery subject to availability; right of technical modifications reserved.

All hardware and software names used are trademarks of their respective manufacturers.

Continued ▶

Druck vom 24. 01.2001 17:00:16 verwivz.doc

Contents

Druck.vom 24. 01.2001 17:00.17 verwvz.doc

Druck vom 24. 01. 2001 17:00.18 einleit

1 Introduction

GINA (**G**eneral **I**nterface for **N**etwork **A**pplications) provides a framework for the implementation and operation of object-oriented, transaction-oriented client/server applications. The GINA-API is an object-oriented solution for the mixed, distributed applications which are encountered everywhere in modern business life.

GINA is suitable for use in many types of client/server environment– for systems which place high demands on the criteria of data consistency and reliability (business-critical applications) as well as for the rightsizing of mainframe-based systems for decentralized online transaction processing (OLTP).

GINA is based on standards and can adapt to individual circumstances. The object-oriented paradigm saves programming time and makes system structures clearer. Modifications and corrections are easier. GINA is a high-performance development environment for distributed and persistent objects.

GINA is an open system and provides connectivity to external OLTP systems in a variety of environments such as the BS2000 or MVS environments.

About this manual

This manual is intended for system administrators who need to install GINA or applications used by GINA. It describes how the communications system and GINA applications are configured.
This manual also deals with the operation of GINA applications in client/server environments.

Developers and programmers of GINA applications may also want to refer to this manual on occasions.

For more information, please contact us at the address below:

Siemens Nixdorf Informationssysteme AG
SBS MPM CPI
Otto-Hahn-Ring 6
81739 München

Fax (089) 636-48 303

E-Mail gina.service@mch20.sbs.de

Structure of this manual

Chapter 1 describes the structure and contents of this manual as well as other documentation on GINA.

Chapter 2 contains a listing of the essential changes since the last version of this manual as well as a brief description of each.

Chapter 3 *Installation and deinstallation*
describes the installation of GINA, including prerequisites.

Chapter 4 *Creating GINA applications*
describes the necessary steps in creating GINA applications.

Chapters 5 ... 8
describe the configuration and administration of GINA applications:

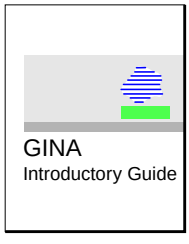
- Chapter 5 *Configuring the Persistency Service*
- Chapter 6 *Configuring T-ORB for openUTM*
- Chapter 7 *Configuring T-ORB for BEA TUXEDO*
- Chapter 8 *Operating GINA applications*

The *Glossary* and *Abbreviations* chapters explain important technical terms and abbreviations.

The *Related publications* section contains a list of manuals and secondary literature.

The table of contents and index simplify the task of finding information.

Documentation on GINA



GINA Introductory Guide

This manual provides a brief summary of the performance characteristics and underlying philosophy of GINA. It also presents the various components which make up GINA.

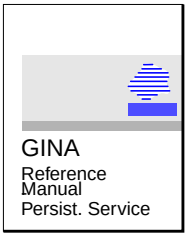
It is aimed at decision-makers who want to assess the possible usefulness of GINA or users who intend to work with GINA and want to become familiar with its structure.

Druck vom 24. 01. 2001 17:00.18 einlieft

GINA Developer Manual

This manual is intended for developers of GINA applications. It provides a detailed description of GINA concepts and gives practical instructions and assistance for use.You should read this manual first as it describes the theory and principles on which GINA is based.

Application developers should be familiar with the fundamentals of the object-oriented paradigm; knowledge of C++ is essential.

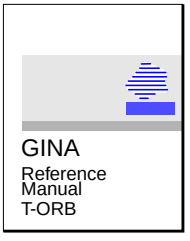


GINA Reference Manual Persistency Service

This is the manual for GINA application programmers. It contains formal descriptions of Persistency Service interfaces set out in alphabetical order.

It also contains descriptions of the associated tools.

Programmers must be familiar with object-oriented programming and must be able to program in C++.
They must be familiar with the concepts of the Persistency Service and Support components which are described in the Developer Manual



GINA Reference Manual T-ORB

This is the manual for GINA application programmers. It contains formal descriptions of T-ORB interfaces set out in alphabetical order.

It also contains descriptions of the associated tools.

Programmers must be familiar with object-oriented programming and must be able to program in C++.
They must be familiar with the concepts of the T-ORB and Support components which are described in the Developer Manual.

The *Related publications* sections of the manuals listed above also provide references to related topics.

Ordering manuals

If you would like to order these manuals please contact your local Siemens office.

Notational conventions used in this manual

This character draws your attention to special features or points of interest; you will also find useful or secondary information there. ❍❍●

Particular attention *must* be paid to the information indicated by this symbol. ❍❍●

Terms that are explained in the text are highlighted in **bold**.

Program code, messages, keywords or class names are indicated by `typewriter text`.

`Italic typewriter text` indicates variables for parameters that you must enter.

Text parts that are to be emphasized are represented by *italics*.

[1] Numbers in square brackets refer to the *Related publications* section.

◊ Rhombuses introduce processing statements.

Druck vom 24. 01. 2001 17:00.19 aender

2 Changes since Version 3

2.1 Interface cancelations

The interfaces listed in the following section were changed in Version 4.0 of GINA. This version contains the new variant. Each section indicates the GINA version as of which the relevant interface or its old variant is no longer supported.

`G_Exception` eliminated

In earlier versions, GINA used the exception handling simulation of the Generic++ class library [11] on some platforms. To facilitate this, the GINA exception classes were derived internally from the Generic class `G_Exception`. This derivation process has had to be eliminated because of a compiler problem.

The `message` and `name` methods derived from `G_Exception` described in the Reference Manual are omitted from GINA Version 4.0 and later.

OQL

Version 3.0 of GINA sees the introduction of a new, improved interface for database queries which is a subset of ODMG-OQL. Up to this version, the new OQL existed in parallel to the old one and the user can decide the variant to be used.

Access via the old OQL will be eliminated as of GINA Version 4.0.

Entering special options in the `mgen2` and `mdiff` generators

As of Version 3.3, special options (e.g. `noansi`, `nohinfo`) for the `mgen2` and `mdiff` generators can be defined in a file, i.e. you no longer need to specify them individually in the call. Instead, you reference an existing file using the `-k` option when you call a generator. All of the special options can be specified in this file.

As of Version 3.3, special options were supported both as call options and via a special option file. As of Version 4, however, special options will only be read from a file.

Changing the names of the iterator methods max/min to maxValue/minValue

The methods max and min in the iterator classes *P*Mibs::MibsFilterIt, *P*Mibs::MibsSeqIt, and VIEWITERATOR(P) were renamed maxValue and minValue respectively in Version 3.0 of GINA in order to prevent conflicts with the max and min macros defined in some environments.

The old API containing the method names max and min was supported as a transitional aid. These methods are inline methods which call the methods maxValue and minValue respectively. You can suppress these methods explicitly using the GINA_WITHOUT_MINMAX compiler switch in order to prevent conflicts with macros of the same name.

The method names max and min will be omitted from GINA Version 4.0 and later.

mgen2: Column aliases (mnemonics for SQL) and PS-DB-API

The algorithm for defining the names of the column aliases as well as the parameters for the functions in the PS-DB-API will be changed as of GINA V5.0.

To avoid name clashes, underscores contained in the names of the specialist attributes will be doubled. In terms of the PS-DB-API, this change will not affect the GINA user as it is the datatype of the individual attributes that is the decisive factor there. In terms of column aliases, this change will affect all SQL queries where a search is to be performed for attributes with an underscore in their names. The underscores in the relevant attribute names must then simply be doubled.

C runtime libraries under WindowsNT

Version 4.0 and above will be shipped with multithreaded libraries only.

Druck vom 24. 01. 2001 17:00:19 aender

2.2 Revisions

Replacement of `idlgen` by `idlgen1`

The `idlgen1` generates two definitions from an interface definition (`x.idl`) specified in CORBA-IDL (Revision 2.2): `x.hi` which defines the data members to be encoded and decoded and `x.hd` which defines the methods to be exported. The interim format `x.hi` serves as input for the MIO generator `miogen2`. The interim format `x.hd` serves as input for the T-ORB generator `dogen2`.


GINA

Druck vom 24. 01.2001 17:00.19 install

3 Installation and deinstallation

This chapter describes how to install and deinstall GINA. Some of the technical information given here is for the purposes of example only, e.g. it may vary partially depending on the details of the operating system.

The current version of GINA can run under:

– UNIX-SVR4
– BS2000/OSD (under special release)
– WindowsNT
– Windows95/98

Further information on your respective system base can be found in the Release Notice supplied with GINA. Please read this Release Notice carefully. ❍❍●

3.1 Requirements

The following third-party products are required to implement the GINA components.

Please note that the products listed may be based on other products, which must then likewise be installed. Up-to-date information on the products required can be found in the Release Notice included in the delivery. ❍❍●

- Generic++ V2.5 [11]

  GINA requires the class library Generic++ V2.5, which is contained in the GINA scope of supply and is installed under the name `libsupport2`.

- *open*UTM V5.0 and *open*UTM-Client V5.0

  For communication and transaction monitoring, the T-ORB server uses the TP monitor UTM. GINA Version 4.0 requires UTM Version *open*UTM (UNIX, NT, BS2000/OSD) V5.0 or later [29].

  To connect non-transaction-monitored applications (T-ORB client) to a transaction-monitored server, the CPIC interface is used.
  This requires the product *open*UTM-Client (UNIX, NT) V5.0A or later [36].

  The software component UTM-D is also required when using GINA with BS2000.

- INFORMIX Dynamic Server 2000 (IDS.2000) V9.2

  The Persistency Service in GINA V3.3 uses INFORMIX as the data storage system. INFORMIX Dynamic Server 2000 [19] (UNIX & NT) and Client SDK 2.40 (UNIX and NT) are required.

  The XA interface [5] is required for the integrated use of T-ORB and the Persistency Service. This interface is only integrated under UNIX in V9.2. Detailed information on coupling T-ORB and the Persistency Service under WindowsNT can be found in chapter 6, *Compiling and linking*, of the Developer Manual [13].

Druck vom 24. 01.2001 17:00.20 install

3.2 Scope of supply and structure of GINA

This section describes the delivery packages and the general delivery structure of GINA Version 4.0.

3.2.1 Delivery package

GINA V4.0 is supplied as a full-feature version (UNIX, NT) and as a partial version (WindowsNT only). The full-feature version contains the following GINA components:

- Persistency Service development, runtime system and PS browser

  The development system of the Persistency Service comprises the generators `mgen1`, `mgen2`, `mgendb`, `mspgen2` and `mdiff`, the runtime system comprises the libraries of the Persistency Service and the Persistency Service/client.

  The PS browser comprises the components `bruno` and `cuno`.

- T-ORB development and runtime system

  The development system of T-ORB comprises the generators `config`, `miogen1`, `miogen2`, `dogen1`, `dogen2` and `idlgen`, the runtime system comprises the libraries of T-ORB and T-ORB/client.

- Support runtime system

  The runtime system of the Support component comprises two libraries, one of which contains support functions for T-ORB and the Persistency Service and the other of which makes available the valid Generic++ class library.

The partial version under WindowsNT contains only the runtime environment (RT version) for a T-ORB client. It is also prepared for use under Windows95 and Windows98. More detailed information can be found in the Release Notice included with the relevant version of GINA.

The delivery medium is a CD.

3.2.2 Licensing of GINA

The tool FLEX*lm* from the company GLOBE*trotter* is used to license GINA. Both the generators and the GINA runtime system are protected by licenses.

Before GINA can be used, the GINA Competence Center must generate the licenses for the machine you require (processor ID) and convey them to you.

If you require an evaluation license, refer to the *Release Notice* for a template which you must return completed to the GINA Competence Center.

Installing licenses

Store the license entries received from the Competence Center in the file `GinaLicense.dat`. This file is a text file. All machines where GINA is installed must be able to access this file. The environment variable `LM_LICENSE_FILE` must be set on each of these machines. This variable contains the path and the name of the license file.

Example (UNIX, csh)

```
setenv LM_LICENSE_FILE /home/usr/license/Ginalicense.dat
```

If the file already exists on your machine, insert in file `license.dat` the lines of relevance to GINA.

You can also specify several license files:

Example (UNIX, csh)

```
setenv LM_LICENSE_FILE /home0/GinaLicense.dat:/home1/GinaLicense.dat
```

Druck vom 24. 01.2001 17:00.20 install

Structure of the license file

The license file has the following structure:

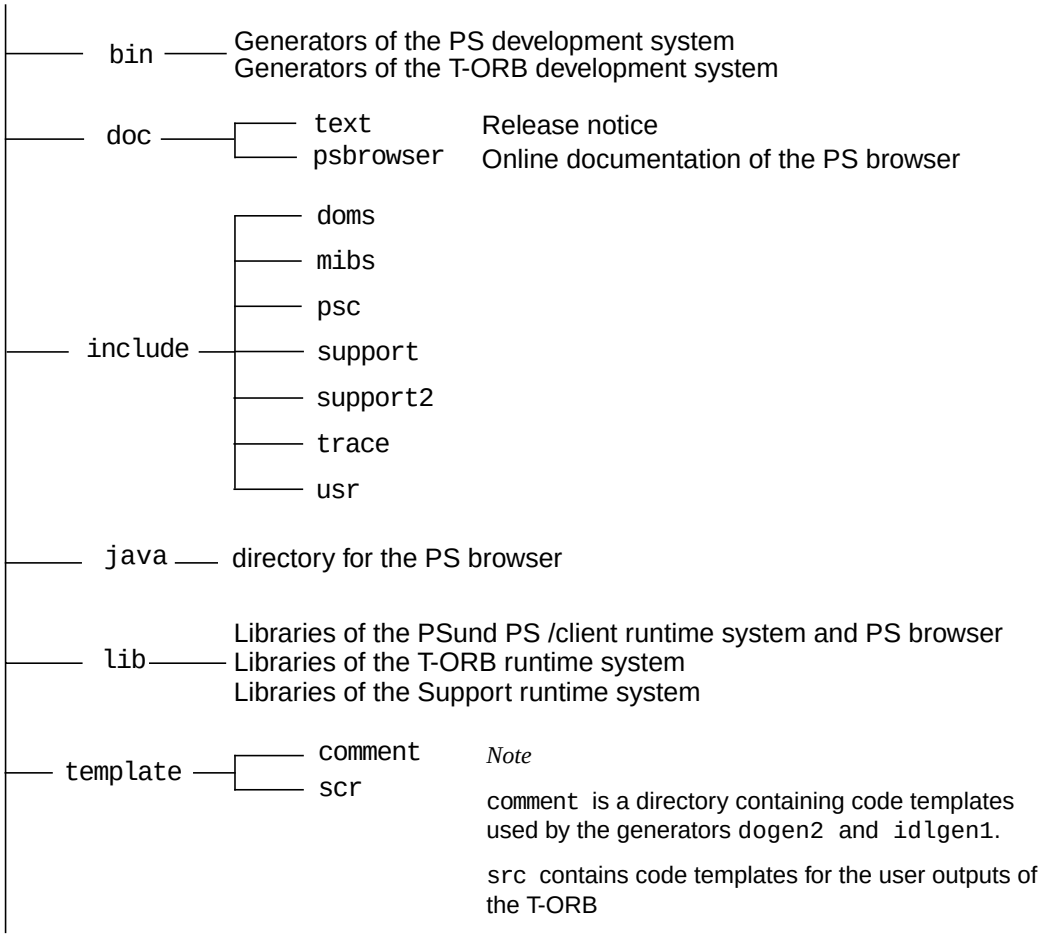
Keyword	License	Daemon	Ver-sion	Valid	N	License code	Proces-sor ID	License-name

Example

```
FEATURE  GinaDEV ginad  1.000 1-jan-0 0  C2348347  2387823 "Gina license"
```

Keyword	Keyword for FLEX*lm*
License	Type of license used (GinaDEV, GinaRTS, TorbDEV, TorbRTS, PsDEV, PsRTS, TorbClient, PsClient)
Daemon	Name of the license daemon
Version	Version number
Valid	Date until which the license is valid
N	Number of licenses (0 = unrestricted license)
License code	20-digit license code
Processor ID	Processor ID which is assigned to the license code
License name	Name of the license

3.2.3 Directory structure



The illustration shows the GINA default directory structure. The actual structure may differ from that shown above.

You should therefore study the Release Notice, which can also be found under the file name `readme.`*`Version`* in the `doc` directory. ❍❍●

Druck vom 24. 01.2001 17:00.21 install

3.3 Installation

The installation procedures depend on the system base. The variants described here are examples only.

You must always perform installation and deinstallation in accordance with the description of your system base or the information in the Release Notice. ❍❍●

GINA is shipped as a package which is created using a system-specific packet assembly procedure (package under UNIX, setup under NT). The full-feature version of GINA V4.0 under UNIX is divided up into subpackages so that you do not have to install the entire product folder. The functional scope of a full-feature version can only be achieved by installing all of the subpackages.

The names and contents of the subpackages as well as the order in which they must be installed can be found in the Release Notice.

Depending on the selected variant, GINA builds on standard products (*open*UTM, INFORMIX). These must be installed using their own installation procedures and are not described here.

If the GINA package is stored on the delivery CD as a file in *compressed* format, it must be copied to an intermediate directory and unpacked prior to installation (refer to the description for the relevant platform).

The following actions, for example, must be carried out to install GINA itself on the platforms listed below.

3.3.1 UNIX (Solaris, SINIX)

Start the installation under UNIX using the command pkgadd.

The installation directory in which GINA is to be installed may not exist before the pkgadd command is called. ❍❍●

1. Log in as `root`.
2. Set up a new user group *`tmns`*.
3. Set up a new user *`gina`*.
4. Create a new directory e.g. `/opt/gina` for GINA:

   ```
   mkdir -p /opt/gina
   chmod 775 /opt/gina
   chown gina:tmns /opt/gina
   ```

5a. Insert the GINA delivery CD in the CD drive and install the *uncompressed* GINA package using the following command:

   pkgadd -d *<CD drive directory>/<gina package file name>*

   You are then prompted to do the following:

   - ◊ Select the GINA subpackage to be installed.
   - ◊ Specify the directory in which GINA is to be installed, e.g. `/opt/gina/`*`GINA`*
   - ◊ Define users for this and the directories below:

     e.g. *`gina`* as userid (default: `root`)

     e.g. *`tmns`* as groupid (default: `root`)
   - ◊ Specify the source directory from which GINA is installed. *<CD drive directory>*

   The GINA package is then installed under `/opt/gina/`*`GINA`*, the directory structure created and the structure filled with the files.

Druck vom 24. 01.2001 17:00.21 install

5b. The following steps are required if the GINA package is stored on the delivery CD in *compressed* format:

◊ Copy the GINA package from the CD to an intermediate directory

cp *<CD drive directory>/<gina package file name>*.gz *<intermediate directory name>*

◊ Unpack the file in the intermediate directory using `gunzip`.

◊ Install the unpacked GINA package

pkgadd -d *<intermediate directory name>/<gina package file name>*

You are then prompted to do the following:

◊ Select the GINA package to be installed.

◊ Specify the directory in which GINA is to be installed,
e.g. `/opt/gina/`*`GINA`*

◊ Define users for this and the directories below:

e.g. *gina* as userid (default: `root`)

e.g. *tmns* as groupid (default: `root`)

◊ Specify the source directory from which GINA is installed.

<intermediate directory name>/<gina package file name>

The GINA package is then installed under `/opt/gina/`*`GINA`*, the directory structure created and the structure filled with the files.

3.3.2 UNIX / HP-UX

See *UNIX (Solaris, SINIX)* on page 16 for steps 1 through 4.

5. Insert the GINA delivery CD into the CD drive and install the *uncompressed* GINA package using the following command:

`swinstall`

The `swinstall` command opens a menu interface via which the aforementioned information is queried.

The GINA package is then installed, the directory structure created and the structure filled with the files.

3.3.3 Windows NT

Installation of the full-feature or partial version (RT version) under Windows NT is performed using the `setup` command, the installation tool is included in the GINA package on the delivery CD.

1. Log in as administrator.
2. Create a new directory for GINA.
3. Insert the GINA delivery CD in the CD drive and start the graphical installation interface using the command

   `<drive>:\setup`

   The dialog box which is displayed queries the path of the installation directory and the components to be installed.

   The GINA package is then installed, the directory structure created and the structure filled with the files.

If, during installation, you selected GINA packages containing the services `DomsEventHandler` and `DomsDynConnectHandler`, they will be automatically activated the next time the system starts (under WindowsNT only).

Further information on this can be found in the Developer Manual [13] in the section entitled *Compiling and linking, Special features under Windows NT.* ❍❍●

If you intend to run GINA on a machine without a network link, you must set the `LM_NO_NETWORK` environment variable in order to avoid the wait times resulting from futile attempts to access a license server. ❍❍●

Special points in relation to the operation of the GINA PS browser

Before the GINA PS browser is called (on UNIX platforms), the directory in which the GINA phases were installed must be included in the `PATH` environment variable.

Example for csh

```
sentenev PATH <gina_install_dir>/:$PATH
```

Furthermore, the environment variable `LD_LIBRARY_PATH` (for Solaris, SINIX) or `SHLIB_PATH` (for HP-UX) must contain the file in which the GINA library is located.

Example for csh

- Solaris, SINIX
  `setenv LD_LIBRARY_PATH <gina_install_dir>/lib:$LD_LIBRARY_PATH`
- HP-UX
  `setenv SHLIB_PATH <gina_install_dir>/lib:$LD_LIBRARY_PATH`

On Windows NT all information necessary to the operation of the PS browser is entered during installation by the setup routine.

That is, the environment variable `GINADIR` points to the file `<gina_install_dir>`, `<%GINADIR%\bin>` is added to the `PATH` variable.

The PS browser is started using the *Start* > *Programs* menu. ❍❍●

Removing `TPStartShut.o`

If a T-ORB application which is to run on a WindowsNT machine uses custom startup and shutdown exits, the dummy implementation of `TPStartShut.o` shipped with GINA must be removed from the libraries

`.../lib/doms.lib` and `.../lib/domsmt.lib`

before this application is linked.

The two scripts `extpss.bat` and `extpssmt.bat` are made available for this.

`extpss.bat` removes the object `TPStartShut.o` from the library `.../lib/doms.lib` and stores it in the library `.../lib/domstpss.lib`.

`extpssmt.bat`
removes the object `TPStartShut.o` from the library `.../lib/domsmt.lib` and stores it in the library `.../lib/domstpssmt.lib`.

The scripts must be executed in the `lib` directory of the GINA installation directory. One prerequisite is that the `%MSVCBIN%` variable must be set. It must point to the `bin` directory of the Microsoft Developer Studio.

Once `TPStartShut.o` has been removed, the library `domstpss.lib` or `domstpssmt.lib` must be specified when linking T-ORB applications which do not use custom startup or shutdown exits.

Further information on special points to be noted when creating a GINA application under NT can be found in the section entitled *Compiling and linking*, *Special features under Windows NT* in the Developer Manual [13]. ❍❍●

3.3.4 BS2000

The following steps are required when installing under BS2000:

1. Log in to BS2000/OSD under the name `$GINA`
2. Read in the tape using `ARCHIVE, IMPORT ..., DEVICE=TAPE-C4`

   The following DVS files are created under the `$GINA` account:

`GINA.PAX`	pax archive containing the POSIX files of the transfer
`LIBxxx`	libraries of the GINA runtime system in PLAM format

3. Copy `GINA.PAX` as a POSIX file and unpack it within a POSIX shell:

   ```
   /START-POSIX-SHELL
   $ cd <gina_install_dir>
   $ bs2cp 'bs2:gina.pax' gina.pax
   $ pax -r -f gina.pax
   ```

A directory `GINA` which contains the subdirectories and files described above is created in `<gina_install_dir>`.

For further information please see the Release Notice.

The compiler C/C++ V3.0B requires the CRTE 2.1 runtime system. Refer to the Release Notice for the C/C++ compiler V3.0A for information on dependencies with other BS2000 system components.

The GINA T-ORB component requires the XDR functionality from the POSIX-NSL 010 software unit. This has been available as a special release since the end of May 1998. ❍❍●

Druck vom 24. 01.2001 17:00.22 install

3.3.5 Environment variables

Certain environment variables must be set when implementing the various components of GINA. Depending on the components implemented, these variables are:

Using the T-ORB on basis of *open*UTM

`UTMPATH` This environment variable contains the path of the installed *open*UTM environment.

`UPICPATH` This environment variable contains the name of the directory with the `upicfile` file. `upicfile` is created using the configuration generator `config` (see chapter 6 on page 43).

This variable need only be set if you are starting a client application in a directory other than that containing the `upicfile` file.

`GINACONFIG` This variable is used to select a directory other than the actual directory for the `servernames` file generated by `config` for the execution of a client application.

Using the T-ORB on basis of BEA TUXEDO

`TUXDIR` This environment variable contains the path to the BEA TUXEDO installation directory.

`TUXCONFIG` This environment variable contains the name of the `TUXCONFIG` configuration file.The absolute path name must be specified here.

This file is created using the `crbincf` script generated by the `config-tux` configuration generator.

`BDMCONFIG` This environment variable contains the name of the `BDMCONFIG` configuration file. The absolute path name must be specified here.

This file is created using the `crbincf` script.

This variable need only be set if you are using the `Domains` component from BEA TUXEDO.

`GINACONFIG` This variable is used to select a directory other than the actual directory for the `servernames` file generated by `config-tux` for the execution of a client application.

Implementation of the Persistency Service

`INFORMIXDIR` This environment variable contains the name of the directory for the installed INFORMIX environment. The environment variable `PATH` must contain an `$INFORMIXDIR/bin` entry.

`INFORMIXSERVER`
This environment variable identifies the current instance of the database server processed by the application database.

Following installation, you should inform the users of the correct values for the variables `UTMPATH`, `INFORMIXDIR` and `INFORMIXSERVER`, or configure their environment accordingly.

When using NLS/GLS functionality, the following variables must also be set:

`DB_LOCALE` If this environment variable is set, then the new databases will be generated on the database server using the locale specified by the variable `DB_LOCALE` instead of the standard locale.

`CLIENT_LOCALE`
This environment variable is needed for the database client and the database server. It must also be set before an application that works with an NLS/ GLS-capable database is started.

No further actions are required for BS2000/OSD.

Refer to the section *Compiling and linking* in the Developer Manual [13] for information on special points to note when configuring the environment under Windows NT. ❍❍●

Druck vom 24. 01.2001 17:00.23 install

3.4 Deinstallation

The installation procedures used depend on the system base; the variants described below are examples only.

Installation and deinstallation should, therfore, always be carried out as described for your system base or in accordance with the information contained in the Release Notice. ❍❍●

System-specific commands are used to deinstall GINA.

3.4.1 UNIX (Solaris, SINIX)

1. Log in as `root`.
2. Switch to the GINA directory:

   `cd /opt/gina/`*`GINA`*
3. Delete all GINA subdirectories and files using the system command:

   `pkgrm` *`gina`*
4. Delete the GINA root directory:

   ```
   cd /
   rmdir /opt/gina/GINA
   ```
5. Remove the *`gina`* user set up at installation.
6. Remove the *`tmns`* user group set up at installation.

3.4.2 UNIX / HP-UX

Deinstallation of GINA under HP-UX is performed using the `swremove` command.

The `swremove` command opens a menu interface via which the information required for deinstallation is queried.

The GINA package is then deinstalled, and the subdirectories and files are deleted.

3.4.3 Windows NT

Deinstallation of GINA under Windows NT, Windows95 and Windows98 is performed by selecting the *Software* icon (*Start > Settings > System Controls)*.

Then select `GINA`, and start the graphical deinstallation interface. The necessary information is queried in the dialog box.

The GINA package is then deinstalled, and the subdirectories and files are deleted.

Druck vom 24. 01.2001 17:00.23 install

3.5 Availability, restrictions

This section provides information on platform-related restrictions, the availability of the GINA components as well as libraries which are missing.

UNIX with CFront compilers

The GINA component PS/client is not available on UNIX systems with CFront compilers (e.g. HP-UX).

WindowsNT

- The configuration generator for T-ORB with the graphical interface (WinConfig) is not available.
- INFORMIX V9.2 does not contain an implementation of the XA interface. For that reason, GINA offers a simulation. This means that INFORMIX cannot currently be used to its full potential with *open*UTM and BEA TUXEDO. For further information please see the enclosed Release Notice.

Further information on this can be found in the Developer Manual [13] in the section entitled *Compiling and linking*, *Special features under Windows NT.*

Windows95/98

- Only the GINA runtime system for T-ORB/client applications (type 4) is available on Windows95/98. This means that a T-ORB/client application can only be run on Windows95. The application must be created in full on WindowsNT, i.e. execution of the GINA generators, compilation, linking, and creation of the configuration.
- Installing GINA

  An RT version for WindowsNT must be used for the GINA installation under Windows95/98.
- C runtime system DLLs required (multithreaded)

  ```
  msvcrt.dll
  msvcirt.dll
  ```

- Configuration file

  The configuration file (the file that reads the `config`) must contain the line

  `OPERATING_SYSTEM("OS_WINNT")`

  for both WindowsNT and Windows95.

- Environment variable

  The `COMPUTERNAME` environment variable is automatically set on WindowsNT, it must be set by the user on Windows95.

BS2000/OSD

The PS/client (client part) as well as the direct database system connection are not available in the runtime system of T-ORB/client under BS2000/OSD.

The `mgendb`, `WinConfig`, `mspgen2` and `idlgen1` as well as the PS browser (`bruno` and `cuno`) generators of the development system are not supplied.

Druck vom 24. 01. 2001 17:00:23 inbtneb

4 Creating GINA applications

GINA applications comprise a range of modules which must be linked as a program before execution time, at the latest. Since GINA represents a framework which offers different functionality in different variants, special activities must be carried out to create an application, depending on its specific type. Further details are provided in the following sections.

4.1 Application variants

The modular structure of GINA results in the following application types:

Type 1 Transaction-monitored communication with integrated local data store

This corresponds to using the full functionality of GINA. Both T-ORB and the Persistency Service are used; object-oriented transactions are supported.

Type 2 Transaction-monitored local data store

This corresponds to using the Persistency Service for applications which do not require transaction-monitored communication in the network and which do not use T-ORB. Transaction-monitored local data storage means that the transaction system of the database is used.

Type 3 Transaction-monitored communication

This corresponds to using T-ORB for applications which do not require transaction-monitored data storage or which do not implement the Persistency Service. Important fields of application for this type involve pure message queuing or connectivity to mainframe applications.

Type 4 Linking applications of types 1 and 3 without transaction monitoring

These are client applications which establish the connection between the user and GINA server applications, e.g. via a GUI.

Type 5 Persistency Service/client applications

In addition to T-ORB/client, these non-transaction monitored applications also use the Persistency Service in the form of the Persistency Service/client.

Type 6 GINA applications associated with a Persistency Service/client application

These use T-ORB and the Persistency Service of GINA jointly.

Type 7 Dynamic client *without* a PS/client component

Type 8 Dynamic Persistency Service/client applications

Application of the type 5 as a dynamic client *with* a PS/client component.

Depending on the type of your application, you may have to specify different libraries when linking.

4.2 Environment

If your system administrator has not already done so, you must at least set the environment variables `INFORMIXDIR` and `UTMPATH`. Please refer to section 3.3.5 on page 21.

You must also know the directory in which GINA has been installed on your system. It is also advisable to read through the Release Notice, as this may also contain information on creating GINA applications. The Release Notice can be found in the subdirectory `doc` of the GINA directory (see section 3.2.3 on page 14).

Druck vom 24. 01. 2001 17:00:24 inbtneb

4.3 The generators

The generators of GINA can be found in the `bin` directory of the GINA installation (see section 3.2 on page 11). These are listed below:

- The generator of the Persistency Service

  `mgen1` for analyzing C++ header files

  `mgen2` for generating the database schema and access methods

  `mgendb` for creating the database

  `mdiff` replaces `mgen2` in the context of schema evolution

- The T-ORB generator

  `config` for configuring T-ORB with the WinConfig graphical user interface for creating `config` input files.

WinConfig is only available on UNIX platforms. ❍❍●

- The T-ORB generator

  `dogen1` for analyzing C++ header files

  `dogen2` for creating global interface, stub, and export interface classes

- The MIO generator

  `miogen1` for analyzing C++ header files

  `miogen2` for creating encryption and decryption methods

- The IDL compiler

  `idlgen` for creating global interface, stub and export interface classes, as well as creating encryption and decryption methods.

The generators of the Persistency Service are described in the Developer Manual [13] and the Reference Manuals [14]. A description of the configuration options can be found in chapter 5 on page 31 of the present manual.

The T-ORB generator `config` is described in chapter 6 on page 43.

4.4 Makefiles

Software development for larger systems, in particular, is much easier to handle with the UNIX program `make`.

The Developer Manual [13] contains a detailed example of a possible makefile configuration.

Druck vom 24.01.2001 17:00:24 per-konf

5 Configuring the Persistency Service

When examining the configuration of the Persistency Service, it is generally a good idea to consult the documentation relating to the underlying database management system. All explanations in this chapter relate to the database management system INFORMIX on the UNIX platform. The environment can be different for other database management systems or on other operating systems.

5.1 Setting up the database

The database is set up in several stages. First of all the system software of the database management system used must be installed as prescribed by the manufacturer [19]. In addition, IDs must be created for the administration of the database system. In this context, it is also important to consider the definition of roles (IDs) in the area of administration and security. Both the database server environment, for example INFORMIX Dynamic Server 2000, and that of the development environment, for example ESQL (in the client SDK), must be installed.

In the next phase, the resources assigned to the database system must be configured. This particularly includes system resources such as

– the number of needed semaphores,
– size and number of shared memory segments,
– configuration of disk memory (raw devices, RAID controller), and the
– definition of connections (connectivity) to the database server.

When using the INFORMIX database system, for example, the entries in the files `/etc/hosts` and `/etc/services` must match those in the `sqlhosts` file. In general, these global activities must be performed under the `root` administrator ID.

The next step is for the database administrator to structure the assigned disk resources (dividing into chunks), applying a structure (dbspaces, blobspaces) which can be used in the definition of the database schema and in the backup and recovery concept. The aim of this activity is to achieve an optimum solution under the aspects of fault tolerance, performance and availability. These activities are supported by appropriate tools, which are supplied with the database system. For example, here INFORMIX offers the `onmode` tool.

To enable these aims to be achieved, knowledge is required on datasets and their dynamics (static, low frequency of change, high rate of change), as well as the access behavior to the data (navigating, value-based). This information is supplied to the Persistency Service generator as customizing[14] input in an optimization process. From this input, the Persistency Service generator produces the optimized definition of the database schema (table layout, indexing, constraints) and the access functions to database tables (see section 5.2 on page 33).

A further task of the database administrator is to configure the database server. Here, the parameters for the operation of the database are defined in the appropriate configuration files and runtime procedures (specific to the selected database system). These parameters include maximum values such as

– the number of transactions permitted in parallel
– the wait time for a response from a remote database server
– name spaces for the current session like `DBSERVERNAME` or `SERVERNUM`
– parameters that affect the execution of the session, e. g. the time interval between two checkpoints.

If the database and transaction monitor are linked via the XA interface, the information required by the participating systems must be configured in appropriate generation and start procedures.

The initialization of the database is particularly important. This includes the initialization of tables from a file which contains the data in ASCII notation, or the transfer of binary-encoded data as an excerpt from a secondary database of the same type. Migration tools which are specific to the database system are available for transferring data from existing databases or files.

Druck vom 24.01.2001 17:00:24 per-konf

5.2 Customizing the database layout

GINA offers you the option of customizing the database layout and the access privileges. One aspect of this relates to the assignment of names of database tables for persistent classes, whereby the default is that the class names of the specialist models are also used to designate the database tables. However, this may cause problems if the maximum permitted length for identifiers is subject to specific restrictions.

For example, table names are limited to 18 characters in the INFORMIX version used. Since GINA itself requires a character as a prefix, class names that are longer than 17 characters must be mapped to a shorter table name. ❍❍●

Another aspect is the definition of the actual storage space for tables in the physical database structures, i.e. the dbspaces. Performance requirements can be considered here, and the optimization functions of the database system can be used.

In terms of access privileges, requirements with regard to access protection are defined at database or table level. The functionality also depends on the concrete database server.

The specifications for this customization are defined in description files. These are made known to the `mgen2` generator using the `-r` and `-t` options [14]. The generator then creates an appropriately modified database schema. The following description files exist:

– the pfx file and
– the tbl file

5.2.1 The pfx file

The description file that is called using the `-r` option, the so-called pfx file, handles the specification of class-wide attribute names.

The attribute names used for the mapping to the database consist of the following components:

classPrefix_attributCounter_extension

classPrefix
: is the part of the field name that is common to all fields in a class. Neither does inheritance change it, i.e. it has the same value in the subclass as in the superclass. It can be specified by the user with the help of the *description file* specified below.

  When using identical views, this prefix is used to generate the attribute names of both the table and also the view belonging to the class.

attributCounter and *extension*
: are always created by mgen2.

Layout of the description file

The description file consists of formatted lines with the following line format.

classname prefix

Classname
: represents a valid C++ identifier.

prefix
: represents a valid SQL identifier. The prefix entries must differ for all classes and can be a maximum of 6 characters in length.

Please note that all entries in the description file are case-sensitive.

If no prefix is found for a specific class, it is automatically generated by the `mgen2` generator. The created prefixes have the following format:

X*dddd* or X*ddddd* with *d*=0,1,2,..,9

The generation starts at X*dddd*=X1024 or X*mmmm*, where *mmmm* represents the highest value for *dddd*+1 found in the description file. That means that if `X10200` was found in the description file, the generation starts at `X10201`.

Once prefixes have been specified by the `mgen2` for all classes, both those found in the description file and the newly generated prefixes are output to the specified description file. The old version of the pfx file is first backed up to a file with the extension `.bak`. An empty or non-existent description file can be specified for use in the creation of the first description file.

Druck vom 24.01.2001 17:00:25 per-konf

5.2.2 The tbl file

The description file that is called using the `-t` option, the so-called tbl file, contains the customizing functions:

- name mapping of database tables and definition of the concrete storage space required by tables (section 1 on page 35),
- fragmentation of a database table (section 2 on page 37),
- name mapping of views (section 3 on page 37),
- versioning of the database (section 4 on page 38) and
- access privileges for tables and views (section 5 on page 38).

The specification of this file, and of all entries in it, is optional.

The structure of the tbl files follows the general structure of the intermediate format files (see the chapter entitled *Tools* in the Reference Manuals [14]). The following line types are supported:

`#` for identifying a comment line
`#| comment text`

`t` for identifying a definition line for the mapping of database table names or for defining the concrete storage space required by tables

`fr` for identifying a definition line for fragmenting a database table

`v` for identifying the version control

`s` for identifying a definition line for access privileges

`vw` for identifying a definition line for the mapping of database view names

The meaning of the individual lines as well as the necessary columns are now described for each customizing function.

1. Name mapping of database tables and definition of the concrete storage space required by tables

   A table line contains the following fields:

   `t | ident | name | resource | firstExtent | nextExtent |`

   Only one definition line of this type is permitted per class.

   Note that all “|” separators must be set.

   `ident` Name of the class in the specialist model.

   Mandatory entry

name Name of the table to which the specialist model classes are to be mapped. The default value is the class name.

Optional entry

resource

Entry to identify a dbspace in which the tables are to be stored. Such a dbspace is a storage area which is based on the physical storage media. Additional storage instructions are entered using the “fr definition lines” and result in the fragmentation of the table.

Assigning tables to storage structures allows for the implementation of optimization strategies for backing up data and for access functions. The default value for storing a table is the dbspace in which the database is generated (`-d` *dbspace* option for the `mgendb` generator [14]).

Optional entry

firstExtent

Entry for defining settings for the size of physically related storage areas, which are reserved with the first entry in the table. Like *resource,* this function serves to optimize the storage layout and enables it to be adapted to the volume of data involved. The minimum, maximum and default values depend on the database system and on the layout of the storage media. They are specified as integer multiples of pages (generally 2 Kb).

Optional entry

nextExtent

Optional entry for defining settings for the size of physically related storage areas, which are reserved with each extension to a table. As with the *firstExtent* entry, this function serves to optimize the storage layout.

Example

```
t | myClassname1 | myNewTableName1 | dbspaceName | 4 | 4 |
t | myClassname2 | myNewTableName2 | | | |
```

Druck vom 24.01.2001 17:00:25 per-konf

2. Fragmentation of a database table

   A fragmentation line contains the following fields:

   `fr | ident | resource |`

   A number of definition lines of this type are permitted per class, at least two `resources` must be defined for the purpose of fragmenting.

   Note that a partitioning of storage space which is taken into account during the fragmentation can also be specified in the `t` definition line.

   ident Name of the class in the specialist model.

   Mandatory entry

   resource Entry to identify a dbspace in which the tables are to be stored. Such a dbspace is a storage area which is based on the physical storage media. The dbspace must exist and can only be assigned once per table. Optimization strategies for access functions can be realized through the fragmentation of tables across several storage structures.

   Mandatory entry

   Example

   `fr| myClassname1 | Dbspace_1|`

   `fr| myClassname1 | Dbspace_2|`

3. Name mapping of database views

   A view line contains the following entries:

   `vw | ident | name`

   Only one definition line of this type is permitted per class.

   ident Name of the class in the specialist model

   Mandatory entry

   name Name of the view to which the class of the specialist model is to be mapped.

   Example

   `vw | myClassname | myNewViewName|`

4. Versioning of the database

   A version line contains the following entries:

   v | *ident*|

   Only one version line is permitted per database.

   ident Version of the database schema (string with a maximum of 30 characters)

   Mandatory entry

   Example

   ```
   v | 1.0 |
   ```

5. Access privileges for tables and identical views

   A privilege line contains the following entries:

   s | *ident* | *name* | *resource* |

   A number of definition lines of this type are permitted per class.

   ident Name of the class in the specialist model

   Mandatory entry

   name Name of the user that is assigned the privilege defined in the *resource* column

   Mandatory entry

   resource Mandatory entry in a definition line for privileges. It names the privilege assigned to the user.

   Value range:

INSERT	allows you to make an object persistent.
DELETE	allows you to delete a persistent object.
UPDATE	allows you to modify a persistent object.
SELECT	allows you to instance a persistent object.

   Only privileges are permitted at table and view level and only those that do not change the schema (ALTER TABLE, CREATE TABLE).

   If the *ident* class is a view class, only the privilege SELECT is permitted.

   Only the CONNECT TO PUBLIC privilege is generated at database level.

Druck vom 24.01.2001 17:00:26 per-konf

Example

```
s | myClassname | hubert | INSERT |
```

If a user is assigned the `DELETE` privilege, he/she also needs the `SELECT` privilege for the purpose of internal consistency checks. ❍❍●

5.2.3 Further options

The mgendb tool can also be used to influence the configuration of the database. In addition to specifications concerning the default database user and NLS/GLS specifications, the -d option can be used to determine a dbspace in which the database tables to be created are to be set up. The mgendb supports the following options:

-d *dbspace*

The named *dbspace* of the database instance is used as the default storage area of the database to be generated. If this option is not set, the database is generated in the root dbspace of the instance. See also [19].

Example
```
mgendb ... -d gina ...
```

-l *ltype*

Specifying this option generates the database in such a way that it is NLS-/GLS-compatible (native language support/global language support), assuming the database server supports this functionality. The concrete language variant (locale) is specified by *ltype*. The syntax of *ltype* depends on the current database server and the operating system used. It is described in the database reference manual or the NLS documentation of the operating system. If the -l option is not set, the database is generated without NLS/GLS.

The following environment variables are set in the mgendb for an NLS-/GLS-compatible database:

CLIENT_LOCALE=*ltype*
DB_LOCALE=*ltype*

The DB_LOCALE environment variable is needed for the database server. If this variable is set, then the new databases will be generated using the locale specified by the variable DB_LOCALE instead of the standard locale.

The CLIENT_LOCALE environment variable is needed for the database client and the database server. It must also be set before an application that works with an NLS-/GLS-capable database is started.

Example
for INFORMIX V9.2 for creating a database with the German character set:

on SunOS	`mgendb -l de_de.8859-1 myDB (LANG=de)`
on SINIX	`mgendb -l de_de.8859-1 myDB (LANG="De_DE.88591@TE")`
on HP-UX	`mgendb -l de_de.8859-1 myDB (LANG=de_DE.iso88591)`

Druck vom 24.01.2001 17:00:26 per-konf

Note
ltype must be a valid locale on the database server platform. If mgendb is called in another operating system environment, errors can occur since *ltype* is not recognized as a valid locale identifier. Therefore mgendb should be called on the same platform as the one on which the database server is running.

`-u` *userName*

A default database user can be entered if this option is specified. The default DB user is identified by *userName* and must be a valid UNIX user. If a default user is specified, no public user has access to the database. If no default user is specified, all users have access to the database.

Example
```
mgendb ... -u ginauser ...
```

Druck vom 24.01.2001 17:00:26 torb-kon

6 Configuring T-ORB for *open*UTM

This chapter describes how to configure communication between GINA applications. The configuration does not require any modifications to the source program. However, it cannot take place while the application is running; rather it must be performed *before* the application *starts*.

The configuration of T-ORB is based on the configuration of *open*UTM [29] and *open*UTM-Client [31]. Knowledge of the respective manuals will therefore aid comprehension. These manuals are listed under *Related publications* at the back of this manual.

6.1 Overview

To operate a distributed system on the basis of GINA, certain information must be known on the desired system structure, the current parameter settings of the host, and the characteristics of the GINA applications.

The configuration tool described here records this information in the form of a general system description, from which it generates the data required by GINA.

The information you must enter can be divided into the following parts (see Figure 1 on page 44):

– specifications on system-wide settings, such as the participating systems

– specifications on host-specific parameters, such as operating system resources

– specifications on application-specific parameters, such as the number of work processes

– specifications on the connections between applications

Practically speaking, some of the parameters can be specified in more than one of these areas. In such cases, the values set higher up in the hierarchy must be taken as default settings. These values are then overwritten by the settings in more specific areas. The desired settings can thus be specified separately for each application if required (customizing).

A connection between two applications is defined by the number of parallel communication channels and the definition of the controlling application of each communication channel. Even if all communication channels are controlled by one application, messages can be

transmitted by both sides. The controlling application simply has priority over the passive application. From a performance point of view, the controlling application should be the one which generally (or more frequently) opens the communication.

The definition of the type and number of all connections between the applications is equivalent to the definition of the communication structure for the entire system.

Figure 1 on page 44 is a symbolic representation of the hierarchies in the definition of this communication structure. It is not drawn to scale to reflect the scope of the individual subdescriptions. Depending on the particular network structure, the definition of the connections between the applications can be much larger than the definition of the applications.

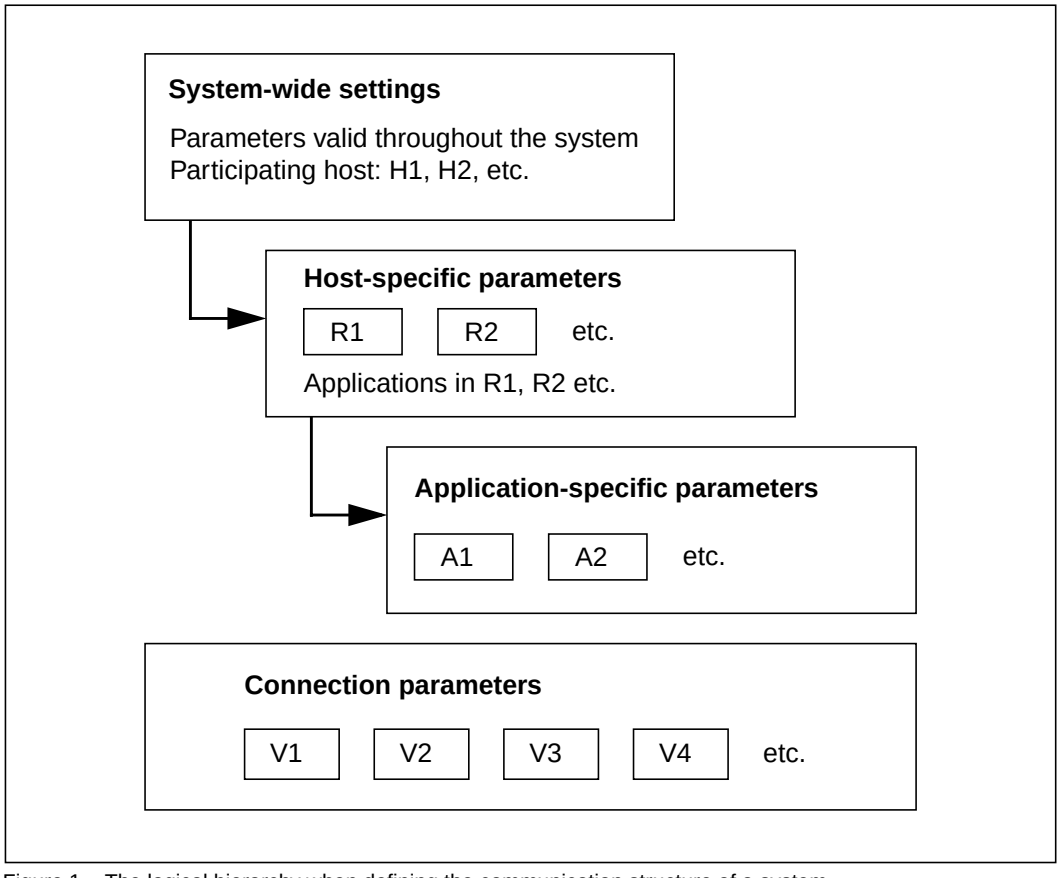


Figure 1 The logical hierarchy when defining the communication structure of a system

Druck vom 24.01.2001 17:00:26 torb-kon

The communication structure of a system can be depicted by a graph with nodes and edges. The nodes correspond to the applications, while the edges represent the communication channels. The first three blocks in the diagram define the nodes of the graph with respect to the specific system, host or application; the last block describes the edges of the graph.

From the input data (which has already been explained), the configuration generator `config` creates the following output data for each application of each host:

– for each application:
  a GINA-specific address file containing all addressable server applications

– for each application:
  a configuration file for the transaction monitor used by GINA

– for each host:
  a list of the necessary TNSX entries

The configuration can be performed on a central computer for the entire system. The files created in the process can then be distributed to the target computers and installed there. The final tasks which must be performed locally are carried out during this installation. Moving the final tasks to the target computers eliminates the need to install the transaction monitor on the generation computer. Because the output data for the generation comprises only text files, the hardware and operating system independence of the configuration process is also guaranteed.

In the first version of the configuration tool, the procedure is that change requests are sent to a central location and that a new configuration process will be implemented from there only. The configuration generator makes it easier to configure the runtime environment for T-ORB and T-ORB/Client. Revision generation is described in section 6.3 on page 88.

A central configuration is recommended for the following reasons:

– It does not make sense to change the configuration on the local target computers because these modifications will be overwritten with the next global update process.

– Another argument against changing the settings locally is that modifications to the hierarchy of the system often affect more than one host. It is precisely changes of this type that require consistency checks, which are not possible on the local level.

6.2 Configuration language

The configuration generator `config` reads a text file which describes the configuration of the entire system in a T-ORB-specific language. This file contains the necessary information on the network protocol, the transaction monitor, T-ORB, and the specialist application.

The elements of the language include keywords, literals, separators and comments. Blanks, tabs, line feeds, form feeds and white spaces are ignored. The characters `//` introduce a comment, the line feed character terminates it.

6.2.1 Statements

The configuration language contains a range of statements which are introduced by keywords; these are explained below. The statements include numerous UTM parameters [31].

The statements may contain two different styles: uppercase letters only or lowercase letters only. ❍❍●

ADMIN

The `ADMIN` statements on the *system level* apply to all hosts, if there is no `ADMIN` statement with the same user ID in the `HOST` statement.
The `ADMIN` statements on the *host level* apply accordingly to all applications, if there is no `ADMIN` statement in the application.

The optional ADMIN statement has the following parameters:

– user ID (LETTER)
– password (LETTER)

Example

```
ADMIN ( "srv1", "srv1" )
```

ADDRESS

The `ADDRESS` statement describes how the foreign *open*UTM application is to be addressed by the GINA application.

It has the following parameters:

– name of the transaction code (LETTER)
– name of the subroutine (LETTER)

Example

```
ADDRESS ( "VSVONREB", "MV1VON" )
```

Druck vom 24.01.2001 17:00.27 torb-kon

The ADDRESS statement describes how the GINA application is to be addressed by the foreign *open*UTM application. It has the following parameters:

– local name of the GINA application (LETTER)
– name of the local transaction code (LETTER)

APPLICATION

The APPLICATION statement describes a client which is connected via T-ORB/Client. It comprises the following components:

– OsId of the application (NUMBER, 1 ... 32767)
– LayerId of the application (NUMBER, 1 ... 32767)
– User-friendly name of the application (LETTER)
– The REMOTE flag can be specified as an option for a client connected via T-ORB/Client. The REMOTE flag classifies a client as a remote client [31].
  If the REMOTE flag is not specified as the last parameter, a client is generated locally if it only has local connections to servers; otherwise, a remote generation is created.
– A user ID and password can be specified. This information is generated automatically if it is not specified explicitly.

Example
```
APPLICATION ( 131, 1, "CmdTrm1" )
```

AREA

The AREA statements on the *system level* apply to all hosts, if there is no AREA statement with the same data area name in the HOST statement.
The AREA statements on the *host level* apply accordingly to all applications, if there is no AREA statement in the application.

The optional AREA statement has one mandatory parameter

– Data area name (LETTER)

and the following optional parameters

– "ACCESS" = "DIRECT" | "INDIRECT" (OS_UNIX, OS_WINNT)
– "LOAD-MODULE" = "lmodname" (OS_BS2000)
– "LOAD" = "STATIC" (OS_BS2000)
– "LOAD" = "(POOL,poolname)" (OS_BS2000)
– "LIB" = "omlname" (OS_BS2000)

Example
```
AREA ( "area1", "ACCESS" = "DIRECT" )
```

ASYN_PRIORITY

The ASYN_PRIORITY statement defines the priority classes for the processing of asynchronous requests (see also the description of the PRIORITIES keyword). The statement has the following format:

```
ASYN_PRIORITY(RELATIVE | ABSOLUTE | EQUAL)
{
  PRIO(1 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
  PRIO(2 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
  PRIO(3 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
  PRIO(4 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
  PRIO(5 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
  PRIO(6 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
  PRIO(7 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
}
```

The ASYN_PRIORITY statement can be used to define up to seven classes (PRIO1 through PRIO7). As part of this process, PRIO1 is mapped to TAC class 10, PRIO2 to TAC class 11, etc. of *open*UTM. TAC class 10 has the second highest priority, TAC class 16 the lowest. TAC class 9 with the highest priority is reserved for the T-ORB.

The RELATIVE, ABSOLUTE and EQUAL attributes of the ASYN_PRIORITY statement were transferred 1:1 from the *open*UTM generation.

These attributes are used to determine the priority with which processes will be distributed to the asynchronous TAC classes containing executable asynchronous requests or interrupted asynchronous requests.

No new pending asynchronous requests will be started once the maximum number of asynchronous operations that can be open simultaneously is reached (set in MAX ASYNTASKS=(...,service_number)), instead an interrupted outstanding asynchronous operation will be selected on the basis of priority and continued.

ABSOLUTE attribute: Absolute priority

A free process will always be assigned to the class with the highest priority, i.e. Class 9, provided that there are pending asynchronous requests or interrupted asynchronous requests for this class. Processes that become free will only process requests from a class with a lower priority if the message queues of all classes with a higher priority do not contain any pending or interrupted asynchronous requests.

Druck vom 24.01.2001 17:00.27 torb-kon

RELATIVE attribute: Relative priority

Free processes will be assigned to higher priority classes more often than lower priority classes provided that there are pending or interrupted requests for these higher priority classes. If there are requests present for all classes, a free process will be assigned to TAC class 9 twice as often as to TAC class 10, and twice as often to TAC class 10 as to TAC class 11, etc.

EQUAL attribute: Equal priority

All classes will be serviced equally if there are requests present. This equal distribution can be disrupted if a class does not contain any pending requests at times or subprogram runs with blocking calls (e.g. KDCS call PGWT) frequently occur in that class.

Further information on the RELATIVE, ABSOLUTE and EQUAL attributes can be found in the *open*UTM documentation V5.0A, *Generating and Handling Applications* [26] in the section entitled *TAC-PRIORITIES – specify priorities of the TAC classes*.

The following TIMER, EVENT, CONTROL and PGWT attributes are optional and can be specified in any order.

TIMER attribute

The T-ORB runtime system has an internal cyclical timer. This is a cyclical order (controlled by means of the CYCLETIME statement of the config generator) that takes over various tasks within T-ORB, e. g. initiation of the EventControl mechanism (see the EVENT attribute).

The TIMER attribute can be used to determine which priority class is to be assigned to the internal cyclical timer. TIMER can only be specified for *one* priority class. If the TIMER attribute is not specified, the class with the average priority (rounded up to the next highest priority) will be assigned to the cyclical timer. If, for example, PRIO1 through PRIO4 is specified, PRIO2 will be assigned to the timer.

EVENT attribute

The T-ORB runtime system maintains the so-called EventControl mechanism internally. This mechanism is responsible for ensuring that requests to T-ORB/client applications that are buffered in the T-ORB application for technical reasons are actually delivered.

The EVENT attribute can be used to determine the priority class in which the EventControl mechanism is to run. EVENT can only be specified for one priority class. If the EVENT attribute is not specified, the class with the average priority (rounded down to the next lowest priority) will be assigned to the EventControl mechanism. If, for example, PRIO3 through PRIO7 is specified, PRIO5 will be assigned to the EventControl mechanism.

This attribute is only needed if T-ORB/client applications are connected to the T-ORB application.

CYCLIC attribute

A so-called administration order is created for each cyclical timed request (DomsClient::cyclicOrder). This administration order makes sure that the cyclical timed request is executed in the selected timeframe.

The CYCLIC attribute can be used to determine the priority class in which these administration orders are to run. CYCLIC can only be specified for *one* priority class. If the CYCLIC attribute is not specified, the class with the average priority (rounded up to the next highest priority) will be assigned to the administration orders. If, for example, PRIO1 through PRIO6 is specified, PRIO3 will be assigned to the administration orders.

PGWT attribute

If a priority class is assigned requests that need to execute callAndWait and execChainAndWait calls, the class must have the appropriate authorization.

The PGWT attribute can be used to grant this authorization to the priority class. PGWT can be specified for each priority class.

BCAMAPPL

The optional BCAMAPPL statement permits the specification of BCAM application names for a TA application. The number of application names required depends on the configuration and is checked by config. If too few application names are specified, config outputs an appropriate error message. A BCAMAPPL statement is only permitted within a TA_APPLICATION statement.

The application names are generated without the BCAMAPPL statement.

Example

```
BCAMAPPL ( "GINA1", "GINA2" )
```

BEST_BCAMAPPL

The keyword BEST_BCAMAPPPL results in a separate BCAMAPPL statement being generated for each connection.

CANCEL

The CANCEL statement of the EVENTCONTROL block defines the interval after which an event on a client expires, i.e. the event has exceeded its delivery time since the CANCEL time, in relation to its start time. The interval is specified by four values for days, hours, minutes, and seconds.
The default value is CANCEL(2,0,0,0).

Druck vom 24.01.2001 17:00:28 torb-kon

The CANCEL statement on *system level* applies to all hosts if there is no CANCEL statement in the HOST statement.
The CANCEL statement on *host level* applies accordingly to all TA applications if there is no CANCEL statement in the TA_APPLICATION statement.

CHECK

The CHECK statement of the EVENTCONTROL block defines the interval after which an event on a client is overdue, i.e. incorporated into the checking mechanism, in relation to its start time. The interval is specified by four values for days, hours, minutes and seconds.
The default value is CHECK(0,1,0,0).

The CHECK statement on *system level* applies to all hosts if there is no CHECK statement in the HOST statement.
The CYCLICTIME statement on the *host level* applies accordingly to all TA applications if there is no CHECK statement in the TA_APPLICATION statement.

CONNECT

The CONNECT entry describes a connection between a T-ORB/Client and a T-ORB-Server. The statement has the following parameters:

– OsId of the first partner (e.g. client)
– LayerId of the first partner
– OsId of the second partner (e.g. server)
– LayerId of the second partner

Example

```
CONNECT ( 221, 1,  204, 1 )
```

CYCLE

The CYCLE statement of the EVENTCONTROL block defines the interval in which the events on the client that have not yet been delivered are checked. The interval is specified by four values for days, hours, minutes and seconds.
The default value is CYCLE(0,0,10,0).

The CYCLE statement on *system level* applies to all hosts if there is no CYCLE statement in the HOST statement.
The CYCLE statement on *host level* applies accordingly to all TA applications if there is no CYCLE statement in the TA_APPLICATION statement.

CYCLICORDER

The CYCLICORDER statement defines the maximum number of cyclical tasks permitted per TA_APPLICATION.
The default value is CYCLICORDER(10).

The CYCLICORDER statement on *system level* applies to all hosts if there is no CYCLICORDER statement in the HOST statement.
The CYCLICORDER statement on *host level* applies accordingly to all TA applications if there is no CYCLICORDER statement in the TA_APPLICATION statement.

CYCLICTIME

The CYCLICTIME statement defines the interval for activation of the application-specific timer. The interval is specified by four values for days, hours, minutes and seconds.
The default value is CYCLICTIME(0,0,0,0).

The CYCLICTIME statement on *system level* applies to all hosts if there is no CYCLICTIME statement in the HOST statement.
The CYCLICTIME statement on *host level* applies accordingly to all TA applications if there is no CYCLICTIME statement in the TA_APPLICATION statement.

DYNAMIC_CONNECT

A DYNAMIC_CONNECT statement on *system level* and on *host level* permits the definition of connections which are used by non-transaction-monitored clients to communicate with transaction-monitored T-ORB applications.

This statement can be used to define connections to several different T-ORB applications.

DYNAMIC_CONNECT((os,layer,number), ...)

The parameters os and layer refer to the OsLayerId in the TA_APPLICATION statement of the transaction-monitored T-ORB application to which connections are to be generated.

The parameter number defines the number of connections.

If there is no DYNAMIC_CONNECT statement for a host, then this host inherits the connections specified at *system level*.

If this list is empty, this means that this host does not inherit any connections from the *system level*.

Druck vom 24.01.2001 17:00.28 torb-kon

EVENTCONTROL

The EVENTCONTROL statement comprises the following components:

- CYCLE
- CHECK
- CANCEL

Each component is optional.

Example

```
EVENTCONTROL
{
CYCLE ( 0, 0, 20, 0 )
CHECK ( 0, 2, 0, 0 )
CANCEL ( 3, 0, 0, 0 )
}
```

FOREIGN_APPLICATION

The FOREIGN_APPLICATION statement describes a foreign *open*UTM application. It comprises the following components:

- OsId of the application (NUMBER, 1 ... 32767)
- LayerId of the application (NUMBER, 1 ... 32767)
- optional: a foreign application number
- user-friendly name of the application (LETTER))

Example

```
FOREIGN_APPLICATION ( 1, 4, "FA1" )
```

FOREIGN_APPLICATION_NUMBER

The FOREIGN_APPLICATION_NUMBER statement summarizes a list of ADDRESS statements: this list is referenced by means of a number. An ADDRESS statement defines the following components:

- FunctionId of the converter function (NUMBER)
- TransactionCode of the application (LETTER)
- TransactionType of the application (LETTER)
- optional: ConverterId of the application (NUMBER).

FOREIGN_SESSION

The FOREIGN_SESSION statement describes connections between a GINA application and a foreign *open*UTM application. It comprises the following components:

– session type (LETTER); the generator currently supports LU6.1
– mapping (optional MAP_SYSTEM parameter)
  MAP=
  controls ASCII/EBCDIC conversion when exchanging unformatted messages with other applications.
  SYSTEM
  *open*UTM converts the data in the message area from ASCII to EBCDIC prior to dispatch or from EBCDIC to ASCII following receipt. The message may only contain printable characters.
  See the *KDCDEF control statement SESCHA* in [26].
– a SESSIONPOINT statement for the GINA application
– a SESSIONPOINT statement for the foreign *open*UTM application

HOST

The HOST statement describes the configuration of a host. The HOST statement comprises the following components:

– host name
– optional: the CMX version of the host (default CMX040)
– optional: the UTM version of the host (default UTM040)
– optional: the flag RESERVE
– Internet address of the host (INTERNETADDRESS)
– shared memory and semaphore key (KEYVECTOR) (not OS_BS2000)
– available port numbers (PORTADDRESS) (not OS_BS2000)
– host-specific customizing statements
  (ADMIN, CYCLICTIME, EVENTCONTROL, MAX, RMXA, START and START_RM)
– description of existing applications

Example

```
HOST ( "Host2" )
{
INTERNETADDRESS ( 127.0.0.2 )
KEYVECTOR ( 5005, 5040 )
PORTADDRESSES ( 2000, 2100 )
...
}
```

Druck vom 24.01.2001 17:00.28 torb-kon

IMPORT

The IMPORT statements on the *system level* apply to all hosts, if there is no IMPORT statement with the same file name in the HOST statement.
The IMPORT statements on the *host level* apply accordingly to all applications, if there is no IMPORT statement with the same parameters in the application.

The optional IMPORT statement has one parameter:

– filename

The IMPORT statement is transformed into the KDCDEF control statement OPTION DATA.

IN_CONVERTER

The IN_CONVERTER statement describes a converter function or converter class method that is called if there is a message from a foreign *open*UTM application.

The IN_CONVERTER statement has the following parameters:

– FunctionId of the converter function or
– ClassId and ClassMethodId of the converter class method

INTERNETADDRESS

The INTERNETADRESS statement is used to specify the current Internet address of the host.

KEYVECTOR

The KEYVECTOR statement contains the keys for the following UTM areas:

– the global application semaphore (SEMKEY)
– the access key for the KAA shared memory segment (KAASHMKEY)
– the access key for communication (IPCSHMKEY)
– the access key for file accesses (CACHESHMKEY)

The KEYVECTOR statement has the following parameters:

– start key
– end key

For the areas of importance, see the CMX documentation [7].

The difference between the start key and end key must be large enough for all applications on the host to have their own key. ❍❍●

The following number of keys must be defined for each application:

– 1 key each for CACHESHMKEY, IPCSHMKEY and KAASHMKEY

– (6 + n + m)/10 semaphore keys for SEMKEY
n = maximum number of work processes
m = maximum number of communication partners connected simultaneously
When calculated, the value must be rounded off to a whole number.

MAX

The MAX statement allows you to customize TP applications.

On *system level*, the MAX statement applies to all hosts if there is no MAX statement of the same name in the HOST statement.
MAX statements on *host level* apply accordingly to all applications.

The optional MAX statement has the following parameters:

– name of the statement (LETTER)
– value of the statement as a string

The statement names APPLINAME, CACHESHMKEY, CONRTIME, DPUTLIMIT1, DPUTLIMIT2, IPCSHMKEY, KAASHMKEY, NB, SEMARRAY and TRMSGLTH must not be specified, as these statements are generated automatically.

If the KDCFILE statement is not specified or if the file name is omitted from the value string, a MAX statement with the name KDCFILE is generated.

MPOOL

The MPOOL statement is only supported for TA applications on a host using the OS_BS2000 operating system. The MPOOL statements on the *system level* apply to all hosts, if there is no MPOOL statement with the same pool name in the HOST statement.
The MPOOL statements on the *host level* apply accordingly to all applications, if there is no MPOOL statement in the application.

The optional MPOOL statement has two mandatory parameters

– Poolname
– LSIZE = poolsize

and the following optional parameters

– ACCESS = READ | WRITE
– LIB = omlname
– PAGE = X'xxxxxxxx'
– SCOPE = GROUP | GLOBAL
– SHARETAB = csectname

Druck vom 24.01.2001 17:00:29 torb-kon

NET_ACCESS

The NET_ACCESS statement describes the manner in which the application is linked to the network.

The NET_ACCESS statements on *system level* apply to all hosts if there is no NET_ACCESS statement with the same name in the HOST statement.
The NET_ACCESS statements on *host level* apply accordingly to all applications.

The optional NET_ACCESS statement has the following parameters:

– SINGLE-THREADED
  Each network connection is administered in a separate network process.
– MULTI-THREADED
  A number of network connections are administered in a single process. Network connections are assigned to network processes via listener IDs that you assign to the application name (BCAMAPPL statement) of your application.
– If you specify MULTI-THREADED, ALL, a LISTENER-ID is specified for each BCAMAPPL statement.
– If you specify MULTI-THREADED, number, number LISTENER-IDs are specified for each BCAMAPPL statement.

The default value is SINGLE-THREADED.

OPERATING_SYSTEM

The optional OPERATING_SYSTEM statement defines the operating system of the host. Possible values are OS_UNIX, OS_WINNT and "OS_BS2000"
The default value is OPERATING_SYSTEM(OS_UNIX).

The OPERATING_SYSTEM statement on *system level* applies to all hosts if there is no OPERATING_SYSTEM statement in the HOST statement.

OUT_CONVERTER

The OUT_CONVERTER statement describes a converter function that is called before a message from a foreign *open*UTM application is delivered.
The optional OUT_CONVERTER statement has the following parameter:

– converter ID of the converter function

PORTADDRESSES

The PORTADDRESSES statement contains the port numbers for the TNSX entries. The statement has the following parameters:

– first port number
– last port number

See the CMX documentation [7] for information on value ranges.

The difference between the start address and end address must be large enough for each BCAMAPPL entry on the host to have its own port number. The following rules apply when calculating the number of BCAMAPPL entries of an application:

– Each CONNECT statement creates two BCAMAPPL entries.
– If the BEST_BCAMAPPL flag is set in the APPLICATION statement, a separate BCAMAPPL entry is generated for each SESSION statement of the application.
– Without the BEST_BCAMAPPL flag, the number of BCAMAPPL entries generated corresponds to the maximum number of SESSION entries specified between the current server application and one of its partners.

❍❍●

PRIORITIES

The PRIORITIES statement has the following format:

```
PRIORITIES
{
  SYNC_PRIORITY(RELATIVE | ABSOLUTE | EQUAL [,free_sync] )
  {
    PRIO(1 [,PGWT] )
    PRIO(2 [,PGWT] )
    PRIO(3 [,PGWT] )
    PRIO(4 [,PGWT] )
    PRIO(5 [,PGWT] )
    PRIO(6 [,PGWT] )
    PRIO(7 [,PGWT] )
     }
  ASYN_PRIORITY(RELATIVE | ABSOLUTE | EQUAL)
  {
    PRIO(1 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
    PRIO(2 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
    PRIO(3 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
    PRIO(4 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
    PRIO(5 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
    PRIO(6 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
```

Druck vom 24.01.2001 17:00:29 torb-kon


```
    PRIO(7 [,TIMER] [,EVENT] [,CYCLIC] [,PGWT] )
    }
  SCHEDULE
  {
    FUNCTION(<FunctionId>,<SyncPriority>,<AsyncPriority>,<Infostring>)
    INSTMETHOD(<ClassId>,<InstMethodIdd>,<SyncPriority>,
                                        <AsyncPriority>,<Infostring>)
    CLASSMETHOD(<ClassId>,<ClassMethodId>,<SyncPriority>,
                                         <AsyncPriority>,<Infostring>)
    CLASS(<ClassId>,<Syncpriority>,<AsyncPriority>,<Infostring>)
  }
}
```

The individual SYNC_PRIORITY, ASYN_PRIORITY and SCHEDULE blocks are described under the relevant keywords.

– The new PRIORITIES statement and the old PRIORITY and SCHEDULE statements are mutually exclusive, i. e. you can only assign one type of priority control to a TA_APPLICATION.

– If you do not specify the PRIORITIES statement, priority control will be effected by means of the function for controlling processing of orders by restricting processes.

– You can specify the PRIORITIES statement at *system*, *host* and *TA application* level. The statement will be passed on from SYSTEM to HOST and from HOST to TA_APPLICATION.

– Both the old priority control and the priority control described here can be used within a T-ORB configuration.

PRIORITY

TAC classes are controlled at the level of the specialist class, class method, instance method and function.

The PRIORITY statement defines three TAC classes with the priorities HIGH, MEDIUM and LOW. This statement summarizes a list of three entries:

```
HIGH   ( tasks [, SYNC_WAIT] )
MEDIUM ( tasks [, SYNC_WAIT] )
LOW    ( tasks [, SYNC_WAIT] )
```

The parameter tasks specifies the maximum number of processes in the application which work concurrently with this TAC class. It depends on the statements MAX TASKS and MAX TASKS-IN-PGWT.

The attribute SYNC_WAIT specifies whether synchronous calls (callAndWait, addToChain and execChainAndWait) are permitted in this TAC class. This attribute can only be assigned to one of the three TAC classes.

The PRIORITY statement may only be used once on *system, host* or *TA application level*.
The PRIORITY statement on the *system level* applies to all hosts, if there is no PRIORITY statement in the HOST statement.
The PRIORITY statement on the *host level* applies accordingly to all TA applications.

REMOTE

The REMOTE keyword classifies a client as a remote client. If the REMOTE flag is not specified as the last parameter, a client is generated locally if it only has local connections to servers; otherwise, a remote generation is created.

REPOSITORY

The REPOSITORY statement defines the file name of a repository in the current directory. This repository contains the generation number and the associated port numbers and keys for each of the OsId and LayerId pair.

If the file exists, these values are read, saved internally, and used during generation. If the file does not exist, the values are generated automatically. The repository is written once generation of these values is concluded.

The repository supports generation for a modified configuration. If an application is removed, the same identifiers are used in the kdcdf file for the remaining applications. The same port numbers are also used for the TNSX entries.

The REPOSITORY statement is optional. If this statement is not specified, no memory is used in the generation.

REREADTIME

The REREADTIME statement has one parameter that defines a time interval in minutes.

REREADTIME(minutes)

The status of the gina.config file of a TA_APPLICATION will be checked every minutes minutes. The file will be read in once more if there has been a change in the status.

The REREADTIME statements at *system level* apply for all hosts unless the HOST statement contains a REREADTIME statement.
The REREADTIME statements at *host level* apply accordingly for all TA applications.

Druck vom 24.01.2001 17:00:30 torb-kon

RMXA

The RMXA statement describes a Resource Manager.

The RMXA statements on *system level* apply to all hosts if there is no RMXA statement of the same name in the HOST statement.
The RMXA statements on *host level* apply accordingly to all applications.

The optional RMXA statement has the following parameters:

– manufacturer name of the Resource Manager
– form of the XA interface
– name of the OML with connection module (OS_BS2000 only)

SCHEDULE

The SCHEDULE statement is used to explicitly assign the specialist classes, class methods, instance methods and functions to the TAC classes with the priorities HIGH, MEDIUM and LOW. This statement summarizes a list of entries:

```
CLASS       ( ClassId, Priority )
CLASSMETHOD ( ClassId, ClassMethodId, Priority )
INSTMETHOD  ( ClassId, InstMethodId, Priority )
FUNCTION    ( FunctionId, Priority )
```

The parameter Priority has the values HIGH, MEDIUM or LOW.

The SCHEDULE statement may only be used once on *system*, *host* or *TA application level*.
The SCHEDULE statement on the *system level* applies to all hosts, if there is no SCHEDULE statement in the HOST statement.
The SCHEDULE statement on the *host level* applies accordingly to all TA applications.

SCHEDULE (PRIORITIES)

The SCHEDULE statement defines the priority assigned to the specialist (synchronous/ asynchronous) processing mechanism (see also the description of the PRIORITIES keyword). The statement has the following format:

```
SCHEDULE
{
  FUNCTION(<FunctionId>,<SyncPriority>,<AsyncPriority>,<Infostring>)
  INSTMETHOD(<ClassId>,<InstMethodIdd>,<SyncPriority>,
                                      <AsyncPriority>,<Infostring>)
  CLASSMETHOD(<ClassId>,<ClassMethodId>,<SyncPriority>,
                                       <AsyncPriority>,<Infostring>)
  CLASS(<ClassId>,<Syncpriority>,<AsyncPriority>,<Infostring>)
}
```

The SCHEDULE statement can be used to assign functions, instance methods, class methods or all methods of a class to a specific priority.

Classes, class methods, instance methods and functions that are not explicitly assigned to a specific category using the SCHEDULE statement will be automatically assigned to the second highest priority class.

The following minimum and maximum values apply to the individual parameters of the SCHEDULE statement::

Parameter	Minimum value	Maximum value
ClassId	1025	SHRT_MAX
ClassMethodId	1	SHRT_MAX
InstMethodId	1	SHRT_MAX
Function	1025	LONG_MAX
SyncPriority	1	7
AsyncPriority	1	7x

SESSION

The SESSION statement describes connections between server applications. The SESSION statement comprises the following components:

– session type (LETTER); at present, the generator supports LU6.1
– mapping (optional MAP_SYSTEM parameter)
  MAP=
  controls ASCII/EBCDIC conversion when exchanging unformatted messages with other applications.
  SYSTEM
  *open*UTM converts the data in the message area from ASCII to EBCDIC prior to dispatch or from EBCDIC to ASCII following receipt. The message may only contain printable characters.
  See the *KDCDEF control statement SESCHA* in [26].
– one SESSIONPOINT statement for each accessible application

Druck vom 24.01.2001 17:00:30 torb-kon

SESSIONPOINT

The SESSIONPOINT statement specifies the number of connections that can be controlled by an application.

The SESSIONPOINT statement has the following parameters:

– OsId of the current application (NUMBER)
– LayerId of the current application (NUMBER)
– number of connections controlled by this application

A SESSIONPOINT statement that describes the connections from a GINA application to a foreign *open*UTM application has the following additional components:

– An ADDRESS statement
  This statement describes the parameters used to specify how the foreign *open*UTM application is to be addressed.
– An optional OUT_CONVERTER statement
  This statement describes a converter function that is called before a message from a foreign *open*UTM application is delivered.

A SESSIONPOINT statement that describes the connections from a foreign *open*UTM application to a GINA application has the following additional components:

– An ADDRESS statement
  This statement describes the parameters used to specify how the GINA application is to be addressed.
– An IN_CONVERTER statement
  This statement describes a converter function or converter class method that is called if there is a message from a foreign *open*UTM application.

START

The START statement allows you to customize TP statements.

The START statements on *system level* apply to all hosts if there is no START statement of the same name in the HOST statement.
The START statements on *host level* apply accordingly to all applications.

The optional START statement has the following parameters:

– name of the statement
– value of the statement as a string

Permissible statement names are TASKS, ASYNTASKS and TASKS-IN-PGWT.

START_RM

The START_RM statements on *system level* apply to all hosts if there is no START_RM statement with the same manufacturer name in the HOST statement.
The START_RM statements on *host level* apply accordingly to all applications.

The optional START_RM statement has the following parameters:

– manufacturer name of the database
– Openstring OS
  If the keyword APPLICATION is written instead of the Openstring parameter, the TA_application name is used as the Openstring.

START_VALUE

The START_VALUE statement defines the first generation number that is used in the generation of identifiers in statements for the kdcdef program. This results in unique reproductions of the OsId and LayerId pair in these identifiers. By specifying different start values, different subsystems that can be combined can be generated. The subsystem connections, however, must still be completed.

The START_VALUE statement is optional. If this statement is not specified, generation of the identifiers starts with the value 1.

If a repository exists, it is used to ascertain the generation numbers, i.e. the START_VALUE statement is ignored.

SYNC_PRIORITY

The SYNC_PRIORITY statement defines the priority classes for the processing of dialog requests (see also the description of the PRIORITIES keyword). The statement has the following format:

```
SYNC_PRIORITY(RELATIVE | ABSOLUTE | EQUAL [,free_sync] )
{
  PRIO(1 [,PGWT] )
  PRIO(2 [,PGWT] )
  PRIO(3 [,PGWT] )
  PRIO(4 [,PGWT] )
  PRIO(5 [,PGWT] )
  PRIO(6 [,PGWT] )
  PRIO(7 [,PGWT] )
}
```

Druck vom 24.01.2001 17:00:31 torb-kon

The SYNC_PRIORITY statement can be used to define up to seven classes (PRIO1 through PRIO7). As part of this process, PRIO1 is mapped to TAC class 2, PRIO2 to TAC class 3 etc. of *open*UTM. TAC class 2 has the second highest priority, TAC class 8 the lowest. TAC class 1 with the highest priority is reserved for the T-ORB.

The RELATIVE, ABSOLUTE, EQUAL attributes of the SYNC_PRIORITY statement and the free_sync parameter were transferred 1:1 from the *open*UTM generation.

These attributes are used to specify the strategy by which free processes will be distributed to the dialog TAC classes containing pending requests. Pending dialog requests only arise if the number of requests retrieved from the request exchange at any one time exceeds the number of processes available for the dialog TAC classes. These pending requests will be written to the request queues of the transaction codes from which they will be read and processed in accordance with their priority by the processes that become free.

ABSOLUTE attribute: Absolute priority

A free process will always be assigned to the class with the highest priority (TAC class 1) provided that there are pending requests for this class. Lower priority classes will only be serviced if there are no pending requests in any of the higher priority classes.

RELATIVE attribute: Relative priority

Free processes will be assigned to higher priority classes more often than lower priority classes provided that there are pending requests for these higher priority classes. If there are requests present for all dialog classes, a free process will be assigned to TAC class 1 twice as often as to TAC class 2, and twice as often to TAC class 2 as to TAC class 3, etc.

EQUAL attribute: Equal priority

All classes will be serviced equally if there are requests present. This equal distribution can be disrupted if a class does not contain any pending requests at times or subprogram runs with blocking calls (e.g. KDCS call PGWT) frequently occur in that class.

free_sync parameter

This parameter allows you to restrict the total number of processes that may process requests to dialog TAC classes relative to the number of processes in the application.

In free_sync specify the number of processes in the application that are to be kept free for processing requests and that do not belong to any dialog TAC class.

Minimum value: 0 (no restriction)

Maximum value: TASKS - 1 (TASKS from MAX statement)

Default: 1

PGWT attribute

If a priority class is assigned requests that need to execute callAndWait and execChainAndWait calls, the class must have the appropriate authorization.

The PGWT attribute can be used to grant this authorization to the priority class. PGWT can be specified for each priority class.

Further information on the RELATIVE, ABSOLUTE and EQUAL attributes as well as on the free_sync parameter can be found in the *open*UTM documentation V5.0, *Generating and Handling Applications* [26] in the section entitled *TAC-PRIORITIES – specify priorities of the TAC classes*.

SYSTEM

The SYSTEM keyword must always be the first keyword in the description. The current description is enclosed in braces, i.e. SYSTEM {...}.

Example SYSTEM {...}.

The SYSTEM statement can be written with a flag in rounded brackets.

Example SYSTEM (TNS_LESS_SESSION){...}

The flag TNS_LESS_SESSION means:
No entries will be generated in the files tnsxin and tnsxdel for the statements SESSION and FOREIGN_SESSION.

TA_APPLICATION

The TA_APPLICATION statement describes a transaction-monitored application on the current host (HOST). It comprises the following components:

– TP application name (LETTER)
– OsId of the application (NUMBER, 1 ... 32767)
– LayerId of the application (NUMBER, 1 ... 32767)
– User-friendly name of the application (LETTER)
– optional: BCAMAPPL name can be assigned exclusively to a GINA application (BEST_BCAMAPPL)

A further form describes a future application with the following parameters:

– OsId of the application (NUMBER, 1 .. 32767)
– LayerId of the application (NUMBER, 1 .. 32767)
– Flag RESERVE

Druck vom 24.01.2001 17:00.31 torb-kon

This statement acts as a wildcard so that a GINA-SESSION between a T-ORB application without the flag RESERVE and a T-ORB application with the flag RESERVE can be defined (see SESSION). A T-ORB application can then be generated from this session in the event of a revision generation. The normal syntax of this statement must then be used. OsId and LayerId may not then be changed.

Customizing statements (ADMIN, CYCLICTIME, EVENTCONTROL, MAX, RMXA, START and START_RM) are permitted after the TA_APPLICATION description of a server.

Example

```
TA_APPLICATION ( "OS1", 1, 1, "OS1" )
{
...
}
```

6.2.2 Lexical structure

This section describes how the configuration generator combines the contents of the configuration file into symbols (like the keywords) for syntax analysis. The description is in the notation used by the UNIX command `lex`.

```
%{

%}

letter                    [a-zA-Z_]
DGS                       [0-9]+
letter_or_digit           [a-zA-Z_0-9]

%%
[ \t\n\v\r\f]+            ;
\/\/.*\n                  ;
\{                        {return(BBOPEN);}
\}                        {return(BBCLOSE);}
\(                        {return(SBOPEN);}
\)                        {return(SBCLOSE);}
\,                        {return(COMMA);}
\=                        {return(EQUAL);}
{DGS}                     {yylval.number=atol(yytext);
                           return(NUMBER);}
{DGS}\.{DGS}\.{DGS}\.{DGS} {strcpy(yylval.string,yytext);
                           return(INADDRESS);};
\"[^"\n]*                 {yytext[yyleng++] = yyinput();
                           yytext[yyleng] = '\0';
                           lettercpy(yylval.string,yytext);
                           return(LETTER);};
{letter}{letter_or_digit}* {return(rwlookup());};
.                         {return(ERROR);};
%%
```

Druck vom 24.01.2001 17:00.31 torb-ken

6.2.3 Syntax

This section describes the syntax of the configuration language in the notation used by the UNIX command yacc.

```
%{

%}

%start sysblock

%union
{
    SessPoint *sessPt;
    long      number;
    char      string[80];
}

%token <number> NUMBER            /* positive integer          */
%token ADDRESS                    /* statement                 */
%token ADMIN                      /* statement                 */
%token APPLICATION                /* statement                 */
%token AREA                       /* statement                 */
%token ASYN_PRIORITY              /* statement                 */
%token BCAMAPPL                   /* statement                 */
%token CANCEL                     /* statement                 */
%token CHECK                      /* statement                 */
%token CLASS                      /* statement                 */
%token CLASSMETHOD                /* statement                 */
%token CONNECT                    /* statement                 */
%token CYCLE                      /* statement                 */
%token CYCLICORDER                /* statement                 */
%token CYCLICTIME                 /* statement                 */
%token DYNAMIC_CONNECT            /* statement                 */
%token EVENTCONTROL               /* statement                 */
%token FOREIGN_APPLICATION        /* statement                 */
%token FOREIGN_APPLICATION_NUMBER /* statement                 */
%token FOREIGN_SESSION            /* statement                 */
%token FUNCTION                   /* statement                 */
%token HOST                       /* statement                 */
%token IMPORT                     /* statement                 */
%token INSTMETHOD                 /* statement                 */
%token INTERNETADDRESS            /* statement                 */
```


```
%token IN_CONVERTER              /* statement                */
%token KEY_VECTOR                /* statement                */
%token MAX_STMT                  /* statement                */
%token MPOOL                     /* statement                */
%token NET_ACCESS                /* statement                */
%token OPERATING_SYSTEM          /* statement                */
%token OUT_CONVERTER             /* statement                */
%token PORTADDRESSES             /* statement                */
%token PRIO                      /* statement                */
%token PRIORITIES                /* statement                */
%token PRIORITY                  /* statement                */
%token REPOSITORY                /* statement                */
%token REREADTIME                /* statement                */
%token RMXA                      /* statement                */
%token SCHEDULE                  /* statement                */
%token SESSION                   /* statement                */
%token SESSIONPOINT              /* statement                */
%token START                     /* statement                */
%token START_RM                  /* statement                */
%token START_VALUE               /* statement                */
%token SYNC_PRIORITY             /* statement                */
%token SYSTEM                    /* statement                */
%token TA_APPLICATION            /* statement                */
%token ABSOLUTE                  /* flag                     */
%token BEST_BCAMAPPL             /* flag                     */
%token CYCLIC                    /* flag                     */
%token EVENT                     /* flag                     */
%token EQUAL                     /* flag                     */
%token HIGH                      /* flag                     */
%token LOW                       /* flag                     */
%token MAP_SYSTEM                /* flag                     */
%token MEDIUM                    /* flag                     */
%token PGWT                      /* flag                     */
%token RELATIVE                  /* flag                     */
%token REMOTE                    /* flag                     */
%token RESERVE                   /* flag                     */
%token SYNC_WAIT                 /* flag                     */
%token TIMER                     /* flag                     */
%token TNS_LESS_SESSION          /* flag                     */
%token <string> INADDRESS        /* Internet address         */
%token <string> LETTER           /* string                   */
%token BBOPEN                    /* {                        */
%token BBCLOSE                   /* }                        */
%token SBOPEN                    /* (                        */
```

Druck vom 24.01.2001 17:00.32 torb-kon


```
%token SBCLOSE                  /* )                          */
%token COMMA                    /* ,                          */
%token EQUALS                   /* =                          */
%token CONFIG_ERROR             /* error code (internal)      */
%%

sysblock       : system
                 BBOPEN
                 sys_statement_list_opt
                 multi_host
                 multi_link_opt
                 BBCLOSE
               ;
system         : SYSTEM
               | SYSTEM
                 SBOPEN
                 TNS_LESS_SESSION
                 SBCLOSE
               ;

sys_statement_list_opt
               : /* empty */
               | sys_statement_list
               ;

sys_statement_list
               : org_statement_list  s_statement_list_opt
               |                     s_statement_list
               ;

org_statement_list
               :                     org_statement
               | org_statement_list org_statement
               ;

s_statement_list_opt
               : /* empty */
               | s_statement_list
               ;

s_statement_list
               :                   s_statement
               | s_statement_list s_statement
               ;
```

```
s_statement    : statement
               | foreign_app_number
               | dynamic_connect
               ;

org_statement  : start_value
               | repository
               | operating_system
               ;

start_value    : START_VALUE SBOPEN NUMBER SBCLOSE
               ;

repository     : REPOSITORY SBOPEN LETTER SBCLOSE
               ;

foreign_app_number
               : foreig_app_number_header
                 BBOPEN address_list BBCLOSE
               ;
foreign_app_number_header
               : FOREIGN_APPLICATION_NUMBER
                 SBOPEN NUMBER SBCLOSE
               ;

address_list   :              address_elm
               | address_list address_elm
               ;

address_elm    : ADDRESS SBOPEN
                 NUMBER COMMA LETTER COMMA LETTER SBCLOSE
               | ADDRESS SBOPEN
                 NUMBER COMMA LETTER COMMA LETTER COMMA NUMBER SBCLOSE
               ;

operating_system
               : OPERATING_SYSTEM SBOPEN LETTER SBCLOSE
               ;
```

Druck vom 24.01.2001 17:00.32 torb-kon


```
statement       : max
                | rm
                | admin
                | start
                | start_rm
                | cyclictime
                | cyclicorder
                | eventcontrol
                | net_access
                | area
                | mpool
                | import
                | priorities
                | priority
                | schedule
                | rereadtime
                ;

cyclictime      : CYCLICTIME SBOPEN
                  NUMBER COMMA
                  NUMBER COMMA
                  NUMBER COMMA
                  NUMBER SBCLOSE
                ;

cyclicorder     : CYCLICORDER SBOPEN NUMBER SBCLOSE
                ;
eventcontrol    : EVENTCONTROL BBOPEN
                               eventcontrol_body
                               BBCLOSE
                ;

eventcontrol_body
                : cycle cyclic_rest
                | check check_rest
                | cancel
                ;

cyclic_rest     : check check_rest
                | check_rest
                ;
```

```
check_rest      : cancel
                | /* empty */
                ;

cycle           : CYCLE SBOPEN
                  NUMBER COMMA
                  NUMBER COMMA
                  NUMBER COMMA
                  NUMBER SBCLOSE
                ;

check           : CHECK SBOPEN
                  NUMBER COMMA
                  NUMBER COMMA
                  NUMBER COMMA
                  NUMBER SBCLOSE
                ;

cancel          : CANCEL SBOPEN
                  NUMBER COMMA
                  NUMBER COMMA
                  NUMBER COMMA
                  NUMBER SBCLOSE
                ;

net_access      : NET_ACCESS SBOPEN LETTER COMMA LETTER SBCLOSE
                | NET_ACCESS SBOPEN LETTER SBCLOSE
                | NET_ACCESS SBOPEN NUMBER SBCLOSE
                ;

area            : AREA SBOPEN LETTER area_parameter_list_opt SBCLOSE
                ;

area_parameter_list_opt
                : /* empty */
                | COMMA area_parameter_list
                ;

area_parameter_list
                :                       area_parameter
                | area_paramter_list area_parameter
                ;
```

Druck vom 24.01.2001 17:00.32 torb-kon


```
area_parameter : LETTER EQUAL LETTER
               ;

mpool          : MPOOL SBOPEN LETTER COMMA mpool_parameter_list SBCLOSE
               ;

mpool_parameter_list
               :                              mpool_parameter
               | mpool_parameter_list COMMA mpool_parameter
               ;

mpool_parameter: LETTER EQUAL LETTER
               ;

import         : IMPORT SBOPEN LETTER SBCLOSE
               ;

priorities     : PRIORITIES BBOPEN priorities_body BBCLOSE
               ;
priorities_body: sync_priority asyn_priority p_schedule
               ;

sync_priority  : SYNC_PRIORITY SBOPEN r_a_e free_sync SBCLOSE
                               BBOPEN prio_list BBCLOSE
               ;

r_a_e          : RELATIVE
               | ABSOLUTE
               | EQUAL
               ;

free_sync      : /* empty */
               | COMMA NUMBER
               ;

prio_list      : prio_element
               | prio_list prio_element
               ;

prio_element   : PRIO SBOPEN NUMBER prio_rest
               ;
```

```
prio_rest      :            SBCLOSE
               | COMMA PGWT SBCLOSE
               ;

asyn_priority  : ASYN_PRIORITY SBOPEN r_a_e SBCLOSE
                               BBOPEN aprio_list BBCLOSE
               ;

aprio_list     : i          aprio_element
               | aprio_list aprio_element
               ;

aprio_element  : PRIO SBOPEN NUMBER aprio_rest
               ;

aprio_rest     :                 SBCLOSE
               | COMMA tecp_list SBCLOSE
               ;

tecp_list      :                 tecp_element
               | tecp_list COMMA tecp_element
               ;

tecp_element   : TIMER
               | EVENT
               | CYCLIC
               | PGWT
               ;

p_schedule     : SCHEDULE BBOPEN p_schedule_list BBCLOSE
               ;

p_schedule_list:                 p_schedule_element
               | p_schedule_list p_schedule_element
               ;
```

Druck vom 24.01.2001 17:00.33 torb-kon


```
p_schedule_element
                : FUNCTION SBOPEN NUMBER COMMA NUMBER COMMA
                                  NUMBER COMMA LETTER SBCLOSE
                | INSTMETHOD SBOPEN NUMBER COMMA NUMBER COMMA
                          NUMBER COMMA NUMBER COMMA LETTER SBCLOSE
                | CLASSMETHOD SBOPEN NUMBER COMMA NUMBER COMMA
                          NUMBER COMMA NUMBER COMMA LETTER SBCLOSE
                | CLASS SBOPEN NUMBER COMMA NUMBER COMMA
                          NUMBER COMMA LETTER SBCLOSE
                ;

priority        : PRIORITY BBOPEN priority_body BBCLOSE
                ;

priority_body   : low_task    low_task_rest
                | medium_task medium_task_rest
                | high_task   high_task_rest
                ;

low_task        : LOW SBOPEN NUMBER opt_sync_wait SBCLOSE
                ;

medium_task     : MEDIUM SBOPEN NUMBER opt_sync_wait SBCLOSE
                ;

high_task       : HIGH SBOPEN NUMBER opt_sync_wait SBCLOSE
                ;

low_task_rest   : medium_task high_task
                | high_task medium_task
                ;

medium_task_rest
                : low_task high_task
                | high_task low_task
                ;

high_task_task  : low_task medium_task
                | medium_task low_task
                ;

opt_sync_wait   : /* empty */
                | COMMA SYNC_WAIT
                ;
```

```
rereadtime     : REREADTIME SBOPEN NUMBER SBCLOSE
               ;

schedule       : SCHEDULE BBOPEN schedule_list BBCLOSE
               ;

schedule_list  :                schedule_element
               | schedule_list schedule_element
               ;

schedule_element
               : CLASS       SBOPEN NUMBER COMMA prio SBCLOSE
               | CLASSMETHOD SBOPEN NUMBER COMMA NUMBER COMMA prio
                             SBCLOSE
               | INSTMETHOD  SBOPEN NUMBER COMMA NUMBER COMMA prio
                             SBCLOSE
               | FUNCTION    SBOPEN NUMBER COMMA prio SBCLOSE
               ;

prio           : LOW
               | MEDIUM
               | HIGH
               ;

ta_attrib      : /* empty */
               | BBOPEN
                 ta_appl_statement_list_opt
                 BBCLOSE
               ;

non_ta_attrib  : /* empty */
               | BBOPEN
                 non_ta_appl_statement_list_opt
                 BBCLOSE
               ;

foreign_attrib : /* empty */
               | BBOPEN
                 foreign_appl_statement_list_opt
                 BBCLOSE
               ;
```

Druck vom 24.01.2001 17:00.33 torb-kon


```
ta_appl_statement_list_opt
                : /* empty */
                | ta_appl_statement_list
                ;

non_ta_appl_statement_list_opt
                : /* empty */
                | non_ta_appl_statement_list
                ;

ta_appl_statement_list
                :                          ta_appl_statement
                | ta_appl_statement_list ta_appl_statement
                ;

non_ta_appl_statement_list
                :                              non_ta_appl_statement
                | non_ta_appl_statement_list non_ta_appl_statement
                ;

ta_appl_statement
                : statement
                | bcamappl_statement
                ;

bcamappl_statement
                : BCAMAPPL SBOPEN bcam_list SBCLOSE
                ;

bcam_list       :                  bcam_element
                | bcam_list COMMA bcam_element
                ;

bcam_element    : LETTER
                ;

non_ta_appl_statement
                : authentication
                ;

foreign_appl_statement_list_opt
                : /* empty */
                | foreign_appl_statement_list
                ;
```

```
foreign_appl_statement_list
                :                                foreign_appl_statement
                | foreign_appl_statement_list foreign_appl_statement
                ;

foreign_appl_statement
                : authentication
                | bcamppl_statement
                ;

rm              : RMXA SBOPEN LETTER COMMA LETTER SBCLOSE
                ;

admin           : ADMIN SBOPEN LETTER COMMA LETTER SBCLOSE
                ;

max             : MAX SBOPEN LETTER COMMA LETTER SBCLOSE
                | max_header max_list_opt SBCLOSE
                ;

max_header      : MAX SBOPEN LETTER EQUAL LETTER
                ;

max_list_opt    : /* empty *?
                | COMMA max_list
                ;

max_list        :               max_element
                | max_list COMMA max_element
                ;

max_element     : LETTER EQUAL LETTER
                ;

start           : START SBOPEN LETTER COMMA LETTER SBCLOSE
                ;

start_rm        : START_RM SBOPEN LETTER COMMA LETTER SBCLOSE
                | START_RM SBOPEN LETTER COMMA APPLICATION
                  SBCLOSE
                ;
```

Druck vom 24.01.2001 17:00.33 torb-kon


```
multi_host     :            hostblock
               | multi_host hostblock
               ;

hostblock      : HOST SBOPEN LETTER SBCLOSE hostbody
                 /* Hostname */
               | HOST SBOPEN LETTER COMMA RESERVE SBCLOSE hostbody
                 /* Hostname, CMX-Version oder UTM-Version */
               | HOST SBOPEN LETTER COMMA LETTER
                 SBCLOSE hostbody
                 /* Hostname, CMX-Version oder UTM-Version */
               | HOST SBOPEN LETTER COMMA LETTER COMMA RESERVE
                 SBCLOSE hostbody
                 /* Hostname, CMX-Version oder UTM-Version */
               | HOST SBOPEN LETTER COMMA LETTER COMMA LETTER
                 SBCLOSE hostbody
                 /* Hostname, CMX-Version, UTM-Version */
               | HOST SBOPEN LETTER COMMA LETTER COMMA LETTER
                 COMMA RESERVE SBCLOSE hostbody
                 /* Hostname, CMX-Version, UTM-Version */
               ;

hostbody       : BBOPEN
                 i_v_a
                 vector_aaddress_opt
                 host_statements
                 multi_applicat_opt
                 BBCLOSE
               ;

i_v_a          : internet internet_rest
               | vector   vector_rest
               | address adress_rest
               ;

address_rest   : internet vector_opt
               | vector internet_opt
               ;

internet_rest  : /* empty */
               | vector address_opt
               | adress vector_opt
               ;
```

```
vector_rest    : internet address_opt
               | address  internet_opt
               ;

internet       : INTERNETADDRESS SBOPEN INADDRESS SBCLOSE
               ;            ;

internet_opt   : /* empty */
               | internet
               ;

vector_opt     : /* empty */
               | vector
               ;

host_statements
               : /* empty */
               | operating_system host_statement_list_opt
               |                  host_statement_list
               ;

host_statement_list_opt
               : /* empty */
               | host_statement_list
               ;

host_statement_list
               : host_statement
               | host_statement_list host_statement
               ;

host_statement : statement
               | dynamic_connect_host
               ;

dynamic_conect_host
               : dynamic_connect opt_dyn_block
               | dynamic_connect_empty
               ;

dynamic_connect_empty
               : DYNAMIC_CONNECT SBOPEN SBCLOSE
               ;
```

Druck vom 24.01.2001 17:00:34 torb-kon


```
dynamic_connect: DYNAMIC_CONNECT SBOPEN dyn_conn_list SBCLOSE
               ;

dyn_conn_list  : /* empty */
               | dyn_conn_list
               ;

dyn_conn_list  :                    dyn_conn_element
               | dyn_conn_list COMMA dyn_conn_element
               ;

dyn_conn_element
               : SBOPEN NUMBER COMMA NUMBER COMMA NUMBER SBCLOSE
               ;

opt_dyn_block  : /* empty */
               | dyn_block
               ;

dyn_block      : BBOPEN authentication BBCLOSE
               ;

multi_applicat_opt
               : /* empty */
               | multi_applicat
               ;

multi_applicat :               applicat
               | multi_applicat applicat
               ;

applicat       : ta_appl           ta_attrib    /* server */
               | non_ta_appl                    /* client */
               | foreign_appl foreign_attrib
               ;

ta_appl        : ta_appl_1
               | ta_appl_2
               ;

ta_appl_1      : ta_appl_best
               | ta_appl_max
               ;
```

```
non_ta_appl    : nonta_appl_remote
               | nonta_appl_1
               ;

ta_appl_best   : TA_APPLICATION
                 SBOPEN
                 LETTER COMMA
                 NUMBER COMMA
                 NUMBER COMMA
                 LETTER COMMA
                 BEST_BCAMAPPL
                 SBCLOSE
               ;

ta_appl_max    : TA_APPLICATION
                 SBOPEN
                 LETTER COMMA
                 NUMBER COMMA
                 NUMBER COMMA
                 LETTER
                 SBCLOSE
               ;

ta_appl_2      : TA_APPLICATION
                 SBOPEN
                 NUMBER COMMA
                 NUMBER COMMA
                 RESERVE
                 SBCLOSE
               ;

nonta_appl_1   : APPLICATION
                 SBOPEN
                 NUMBER COMMA
                 NUMBER COMMA
                 LETTER
                 SBCLOSE
               | APPLICATION
                 SBOPEN
                 NUMBER COMMA
                 NUMBER COMMA
```

Druck vom 24.01.2001 17:00:34 torb-kon


```
                    LETTER COMMA
                    LETTER COMMA /* userid */
                    LETTER       /* passwd */
                    SBCLOSE
                  ;

nonta_appl_remote
                  : APPLICATION
                    SBOPEN
                    NUMBER COMMA
                    NUMBER COMMA
                    LETTER COMMA
                    REMOTE
                    SBCLOSE
                  | APPLICATION
                    SBOPEN
                    NUMBER COMMA
                    NUMBER COMMA
                    LETTER COMMA
                    REMOTE COMMA
                    LETTER COMMA   /* userid */
                    LETTER         /* passwd */
                    SBCLOSE
                  ;

foreign_appl      : FOREIGN_APPLICATION
                    SBOPEN
                    NUMBER COMMA
                    NUMBER COMMA
                    LETTER SBCLOSE
                  | FOREIGN_APPLICATION
                    SBOPEN
                    NUMBER COMMA
                    NUMBER COMMA
                    LETTER COMMA LETTER SBCLOSE
                  | FOREIGN_APPLICATION
                    SBOPEN
                    NUMBER COMMA
                    NUMBER COMMA
                    NUMBER COMMA      /* foreign application number */
                    LETTER SBCLOSE
                  | FOREIGN_APPLICATION
                    SBOPEN
```

```
                  NUMBER COMMA
                  NUMBER COMMA
                  NUMBER COMMA       /* foreign application number */
                  LETTER COMMA LETTER SBCLOSE
                ;

vector          : KEYVECTOR SBOPEN NUMBER COMMA NUMBER
                  SBCLOSE
                ;

address         : PORTADDRESSES SBOPEN NUMBER COMMA
                  NUMBER SBCLOSE
                ;

multi_link_opt : multi_link
               | /* empty */
               ;

multi_link      :            link
                | multi_link link
                ;

link            : sessblock
                | connect
                ;

connect         : CONNECT
                  SBOPEN
                  NUMBER COMMA NUMBER COMMA NUMBER COMMA
                  NUMBER
                  SBCLOSE
                ;

sessblock       : session_header
                  BBOPEN sesspoint sesspoint BBCLOSE
                | foreign_session_header
                  BBOPEN foreign_sesspoint
                         foreign_sesspoint BBCLOSE
                ;

foreign_session_header
                : FOREIGN_SESSION SBOPEN LETTER SBCLOSE
                | FOREIGN_SESSION SBOPEN LETTER COMMA MAP_SYSTEM SBCLOSE
                ;
```

Druk vom 24.01.2001 17:00:34 torb-ken


```
session_header : SESSION SBOPEN LETTER SBCLOSE
               | SESSION SBOPEN LETTER COMMA MAP_SYSTEM SBCLOSE
               ;

foreign_sesspoint
               : foreign_sesspoint_header
                 BBOPEN address_converter BBCLOSE
               | foreign_sesspoint_header
               ;

foreign_sesspoint_header
               : SESSIONPOINT
                 SBOPEN
                 NUMBER COMMA NUMBER COMMA NUMBER
                 SBCLOSE
               ;

address_converter
               : foreign_address converter_opt
               ;

foreign_address
               : ADDRESS SBOPEN LETTER COMMA LETTER SBCLOSE
               ;

converter_opt  : /* empty */
               | converter
               ;

converter      : IN_CONVERTER SBOPEN NUMBER SBCLOSE
               | IN_CONVERTER SBOPEN NUMBER COMMA NUMBER SBCLOSE
               | OUT_CONVERTER SBOPEN NUMBER SBCLOSE
               ;

sesspoint      : SESSIONPOINT
                 SBOPEN
                 NUMBER COMMA NUMBER COMMA NUMBER
                 SBCLOSE
               ;
%%
```

6.3 Revision generation

The purpose of the revision generation is as follows: when the configuration is revised, the generation for the applications not affected by the revision remain the same.

Prerequisites

The use of a repository is an absolute must for this revision generation (see the statement *REPOSITORY* on page 60). The repository must be created when generating the previous version or updated to the new status.

The applications use host resources such as authorization keys (KEYVECTOR) and port numbers (PORTADDRESSES). The applications not affected by a revision must use the same resources. The authorization keys and port numbers are stored in the repository for each application.

When creating the `kdcdf` script, the generator `config` must generate unique identifiers for each application. The same identifiers must then be used in a revision generation. In the case of a first generation with an empty repository, a generation number is written to the repository for each application so that this number will be used to generate the identifiers (see the statement *START_VALUE* on page 64).

The following further characteristics or values are entered in the repository for each server application:

– `KDCFILE, DOUBLE`
  If the `KDCFILE` is maintained in duplicate for security reasons, the characteristic `DOUBLE` is stored and can then be taken into account during the next generation.

– `PGPOOLFS`
  The `PGPOOLFS` parameter defines the number of files across which the page pool is distributed. This number is entered in the repository and can thus be evaluated during the next generation.

– `RECBUFFS`
  `RECBUFFS` defines the number of files across which the restart area is to be distributed. This number is stored in the repository and read during the next generation.

Druck vom 24.01.2001 17:00.35 torb-kon

Performing the revision generation

The user must perform the following steps when revising the configuration so that no user data is lost from an application:

– Adapt the configuration file
  The modifications to the configuration, e.g. new hosts, new applications and connections must be defined in the input file of the configuration generator `config`.

– Perform the generation
  The generator reads the configuration file and generates, among other things, `kdcdf` scripts.

The following steps must be performed on all hosts affected by the modifications:

– Terminate the application normally
  The application must be terminated using the command `kdcshut normal` or `kdcdshut warn` so that the files `KDCA`, `R01A`, `...`, `P01A`, `...`, `KDCB`, `R01B`, `...`, `P01B`, `...` are in a consistent state.

– Execute the `kdcdf` script
  The files `KDCA`, etc. are first copied to the directory `old`.
  The files `KDCA`, etc. are then recreated using the *open*UTM utility routine `kdcdef` [29].
  This *open*UTM utility routine `kdcupd` transfers the data from the old files to the newly generated files.

– Restart the applications
  The `shell` script `utmstart.multi` is one of many which is generated when the `kdcdf` script is executed. This `shell` script must be executed.

6.4 Sample configuration file

This section illustrates the syntactic structure using a sample configuration.

```
SYSTEM
{
    //
    // system-wide MAX statements.
    //
    MAX("TASKS","7")
    MAX("ASYNTASKS","4")
    MAX("PGPOOL","(200,80,96)")
    MAX("RECBUF","(32,4096)")
    MAX("TASKS-IN-PGWT","3")

    RMXA("INFORMIX","C")

    ADMIN("upicadm","valentin")
    ADMIN("admin","4711")

    START_RM("INFORMIX",APPLICATION)

    //
    // first operating system of the system.
    // in this example the hostname is
    // used as symbolic name.
    //
    HOST("kotw002")
    {
        //
        // internetaddress of first host.
        //
        INTERNETADDRESS(192.200.94.8)

        //
        // available shared memory and semaphore keys.
        //
        KEYVECTOR(5011,5047)
```

Druck vom 24.01.2001 17:00:35 torb-kon


```
        //
        // available port addresses.
        //
        PORTADDRESSES(10111,10126)
        //
        // host wide MAX statements.
        //
        MAX("LSSBS","200")
        MAX("TASKS","6")

        //
        // first application of first operating system.
        // a1 is the utm known application name.
        // buslay is the user friendly name.
        // the third parameter is optional for BCAMAPPL
        // optimization.
        //
        TA_APPLICATION("a1",1,1,"buslay")
        {
            //
            // application wide MAX statements
            //
            MAX("TASKS","5")
        }

        TA_APPLICATION("a2",1,2,"servlay")


    HOST("kotw005")
     {
        INTERNETADDRESS(192.200.94.4)
        KEYVECTOR(5000,5015)
        PORTADDRESSES(10114,10135)

        TA_APPLICATION("a1",1,3,"nwlay")

        TA_APPLICATION("a2",1,4,"nellay")

        APPLICATION(1,5,"client1","userid","password")

        APPLICATION(1,6,"client2")

        APPLICATION(1,7,"client3",REMOTE)
    }
```

```
//
// first session of the system between application a2
// of kotw002
// and a1 of kotw005.
// a2 controls 3 connections
// a1 controls 1 connection
//
SESSION("LU6.1")
{
    SESSIONPOINT(1,1, 3)
    SESSIONPOINT(1,2, 1)
}

SESSION("LU6.1")
{
    SESSIONPOINT(1,1, 1)
    SESSIONPOINT(1,3, 2)
}

SESSION("LU6.1")
{
    SESSIONPOINT(1,2, 3)
    SESSIONPOINT(1,3, 2)
}

SESSION("LU6.1")
{
    SESSIONPOINT(1,3, 2)
    SESSIONPOINT(1,4, 4)
}

//
// connection between client1 of kotw005 and a1
// of kotw005
// client is configured local
//
CONNECT(1,5, 1,3)
```

Druk vom 24.01.2001 17:00:35 torb-ken


```
        //
        // client2 of kotw005 is configured remote because
        // of remote server 1,1
        //
        CONNECT(1,1, 1,6)
        CONNECT(1,6, 1,3)

        //
        // client3 of kotw005 is configured remote because
        // of REMOTE statement in the client3 definition
        //
        CONNECT(1,7, 1,3)
}
```

6.5 Call and options

The configuration generator `config` is called as follows:

`config [-a] [-d] [-k nohinfo] [-m] [-r] [-u] [-v] [-V] configfile`

`-a` The `all` option contains the `-d` and `-r` options that are described below. If no option is set in the call, `-a` is assumed, i.e. all of the files are generated.

`-d` If the `development` option is set, the configuration generator creates files required in the application development process (see below for a description of the individual files).

`-k nohinfo`
The `-k nohinfo` option suppresses output of the time of generation in the generated files.

`-m` If the `mainframe` option is set, the configuration generator creates identifiers in the file `kdcdf` made up solely of uppercase letters and digits. This option must be specified if the configuration description includes BS2000/OSD hosts.

`-r` If the `runtime` option is set, the configuration generator creates files required to run a distributed application (see below for a description of the individual files).

`-u` The `-u` option outputs the call syntax.

`-v` With the `verbose` option, generator messages are output to `stdout`.

`-V` If `-V` is specified, only the version message is output to `stdout` and execution of the program is terminated.

For *`configfile`*, you must specify the name of your configuration file.

Druck vom 24.01.2001 17:00:36 torb-kon

6.6 Generated files

The configuration generator `config` analyzes a description file and generates from it resource files and configuration scripts for the runtime environment of the distributed system. Some of the generated files are host-specific and some are application-specific. `config` creates a directory tree in the current directory. The basic structure of this tree is illustrated in Figure 2 below.

There is a subdirectory for each host under the root `system`. Each of these host subdirectories bears as its name the descriptive name from the `HOST` statement. All of these subdirectories also have their own subdirectories bearing the names of the applications which are to run on the respective hosts. No directory will be created for a host marked with the `RESERVE` flag. In the same way no directory will be created for a `TA_APPLICATION` labeled with the `RESERVE` flag.

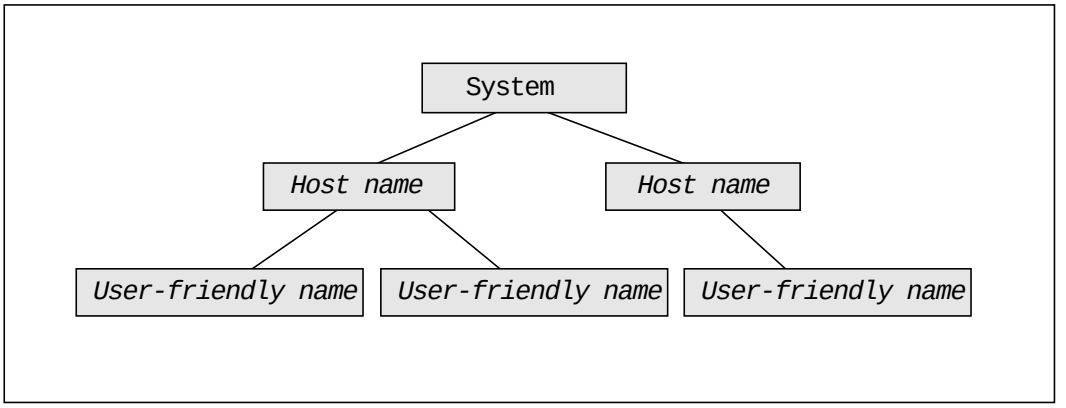


Figure 2 Model of the directory structure as created by `config`

Most of the files created by `config` are incorporated directly into the runtime environment of the distributed application. In addition, `config` generates a script for each T-ORB application which calls the configuration tools of the underlying transaction monitor and in the process creates files which are needed to develop or run the distributed system.

The script is generated in accordance with the operating system of the host on which the relevant T-ORB application is to run and must be executed under this operating system.

– A C shell script with the file name `kdcdf` is generated for a UNIX host, i.e. the shell `/bin/csh` must exist on the relevant host.

– A batch processing file `kdcdf.bat` is generated for the WindowsNT operating system.

– An SDF command procedure `KDCDF` is generated for BS2000/OSD.

The call options specified for the configuration generator result in different variants of the script being generated. These different variants generate different files:

– If `kdcdf` was generated using the `config -d` development option, it generates source files for program parts which are needed to link the T-ORB application to the transaction monitor.

– If `kdcdf` was generated using the `config -r` development option, it generates configuration and runtime files for the transaction monitor, as well as some utility procedures.

– If `kdcdf` was generated using the “all” option (`config -a`), it generates the same files as the development and runtime variants combined.

In the case of an external *open*UTM application, the `kdcdf` file is not generated as a script, but rather as a file containing *open*UTM configuration statements. It must be included in the configuration process of the external *open*UTM application. ❍❍●

Druck vom 24.01.2001 17:00.36 torb-kon

6.6.1 Generated files for UNIX hosts

6.6.1.1 Development option

The development variant of the C shell script `kdcdf` generates the following C source files:

- `GinaRoot.c`
- `OwnMsgs.c`

This files must be compiled using a C compiler and linked to the T-ORB application (see the GINA Developer Manual [13], chapter *Compiling and linking*).

The name of the Resource Manager manufacturer from the `RMXA` statement is incorporated into the `GinaRoot` source. If this statement is modified, `GinaRoot` must be regenerated and compiled and the application must be linked once more. ❍❍●

6.6.1.2 Runtime option

TNSX configuration

Two text files, `tnsxin` and `tnsxdel`, are created for each host. These files simplify the process of configuring the Transport Name Service TNSX. The file `tnsxin` contains all of the information required to make the TNSX entries for setting up the connections with other hosts. It is processed using the command `tnsxcom -S tnsxin`. In the event of a deinstallation or reconfiguration, these TNSX entries can be deleted again using `tnsxcom -S tnsxdel`.

`upicfile`

A file `upicfile` is generated for each T-ORB/client application. This file acts as a kind of server directory for addressable servers. For a static T-ORB/client (`APPLICATION`) this file is created in the directory which is assigned to the application; for dynamic T-ORB/clients (`DYNAMIC_CONNECT`) the file is created in the corresponding `HOST` directory.

`gina.config`

A file `gina.config` is generated for each T-ORB application (`TA_APPLICATION`), static T-ORB/client application (`APPLICATION`) and for dynamic T-ORB clients (`DYNAMIC_CONNECT`). It acts as a directory for addressable applications. Among other things, the file also contains intervals for an application-specific timer and for controlling events.

For T-ORB applications and static T-ORB clients, this file is generated in the directory assigned to the application; for dynamic clients the file is created in the directory of the host on which the dynamic clients are configured.

The gina.config file must normally be copied to the directory where the application is called. You can, however, choose a directory other than "." using the environment variable GINACONFIG.

gina.dynamic

A file gina.dynamic is generated for each host for which dynamic T-ORB clients are configured (statement DYNAMIC_CONNECT). This file contains information on the dynamic connections and is used by DomsDynConnectHandler (see *Dynamic Connection Handler* on page 197).

Configuration data for the transaction monitor

If the kdcdf script was created with the runtime option config -r, it generates a file KDCA and possibly other elements of KDCFILE. This data is configuration data for *open*UTM. The kdcdf script must be called in the directory where the KDCA file will reside when the application is running. MAX statements in the configuration description can be used to influence the files other than KDCA which are generated [26]. If the file KDCA already exists when kdcf is called, it is backed up under the name old/KDCA (UNIX).

Start and administration scripts

The runtime variant of the kdcdf script which was created using config -r generates procedures for starting and administering a T-ORB application when called. These procedures are generated as scripts for the C shell, i.e. the shell /bin/csh must be installed on the relevant host.

- utmstart.multi
  Start procedure for the application
- utmstart.single
  Start procedure for the debug variant of the application
- start
  Start parameters for *open*UTM
- dtp
  Procedure for *open*UTM administration

Druck vom 24.01.2001 17:00:36 torb-kon

6.6.2 Generated files for WindowsNT hosts

6.6.2.1 Development option

The development variant of the batch processing file `kdcdf.bat` generates the following C source files:

- `GinaRoot.c`
- `OwnMsgs.c`

This files must be compiled using a C compiler and linked to the T-ORB application (see the GINA Developer Manual [13], chapter *Compiling and linking*).

The name of the Resource Manager manufacturer from the `RMXA` statement is incorporated into the `GinaRoot` source. If this statement is modified, `GinaRoot` must be regenerated and the application must be linked once more. ❍❍●

6.6.2.2 Runtime option

TNSX configuration

PCMX Transport Name Service Source Files with the names `tnsxin.tns` and `tnsxdel.tns` are created for each host. These files simplify the process of configuring the Transport Name Service TNSX. The file `tnsxin.tns` contains all of the information required to make the TNSX entries for setting up the connections with other hosts. The data is entered when the file is executed. In the event of a deinstallation or reconfiguration, these TNSX entries can be deleted again using `tnsxdel.tns`.

`upicfile`

A file `upicfile` is generated for each T-ORB/client application. This file acts as a kind of server directory for addressable servers. For a static T-ORB/client (`APPLICATION`) this file is created in the directory which is assigned to the application; for dynamic T-ORB/clients (`DYNAMIC_CONNECT`) the file is created in the corresponding `HOST` directory.

`gina.config`

A file `gina.config` is generated for each T-ORB application (`TA_APPLICATION`), static T-ORB/client application (`APPLICATION`) and for dynamic T-ORB clients (`DYNAMIC_CONNECT`). It acts as a directory for addressable applications. Among other things, the file also contains intervals for an application-specific timer and for controlling events.

For T-ORB applications and static T-ORB clients, this file is generated in the directory assigned to the application; for dynamic clients the file is created in the directory of the host on which the dynamic clients are configured.

The `gina.config` file must normally be copied to the directory where the application is called. You can, however, choose a directory other than "`.`" using the environment variable `GINACONFIG`.

`gina.dynamic`

A file `gina.dynamic` is generated for each host for which dynamic T-ORB clients are configured (statement `DYNAMIC_CONNECT`). This file contains information on the dynamic connections and is used by `DomsDynConnectHandler` (see *Dynamic Connection Handler* on page 197).

Configuration data for the transaction monitor

If the `kdcdf.bat` script was created with the runtime option `config -r`, it generates a file `KDCA` and possibly other elements of `KDCFILE`. This data is configuration data for *open*UTM. The `kdcdf.bat` script must be called in the directory where the `KDCA` file will reside when the application is running. Before the call, ensure that the `%PATH%` variable contains `<install-dir>\bin` (where `<install-dir>` is the GINA installation directory). `MAX` statements in the configuration description can be used to influence the files other than `KDCA` which are generated [26]. If the file `KDCA` already exists when `kdcf.bat` is called, it is backed up under the name `old\KDCA` (UNIX).

Start and administration scripts

The runtime variant `of kdcdf.bat` (created using `config -r`) generates scripts for starting and administering a T-ORB application when called.

– `utmstart.multi.bat`
  Start procedure for the application
– `utmstart.single.bat`
  Start procedure for the debug variant of the application
– `start`
  Start parameters for *open*UTM
– `dtp.bat`
  Procedure for *open*UTM administration

Druck vom 24.01.2001 17:00.37 torb-kon

6.6.3 Generated files for BS2000/OSD hosts

6.6.3.1 Development option

The development variant of the command procedure KDCDF generates assembler source files for the following program parts:

- GinaRoot
- OwnMsgs

The script KDCDF automatically calls a file assembly routine and writes the objects created to an LMS library as the LLM elements GINART and OWNMSGS. The LMS library has the name *TP_application_name*.LIB. From there they must be linked to the T-ORB application. A different library name to the TP application name can be selected via a call parameter of the KDCDF script.

The name of the Resource Manager manufacturer from the RMXA statement is incorporated into the GinaRoot source. If this statement is modified, GinaRoot must be regenerated and the application must be linked once more. ❍❍●

6.6.3.2 Runtime option

gina.config

A file GINA.CONFIG.*TP_application_name* is generated for each T-ORB application (TA_APPLICATION) and static T-ORB/client application (APPLICATION). It acts as a directory for addressable applications. Among other things, the file also contains intervals for an application-specific timer and for controlling events.

Configuration data for the transaction monitor

If the kdcdf script was created with the runtime option config -r, it generates a file KDCA and possibly other elements of KDCFILE. This data is configuration data for *open*UTM. The kdcdf script must be called in the directory where the KDCA file will reside when the application is running. MAX statements in the configuration description can be used to influence which files other than KDCA are generated [27].

KDCA must also exist as a DMS file under BS2000/OSD. To avoid naming conflicts, the file name is prefixed with the string GINA and the TP application name from the configuration description, i.e. GINA.*TP_application_name.*KDCA.

If the file KDCA already exists when kdcf is called, it is backed up under the name old/KDCA (UNIX) or old\KDCA (WindowsNT) or OLD.GINA.*TP_application_name*.KDCA (BS2000/OSD).

Start and administration scripts

The runtime variant of the procedure `KDCDF` (created using `config -r`) creates an ENTER file with the name `START.`*`TP_application_name`* when called. This file is used to start a T-ORB application. It must be copied as a DMS file to the host on which the application is to run.

6.6.4 Example

The file structure illustrated in Figure 3 is an example of the structure created for the sample configuration file on page 90.

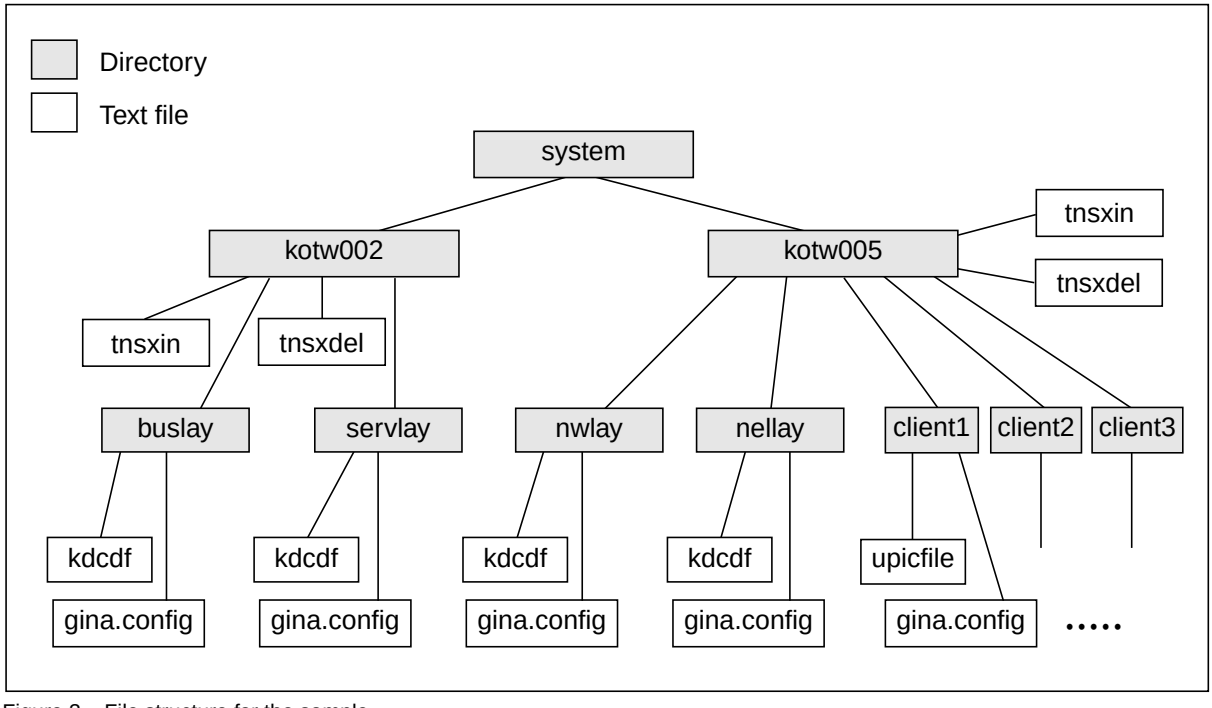


Figure 3 File structure for the sample

Druck vom 24.01.2001 17:00:38 torb-kon

6.7 Creating a configuration file using WinConfig

The configuration file described in the preceding sections contains all the necessary information on the desired system structure and can be created either with an editor or using the WinConfig graphical user interface.

The WinConfig graphical user interface is only available on UNIX platforms and only for *open*UTM. ❍❍●

WinConfig contains a direct connection to the GINA configuration generator config. This means that the configuration process can be executed almost entirely within the graphical user interface. The user controls the creation of a configuration file and the calling of the configuration generator by using WinConfig.

The individual configuration elements (HOST, TA-APPLICATION, APPLICATION, FOREIGN_APPLICATION, SESSION, CONNECTION and FOREIGN_SESSION) of the configuration language from section 6.2 on page 46 are represented in WinConfig by icons and lines. Clicking on the icons or lines opens dialog windows in which you can enter or change the relevant parameters.

The next sections frequently use the terms “TA application”, “non-TA application”, “foreign application”, “session”, “connection” and “foreign session”. In the context described, they mean the following:

TA application	Transaction-monitored application that is connected via T-ORB. In the configuration language, TA applications are represented by the TA_APPLICATION statement.
Non-TA application	Transaction-free application (e. g. GUI clients) that is connected via T-ORB/Client. In the configuration language, non-TA applications are represented by the APPLICATION statement.
Foreign application	A foreign *open*UTM application. In the configuration language, foreign applications are represented by the FOREIGN_APPLICATION statement.
Session	Connection between two transaction-monitored applications. In the configuration language, sessions are represented by the SESSION statement.
Connection	Connection between a transaction-monitored application and a transaction-free application. In the configuration language, connections are represented by the CONNECTION statement.
Foreign session	Connection between a transaction-monitored application and a foreign *open*UTM application. In the configuration language, foreign sessions are represented by the FOREIGN_SESSION statement.

The configuration file created using WinConfig is the input for the GINA configuration generator config. The configuration generator creates the files required for configuring the entire system based on the information provided in the configuration file (see section 6.5 on page 94).

This chapter explains the steps for working with the WinConfig graphical user interface. The description requires knowledge of the configuration language specified in section 6.2 on page 46.

6.7.1 Calling WinConfig

The WinConfig graphical user interface is called as follows:

WinConfig [-V] | [*configfile*]

The interface can be started with or without a configuration file. If a configuration file is specified in the call, WinConfig tries to load this file immediately after it is started.

If the -V option is specified, only the version message is output to stdout and execution of the program is terminated.

No environment variables or X resources must be set. However, the font, if required, can be specified via the X resource WinConfig*fontList.

Druck vom 24.01.2001 17:00:38 torb-ken

6.7.2 Elements of the graphical user interface

After startup, the WinConfig main window is displayed (see Figure 4):

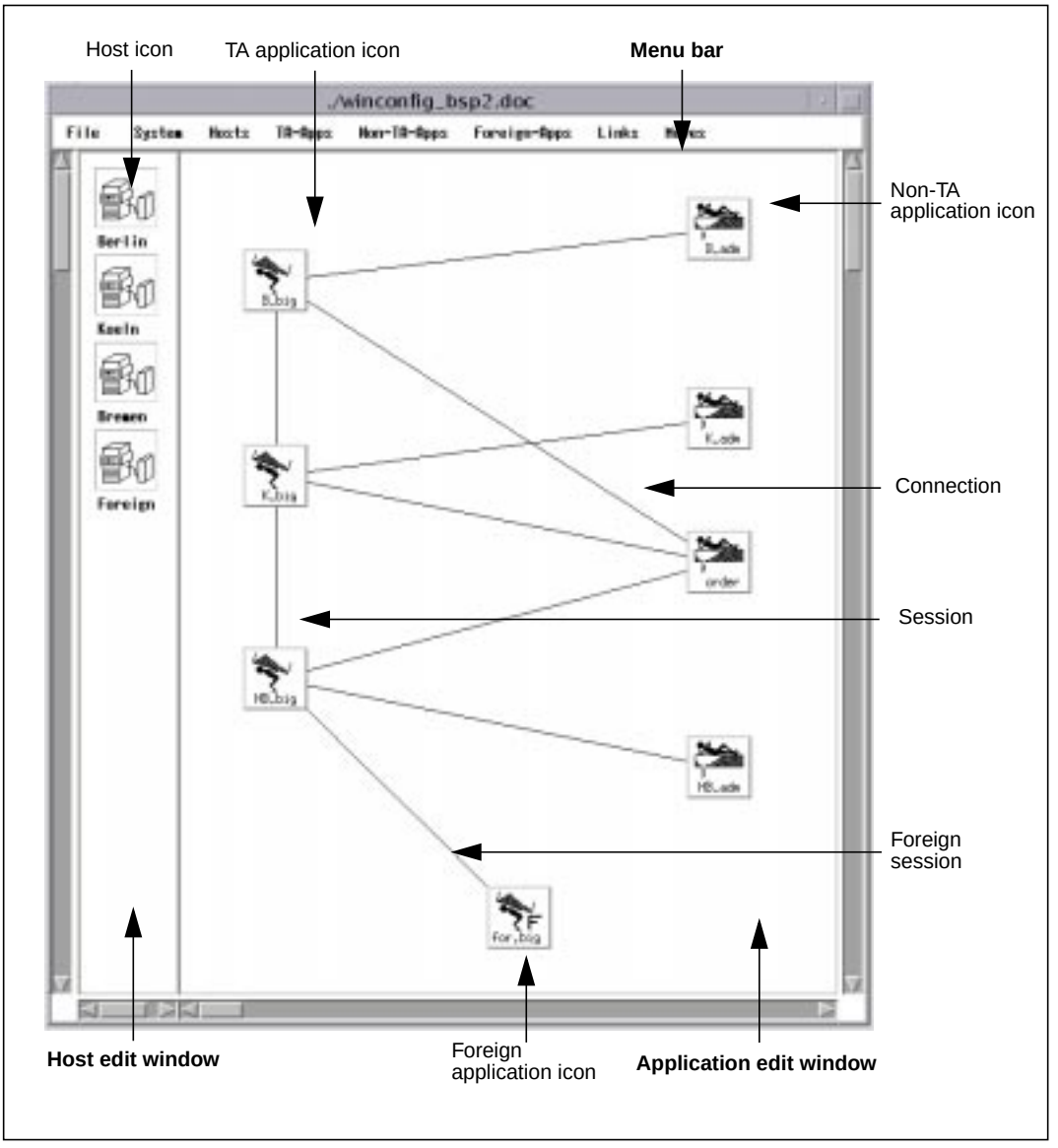


Figure 4 WinConfig: Main window (with a sample configuration)

The WinConfig main window consists of two edit windows with the associated scroll bars and a menu bar.

Host edit window — For information on editing the hosts in a configuration (physical view), see section 6.7.2.1 on page 110.

Application edit window — For information on editing the applications (TA applications, non-TA applications and foreign applications), as well as their sessions, connections, and foreign sessions (logical view), see section 6.7.2.2 on page 112.

Menu bar — Contains the File menu with general file operations, the System, Hosts, TA-Apps, Non-TA-Apps, Foreign-Apps menus for modifying customizing settings, the Links menu for the multiple generation and multiple deletion of sessions, connections and foreign sessions, as well as the Moves menu for changing the graphical layout; see section 6.7.2.3 on page 120.

If the Autoscroll option is activated in the pop-up menu of the right mouse button, WinConfig scrolls in the application edit window automatically if the cursor goes beyond the edge of this area. In the default case, scroll bars are used for scrolling.

The configuration example illustrated in Figure 4 on page 105 contains the four hosts Berlin, Koeln (= Cologne), Bremen and Foreign. The following applications are running on the relevant hosts:

Berlin — Transaction-monitored application B.big, non-transaction-monitored application B.adm.

Koeln — Transaction-monitored application K.big, non-transaction-monitored applications K.adm and order.

Bremen — Transaction-monitored application HB.big, non-transaction-monitored application HB.adm.

Foreign — The foreign *open*UTM application for.big.

Druck vom 24.01.2001 17:00:39 torb-kon

There are a total of six connections, two sessions, and a foreign session in this configuration example:

Connections
: The non-transaction-monitored applications `B.adm`, `K.adm` and `HB.adm` are connected locally with the transaction-monitored applications `B.big`, `K.big` or `HB.big` respectively.
The non-transaction-monitored application `order` is connected with each of the three transaction-monitored applications.

Sessions
: There are connections between the transaction-monitored applications `B.big` and `K.big`, as well as between `K.big` and `HB.big`.

Foreign session
: Connection between the transaction-monitored application `HB.big` and the foreign *open*UTM application `for.big`.

The parameters of hosts, TA applications, non-TA applications and foreign applications are edited using dialog boxes. Clicking on a host icon or application icon opens the relevant dialog box.

Figure 5 on page 108 shows the dialog boxes for the host `Berlin`, for the non-TA application `order`, for the TA application `HB.big`, and for the foreign application `for.big`.

The parameters of sessions, connections and foreign sessions can also be edited. Clicking on a connecting line opens appropriate dialog boxes; see the dialog boxes for `HB.big - K.big` and `HB.big - order` in Figure 5 on page 108.

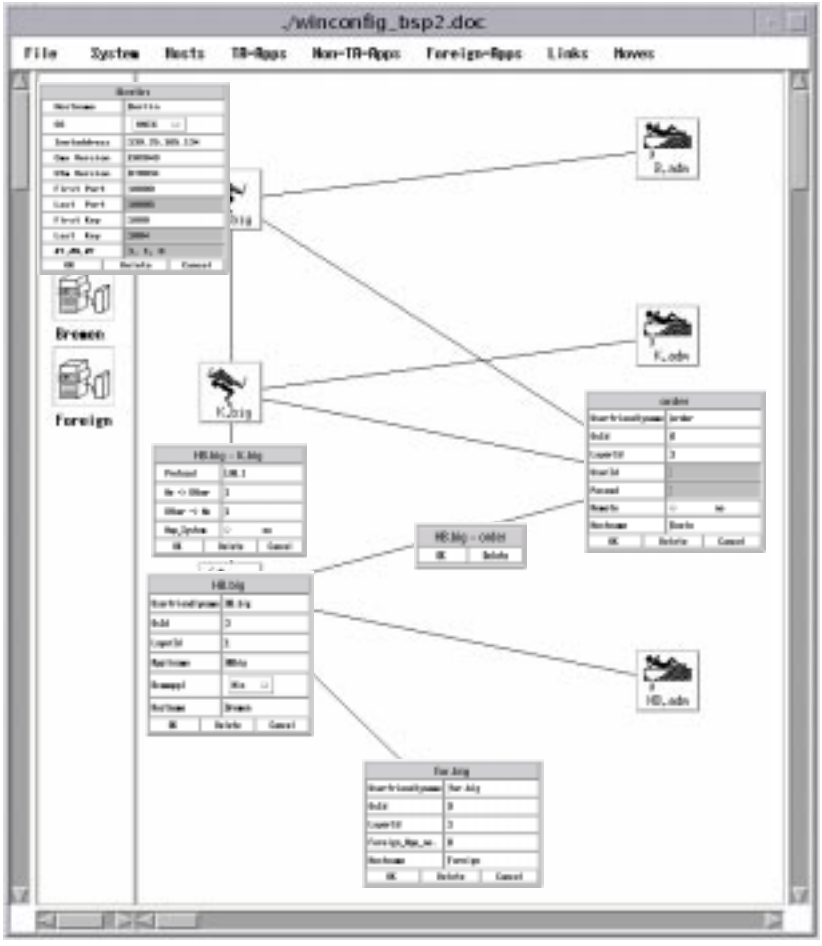


Figure 5 WinConfig: main window with open dialog boxes

With few exceptions, all WinConfig dialog windows are based on the same model: the left-hand column in the window shows the parameter names, and the right-hand column shows the input fields with the relevant values.

Parameter input is terminated by clicking with the left mouse button on OK, Delete or Cancel. The dialog window is closed.

Druck vom 24.01.2001 17:00:40 torb-kon

Figure 6 shows the WinConfig main window with a graphical representation of the configuration file from section 6.4 on page 90. The individual application icons were positioned manually, see also the File>Open menu item on page 121.

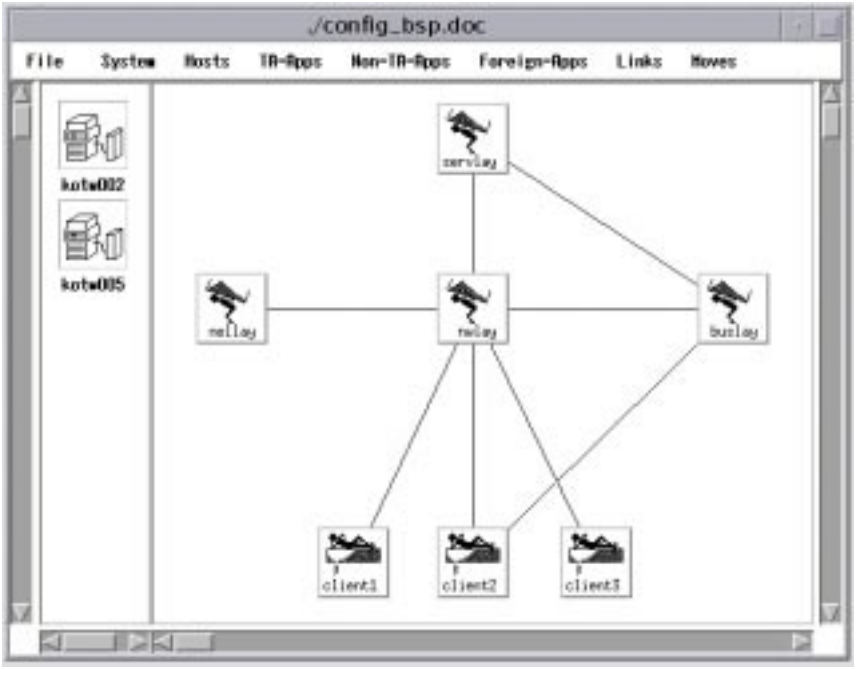


Figure 6 WinConfig: main window with the loaded configuration file from section 6.4

The next sections describe the individual elements of the graphical user interface and the corresponding dialog windows in detail.

6.7.2.1 Host edit window

Each host in a configuration is represented in the host edit window by a host icon. A label underneath the icon shows the name of the host.

Each new host is generated using the pop-up menu of the right mouse button. However, newly generated hosts do not yet have a label underneath the icon showing the host's name. WinConfig arranges all host icons one below the other in the host edit window.

The host parameters can be edited by clicking on the host icon with the left mouse button. The following dialog window is opened via the host icon:

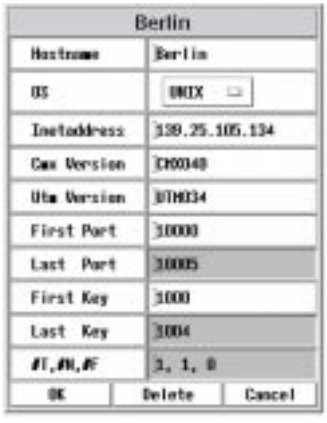


Figure 7 Dialog window: Host parameters

The Last Port, Last Key and #T, #N, #F input fields cannot be edited. WinConfig calculates their values automatically.

The following table describes the dialog window entries; the "Statement" column specifies the corresponding statement that is generated by WinConfig when saving to a configuration file.

Parameter name	Value	Default	Statement
Hostname	Name of the host	none	HOST
OS	Name of the operating system, possible values: UNIX, WINNT, BS2000 Selected via a pull-down menu	-	OPERATING_SYSTEM
Inetaddress	Internet address of the host	none	INTERNETADDRESS
Cmx Version	CMX version of the host	CMX040	HOST
Utm Version	UTM version of the host	UTM040	HOST

Druck vom 24.01.2001 17:00:41 torb-kon

Parameter name	Value	Default	Statement
First Port	First port number for the TNSX entries	10000	PORTADDRESSES
Last Port	Last port number for the TNSX entries This parameter value is calculated automatically by WinConfig. The calculation is explained in section 6.2 on page 46. WinConfig only uses the new value if it is larger than the old value.	10000	PORTADDRESSES
First Key	Start key for the KEYVECTOR statement (shared memory and semaphore key)	1000	KEYVECTOR
Last Key	End key for the KEYVECTOR statement This parameter value is calculated automatically by WinConfig. The calculation is explained in section 6.2 on page 46. WinConfig only uses the new value if it is larger than the old value.	1000	KEYVECTOR
#T, #N, #F	Number of TA, non-TA and foreign applications running on the host This parameter value is generated automatically by WinConfig.	0,0,0	none

Hosts with the BS2000 operating system do not require any port addresses or KEYVECTOR statements. For this reason, the entries First Port, Last Port, First Key and Last Key in the dialog window cannot be edited for BS2000 hosts.

The buttons in the dialog window execute the following actions:

OK — The entered parameter values are confirmed.

Delete — The corresponding host is deleted, i.e. the host icon in the host edit window disappears.

Cancel — The entered parameter values are discarded again.

An application (represented by an icon in the application edit window) that is running on a host and whose dialog window is open is displayed by WinConfig with a frame around it. In addition, changing the host name in the host dialog window automatically also changes the host name of all of the applications on this host.

6.7.2.2 Application edit window

a) Editing non-TA application parameters

Each non-TA application in a configuration is represented in the application edit window by an icon. The name of the non-TA application is displayed within the icon. The position of an icon within the edit window can be changed by clicking once on the relevant icon using the middle mouse button. The icon tracks the movement of the mouse until you once again press the middle mouse button.

Each new non-TA application is generated using the pop-up menu of the right mouse button. The new non-TA application is represented by an icon that is positioned at the current mouse position within the application edit window. Newly generated non-TA applications do not yet have a name within the icon.

The parameters can be edited by double-clicking on the non-TA application icon with the left mouse button. The following dialog window is opened via the icon:

Figure 8 Dialog window: Non-TA application parameters

The UserId and Passwd input fields have a different background color to the other input fields when the window is first called. This means that these values cannot initially be changed. The values can only be changed after calling the Non-TA-Apps>Admin... menu.

The following table describes the dialog window entries; the “Statement” column specifies the corresponding statement that is generated by WinConfig when saving to a configuration file.

Druck vom 24.01.2001 17:00.42 torb-kon

Parameter name	Value	Default	Statement
Userfriendly-name	User-friendly name of the non-TA application	none	APPLICATION
OsId	Operating system ID of the non-TA application	is generated by WinConfig	APPLICATION
LayerId	Layer ID of the non-TA application	is generated by WinConfig	APPLICATION
UserId	User ID	none	APPLICATION
Passwd	Password	none	APPLICATION
Remote	Possible values: remote / not remote see page 47 Selected via a toggle button	not remote	APPLICATION
Hostname	Name of the host on which the non-TA application is running	none	HOST

The simplest way of entering the host name is to perform a drag-and-drop operation on the host label:

◊ Click on the label of a host icon with the middle mouse key and drag the label into the Hostname input field by holding down the mouse key.

The buttons in the dialog window execute the following actions:

OK — The entered parameter values are confirmed.

Delete — The corresponding non-TA application is deleted, i.e. the icon in the application edit window disappears.

Cancel — The entered parameter values are discarded again.

b) Editing TA application parameters

Each TA application in a configuration is represented in the application edit window by an icon. The name of the TA application is displayed within the icon. The position of an icon within the application edit window can be changed by clicking once on the relevant icon with the middle mouse button. The icon tracks the movement of the mouse until you once again press the middle mouse button.

Each new TA application is generated using the pop-up menu of the right mouse button. The new TA application is represented by an icon that is positioned at the current mouse position within the application edit window. Newly generated TA applications do not yet have a name within the icon.

The parameters can be edited by clicking on the TA application icon with the left mouse button. The following dialog window is opened via the icon:

Figure 9 Dialog window: TA application parameters

The following table describes the dialog window entries; the “Statement” column specifies the corresponding statement that is generated by WinConfig when saving to a configuration file.

Parameter name	Value	Default	Statement
Userfriend-lyname	User-friendly name of the TA application	none	APPLICATION
OsId	Operating system ID of the TA application	is generated by WinConfig	APPLICATION
LayerId	Layer ID of the TA application	is generated by WinConfig	APPLICATION
Appliname	Name of the TA application	none	APPLICATION
Bcamappl	Possible values: Min, Best, Thread Selected via a pull-down menu	Min	APPLICATION
Hostname	Name of the host on which the TA application is running	none	HOST

Druck vom 24.01.2001 17:00.43 torb-kon

The simplest way of entering the host name is to perform a drag-and-drop operation on the host label:

◊ Click on the label of a host icon with the middle mouse button and drag the label into the Hostname input field by holding down the mouse key.

The buttons in the dialog window execute the following actions:

OK	The entered parameter values are confirmed.
Delete	The corresponding TA application is deleted, i.e. the icon in the application edit window disappears.
Cancel	The entered parameter values are discarded again.

c) Editing foreign application parameters

Each foreign application in a configuration is represented in the application edit window by an icon. The name of the foreign application is displayed within the icon. The position of an icon within the application edit window can be changed by clicking once on the relevant icon with the middle mouse button. The icon tracks the movement of the mouse until you once again press the middle mouse button.

Each new foreign application is generated using the pop-up menu of the right mouse button. The new foreign application is represented by an icon that is displayed at the current mouse position within the application edit window. Newly generated foreign applications do not yet have a name within the icon.

The parameters can be edited by clicking on the foreign application icon with the left mouse button. The following dialog window is opened via the icon:

Figure 10 Dialog window: Foreign application parameters

The following table describes the dialog window entries; the “Statement” column specifies the corresponding statement that is generated by WinConfig when saving to a configuration file.

Parameter name	Value	Default	Statement
Userfriendly-name	User-friendly name of the foreign application	none	FOREIGN_ APPLICATION
OsId	Operating system ID of the foreign application	is generated by WinConfig	FOREIGN_ APPLICATION
LayerId	Layer ID of the foreign application	is generated by WinConfig	FOREIGN_ APPLICATION
Foreign_App_no	Foreign application number If 0 is entered, WinConfig does not create any corresponding parameter in the FOREIGN-APPLICATION statement.	0	FOREIGN_ APPLICATION
Hostname	Name of the host on which the foreign application is running	none	HOST

The simplest way of entering the host name is to perform a drag-and-drop operation on the host label:

◊ Click on the label of a host icon with the middle mouse button and drag the label into the Hostname input field by holding down the mouse key.

The buttons in the dialog window execute the following actions:

OK — The entered parameter values are confirmed.

Delete — The corresponding foreign application is deleted, i.e. the icon in the application edit window disappears.

Cancel — The entered parameter values are discarded again.

Druck vom 24.01.2001 17:00.43 torb-ken

d) Editing sessions

A session between two TA applications is represented graphically by a connecting line between the two icons. A session between two TA applications is generated by clicking once on the two relevant icons with the left mouse button.

The session parameters can be edited by clicking on the connecting line with the left mouse button. A dialog window that is positioned on the first icon that was clicked on is displayed. This TA application is then the “Me application”. The other TA application represents the “Other application”.

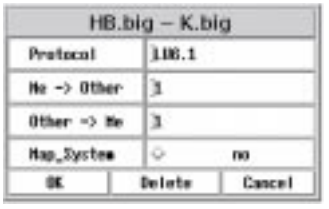


Figure 11 Dialog window: Session parameters

The following table describes the dialog window entries; the “Statement” column specifies the corresponding statement that is generated by WinConfig when saving to a configuration file

Parameter name	Value	Default	Statement
Protocol	Name of the protocol used	LU6.1	SESSION
Me -> Other	Number of connections controlled by the Me application	1	SESSION-POINT
Other -> Me	Number of connections controlled by the Other application	1	SESSION-POINT
Map_System	Enable/disable mapping, see section *SESSION* on page 62. Possible values: yes / no Selected via a toggle button	no	SESSION

The buttons in the dialog window execute the following actions:

OK The entered parameter values are confirmed.

Delete The corresponding session is deleted, i.e. the line between the two icons in the application edit window disappears.

Cancel The entered parameter values are discarded again.

e) Editing connections

A connection between a TA application and a non-TA application is represented graphically by a connecting line between the two icons. A connection between a TA application and a non-TA application is generated by clicking once on the two relevant icons with the left mouse button.

A connection can be editing by clicking on the connecting line with the left mouse button. A dialog window that is positioned on the first icon that was clicked on is displayed.

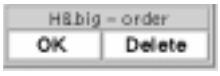


Figure 12 Dialog window: Connection

A connection itself has no parameter values. You can only confirm it in the dialog window with OK or delete it by pressing Delete.

f) Editing foreign sessions

A foreign session between a foreign application and a TA application is represented graphically by a connecting line between the two icons. A foreign session between a foreign session and a TA application is generated by clicking once on the two relevant icons with the left mouse button.

The foreign session parameters can be edited by clicking on the connecting line with the left mouse button. A dialog window that is positioned on the first icon that was clicked on is displayed. This application is then the “Me application”. The other application represents the “Other application”.

Figure 13 Dialog window: Foreign session parameters

Druck vom 24.01.2001 17:00.44 torb-kon

The following table describes the dialog window entries; the “Statement” column specifies the corresponding statement that is generated by WinConfig when saving to a configuration file.

Parameter name	Value	Default	Statement
Protocol	Name of the protocol used	LU6.1	FOREIGN_SESSION
Me -> Other	Number of connections controlled by the Me application	1	SESSION-POINT
Other -> Me	Number of connections controlled by the Other application	1	SESSION-POINT
AddressMe	Specifies how the other foreign or TA application is to be addressed (see section 6.2 on page 46). The two parameter values must be separated by a dot.	none	ADDRESS
AddressOther	Specifies how the other foreign or TA application is to be addressed (see section 6.2 on page 46). The two parameter values must be separated by a dot.	none	ADDRESS
In_Converter	Function ID of the converter function or Class ID and ClassMethod ID of the converter class method (see section 6.2 on page 46). When entering a Class ID and a ClassMethod ID, they must be separated by a dot.	none	IN_CONVERTER
Out_Converter	Converter ID of the converter function (see section 6.2 on page 46). If you enter 0, WinConfig does not create an OUT_CONVERTER statement.	0	OUT_CONVERTER
Map_System	Enable/disable mapping (see section *FOREIGN_SESSION* on page 54) Possible values: yes / no Selected via a toggle button	no	FOREIGN_SESSION

The buttons in the dialog window execute the following actions:

OK	The entered parameter values are confirmed.
Delete	The corresponding foreign session is deleted, i.e. the line between the two icons in the application edit window disappears.
Cancel	The entered parameter values are discarded again.

6.7.2.3 WinConfig menu bar

The menu bar contains the File, System, Hosts, TA-Apps, Non-TA-Apps, Links and Moves menus. The functionality of the menus can be roughly divided into four function areas:

File

General WinConfig operations
e. g. terminating an application, loading/saving a configuration file, starting the GINA configuration generator config.

System, Hosts, TA-Apps, Non-TA-Apps, Foreign-Apps

Modification of customizing settings for the various hierarchies (system-host-application).

The customizing information specified for the system is passed on to the hosts as default values. In addition, all applications on a host inherit that host's customizing information. Inherited customizing information can be overwritten using the customizing menus. WinConfig only shows the modified customizing information for the relevant hosts or applications. The consistency of the customizing information is not completely checked in Version 2.0 of WinConfig.

All customizing statements mentioned in this section are described in more detail in section 6.2 on page 46.

Links

Multiple generation or multiple deletion of sessions, connections and foreign sessions as well as the hiding and showing of specific sessions, connections and foreign sessions.

Moves

Functionality for changing the graphical layout of the configuration currently being displayed.

Druck vom 24.01.2001 17:00:46 torb-kon

File menu

New

All customizing settings are assigned defaults which means that parameter input for the hosts, non-TA applications, TA applications, foreign applications, sessions, connections and foreign sessions is sufficient to configure a system. The default values can be queried and changed via the System menu item in the individual customizing menus.

Open...

Loads a configuration file. A selection box for selecting the file name is displayed. Configuration files that were not created using WinConfig can also be loaded if they are syntactically correct. In this case, the position information for the application icon is missing and all application icons are placed one on top of the other in the upper left corner of the application edit window. The application icons must be dragged to the required position using the mouse.

Once the configuration file is loaded, the path name of the file is displayed in the title line of the WinConfig main window.

Save

Saves the configuration file currently being edited. The name of the file is displayed in the title line of the WinConfig main window.

If WinConfig was started without parameters or the File>New menu item was called, WinConfig: New is displayed in the title line of the main window. In this case, calling Save has the same effect as calling SaveAs....

SaveAs...

Saves the configuration description currently being edited to a file. A selection box for entering the file name is displayed.

Config

This menu item starts the GINA configuration generator config with the configuration file displayed in the title line of the main window. An exact description of the config call options can be found in section 6.5 on page 94.

The Config menu item contains four submenu items:

Config>Check Syntax

The configuration file is checked for completeness.
A configuration file is complete if all of the parameter values for hosts, TA applications, non-TA applications, foreign applications, sessions and foreign sessions are entered. The GINA configuration generator config only accepts complete configuration files as input.

Config>Generate all

Calls the GINA configuration generator config with the option -a.

Config>Generate runtime

Calls the GINA configuration generator config with the option -r.

Config>Generate development

Calls the GINA configuration generator config with the option -d.
If the configuration includes BS2000 hosts, config (when calling the menu items Config>Generate all, Config>Generate runtime, Config>Generate development) is also called with the option -m.

Quit

Terminates WinConfig.

Druck vom 24.01.2001 17:00:46 torb-kon

System menu

This menu can be used to modify the customizing settings for the system.

`Init System...`

This menu entry permits the input of system-specific data for START_VALUE, REPOSITORY and OPERATING_SYSTEM statements.

Three input fields are displayed in the dialog window when this menu item is activated:

Figure 14 Dialog window: Init System

The following table describes the dialog window entries; the “Statement” column specifies the corresponding statement that is generated by WinConfig when saving to a configuration file.

Parameter name	Value	Default	Statement
OS	Name of the operating system Possible values: UNIX, WINNT, BS2000 Selected via a pull-down menu	UNIX	OPERATING_SYSTEM
Startvalue	Start value of the first generation number that is used in the generation of identifiers (see section 6.2 on page 46)	0	START_VALUE
Repository	Name of the repository file (see section 6.2 on page 46)	none	REPOSITORY

The buttons in the dialog window execute the following actions:

OK	The entered parameter values are confirmed.
Delete	All parameter values are deleted.
Cancel	The entered parameter values are discarded again.

Foreign-TA-App-Numbers...

This menu item is used to input FOREIGN_APPLICATION_NUMBER statements.

A FOREIGN_APPLICATION_NUMBER statement summarizes a list containing ADDRESS statements. This list is referenced by a number (see section 6.2 on page 46).

The following list window is displayed when this menu item is called:

Figure 15 Dialog window: Foreign-TA-App-Numbers

The list window displays the current FOREIGN_APPLICATION_NUMBER settings. There are no default settings. The statements displayed in the list window are transferred to a configuration file when they are saved.

The current settings can be modified using the Del Statement and Add Statement buttons.

- Deleting a statement

◊ Mark a FOREIGN_APPLICATION_NUMBER statement in the display area with the left mouse button.

◊ Click on Del Statement with the left mouse button.

- Inserting a statement

◊ Enter the parameters for the FOREIGN_APPLICATION_NUMBER statement in the input fields beneath the display area.

The meaning of the input fields is as follows:

Input field 1: Number of the FOREIGN_APPLICATION_NUMBER statement

The next input fields represent an ADDRESS statement for the FOREIGN_APPLICATION_NUMBER statement with the number specified in the first parameter:

Input field 2: FunctionID of the converter function

Input field 3: TransactionsCode of the application

Druck vom 24.01.2001 17:00:47 torb-ken

Input field 4: TransactionsType of the application

Input field 5: optional, ConverterId of the application

◊ Click on Add Statement with the left mouse button.

Terminate the input of FOREIGN_APPLICATION_NUMBER statements with OK or Cancel. The dialog window closes.

The buttons execute the following actions:

OK	The entered FOREIGN_APPLICATION_NUMBER statements are confirmed.
Cancel	The modified FOREIGN_APPLICATION_NUMBER statements are discarded again.

MaxState...

Customizes the MAX statements for the system.

The following list window is displayed when this menu item is called:

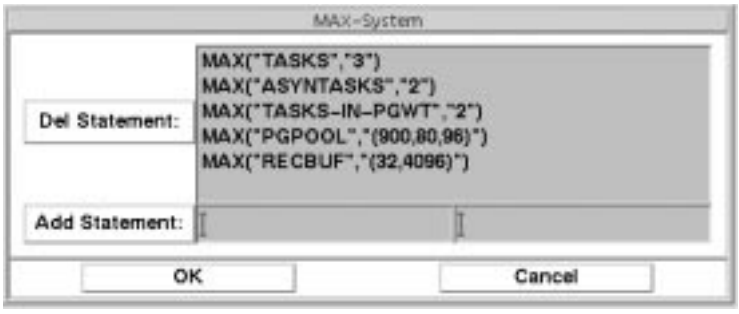


Figure 16 Dialog window: MAX system

The list window always displays the current MAX settings for the system. Figure 15 shows the default settings. The statements displayed in the list window are transferred to a configuration file when they are saved.

The current settings can be modified using the Del Statement and Add Statement buttons.

- Deleting a statement

◊ Mark a MAX statement in the display area with the left mouse button.

◊ Click on Del Statement with the left mouse button.

- Inserting a statement

◊ Enter the first and second parameters in the left or right input field beneath the display area.
WinConfig always expects that both parameters will be entered.

◊ Click on Add Statement with the left mouse button.

Terminate the input of MAX statements with OK or Cancel. The dialog window closes.

The buttons execute the following actions:

OK	The entered MAX statements are confirmed.
Cancel	The modified MAX statements are discarded again.

Druck vom 24.01.2001 17:00.47 torb-ken

Admin...

Customizes the ADMIN statement for the system.

The following list window is displayed when this menu item is called:

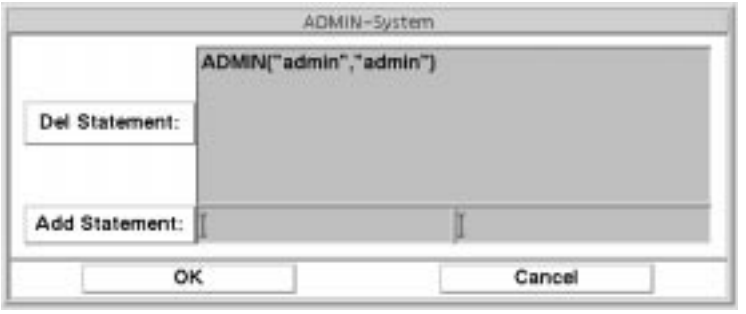


Figure 17 Dialog window: Admin system

The list window always displays the current ADMIN settings for the system. Figure 17 shows the default setting. The statements displayed in the list window are transferred to a configuration file when they are saved.

The current settings can be modified using the Del Statement and Add Statement buttons.

- Deleting a statement

◊ Mark an ADMIN statement in the display area with the left mouse button.

◊ Click on Del Statement with the left mouse button.

- Inserting a statement

◊ Enter the first and second parameters in the left or right input field beneath the display area.
WinConfig always expects that both parameters will be entered.

◊ Click on Add Statement with the left mouse button.

The buttons in the list window execute the following actions:

OK	The entered ADMIN statements are confirmed.
Cancel	The modified ADMIN statements are discarded again.

Start...

The Start menu item permits the customizing of the START and START_RM statements.

A dialog window with five input fields is displayed when this menu item is activated:

Figure 18 Dialog window: Start system

The following table describes the dialog window entries; the “Statement” column specifies the corresponding statement that is generated by WinConfig when saving to a configuration file.

Parameter name	Value	Default	Statement
Tasks	Number of work processes	none	START
Asyntasks	Number of asynchronous work processes	none	START
Tasks-in-Pgwt	Number of tasks for PGWT tasks	none	START
DBVendor	Name of the database vendor Possible values: Informix, Oracle, UDS or None Selected via a pull-down menu	Informix	START_RM
DBName	Name of the database	none	START_RM

If a database vendor is defined in line 4 of the dialog box but line 5 is left blank, the application name of the corresponding GINA application is the database name.

The buttons execute the following actions:

OK The entered parameter values are confirmed.

Delete All parameter values are deleted.

Cancel The modified parameter values are discarded again.

Druck vom 24.01.2001 17:00:48 torb-kon

Rmxa...

Customizes the RMXA statements.

A dialog window with two input fields is displayed when this menu item is activated:

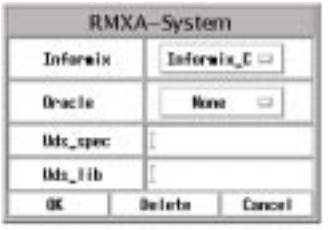


Figure 19 Dialog window: RMXA system

The following table describes the dialog window entries; the “Statement” column specifies the corresponding statement that is generated by WinConfig when saving to a configuration file.

Parameter name	Value	Default	Statement
Informix	XA interface used Possible values: Informix_C (XA-CAE interface), Informix-P (XA-P interface), None	Informix_C	RMXA
Oracle	XA interface used Possible values: Oracle_C (XA-CAE interface), Oracle-P (XA-P interface), None	None	RMXA
Uds_spec	Entry name of the database, if UDS is used as the database system	None	RMXA
Uds_lib	Name of the OML with connection module, if UDS is used as the database system	None	RMXA

The buttons in the dialog box execute the following actions:

OK The entered parameter values are confirmed.

Delete All parameter values are deleted.

Cancel The modified parameter values are discarded again.

Timer...

Customizes the statements for CYCLICTIME, CYCLICORDER, CYCLE, CHECK and CANCEL.

The following dialog window is displayed when this menu item is called:

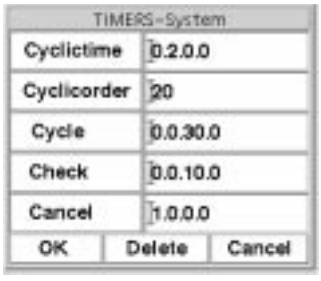


Figure 20 Dialog window: Timer system

The following table describes the dialog window entries; the “Statement” column specifies the corresponding statement that is generated by WinConfig when saving to a configuration file.

Parameter name	Value	Default	Statement
Cyclictime	Time interval for activation of the application-specific timer	none	CYCLICTIME
Cyclicorder	Maximum permitted cyclical tasks	none	CYCLICORDER
Cycle	Time interval in which events on the client that have not yet been delivered are checked	none	CYCLE
Check	Time interval after which an event on a client is incorporated in the check mechanism	none	CHECK
Cancel	Time interval after which an event on a client expires	none	CANCEL

The time intervals are entered using four values for days, hours, minutes, seconds. Note that the individual values must be separated by a dot.

The buttons in the dialog box execute the following actions:

OK The entered parameter values are confirmed.

Delete All parameter values are deleted.

Cancel The modified parameter values are discarded again.

Druck vom 24.01.2001 17:00.49 torb-kon

`Netaccess...`

Customizes the `NETACCESS` statement.

The following dialog window is opened when this menu item is called:

Figure 21 Dialog window: `Netaccess system`

The following table describes the dialog window entries; the “Statement” column specifies the corresponding statement that is generated by `WinConfig` when saving to a configuration file.

Parameter name	Value	Default	Statement
`Netaccess`	Linking of the application to the network. Possible values: `S-Thread` (SINGLE-THREADED), `M-Thread` (MULTI-THREADED), `M-Thread-All` (MULTI-THREADED,ALL)	-	NETACCESS

The buttons in the dialog window execute the following actions:

`OK` The entered parameter values are confirmed.

`Delete` All parameter values are deleted.

`Cancel` The modified parameter values are discarded again.

Priority...

The menu item is used to customize the PRIORITY statement in line with TAC class control.

The following dialog window is displayed when this menu item is called:

Figure 22 Dialog window: Priority-System

The following table describes the dialog window entries; the “Statement” column specifies the corresponding statement that is generated by WinConfig when saving to a configuration file.

Parameter name	Value	Default	Statement
High	Maximum number, the number of processes in the application that can work concurrently with the TAC class with the priority HIGH.	0	PRIORITY
Medium	Maximum number, the number of processes in the application that can work concurrently with the TAC class with the priority MEDIUM.	0	PRIORITY
Low	Maximum number, the number of processes in the application that can work concurrently with the TAC class with the priority LOW.	0	PRIORITY
Sync_Wait	Specifies for which of the three TAC classes synchronous calls are permitted. Possible values: HIGH, MEDIUM, LOW Selected via a pull-down menu	-	PRIORITY

Druck vom 24.01.2001 17:00.49 torb-kon

The buttons in the dialog window execute the following actions:

OK	The entered parameter values are confirmed. If the three parameter values for High, Medium, Low are equal to 0, WinConfig does not generate a PRIORITY statement. WinConfig outputs an error message if there is at least one parameter value equal to 0 and at least one parameter value not equal to 0.
Delete	All parameter values are deleted.
Cancel	The modified parameters are discarded again.

Schedule...

This menu item permits the input of entries for the SCHEDULE statement.

The SCHEDULE statement is used to explicitly assign the specialist classes, class methods, instance methods and functions to the TAC classes with the priorities HIGH, MEDIUM and LOW (see section 6.2 on page 46).

The following list window is displayed when this menu item is called:

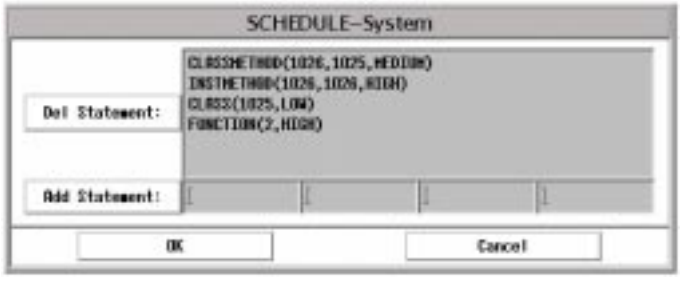


Figure 23 Dialog window: SCHEDULE-System

The list window displays the current SCHEDULE statement entries for the system. There are no default settings. The elements displayed in the list window are transferred to a configuration file (within the Schedule block) when saved.

The current settings can be modified using the Del Statement and Add Statement buttons.

- Deleting an entry

◊ Mark an entry in the display area with the left mouse button.

◊ Click on Del Statement with the left mouse button.

- Inserting an entry

◊ Insert the entry for the SCHEDULE statement in the input fields beneath the display area.

The meaning of the input fields is as follows:

Input field 1	Allows you to enter one of the keywords CLASS, CLASSMETHOD, INSTMETHOD or FUNCTION.
Input field 2	Allows you to enter the ClassId (not required if FUNCTION was entered in the first input field).
Input field 3	Allows you to enter the ClassmethodID, InstMethodId, FunctionId (not required if CLASS was entered in the first input field CLASS).

Druck vom 24.01.2001 17:00:50 torb-kon

Input field 4 — Allows you to enter one of the keywords LOW, MEDIUM or HIGH (to define the TAC class).

◊ Click on Add Statement with the left mouse button.

The buttons execute the following actions:

OK	The entered entries for the SCHEDULE statement are confirmed.
Cancel	The modified entries for the SCHEDULE statement are discarded again.

Import...

This menu item allows you to customize the IMPORT statement.

The following list window is displayed when this menu item is called:

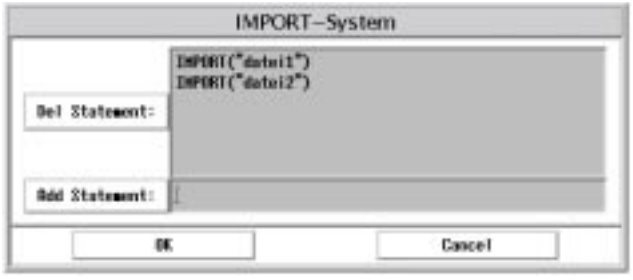


Figure 24 Dialog window: IMPORT-System

The list window always displays the current IMPORT statements for the system. There are no default settings. The statements displayed in the list window are transferred to a configuration file when they are saved.

The current settings can be modified using the Del Statement and Add Statement buttons.

- Deleting a statement

◊ Mark an IMPORT statement in the display area with the left mouse button.

◊ Click on Del Statement with the left mouse button.

- Inserting a statement

◊ Enter the parameter (filename) for the IMPORT statement in the input field beneath the display area.

◊ Click on Add Statement with the left mouse button.

The buttons execute the following actions:

OK	The entered IMPORT statements are confirmed.
Cancel	The modified IMPORT statements are discarded again.

Druck vom 24.01.2001 17:00:50 torb-kon

Area...

This menu item allows you to customize the AREA statement.

The following list window is displayed when this menu item is called:

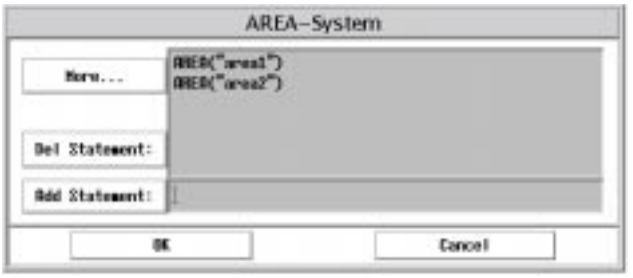


Figure 25 Dialog window: AREA-System

The list window always displays the current AREA statements for the system. There are no default settings. The statements displayed in the list window are transferred to a configuration file when they are saved.

The current settings can be modified using the Del Statement and Add Statement buttons.

- Deleting a statement

◊ Mark an AREA statement in the display area with the left mouse button.

◊ Click on Del Statement with the left mouse button.

- Inserting a statement

◊ Enter the mandatory parameter (data area name) for the AREA statement in the input field beneath the display area.

◊ Click on Add Statement with the left mouse button.

Further optional parameters for this AREA statement can be entered by marking an AREA statement and clicking on More... with the left mouse button. A further dialog window is displayed:

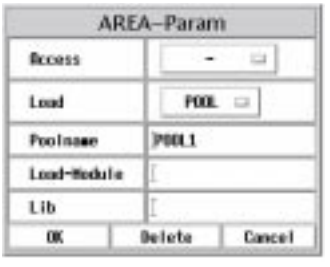


Figure 26 Dialog window: AREA-Param

The following table describes the dialog window entries; the “Statement” column specifies the corresponding statement that is generated by WinConfig when saving to a configuration file.

Parameter name	Value	Default	Statement
Access	Mode of access to the additional data area. Possible values: DIRECT INDIRECT Selected via a pull-down menu. (This parameter only exists for OS_UNIX or OS_WINNT.)	-	AREA
Load	Specifies when and where the data range is to be loaded. Possible values: STATIC POOL Selected via a pull-down menu. You must specify a poolname with POOL. (This parameter only exists for OS_BS2000.)	-	AREA
Poolname	Name with which the data area in the common memory pool is loaded. (This parameter only exists for OS_BS2000.)	None	AREA
Load-Module	Name of the load module in which the module (i.e. the data area which can be used jointly) is linked. (This parameter only exists for OS_BS2000.)	None	AREA
Lib	Program library from which the module is to be dynamically loaded or linked. (This parameter only exists for OS_BS2000.)	None	AREA

The buttons in the dialog window execute the following actions:

OK The entered additional parameter values are confirmed.

Delete All additional parameter values are deleted.

Cancel The modified parameters are discarded again.

Druck vom 24.01.2001 17:00:51 torb-kon

The buttons in the list window execute the following actions:

OK	The entered AREA statements are confirmed.
Cancel	The modified AREA statements are discarded again.

Mpool...

This menu item allows you to customize the Mpool statement. This statement is only of relevance for BS2000 hosts.

The following list window is displayed when this menu item is called:

Figure 27 Dialog window: Mpool-System

The list window always displays the current MPOOL statements for the system. There are no default settings. The statements displayed in the list window are transferred to a configuration file when they are saved.

The current settings can be modified using the Del Statement and Add Statement buttons.

- Deleting a statement

◊ Mark an MPOOL statement in the display area with the left mouse button.

◊ Click on Del Statement with the left mouse button.

- Inserting a statement

◊ Enter the two mandatory parameters (name of the common memory pool, number of 64K memory segments in the pool) for the MPOOL statement in the two input fields beneath the display area.

◊ Click on Add Statement with the left mouse button.

Further optional parameters for this MPOOL statement can be entered by marking an MPOOL statement and clicking on More... with the left mouse button. A further dialog window is displayed:

Figure 28 Dialog window: MPOOL-Param

The following table describes the dialog window entries; the “Statement” column specifies the corresponding statement that is generated by WinConfig when saving to a configuration file.

Parameter name	Value	Default	Statement
Access	Defines the access authorization. Possible values: DIRECT INDIRECT Selected via a pull-down menu.	-	MPOOL
Lib	Identifies the object module library when switching programs using KDCLOAD.	None	MPOOL
Page	Sedecimal address in the form X'xxxxxxxxx'	None	MPOOL
Scope	Defines the scope of the common memory pool. Possible values: GLOBAL GROUP Selected via a pull-down menu.	-	MPOOL
Sharetab	CSECT name of the KDCSHARE table to be loaded in this common memory pool.	None	MPOOL

The buttons in the dialog window execute the following actions:

OK The entered additional parameter values are confirmed.

Delete All additional parameter values are deleted.

Cancel The modified parameters are discarded again.

Druck vom 24.01.2001 17:00:52 torb-kon

The buttons in the list window execute the following actions:

OK	The entered MPOOL statements are confirmed.
Cancel	The modified MPOOL statements are discarded again.

Dyn_Connect

This menu item supports the definition of connections which are to be used by non-transaction-monitored dynamic clients for communication with transaction-monitored T-ORB applications (see section 6.2 on page 46).

The Dyn_Connect menu item contains two submenu items:

Dyn_Connect>System...

This menu item allows you to customize the DYNAMIC_CONNECT statement.

The following list window is displayed when this menu item is called:

Figure 29 Dialog window: Dyn_Connect-System

The list window always displays the current entries for the DYNAMIC_CONNECT statement for the system. There are no default settings. The entries displayed in the list window are transferred to a configuration file as parameters of the DYNAMIC_CONNECT statement when they are saved.

The current entries can be modified using the Del Statement and Add Statement buttons.

- Deleting an entry

◊ Mark an entry in the display area with the left mouse button.

◊ Click on Del Statement with the left mouse button.

- Inserting an entry

◊ Insert the entries for the DYNAMIC_CONNECT statement in the two input fields beneath the display area.

The meaning of the two input fields is as follows:

Input field 1	Application name of the TA_APPLICATION for which connections are to be generated.
Input field 2	Number of connections.

◊ Click on Add Statement with the left mouse button.

The buttons in the list window execute the following actions:

OK	The entered entries are confirmed.
Cancel	The modified entries are discarded again.

Druck vom 24.01.2001 17:00:52 torb-kon

Dyn_Connect>not_inherit_for_hosts...

This menu item allows you to input the hosts which are not to inherit the DYNAMIC_CONNECT statement from the system.

The following list window is displayed when this menu item is called:

Figure 30 Dialog window: Hosts (not inherit DYNAMIC_CONNECT)

The list window always displays all hosts which are not to inherit the DYNAMIC_CONNECT statement from the system, i.e. when saving to a configuration file, WinConfig generates an empty DYNAMIC_CONNECT statement in the respective host block (if there are no host-specific SCHEDULE entries). There are no default settings.

The current settings can be modified using the Del Statement and Add Statement buttons.

- Deleting a host entry

◊ Mark a host name in the display area with the left mouse button.

◊ Click on Del Statement with the left mouse button.

- Inserting a host entry

◊ Enter the host name beneath the display area.

◊ Click on Add Statement with the left mouse button.

The buttons in the list window execute the following actions:

OK	The entered host name are confirmed.
Cancel	The modified host name are discarded again.

Hosts menu

This menu allows you to modify the customizing settings for the hosts.

MaxState...

This menu item allows you to customize MAX statements for all hosts whose icons are open.
The MAX statements are input in the same way as for the menu item System>MaxState....
There are no default values.

Admin...

This menu item allows you to customize ADMIN statements for all hosts whose icons are open.
The ADMIN statements are input in the same way as for the menu item System>Admin....
There are no default values.

Start...

This menu item allows you to customize START and START_RM statements for all hosts whose icons are open.
The statements are input in the same way as for the menu item System>Start....
There are no default values.

Rmxa...

This menu item allows you to customize RMXA statements for all hosts whose icons are open.
The statements are input in the same way as for the menu item System>Rmxa....
There are no default values.

Timer...

This menu item allows you to customize CYCLICTIME, CYCLICORDER, CYCLIC, CHECK and CANCEL statements for all hosts whose icons are open.
The statements are input in the same way as for the menu item System>Timer....
There are no default values.

Netaccess...

This menu item allows you to customize the NETACCESS statement for all hosts whose icons are open.
The statements are input in the same way as for the menu item System>Netaccess....
There are no default values.

Druck vom 24.01.2001 17:00:53 torb-kon

`Priority...`

This menu item allows you to customize the `PRIORITY` statement for all hosts whose icons are open.
The statements are input in the same way as for the menu item `System>Priority...`.
Default values are the same as for `System`.

`Schedule...`

This menu item allows you to customize the `SCHEDULE` statement for all hosts whose icons are open.
The entries are input in the same way as for the menu item `System>Schedule...`.
There are no default values.

`Import...`

This menu item allows you to customize the `IMPORT` statement for all hosts whose icons are open.
The statements are input in the same way as for the menu item `System>Import...`.
There are no default values.

`Area...`

This menu item allows you to customize the `AREA` statement for all hosts whose icons are open.
The statements are input in the same way as for the menu item `System>Area...`.
There are no default values.

`Mpool...`

This menu item allows you to customize the `MPOOL` statement for all hosts whose icons are open.
The statements are input in the same way as for the menu item `System>Mpool...`.
There are no default values.

`Dyn_Connect...`

This menu item allows you to customize the `DYNAMIC_CONNECT` statement for all hosts whose icons are open.
The entries are input in the same way as for the menu item `System>Dyn_Connect>System...`.
There are no default values.

TA-Apps menu

This menu allows you to modify the customizing settings for the TA applications.

`MaxState...`

This menu item allows you to customize `MAX` statements for all TA applications whose icons are open. The `MAX` statements are input in the same way as for the menu item `System>MaxState...`.
There are no default values.

`Admin...`

This menu item allows you to customize `ADMIN` statements for all TA applications whose icons are open. The `ADMIN` statements are input in the same way as for the menu item `System>Admin...`.
There are no default values.

`Start...`

This menu item allows you to customize `START` and `START_RM` statements for all TA applications whose icons are open. The `START` and `START_RM` statements are input in the same way as for the menu item `System>Start...`.
There are no default values.

`Rmxa...`

This menu item allows you to customize `RMXA` statements for all TA applications whose icons are open. The statements are input in the same way as for the menu item `System>Rmxa...`.
There are no default values.

`Timer...`

This menu item allows you to customize `CYCLICTIME`, `CYCLICORDER`, `CYCLIC`, `CHECK` and `CANCEL` statements for all TA applications whose icons are open. The statements are input in the same way as for the menu item `System>Timer...`.
There are no default values.

`Netaccess...`

This menu item allows you to customize the `NETACCESS` statement for all TA applications whose icons are open. The statements are input in the same way as for the menu item `System>Netaccess...`.
There are no default values.

Druck vom 24.01.2001 17:00:53 torb-kon

`Priority...`

This menu item allows you to customize the `PRIORITY` statement for all TA applications whose icons are open. The statements are input in the same way as for the menu item `System>Priority...`.
Default values are the same as for `System`.

`Schedule...`

This menu item allows you to customize the `SCHEDULE` statement for all TA applications whose icons are open. The statements are input in the same way as for the menu item `System>Schedule...`.
There are no default values.

`Import...`

This menu item allows you to customize the `IMPORT` statement for all TA applications whose icons are open. The statements are input in the same way as for the menu item `System>Import...`.
There are no default values.

`Area...`

This menu item allows you to customize the `AREA` statement for all TA applications whose icons are open. The statements are input in the same way as for the menu item `System>Area...`.
There are no default values.

`Mpool...`

This menu item allows you to customize the `MPOOL` statement for all TA applications whose icons are open. The statements are input in the same way as for the menu item `System>Mpool...`.
There are no default values.

Bcamappl...

This menu item allows you to customize the BCAMAPPL statement.

When this menu item is called, the following list window is displayed for each TA application whose icon is open:

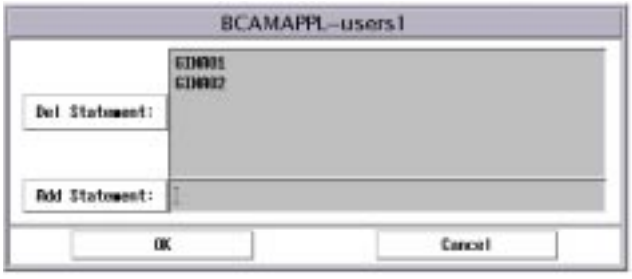


Figure 31 Dialog window: BCAMAPPL

The list window displays the current BCAMAPPL parameters for a specific TA application. There are no default settings. The BCAMAPPL parameters displayed in the list window are transferred to a configuration file when they are saved.

The current parameters can be modified using the Del Statement and Add Statement buttons.

- Deleting a statement

◊ Mark a BCAMAPPL parameter in the display area with the left mouse button.

◊ Click on Del Statement with the left mouse button.

- Inserting a parameter

◊ Enter the parameter (appliname) for the BCAMAPPL statement in the input field beneath the display area.

◊ Click on Add Statement with the left mouse button.

The buttons in the list window execute the following actions:

OK	The entered BCAMAPPL parameters are confirmed.
Cancel	The modified BCAMAPPL parameters are discarded again.

Druck vom 24.01.2001 17:00:54 torb-kon

Non-TA-Apps menu

This menu permits the input of a UserId and Password for non-TA applications.

Admin...

The optional input of a UserId and Password is enabled in the non-TA application window for all non-TA applications whose icons are open.

Foreign-Apps menu

This menu can be used to modify the customizing settings for the foreign applications.

Bcamappl...

This menu item allows you to customize the BCAMAPPL statement for all foreign applications whose icons are open. The parameters of the BCAMAPPL statement are input in the same way as for the menu item TA-Apps>Bcamappl... .

There are no default values.

Links menu

This menu supports the simultaneous generation/deletion of a number of sessions, connections and foreign sessions as well as the hiding and showing of selected sessions, connections and foreign sessions.

Link

This menu item contains the two submenus set Filter and Filter off:

Link>set Filter

For a better overview with larger configurations, all sessions, connections and foreign sessions of applications whose dialog boxes are not open can be hidden. This is done by calling the Link>set Filter menu.

Link>Filter off

Calling the Link>Filter off menu disables the filter and redisplays all sessions, connections and foreign sessions.

Multi Link

This menu item contains the two submenus Create Some and Erase Some. These two submenus can be used to generate or delete groups of sessions, connections and foreign sessions in a targeted manner.

Link>Create Some

The following dialog window opens when the Link>Create Some menu is called:

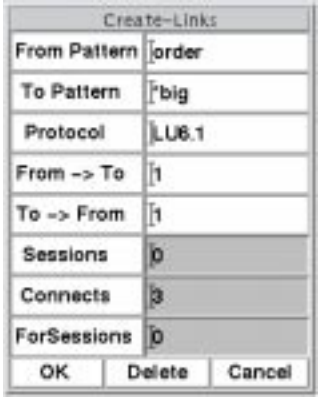


Figure 32 Dialog window: Create links

Patterns can be entered in each of the From Pattern and To Pattern input fields that define a group of applications for which sessions, connections or foreign sessions are to be created.

WinConfig supports the following metacharacters in the patterns:

* Any number of characters

? Exactly one character

. ORing of two patterns

For example, the a*.b? pattern represents all applications that start with an a as well as all applications that start with a b, followed by any other letter.

The third input field defines the protocol used (default: LU6.1). The From->To and To-> From input fields specify the number of connections checked by the source application or target application respectively (default: 1 each). These two parameters are only of importance for sessions and foreign sessions. There are of no importance for a connection.

WinConfig displays the number of sessions, connections or foreign sessions created using the Create links dialog in the sixth, seventh and eighth lines of the window. This information cannot be edited – it is calculated automatically.

Druck vom 24.01.2001 17:00:55 torb-kon

The buttons in the dialog window execute the following actions:

OK	Generates all of the sessions/connections/foreign sessions between applications that were specified by the pattern entered.
Delete	Deletes the values entered by the user.
Cancel	Closes the dialog window.

Link>Erase Some

The following dialog window opens when the Link>Erase Some menu is called:

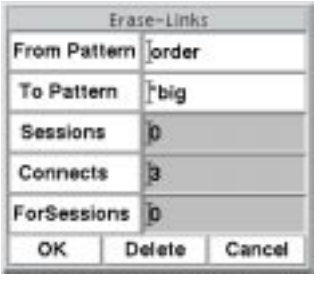


Figure 33 Dialog window: Erase links

Patterns can be defined in each of the From Pattern and To Pattern input fields that define a group of applications for which sessions, connections or foreign sessions are to be deleted. WinConfig recognizes the same metacharacters (*, ? and .) in the patterns as in the Link>Create Some menu item.

WinConfig outputs the number of sessions, connections or foreign sessions deleted using the Erase links dialog in the third, fourth and fifth lines of the window. This information cannot be edited – it is calculated automatically.

The buttons in the dialog window execute the following actions:

OK	Generates all of the sessions/connections/foreign sessions between applications that were specified by the pattern entered.
Delete	Deletes the values entered by the user.
Cancel	Closes the dialog window.

Moves menu

This menu allows the user to change the layout of the configuration file displayed in the WinConfig main window.

Move Param

Among other things, the Moves menu permits a group of applications to be shifted using the Right, Left, Up, Down menu items and the cursor keys. The Move Param... menu item is used to define which applications belong to a group.

The following dialog window is opened when this menu item is called:

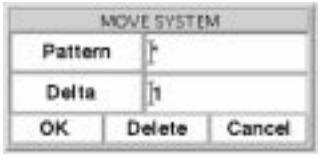


Figure 34 Dialog window: Move Param

A pattern can be entered in the upper input field by means of which the group of applications is specified. WinConfig recognizes the same metacharacters (*, ? and .) in a pattern as in the Link>Create Some menu item (see page 150).

In the lower input field, the user defines the distance by which a group of applications is to be moved. The default value is 1.

The buttons in the dialog box execute the following actions:

OK	The entered parameter values are confirmed.
Delete	All parameter values are deleted.
Cancel	The modified parameter values are discarded again.

Right, Left, Up, Down

You can use these menu items to move a group of applications (defined by the Move Param... menu item, see page 152) to the right, left, top or bottom by a specific delta (also specified using Param...). The movement can also be performed using the cursor keys.

Grid on | off

You can move a group of applications in grid formation if you call the Grid on | off menu. A mark at the lower edge of the open Moves menu (beside the Grid on | off menu item) shows that grid formation is enabled.If you call this menu item again, the grid format is disabled and the mark disappears again.

Druck vom 24.01.2001 17:00:55 torb-kon

6.7.3 Mouse key assignments and mouse actions

This section summarizes once again all of the mouse actions used by WinConfig and their effect. The most important mouse actions have already been explained in the preceding sections.

Physical mouse key	Position of the mouse cursor	Mouse action	Effect
Left mouse button	Menu bar	Press	Opens the menu
Left mouse button	Open menu	Drag to a menu item	Executes the action of the selected menu item
Left mouse button	Toggle button	Click	Selects one of the alternatives
Left mouse button	Pushbutton	Click	Executes the relevant action
Left mouse button	Host icon	Click	Opens the host dialogue window
Left mouse button	TA, non-TA or foreign application icon	Double-click	Opens the TA, non-TA or foreign application dialogue window
Left mouse button	TA, non-TA or foreign application icon	Click, drag to another application icon, click again	Configures a session, connection or foreign session between two application icons
Left mouse button	Connecting line between two application icons	Click	Opens a session, connection or foreign session dialogue window
Middle mouse button	TA, non-TA or foreign application icon	Click	The icon tracks the cursor movement until the next mouse click
Middle mouse button	Host icon label	Press and drag to an input field	Copies the label to the input field
Right mouse button	WinConfig window	Press	Pop-up menu for generating a new host, a new TA, non-TA or foreign application with WinConfig or for autoscrolling

Physical mouse key	Position of the mouse cursor	Mouse action	Effect
Right mouse button	Pop-up menu for generating a new host, a new TA, non-TA or foreign application or for autoscrolling	Drag to the menu item `New Host`, `New TA-App`, `New Non-TA-App`, `New Foreign-App` or `Autoscroll`	Depending on the menu selected, a new host, TA, non-TA or foreign application will be generated or autoscrolling activated

Druck vom 24. 01.2001 17:00:56 torbkomtux

7 Configuring T-ORB for BEA TUXEDO

This chapter describes how to configure communication between GINA applications. The configuration does not require any modifications to the source program. However, it cannot take place while the application is running; rather it must be performed *before* the application *starts*.

The configuration of T-ORB is based on the configuration of BEA TUXEDO. Knowledge of the respective manuals will therefore aid comprehension. These manuals are listed under *Related publications* at the back of this manual.

The `WinConfig` graphical user interface is only available on UNIX platforms and only for *open*UTM. ❍❍●

7.1 Overview

To operate a distributed system on the basis of GINA, certain information must be known on the desired system structure, the current parameter settings of the host, and the characteristics of the GINA applications.

The configuration tool described here records this information in the form of a general system description, from which it generates the data required by GINA.

The information you must enter can be divided into the following parts (see Figure 35 on page 156):

– specifications on system-wide settings, such as the participating systems
– specifications on host-specific parameters, such as operating system resources
– specifications on application-specific parameters, such as the number of work processes

Practically speaking, some of the parameters can be specified in more than one of these areas. In such cases, the values set higher up in the hierarchy must be taken as default settings. These values are then overwritten by the settings in more specific areas. The desired settings can thus be specified separately for each application if required (customizing).

If parameters are specified more than once at system level, they are then valid for the rest of the system level or until a new parameter is specified.

Figure 35 is a symbolic representation of the hierarchies in the definition of the system. It is not drawn to scale to reflect the scope of the individual sub-descriptions.

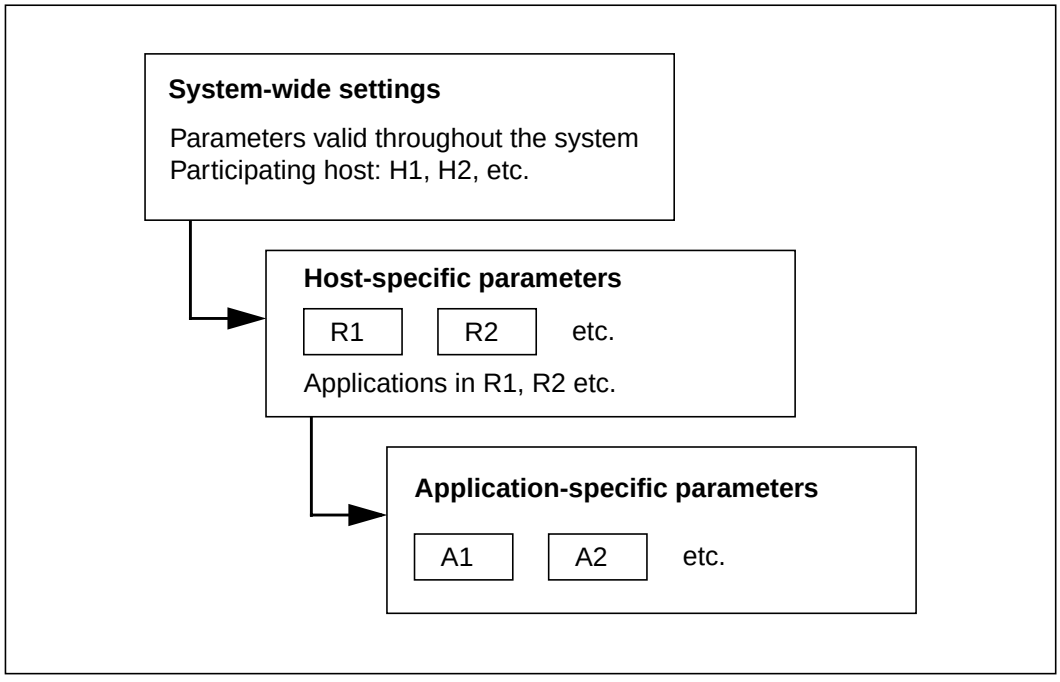


Figure 35 The logical hierarchy when defining the communication structure of a system

The communication structure of a system can be depicted by a graph with nodes and edges. The nodes correspond to the applications, while the edges represent the communication channels. The blocks in the diagram define the nodes of the graph with respect to the specific system, host or application. The edges of the graph represent the applications' connections, with each application connected to all of the other applications.

From the input data (which has already been explained), the configuration generator `config-tux` creates the following output data:

– for each host
  a GINA-specific address file containing all addressable server applications.

– for each application
  a GINA-specific address file containing all addressable server applications and client applications.

– for each transaction-monitored application
  a GINA-specific environment file preset with the definition of the `GINACONFIG` environment variable.

Druck vom 24.01.2001 17:00:57 torbkomtux

– for the MASTER host
  a configuration file `ubbconfig` for the BEA TUXEDO transaction monitor and the shell scripts `crbincf` and `crtlogs`.

– for each host
  a shell script `crdevqu` for setting up the “Queue Device”, “Queue Space” and “Queues” and for generating the database transaction manager.

The configuration can be performed on a central computer for the entire system. The files created in the process must then be distributed to the target computers and installed there. The final tasks which must be performed locally are carried out during this installation. Moving the final tasks to the target computers eliminates the need to install the transaction monitor on the generation computer. Because the output data for the generation comprises only text files, the hardware and operating system independence of the configuration process is also guaranteed. You must observe the different ways in which UNIX and WindowsNT display the end of a line in a text file if the generation computer and the target computer are running different operating systems.

In the first version of the configuration tool, the procedure is that change requests are sent to a central location and that a new configuration process will be implemented from there only. The configuration generator makes it easier to configure the runtime environment for T-ORB and T-ORB/Client. Revision generation is described in section 7.3 auf Seite 174.

A central configuration is recommended for the following reasons:

– It does not make sense to change the configuration on the local target computers because these modifications will be overwritten with the next global update process.

– Another argument against changing the settings locally is that modifications to the hierarchy of the system often affect more than one host. It is precisely changes of this type that require consistency checks, which are not possible on the local level.

7.2 Configuration language

The configuration generator `config-tux` reads a text file which describes the configuration of the entire system in a T-ORB-specific language. This file contains the necessary information on the transaction monitor, the T-ORB, and the specialist application.

The elements of the language include keywords, literals, separators and comments. Blanks, tabs, line feeds, form feeds and white spaces are ignored. The characters `//` introduce a comment, the line feed character terminates it.

7.2.1 Statements

The configuration language contains a range of statements which are introduced by keywords; these are explained below. The statements may include numerous BEA TUXEDO parameters.

The statements may contain two different styles: uppercase letters only or lowercase letters only. ❍❍●

APPLICATION

The `APPLICATION` statement describes a client which is connected via T-ORB/Client. It comprises the following components:

– `OsId` of the application (`NUMBER, 1 ... 32767`)
– `LayerId` of the application (`NUMBER, 1 ... 32767`)
– User-friendly name of the application (`LETTER`)

Example

```
APPLICATION ( 131, 1, "CmdTrm1" )
```

CM_APPLICATION

The `CM_APPLICATION` statement describes a conversational mode application on the current host (`HOST`). It comprises the following components:

– TP application name (`LETTER`, max. 30 characters)
– `OsId` of the application (`NUMBER, -128 ... -32767`)
– `LayerId` of the application (`NUMBER, 1 .. .32767`)
– User-friendly name of the application (`LETTER`)

Customizing statements (`MAX` statements `only`) are permitted after the `CM_APPLICATION` description of a server.

Druck vom 24. 01.2001 17:00:57 torbkomtux

Example
```
CM_APPLICATION ( "CM1",-330, 1, "CM1" )
{
...
}
```

CM_PREFIX

The CM_PREFIX statement defines a unique character string for each host which is used as a prefix when generating the parts resulting from conversational mode applications.

The CM_PREFIX statement is mandatory if conversational mode applications are to be assigned to the host. The string may not exceed 5 characters and must start with a letter.

Example
```
CM_PREFIX ( "mch" )
```

CM_APPLICATIONS

The CM_APPLICATIONS statement defines a list of conversational mode applications at *system level*. Each such list must be assigned a different number and can be copied to a host using the statement USE_CM_APPLICATIONS.

Example
```
CM_APPLICATIONS ( 20 )
{
  CM_APPLICATION ( "CM20",-220, 1, "CM20" )
  {
  ...
  }

...
}

:
```

HOST

The HOST statement describes the configuration of a host. The HOST statement comprises the following components:

- host name, max. 30 characters
- optional parameter MASTER or BACKUP
- Internet address of the host (INTERNETADDRESS)
- shared memory and semaphore key (KEYVECTOR)
- available port numbers (PORTADDRESS)
- host-specific customizing statements (MAX, OPENINFO)
- description of existing applications

Example

```
HOST ( "Host2" )
{
INTERNETADDRESS ( 127.0.0.2 )
KEYVECTOR ( 5005, 5040 )
PORTADDRESSES ( 2000, 2100 )
...
}
```

INTERNETADDRESS

The INTERNETADRESS statement is used to specify the current Internet address of the host. or the relative domain name of the domain name system.

KEYVECTOR

The KEYVECTOR statement contains the keys for the BEA TUXEDO areas:

– a key to identify IPC resources for the “Queue Space” for each host with server applications

The KEYVECTOR statement has the following parameters:

– start key
– end key

MAX

The MAX statement allows you to customize TP applications.

At *system level*, the MAX statement applies to all hosts if there is no MAX statement of the same name in the HOST statement.

The MAX statements with the names CM_IDLETIME, CM_IDLETIME_MS, ENVDIR, GINACONFIGDIR, MIN, MAX, REQUESTQUEUE,TMQF_IDLETIME and TMQF_TRANSACTIONTIME at *host level* apply accordingly to all server applications.

The optional MAX statement has the following parameters:

– statement name (LETTER)
– value of the statement as a string

The statement name IPCKEY must be defined at *system level*.

If the configuration generator config-tux is called with the option -s, then GINACONFIGDIR must be defined at *system level*.

Druck vom 24.01.2001 17:00:57 torbkomtux

The statement names APPDIR (78), ENVDIR, GINACONFIGDIR, TLOGDEVICE (64), TUXCONFIG (64), TUXDIR (78) and QUEUEFILE (78) must be defined indirectly or directly for all server machines. The numbers in brackets indicate the maximum number of characters permitted in the parameter value.

The statement names ENVFILE, LDBAL, LMID, MASTER, MODEL, OPTIONS and TYPE may *not* be specified as these statements are generated automatically.

The statement names MAXDRT, MAXRFT and MAXRTDATA may *not* be specified as this functionality is not used.

The statement names MESSAGES and PAGES are used to define the corresponding parameters of the qspacecreate utility routine.
The default values are "MESSAGES" = "100" and "PAGES" = "2048".

The statement name REQUESTQUEUE is used to control generation of the parameters REPLYQ and RQADDR for each server application in the file ubbconfig.
The default value is "REQUESTQUEUE" = "N".
The REPLYQ parameter specifies whether a separate REPLY-QUEUE will be set up (values: Y or N).
See *TUXEDO Administration Guide* [2], Section 3 *Services, Servers & Server Groups*.

The statement name TMQF_IDLETIME is used to control generation of the parameter -i idltime for each TMQFORWARD server in the file ubbconfig. -i idletime is not generated if this statement is omitted.

The statement name TMQF_TRANSACTIONTIME is used to control generation of the parameter -t transactiontime for each TMQFORWARD server in the file ubbconfig. -t transactiontime is not generated if this statement is omitted.
See *TMQFORWARD – TUXEDO System/T Message Forwarding Server* in [1].

The statement name CM_IDLETIME is used to control generation of the parameter -i idltime for each conversational mode server in the file ubbconfig. -i idletime is not generated if this statement is omitted.

The statement name CM_IDLETIME_MS is used to control generation of the parameter -I idltime for each conversational mode server in the file ubbconfig. -I idletime is not generated if this statement is omitted. CM_IDLETIME and CM_IDLETIME_MS are mutually exclusive.

Example MAX ("CM_IDLETIME_MS" = "570")

OPENINFO

The optional statement `OPENINFO` defines the Resource Manager. The parameters contain the following information:

– the name of the database manufacturer
– the keyword `APPLICATION` or a string
  The current application name is used if `APPLICATION` is specified.
– the optional parameter `TMS_NAME`
  The default values is `"TMS_INFX72"`.

The `OPENINFO` statement at *system level* applies to all hosts unless the `HOST` statement contains an `OPENINFO` statement.
The `OPENINFO` statement at *host level* applies to all TA applications unless the `TA_APPLICATION` statement contains an `OPENINFO` statement.

Example `OPENINFO ( "INFORMIX-OnLine" = APPLICATION, "TMS_INFX3")`

OPERATING_SYSTEM

The optional `OPERATING_SYSTEM` statement defines the operating system of the host. Possible values are `"OS_UNIX"` and `"OS_WINNT"`.
The default value is `OPERATING_SYSTEM( "OS_UNIX" )`.

The `OPERATING_SYSTEM` statement on *system level* applies to all hosts if there is no `OPERATING_SYSTEM` statement in the `HOST` statement.

PORTADDRESSES

The `PORTADDRESSES` statement contains the port numbers. The statement has the following parameters:

– first port number
– last port number

Example `PORTADDRESSES ( 5000, 5005 )`

REPOSITORY

The `REPOSITORY` statement defines the file name of a repository in the current directory. This repository contains the generation number for each of the `OsId` and `LayerId` pair.

If the file exists, these values are read, saved internally, and used during generation. If the file does not exist, the values are generated automatically. The repository is written once generation of these values is concluded.

Druck vom 24.01.2001 17:00:58 torbkomtux

The repository supports generation for a modified configuration. If an application is removed, the same identifiers are used in the ubbconfig file for the remaining applications.

The REPOSITORY statement is optional. If this statement is not specified, no memory is used in the generation.

Example REPOSITORY ("repository")

START_VALUE

The START_VALUE statement defines the first generation number that is used in the generation of identifiers in statements for the ubbconfig file. This results in unique reproductions of the OsId and LayerId pair in these identifiers. By specifying different start values, different subsystems that can be combined can be generated.

The START_VALUE statement is optional. If this statement is not specified, generation of the identifiers starts with the value 1.

If a repository exists, it is used to ascertain the generation numbers, i.e. the START_VALUE statement is ignored.

Example START_VALUE (240800312)

SYSTEM

The SYSTEM keyword must always be the first keyword in the description. The current description is enclosed in braces, i.e. SYSTEM {...}.

The flag DEBUG can be specified as an option: SYSTEM(DEBUG){...}. Scripts for WindowsNT, for example, will then be generated with ECHO ON.

TA_APPLICATION

The TA_APPLICATION statement describes a transaction-monitored application on the current host (HOST). It comprises the following components:

– TP application name (LETTER, max. 30 characters)
– OsId of the application (NUMBER, 1 ... 32767)
– LayerId of the application (NUMBER, 1 ... 32767)
– User-friendly name of the application (LETTER)

Customizing statements (MAX and OPENINFO) are permitted after the TA_APPLICATION description of a server.

Example

```
TA_APPLICATION ( "OS1", 1, 1, "OS1" )
{
...
}
```

USE_CM_APPLICATIONS

The optional statement `USE_CM_APPLICATIONS` on the current machine (`HOST`) copies the conversational mode applications from the *system level* on this machine.

Example

```
USE_CM_APPLICATIONS ( 50, 55 )
```

Druck vom 24. 01.2001 17:00:58 torbkomtux

7.2.2 Lexical structure

This section describes how the configuration generator combines the contents of the configuration file into symbols (like the keywords) for syntax analysis. The description is in the notation used by the UNIX command `lex`.

```
%{

%}

letter                    [a-zA-Z_]
DGS                       [0-9]+
letter_or_digit           [a-zA-Z_0-9]

%%
[ \t\n\v\r\f]+            ;
\/\/.*\n                  ;
\{                        {return(BBOPEN);}
\}                        {return(BBCLOSE);}
\(                        {return(SBOPEN);}
\)                        {return(SBCLOSE);}
\,                        {return(COMMA);}
\=                        {return(EQUAL);}
\-                        {return(MINUS);}
{DGS}                     {yylval.number=atol(yytext);
                           return(NUMBER);}
{DGS}\.{DGS}\.{DGS}\.{DGS} {strcpy(yylval.string,yytext);
                           return(INADDRESS);};
\"[^"\n]*                 {yytext[yyleng++] = yyinput();
                           yytext[yyleng] = '\0';
                           lettercpy(yylval.string,yytext);
                           return(LETTER);};
{letter}{letter_or_digit}* {return(rwlookup());};
.                         {return(CONFIG_ERROR);};
%%
```

7.2.3 Syntax

This section describes the syntax of the configuration language in the notation used by the UNIX command yacc.

```
%{

%}

%start sysblock

%union
{
    long      number;
    char      string[80];
}

%token <number> NUMBER           /* positive integer          */
%token BACKUP                    /* flag                      */
%token DEBUG                     /* flag                      */
%token CM_PREFIX                 /* statement                 */
%token SYSTEM                    /* statement                 */
%token HOST                      /* statement                 */
%token APPLICATION               /* statement                 */
%token CM_APPLICATION            /* statement                 */
%token TA_APPLICATION            /* statement                 */
%token MASTER                    /* flag                      */
%token MAX_STMT                  /* statement                 */
%token KEYVECTOR                 /* statement                 */
%token OPENINFO                  /* statement                 */
%token PORTADDRESSES             /* statement                 */
%token INTERNETADDRESS           /* statement                 */
%token REPOSITORY                /* statement                 */
%token START_VALUE               /* statement                 */
%token OPERATING_SYSTEM          /* statement                 */
%token UFN                       /* flag                      */
%token USE_CM_APPLICATIONS       /* statement                 */
%token <string> INADDRESS        /* Internet address          */
%token <string> LETTER           /* string with  a..Z,0..9, _ */
%token BBOPEN                    /* {                         */
%token BBCLOSE                   /* }                         */
%token SBOPEN                    /* (                         */
%token SBCLOSE                   /* )                         */
```

Druck vom 24.01.2001 17:00:58 torbkomtux


```
%token COMMA                    /* ,                          */
%token EQUAL                    /* =                          */
%token MINUS                    /* -                          */
%token CONFIG_ERROR             /* error code (internal)      */
%%

sysblock       : system
                 BBOPEN
                 sys_statement_list_opt
                 multi_host
                 BBCLOSE
               ;

system         : SYSTEM
               | SYSTEM SBOPEN DEBUG SBCLOSE
               ;

sys_statement_list_opt
               : /* empty */
               | sys_statement_list
               ;

sys_statement_list
               : org_statement_list  s_statement_list_opt
               |                     s_statement_list
               ;

org_statement_list
               :                     org_statement
               | org_statement_list org_statement
               ;

s_statement_list_opt
               : /* empty */
               | s_statement_list
               ;

s_statement_list
               :                  s_statement
               | s_statement_list s_statement
               ;
```

```
s_statement    : statement
               | conv_mode_appls
               ;

org_statement  : start_value
               | repository
               | operating_system
               ;

start_value    : START_VALUE SBOPEN NUMBER SBCLOSE
               ;

repository     : REPOSITORY SBOPEN LETTER SBCLOSE
               ;

operating_system
               : OPERATING_SYSTEM SBOPEN LETTER SBCLOSE
               ;

statement      : max
               | openinfo
               ;

conv_mode_appls: conv_mode_appl_header conv_mode_appls_body
               ;

conv_mode_appls_header
               : CM_APPLICATIONS SBOPEN NUMBER SBCLOSE
               ;

conv_mode_appls_body
               : BBOPEN cm_appl_list BBCLOSE
               ;

cm_appl__list  :              cm_appl_s
               | cm_appl__list cm_appl_s
               ;

cm_appl_s      : cm_appl cm_attrib
               ;
```

Druck vom 24. 01.2001 17:00:59 torbkomtux


```
ta_attrib      : /* empty */
               | BBOPEN
                 ta_appl_statement_list_opt
                 BBCLOSE
               ;

ta_appl_statement_list_opt
               : /* empty */
               | ta_appl_statement_list
               ;

ta_appl_statement_list
               :                          ta_appl_statement
               | ta_appl_statement_list ta_appl_statement
               ;

ta_appl_statement
               : statement
               ;

cm_attrib      : /* empty */
               | BBOPEN
                 cm_appl_statement_list_opt
                 BBCLOSE
               ;

cm_appl_statement_list_opt
               : /* empty */
               | cm_appl_statement_list
               ;

cm_appl_statement_list
               :                          cm_appl_statement
               | cm_appl_statement_list cm_appl_statement
               ;

cm_appl_statement
               : max
               ;
```

```
openinfo       : OPENINFO SBOPEN LETTER COMMA LETTER SBCLOSE
               | OPENINFO SBOPEN LETTER COMMA APPLICATION SBCLOSE
               | OPENINFO SBOPEN LETTER COMMA LETTER COMMA
                                              LETTER SBCLOSE
               | OPENINFO SBOPEN LETTER COMMA APPLICATION COMMA
                                              LETTER SBCLOSE
               ;

max            : max_header max_list_opt SBCLOSE
               ;

max_header     : MAX_STMT SBOPEN LETTER EQUAL LETTER
               ;

max_list_opt   : /* empty */
               | COMMA max_list
               ;

max_list       :                max_element
               | max_list COMMA max_element
               ;

max_element    : LETTER EQUAL LETTER
               ;

multi_host     :            hostblock
               | multi_host hostblock
               ;

hostblock      : hostblock1 hostblock2 hostblock3
               ;

hostblock1     : HOST SBOPEN LETTER SBCLOSE
                 /* Hostname */
               | HOST SBOPEN LETTER COMMA MASTER SBCLOSE
                 /* Hostname, MASTER */
               | HOST SBOPEN LETTER COMMA BACKUP SBCLOSE
                 /* Hostname, BACKUP */
               ;
```

Druck vom 24. 01.2001 17:00:59 torbkomtux


```
hostblock2      : BBOPEN
                  i_v_a
                  host_statements_opt
                  multi_applicat_opt
                  BBCLOSE
                ;

i_v_a           : /* empty */
                | internet internet_rest
                | vector vector_rest
                | address adddress_rest
                ;

internet_rest   : vector address
                | address vector
                ;

vector_rest     : internet address
                | address internet
                ;

address_rest    : internet vector
                | vector internet
                ;

hostblock3      : /* empty */
                | after_host_statements
                ;

after_host_statements
                :                       after_statement
                | after_host_statements after_statement
                ;

after_statement
                : statement
                | operating_system
                ;

internet        : INTERNETADDRESS SBOPEN INADDRESS SBCLOSE
                | INTERNETADDRESS SBOPEN LETTER SBCLOSE
                ;
```

```
host_statements_opt
                : /* empty */
                | operating_system host_statement_list_opt
                |                  host_statement_list
                ;

host_statement_list_opt
                : /* empty */
                | host_statement_list
                ;

host_statement_list
                : host_statement
                | host_statement_list host_statement
                ;

host_statement  : statement
                | cm_prefix
                | use_cm_applications
                ;

cm_prefix       : CM_PREFIX SBOPEN LETTER SBCLOSE
                :

use_cm_applications
                : USE_CM_APPLICATIONS SBOPEN number_list SBCLOSE
                ;

number_list      :                   NUMBER
                 | number_list COMMA NUMBER
                 ;

multi_applicat_opt
                : /* empty */
                | multi_applicat
                ;

multi_applicat :                applicat
                | multi_applicat applicat
                ;
```

Druck vom 24.01.2001 17:00:59 torbkomtux


```
applicat       : ta_appl ta_attrib    /* server */
               | non_ta_appl          /* client */
               | cm_appl cm_attrib    /* conv. mode */
               ;

ta_appl        : TA_APPLICATION
                 SBOPEN
                 LETTER COMMA
                 NUMBER COMMA
                 NUMBER COMMA
                 LETTER
                 SBCLOSE
                              SBCLOSE
              ;

nonta_appl     : APPLICATION
                 SBOPEN
                 NUMBER COMMA
                 NUMBER COMMA
                 LETTER
                 SBCLOSE
               ;

cm_appl        : CM_APPLICATION
                 SBOPEN
                 LETTER COMMA
                 MINUS NUMBER COMMA
                 NUMBER COMMA
                 LETTER
                 SBCLOSE
               ;

vector         : KEYVECTOR SBOPEN NUMBER COMMA NUMBER
                 SBCLOSE
               ;

address        : PORTADDRESSES SBOPEN NUMBER COMMA
                 NUMBER SBCLOSE
               ;

%%
```

7.3 Revision generation

The purpose of the revision generation is as follows: when the configuration is revised, the generation for the applications not affected by the revision remain the same.

Prerequisites

The use of a repository is an absolute must for this revision generation (see the statement *REPOSITORY* on page 162). The repository must be created when generating the previous version or updated to the new status.

When creating the `ubbconfig` file, the generator `config-tux` must generate unique identifiers for each application. The same identifiers must then be used in a revision generation like in the first generation. In the case of a first generation with an empty repository, a generation number is written to the repository for each application so that this number will be used to generate the identifiers (see the statement *START_VALUE* on page 163).

Performing the revision generation

The user must perform the following steps when revising the configuration so that no user data is lost from an application:

– Adapt the configuration file
  The modifications to the configuration, e.g. new hosts and new applications must be defined in the input file of the configuration generator `config-tux`.

– Perform the generation
  The generator reads the configuration file and generates, among other things, the generation file `ubbconfig` and shell scripts.

The following steps must be performed on all hosts affected by the modifications:

– Terminate the application normally
  The application must be terminated using the command `tmshutdown -y` on the MASTER host.

– Execute scripts

– Restart the applications

Druck vom 24. 01.2001 17:01:00 torbkomtux

7.4 Sample configuration file

This section illustrates the syntactic structure using a sample configuration.

```
SYSTEM
{
    //
    // system-wide MAX statements.
    //
   MAX( "IPCKEY" = "65024" )
   MAX( "APPDIR" = "/tmp/appdir" )
   MAX( "ENVDIR" = "/tmp/appdir/envdir" )
   MAX( "GINACONFIGDIR" = "/tmp/appdir/ginaconfigdir" )
   MAX( "TUXCONFIG" = "/tmp/tuxconfig/TUXCONFIG" )
   MAX( "TUXDIR" = "/opt/TUXEDO" )
   MAX( "QUEUEFILE" = "/tmp/queuefile/QUE" )
   MAX( "MIN" = "2" )
   MAX( "MESSAGES" = "111" )



    //
    // first operating system of the system.
    // in this example the hostname is
    // used as symbolic name.
    //
    HOST("kotw002", BACKUP)
    {
        //
        // internetaddress of first host.
        //
        INTERNETADDRESS(192.200.94.8)

        //
        // available shared memory and semaphore keys.
        //
        KEYVECTOR(5011,5047)

        //
        // available port addresses.
        //
        PORTADDRESSES(10111,10126)
```

```
            //
            // first application of first operating system.
            // a1 is the utm known application name.
            // buslay is the user friendly name.
            //
            TA_APPLICATION("a1",1,1,"buslay")
            {

            }

            TA_APPLICATION("a2",1,2,"servlay")
        }

        HOST("kotw005")
        {
            INTERNETADDRESS(192.200.94.4)
            KEYVECTOR(5000,5015)
            PORTADDRESSES(10114,10135)

            TA_APPLICATION("a1",1,3,"nwlay")

            TA_APPLICATION("a2",1,4,"nellay")

            APPLICATION(1,5,"client1")

            APPLICATION(1,6,"client2")

            APPLICATION(1,7,"client3")
        }
    }
```

Druck vom 24. 01.2001 17:01:00 torbkomtux

7.5 Call and options

The configuration generator `config-tux` is called as follows:

`config-tux [-s] [-u] [-v] [-V] configfile`

`-s` The `-s` option outputs the addressing and name server data in a compact format.

`-u` The `-u` option outputs the call syntax.

`-v` With the `verbose` option, generator messages are output to `stdout`.

`-V` If `-V` is specified, only the version message is output to `stdout` and execution of the program is terminated.

For *`configfile`*, you must specify the name of your configuration file.

7.6 Generated files

The configuration generator `config-tux` analyzes a description file and generates from it resource files and configuration scripts for the runtime environment of the distributed system. Some of the generated files are host-specific and some are application-specific. `config-tux` creates a directory tree in the current directory. The basic structure of this tree is illustrated in Figure 36.

There is a subdirectory for each host under the root `system`. Each of these host subdirectories bears as its name the descriptive name from the `HOST` statement. All of these subdirectories also have their own subdirectories bearing the names of the applications which are to run on the respective hosts.

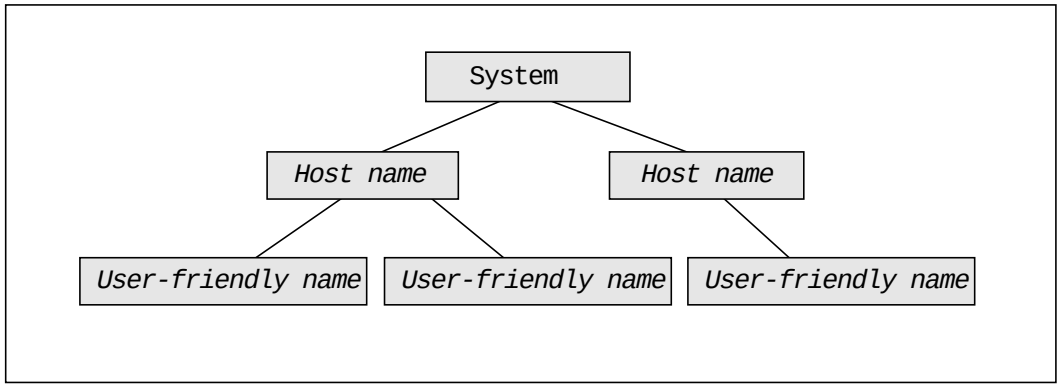


Figure 36 Model of the directory structure as created by `config`

The files created by `config-tux` are incorporated directly into the runtime environment of the distributed application.

The scripts are generated in accordance with the operating system of the host on which the relevant T-ORB application is to run and must be executed under this operating system.

– The configuration file `ubbconfig` and the scripts `crbincf` and `crtlogs` are created for the MASTER host, i. e. the shell `/bin/sh` must exist on the relevant UNIX host (suffix `.sh`) or the command line interpreter must exist on the WindowsNT host (suffix `.cmd`).

– The script `crdevqu` is created for the other server hosts.

– Batch processing files `*.cmd` are generated for the WindowsNT operating system.

Certain conditions must be satisfied before the scripts are executed:

– The directories and files specified using the parameters `APPDIR`, `ENVDIR`, `GINACONFIGDIR`, `TLOGDEVICE`, `TUXCONFIG` and `QUEUEFILE` must be set up on all server hosts.

Druck vom 24.01.2001 17:01:00 torbkomtux

– The BEA TUXEDO daemon process `tlisten` must be started on all server hosts.

– The scripts must be executed in the order `crbincf` (MASTER), `crdevqu` (MASTER and then on all server hosts) and `crtlogs` (MASTER). Make sure that each one executes correctly.

7.6.1 Generated files for UNIX hosts

`readme`

The `readme` file is generated in the `system` directory. This file contains a list of the server host names with the related port numbers of the BEA TUXEDO system processes `tlisten` and `WSL`.

`gina.config`

- Generation *without* the `-s` option

  A file `gina.config` is generated for each T-ORB application (`TA_APPLICATION`) and T-ORB/client application (`APPLICATION`). It acts as a directory for addressable applications.

  For T-ORB applications and T-ORB clients, this file is generated in the directory assigned to the application.

- Generation *with* the `-s` option

  The file `gina.config` is the same for all server hosts and is therefore created once in the `system` directory.

  This file is not created for T-ORB applications and T-ORB clients.

`<user_friendly_name>.evf`

- Generation *without* the `-s` option

  For T-ORB applications, this file is generated in the directory assigned to the application. It contains the definition of the `GINACONFIG` environment variable.

  `GINACONFIG=<gina_config_path>/<user_friendly_name>`

  The application reads this file after it starts and sets the environment variables.

- Generation *with* the `-s` option

  The file is created with the name `gina.evf` in the host directory.

ubbconfig

The ubbconfig configuration file is generated in the directory assigned to the MASTER host. It contains configuration data for the BEA TUXEDO transaction monitor.

crbincf.sh

The script crbincf.sh calls the BEA TUXEDO utility routine tmloadcf and should only be executed on the MASTER host.

crdevqu.sh

The script crdevqu.sh is called with one parameter. The values which the parameter can assume are -y or -d or -q or -t, where -y contains the values -d, -q and -t.

If you specify -d, the BEA TUXEDO utility routine tmadmin (function crdl) will be called, if you specify -q, qmadmin (functions crdl, qspacecreate and qcreate) will be called and if you specify -t, buildtms will be called. The script must be executed on all server hosts, first on the MASTER host, then on the other server hosts. The files needed for queues and transaction logs are generated during execution of the script.

crtlogs.sh

The script crtlogs.sh calls the BEA TUXEDO utility routine tmadmin and is only to be executed on the MASTER host. The log files for backing up the transactions (function crlog) for all server hosts are set up during execution of the script.

Druck vom 24. 01.2001 17:01.01 torbkomtux

7.6.2 Generated files for WindowsNT hosts

readme

The readme file is generated in the system directory. This file contains a list of the server host names with the related port numbers of the BEA TUXEDO system processes tlisten and WSL.

gina.config

- Generation *without* the -s option

A file gina.config is generated for each T-ORB application (TA_APPLICATION) and T-ORB/client application (APPLICATION). It acts as a directory for addressable applications.

For T-ORB applications and T-ORB clients, this file is generated in the directory assigned to the application.

- Generation *with* the -s option

  The file gina.config is the same for all server hosts and is therefore created once in the system directory.

  This file is not created for T-ORB applications and T-ORB clients.

<user_friendly_name>.evf

- Generation *without* the -s option

  For T-ORB applications, this file is generated in the directory assigned to the application. It contains the definition of the GINACONFIG environment variable.

  GINACONFIG=<gina_config_path>/<user_friendly_name>

  The application reads this file after it starts and sets the environment variables.

- Generation *with* the -s option

  The file is created with the name gina.evf in the host directory.

ubbconfig

The ubbconfig configuration file is generated in the directory assigned to the MASTER host. It contains configuration data for the BEA TUXEDO transaction monitor.

crbincf.cmd

The script crbincf.cmd calls the BEA TUXEDO utility routine tmloadcf and should only be executed on the MASTER host.

crdevqu.cmd

The script crdevqu.cmd calls the BEA TUXEDO utility routines tmadmin (function crdl), qmadmin (functions crdl, qspacecreate and qcreate) and buildtms. The script must be executed on all server hosts, first on the MASTER host, then on the other server hosts. The files needed for queues and transaction logs are generated during execution of the scripts. The WindowsNT version of this script does not need any parameters, i. e. -y is the default value here.

crtlogs.cmd

The script crtlogs.cmd calls the BEA-TUXEDO utility routine tmadmin and is only to be executed on the MASTER host. The log areas for backing up the transactions (function crlog) for all server hosts are set up during execution of the script.

The three scripts crbincf.cmd, crdevqu.cmd and crtlogs.cmd must be executed in this order.

7.6.3 Example

The file structures illustrated in Figure 37 and 38 is an example of the structure created for the sample configuration file on page 175.

Druck vom 24. 01.2001 17:01.01 torbkomtux

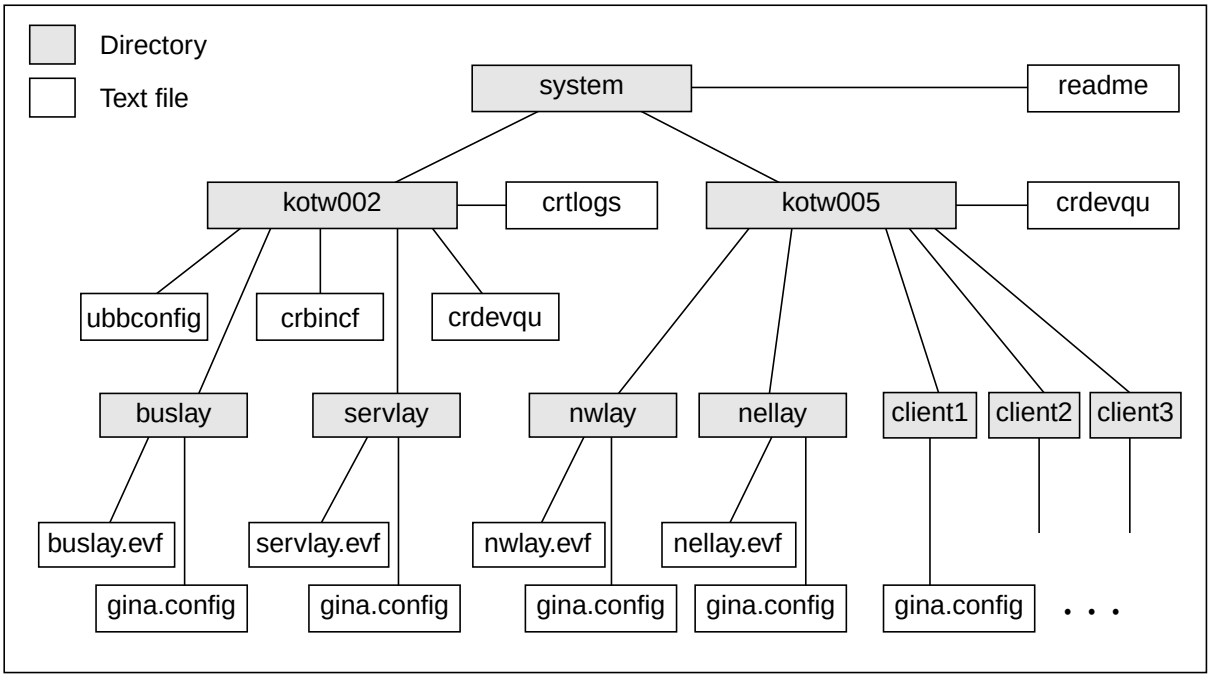


Figure 37 File structure for the example of generation without the `-s` option

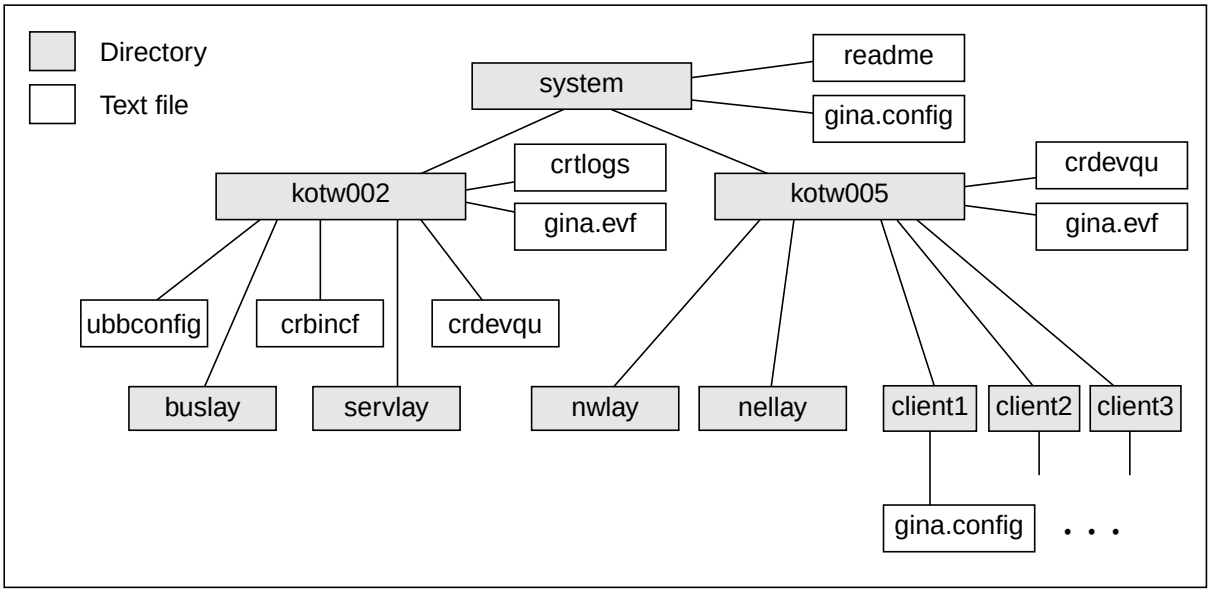


Figure 38 File structure for the example of generation with the `-s` option

7.7 BEA TUXEDO domains

A BEA TUXEDO domain refers to an administrative unit, a BEA TUXEDO application, which is independent of other BEA TUXEDO applications. These independent domains can, however, communicate with each other using gateways, i. e. services which offer and export a domain can be imported into another domain and then called.

A BEA TUXEDO application (domain) refers to a system where all BEA TUXEDO services made known through the configuration are managed from a central location. Several machines can be part of a single BEA TUXEDO application.

Domains allow one large BEA TUXEDO application to be split up into smaller, manageable parts.

The information you must enter can be divided into four parts (see Figure 35 on page 156):

– specifications on system-wide settings, such as the participating domains
– specifications on domain-wide settings, such as the participating hosts
– specifications on host-specific parameters, such as operating system resources
– specifications on application-specific parameters, such as the number of work processes

Practically speaking, some of the parameters can be specified in more than one of the four areas. In such cases, the values set higher up in the hierarchy must be taken as default settings. These values are then overwritten by the settings in more specific areas. The desired settings can thus be specified separately for each application if required (customizing).

If parameters are specified more than once at system level, they are then valid for the rest of the system level or until a new parameter is specified.

Figure 35 on page 156 is a symbolic representation of the hierarchies in the definition of the system. It is not drawn to scale to reflect the scope of the individual sub-descriptions.

Druck vom 24.01.2001 17:01.01 torbkomtux

7.7.1 Statements

DOMAIN

The DOMAIN statement defines the name of the domain and describes its configuration.

– domain name, max. 30 characters

– domain-specific customizing statements
(MAX, OPENINFO)

– description of the host

Example

```
DOMAIN ( "Domain1" )
{
START_VLAUE(
MAX ...
HOST ("Host1",MASTER)
  {
  ...
  }
...
}
```

EXPORT

The keyword EXPORT attributes a TA_APPLICATION (...,EXPORT). This application will then be exported by the domain. Hosts can also be attributed in this way:
HOST (...,EXPORT). All server applications will then export this host.

IMPORT

The optional statement IMPORT in the current domain (DOMAIN) imports server applications from other domains. The applications are defined by the numeric pair OsId and LayerId. The user-friendly name of the application or a host name can also be used instead of the numeric pair. If a host name is specified, all exported server applications of this host will be imported.

Example

```
IMPORT (( 50, 55 ), (47, 11), ... )
IMPORT (UFN = "userfn", HOST = "Host2", ... )
```

7.7.2 Syntax

```
sysblock        : SYSTEM
                  BBOPEN
                  sys_statement_list_opt
                  system_body
                  BBCLOSE
                ;

system_body     : multi_host
                | multi_domain
                ;

multi_domain    :             domain_block
                | multi_domain domain_block
                ;

domain_block    : domain_block1 domain_block2 domain_block3
                ;

domain_block1   : DOMAIN SBOPEN LETTER SBCLOSE
                ;

domain_block2   : BBOPEN domain_statement_list multi_host BBCLOSE
                ;

domain_block3   : /* empty */
                | after_domain_statement_list
                ;

domain_statement_list
                :                       domain_statement
                | domain_statement_list domain_statement
                ;

domain_statement
                : statement
                | operating_system
                | start_value
                | import
                | conv_mode_appls
                ;
```

Druck vom 24.01.2001 17:01:02 torbkomtux


```
after_domain_statement_list
                : after_domain_statement
                | after_domain_statement_list after_domain_statement
                ;

after_domain_statement
                : statement
                | operating_system
                ;

import          : IMPORT SBOPEN import_list SBCLOSE
                ;

import_list     :                   import_element
                | import_list COMMA import_element
                ;

import_element : SBOPEN NUMBER COMMA NUMBER SBCLOSE
                | HOST = LETTER
                | UFN = LETTER
                | LETTER /* user friendly name */
                ;
```

7.7.3 Example of a configuration file with domains

```
//
// Configuration file for the DOMSTEST for NT
// File:         %M%
// Version:      %I% %E%
// Description:  source file for config-tux
// Changes:      Adapted to GINA on TUXEDO
//               Adapted to Windows NT
//

SYSTEM
{
    REPOSITORY("repository")
    OPERATING_SYSTEM("OS_WINNT")
    MAX("IPCKEY"="0x0e011")
    MAX("TUXDIR"="c:\tuxedo")
    MAX("TUXCONFIG"="c:\domaintest\bin\tuxconfig")
    MAX("BDMCONFIG"="c:\domaintest\bin\bdmconfig")
    MAX("TLOGDEVICE"="c:\domaintest\bin\TLOG")
    MAX("QUEUEFILE"="c:\domaintest\bin\QUE")
    MAX("APPDIR"="c:\domaintest\bin")
    MAX("GINACONFIGDIR"="c:\domaintest\bin")
    MAX("ENVDIR"="c:\domaintest\bin")
    MAX("MAX"="10")
    MAX("REQUESTQUEUE"="Y")
    DOMAIN("dom_nt")
    {
      MAX("IPCKEY"="0x0e011")
      START_VALUE(125)
      IMPORT((2,1),(3,1))
      HOST("M19041PP",MASTER)
      {
          INTERNETADDRESS("m19041pp")
          KEYVECTOR(6210,6299)
          PORTADDRESSES(4049,4055)
          OPERATING_SYSTEM("OS_WINNT")
          TA_APPLICATION("DT_OS",1,1,"dt_os1",export)
          {
              MAX("MIN"="2")
          }
```

Druck vom 24.01.2001 17:01:02 torbkomtux

```
            APPLICATION(1,9,"client1")
            APPLICATION(2,9,"client2")
            APPLICATION(3,9,"client3")
        }
    }

    DOMAIN("dom_sun")
    {
        MAX("IPCKEY"="0x0e011")
        START_VALUE(250)
        IMPORT((1,1))
        MAX("GINACONFIGDIR"="/tmp/domaintest/bin")
        MAX("ENVDIR"="/tmp/domaintest/bin")

        HOST("m61930pp",MASTER)
        {
            INTERNETADDRESS("m61930pp")
            KEYVECTOR(6210,6299)
            PORTADDRESSES(4049,4055)
            OPERATING_SYSTEM("OS_UNIX")

            MAX("TUXDIR"="/opt/TUXEDO")
            MAX("TUXCONFIG"="/tmp/domaintest/bin/tuxconfig")
            MAX("BDMCONFIG"="/tmp/domaintest/bin/bdmconfig")
            MAX("TLOGDEVICE"="/tmp/domaintest/bin/TLOG")
            MAX("QUEUEFILE"="/tmp/domaintest/bin/QUE")
            MAX("APPDIR"="/tmp/domaintest/bin")

            TA_APPLICATION("DT_OS",2,1,"dt_os2",export)
            {
                MAX("MIN"="2")
            }
            TA_APPLICATION("DT_OS",3,1,"dt_os3",export)
            {
                MAX("MIN"="2")
            }
            APPLICATION(1,9,"client1")
            APPLICATION(2,9,"client2")
            APPLICATION(3,9,"client3")
        }
    }

}
```

7.7.4 Generated files

dmconfig

The configuration file `dmconfig` is generated in the directory assigned to the MASTER host. This file contains configuration data for the BEA TUXEDO domain.

The scripts `crbincf.sh` and `crbincf.cmd` call the BEA TUXEDO utility routine `dmloadcf`.

Figure 39 illustrates the file structure for the sample configuration file on page 188.

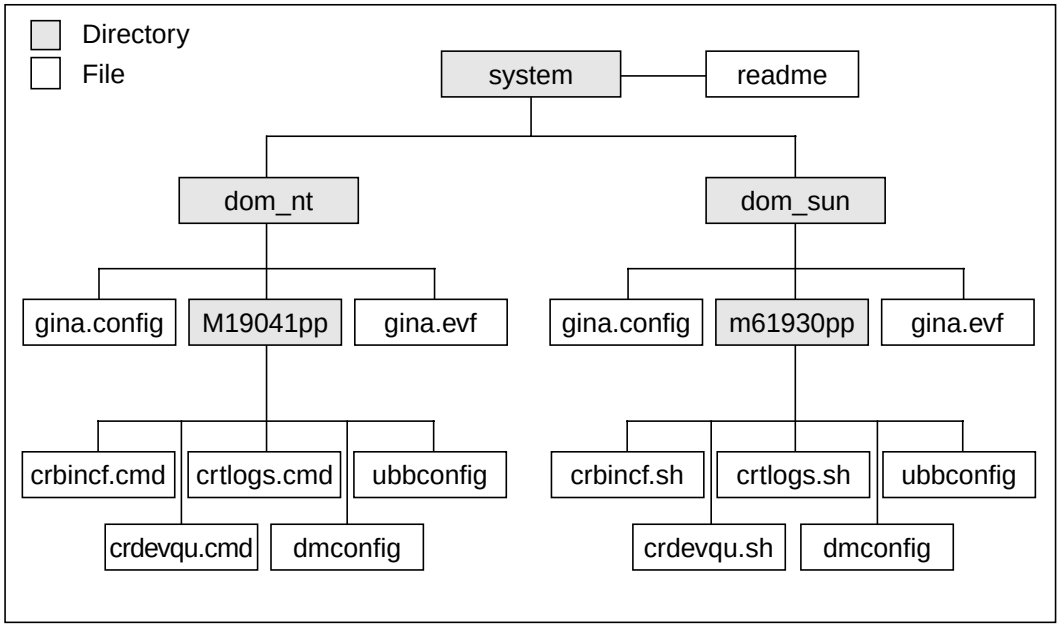


Figure 39 File structure for the example of a configuration file with domains (generation with the `-s` option)

Druck vom 24.01.2001 17:01:03 torbkomtux

7.7.5 Special points

CM_APPLICATIONS

The CM_APPLICATIONS statement defines a list of conversational mode applications at *domain level*.

MAX

The MAX statement with the name BDMCONFIG (64) must be defined for the MASTER host. The BDMCONFIG environment variable is used by BEA TUXEDO to locate the file BDMCONFIG (binary).

REPOSITORY

If a REPOSITORY statement was specified at system level, then a separate repository file with the name

```
< String from the repository statement > . < domain name > .rep
```

will be created for each domain. The CM applications will be stored in the repository with the name

```
< String from the repository statement >  .rep
```

If these repositories are present when a new generation operation is carried out, they are read in prior to analysis of the input and recreated as part of a successful generation.

START_VALUE

The START_VALUE statement is a mandatory statement for each domain. A different value must be specified for each domain to facilitate the import and/or export of applications, i. e. the difference must exceed 99. The START_VALUE statement at system level defines a generation number which is used when generating all conversational mode applications.

Call and options

The following options are supported in connection with the generation of domains:

`-i repository-file`
: Imports the repository file. The repository file describes a domain from which server applications are imported. This domain is not part of the input file.


GINA

Druck vom 24. 01.2001 17:01:03 sys-adm

8 Operating GINA applications

This chapter provides an introduction to the runtime administration of configured GINA applications.

8.1 Communication administration

8.1.1 Communication structure of a server application

Communication in GINA is based on the *open*UTM transaction monitor. Its functionality with respect to transaction-monitored message interchange and restart is thus inherited in the event of error. With the support of the XA interface, data storage is also integrated in this transaction bracket. Knowledge of the administration and diagnostic options offered by UTM is required for the successful operation of (networked) GINA applications.

A precise description of the concepts involved can be found in the Developer Manual [13]. The description of the network in the configuration generator determines which partners can communicate with each other and in what way (chapter 6 on page 43).

8.1.2 Communication structure of a client application

Unlike the transaction-monitored communication between server applications, client applications (generally as GUI applications) connect to server applications using non-transaction-monitored protocols. Here too, knowledge is required of the underlying *open*UTM-Client product and the diagnostic tools. A precise description can be found in the corresponding manuals on *open*UTM and *open*UTM-Client or in the communication protocols used by *open*UTM [29], [31].

8.2 DB administration

The database system INFORMIX Dynamic Server 2000 Version 9.2 is used for data storage within the Persistency Service. This results in a range of tasks for DB administration which can be completed with the aid of appropriate utilities. As already described, the transaction monitor and the database are synchronized by the XA protocol. It must be ensured that this link is supported in the current installation.

The main tasks in operating the database lie in the area of precautionary backup and tuning in the distribution of data on the available storage media.

In this case too, knowledge is required of the respective service functions in the database system, which are described in the relevant manuals.

8.2.1 Security management

The database system allows you to log events in an audit procedure in order to obtain information on the activities of users or user groups, particularly in relation to attempts to acquire rights for themselves or others. The ID of the database administrator (`informix`) should always be monitored by the audit function. In Version 9.2 of INFORMIX Dynamic Server 2000, the functionality of the database administrator can also be distributed over a number of roles in order to improve the separation of administration and security functions. The actual steps involved are described in the relevant manuals [21]. Moreover, access rights to database objects (tables, procedures) can be assigned on the level of the GRANT and REVOKE function. A precise description can be found in [16].

8.2.2 Data backup

This aspect is concerned with the use of backed up data to fully or partially recover a database which can no longer be used due to the failure of a disk drive, for example. This type of backup archive is created and subsequently recovered with the aid of utilities which are provided by the database system.

The task of the database administrator is to minimize the effort involved, whilst working on the basis of restoring a database securely. The backup can be installed as an automatic process, whereby utilities of the INFORMIX database system, such as `onarchive`, can be used in conjunction with `onautovop`.[1)]

An important function in increasing availability lies in the use of RAID memory or in the utilization of the database function to mirror disk areas (chunks).

1) The `dbexport` and `dbimport` commands from INFORMIX-OnLine do not back up the full information on the status of the XA interface.
You should therefore never use these commands to back up data.

Druck vom 24. 01.2001 17:01:04 sys-adm

Should a recovery procedure be required, manual interception to diagnose the error on the one hand and rectify it on the other (replace the faulty component) is generally necessary. Only then can the data recovery be performed from the backup archives using the appropriate programs (on`archive`). This process can take a long time, during which the database can only be partially used or not at all. If this cannot be tolerated in reality (error-protected 24-hour operation), additional replica databases can be installed on a secondary system, with the option of automatically switching over to the replica in the event of error. Of course, this also means that appropriate access possibilities (connections) are required to communicate with this secondary system.

A detailed description of these topics can be found in the appropriate manuals for the database system [19], [20], [21].

8.2.3 Logging database errors

GINA allows you to log INFORMIX errors in a file for the purpose of debugging during the development phase.

If the `GINA_DBPROT` environment variable is set when the application is started, database errors at SQL level are logged to the file specified by the value in this variable. In the case of errors in filter iterators, the relevant select statement is output before the INFORMIX error message.

`GINA_DBPROT` must contain a valid file name. The file is created automatically if it does not yet exist. If the file already exists, new error messages are simply appended to the existing file.

8.3 Starting and stopping GINA applications

8.3.1 Environment variables

Please note that certain environment variables must be set so that GINA applications can run. These variables are explained in section 3.3.5 on page 21.

8.3.2 Transaction-monitored applications

A GINA application which is based on the T-ORB service is made available by a start process. This means that authorized partners can connect to the application following a successful start in order to use its functions. The start process itself is implemented by the shell script `utmstart.multi`. The script is created during the production process of an application with the aid of the configuration generator. Setting the `-r` option creates the `kdcdf` script, whose execution then results in the creation of the start script.

An application is terminated using the administration (see section 8.4 on page 199).

Before a GINA application is started, the database instance with which the application works must be available. With INFORMIX, this can be achieved using the interactive program `onmonitor` or the program `oninit`. These programs and the necessary environment variables are described in [21].

8.3.3 Non-transaction-monitored applications

When using a database with NLS functionality, you must ensure that the appropriate environment variables (`DBNLS`, `LANG`) are set. A description can be found in [18].

Event Handler

The `DomsEventHandler` (contained in the `bin` directory of the GINA installation) must be available as a daemon process/NT service for all machines running non-transaction-monitored applications. We recommend that this daemon process be started in the boot script.

The `DomsEventHandler` is required for local communication to the T-ORB/Client application; the daemon process informs this application that a message has arrived at the GINA application. A description of the interoperation of the participating partners can be found in the Developer Manual [13]. Communication via the Event Handler is based on the TCP/IP protocol and thus requires one of corresponding entries in the system file

Druck vom 24. 01.2001 17:01:04 sys-adm

UNIX	`/etc/services`
NIS	`/etc/yp/services`
WindowsNT	`%WINDIR%\system32\drivers\etc\services`

The entry comprises the following parts:

Name	domsehd
Port#	free, nonprivileged port number
Type	`tcp`

Example

The following entry would be made for port 7419:

```
domsehd   7419/tcp   # DomsEventHandler
```

Dynamic Connection Handler

The `DomsDynConnectHandler` (resides in the `bin` directory of the GINA installation directory) must be available as a daemon process for all machines on which dynamic T-ORB/client applications (type 7, 8) run.

The Dynamic Connection Handler needs the file `gina.dynamic` in order to run. By default it expects this file to be in the directory in which it (the Dynamic Connection Handler) is started. The environment variable `GINA_DYNAMIC` can be used to specify the name of a different directory in which the file `gina.dynamic` is to be read.

Communication with the `DomsDynConnectHandler` is based on the TCP/IP protocol and therefore requires an appropriate entry in the system file `/etc/services` or in `/etc/yp/services` when using `NIS`.

The entry comprises the following elements:

Name	domschd
Port#	Free, non-privileged port number
Type	`tcp`

Example

The following entry would be required for the port 7420:

```
domschd   7420/tcp   # DomsDynConnectHandler
```

Event Handler and Dynamic Connection Handler under WindowsNT

The `DomsEventHandler` and the `DomsDynConnectHandler` are installed under WindowsNT as services which are automatically started when the system starts. The port number must be entered as described above in the file:

`C:\WINNT\system32\drivers\etc\Services`

Make sure that the lines end with an end of line character.

Dynamic T-ORB/clients

Dynamic clients need the files `gina.config` and `upicfile` to run (see section 6.5 on page 94). By default, a dynamic client looks for these files in the directory in which it is started.

The environment variables `GINACONFIG` and `UPICPATH` can be used to select a directory other than "**.**" for the file `gina.config` or `upicfile`.

Druck vom 24. 01.2001 17:01:04 sys-adm

8.4 Administering GINA applications

8.4.1 TP monitor

While an application is active, the administrator can decide to query information on the status and load or to interrupt the sequence. He/she uses the `dtp` script that is also created by the configuration generator to perform these activities. The administrator then gets a local dialog interface to the application by means of which he/she can execute control functions, for example the controlled termination of the application as well as the attachment and reattachment of resources. In this context, it is important to consider the possibility of customizing applications to changing environments, such as the connection of new partners or new functions.

A description of the complete range of functions can be found in the *open*UTM administration User Guide [27].

8.4.2 Cyclical timer

When a transaction-monitored application is started, T-ORB activates a cyclical timer that activates monitoring of the alarms (cf. section 8.4.3 on page 200) at specific intervals. The interval is specified by the `CYCLICTIME(d,h,m,s)` parameter of the T-ORB generator `config` when the configuration is being created. The T-ORB generator `config` generates the `gina.config` file for each transaction-monitored application in which the interval for the cyclical timer is stored (see Figure 40).

```
CTL1 0 0 10 0    // corresponds to CYCLICTIME(0,0,10,0)
```

Figure 40 Time interval of the cyclical timer in `gina.config`

If the interval is to be changed while the transaction-monitored application is running, the `gina.config` file must be modified using a suitable editor (e.g. `vi`) and then the `DNEWCYCA` administration command activated using the administration facility (see section 8.4.1 on page 199).

8.4.3 Monitoring alarms

The delivery of an alarm to a non-transaction-monitored application is broken up into two parts. The partner is first informed of the event “Message n exists” via the auxiliary process EventHandler, and then the message is collected.

The termination of the non-transaction-monitored partner or the EventHandler, for example as a result of a power failure, can result in alarms being lost during delivery.

To prevent this, T-ORB provides a mechanism that monitors the delivery of alarms to non-transaction-monitored partners.

T-ORB checks at certain intervals whether alarms that were announced to the partner via the EventHandler have actually been collected. If this is not the case, T-ORB announces them again.

The time intervals are defined by the `CYCLE(d,h,m,s)`, `CHECK(d,h,m,s)` and `CANCEL(d,h,m,s)` parameters in the `EVENTCONTROL` statement of the T-ORB generator `config` when the configuration is being created. For each transaction-monitored application, the T-ORB generator `config` generates the `gina.config` file, in which the intervals for the monitoring of alarms are stored (see Figure 41).

```
CTL2 0 0 10 0 100    // corresponds to CHECK(0,0,10,0)
CTL3 0 1 0 0         // corresponds to CHECK(0,1,0,0)
CTL4 2 0 0 0         // corresponds to CANCEL(2,0,0,0)
```

Figure 41 Time intervals for the monitoring of alarms

If these time intervals are to be changed while the transaction-monitored application is running, the `gina.config` file must be modified using a suitable editor (e.g. `vi`) and then the `DNEWCYCA` administration command activated using the administration facility (cf. section 8.4.1 on page 199).

Druck vom 24. 01.2001 17:01:05 sys-adm

8.4.4 Cyclical tasks

The number of cyclical tasks that a transaction-monitored application can place at one time is limited. It is defined by the `CYCLICORDER(n)` parameter of the T-ORB generator `config` when the configuration is being created. For each transaction-monitored application, the T-ORB generator `config` generates the `gina.config` file, in which the number of cyclical tasks (cf. Figure 42) is stored.

```
COL 10            // corresponds to CYCLICORDER(10)
```

Figure 42 Number of cyclical tasks

If this number is to be changed while the transaction-monitored application is running, the `gina.config` file must be modified using a suitable editor (e.g. `vi`) and then the `DNEWORDA` administration command activated using the administration facility.

Druck vom 24. 01.2001 17:01:05 fachwort

Glossary

action point

A number of remote subcalls can be started in the T-ORB procedure “action point”. Given that the →server process is subsequently released, an action point is always a potential →end of transaction, a potential →monitoring and restart point.

The results of all subcalls are evaluated by the `Continuation` of the action point. The continuation is a parallel callback which is only started when all the results have been received.

A →client cannot recognize whether the →server is carrying out an action point. A server cannot recognize whether it was called from an action point.

agent

see →server

annotation

The parsers of the GINA development systems sometimes need more information than can be formulated in C++. GINA comments are used for this purpose. They are ignored by the compiler but not by the GINA parsers. These comments contain a sequence of expressions that are referred to as annotations.

application

A GINA application is a transaction-monitored application under T-ORB. A GINA application can include a number of →server processes. A non-transaction-monitored application can be connected to a transaction-monitored application using T-ORB/Client.

asynchronous request/call

When you use the T-ORB procedure “asynchronous call”, the →client does not wait for the call to be executed and the result returned. Possible results are thus discarded (see also →notification).

Under T-ORB, even asynchronous requests/calls undergo transaction monitoring, i.e. they are executed in an independent transaction *after* a successful end of transaction (commit). The asynchronous call under T-ORB is a relative →time request with zero delay.

callback

A procedure is called when an event occurs in an event-monitored environment (e.g. in an X application). This procedure is called a callback function, or simply only callback.

class method

A class method is a method which is used on the class, rather than on an object. In C++ we also talk about `static` methods.

client

In a client/server system, a client makes requests/calls to a →server. It requests a →service from the server. A client can also be the server of a client.

In addition to transaction-monitored clients, non-transaction-monitored clients, e.g. X applications, can also be connected under T-ORB using T-ORB/Client.

client stub

In DCE terminology, a client stub is the same as a →stub under GINA.

commit

see →end of transaction

cursor

A cursor can be used to scroll through a list or set one element at a time. An important special case is a database cursor, which is used to scroll through the result of a database query.

customizing

This describes the feature which allows the user to carry out individual customizations.

The following mechanisms are available in the GINA environment:

→Persistency Service
In the persistency framework, the customizing concept allows you to embed external classes from a class library as complete modules. Customizing to specific DBMS is also possible; these special features offered can be exploited.

→T-ORB
The user-specific generation of the runtime environment in the T-ORB framework using a special description language, whereby the description of the environment is transformed accordingly using a generator.

Druck vom 24. 01.2001 17:01:05 fachwort

data link point

A data link point is an →end of a transaction using commit. Transaction-monitored changes become visible and permanent outside the transaction at a data link point (see ACID property of →transactions).

data reference

A data reference `DREF(P)` is a special →global reference. It cannot be dereferenced. A data reference can however be used as an object, a data member, an argument or a result.

distributed transaction

Distributed processing is when a →transaction covers a number of processes, generally on a number of machines. In this case, a number of →local transactions from different →resource managers are grouped together by a →transaction monitor to form a →global transaction.

end of transaction

At the end of a transaction, the entire operation must either be confirmed (commit) or reversed (rollback). A (local) end of transaction occurs at the end of a transaction-monitored call or at an →action point, which is a →data link point.

In →distributed transaction processing, all the local end of transactions only come into effect when a global end of transaction occurs. Since the →server process is automatically released under T-ORB when an end of transaction occurs, the end of transaction is not only a →data link point for the database, but also a →restart point for the operation.

exception handling

In addition to indicating processing errors using special return codes in the case of method calls, exception situations in GINA are indicated by initiating exceptions. This procedure is used in particular when resource bottlenecks occur.

external interface

A method in the external interface of a →server is called using a remote request/call. The external interface method decodes the parameters, calls the technical operation and encodes the results which are sent back to the →client. In DCE we call this a server stub, in CORBA we talk about a skeleton.

framework

A framework refers to a general framework in which the application constructor adds application-specific code as modules.

general reference

A general reference `GREF(P)` is a →global reference. The →smart pointer `GREF(P)` can refer to a local →persistent object or a stub object. A method call is handled accordingly using either a local or a →remote call.

global interface

The global interface of a class is the subset of the methods of the class which can also be called →remotely.

global reference

A global reference is used in GINA to address an object, which is (possibly) remote. There are different types of global references – pure →data references or →smart pointers in the form of →remote references or →general references. A global reference can be reduced to a local Persistency Service reference on the target application.

global transaction

In a →distributed transaction, a →transaction monitor groups together a number of →local transactions to form a global transaction.

When the end of the global transaction is reached, either all the local transactions are rolled back, or all the local transactions are terminated with a commit. The local transactions do not therefore behave like nested transactions.

inheritance

In object-oriented languages, a derived class inherits the interface of its base classes. In C++, the implementation of the base classes is also inherited together with the interface.

instance method

An instance method is a method which is used on a certain instance, i.e. an object, of a class. These are also called object methods or simply methods.

instantiation

In object-oriented database systems, an object is generally accessed directly using a reference. In Persistency Service, the persistent object is instantiated when it is first accessed, i.e. it is loaded into the process address space.

job submitter

see →client

Druck vom 24. 01.2001 17:01:05 fachwort

job receiver

see →server

join

We talk about a join when two or more database tables (i.e. persistent classes) are linked for a database query.

lazy evaluation

Lazy evaluation is when an expression can only ever be evaluated to the extent that is required, as dictated by its use.

In connection with Persistency Service, the procedures used to access persistent objects follow a lazy evaluation strategy. When an object is accessed, the directly dereferenced object is loaded, rather than the transitive shell of all the references subobjects.

local transaction

A transaction which is restricted to one process is called a local transaction. A →distributed transaction includes all the local transactions.

navigating access

In object-oriented database systems, an object is generally accessed directly using a reference. If this is done on a number of levels, it is called navigating access.

notification

In a hierarchical client/server architecture, only one client can instruct one server. However, in some cases, e.g. for alarms, the reverse information flow is required. In these cases, the server sends a notification to the client. Technically, this is done using an →asynchronous call.

Even the results of →asynchronous calls and →time requests can be presented to the client in the form of a notification. This must be modeled technically.

object-oriented transaction

The transaction principle in GINA is object-oriented. This means that, in contrast to the conventional transaction concept, both the →persistency of objects and communication between objects in the form of calls using the T-ORB undergo transaction monitoring.

Persistency Service

The Persistency Service provides the user programs with an object-oriented view of the related database system.

persistent

A persistent object, unlike a →transient object, has a life span which does not depend on a process. In contrast to this, the life span of a transient object is limited by the life span of its process.

polymorphy

In object-oriented languages, a derived class inherits the interface of its base class(es). Polymorphy is when an object of the derived class can be used instead of an object of a base class.

In C++ we use C++ references and C++ pointers here. A pointer to a base class can refer to an object of a derived class. In order to call a method of the derived class in this case, it must already be declared as virtual in the base class. The method is then linked dynamically at runtime, depending on the current object (*dynamic binding*).

proxy

see →stub

reference

1. An alias for an object is called a reference in C++. If you access a reference, you actually access the referenced object. The reference is thus used in the same way as an object, from the syntactic point of view.
2. The ODMG uses the term reference to refer to →smart pointers, which are used to access a persistent object by dereferencing it. In C++, a reference of this kind, like a pointer, is dereferenced using the operators `->` and `*`. Persistency Service uses local references *P*`Mibs::MibsRef` to access local persistent objects, and T-ORB uses →global references to address remote objects.

referential integrity

Referential integrity relates to the relations between objects. With the exception of the →0 reference, an object can only contain references to an existing object. An object which is deleted can no longer be referenced by another object. Persistency Service and Database systems can monitor referential integrity.

remote call

A remote call, i.e. a call between two processes (generally on different machines) is implemented by sending and receiving messages.

Druck vom 24. 01.2001 17:01:06 fachwort

remote reference

A remote reference `RREF(P)` is a special global reference. The →smart pointer always refers to a stub object. Given this, a method call using a remote reference is always handled using a →remote call. Remote references occur on the client side in strict client/server hierarchies.

request

Request is a synonym for call.

resource manager

Resource managers, which manage their resources locally using transaction monitoring, are only used in relation to distributed transaction processing. The most common example of a resource manager of this kind is a database.

restart point

An →end of transaction or a →data link point is a restart point under GINA. In the case of a rollback, the operation is restarted at the restart point using the same request/call.

rollback

see →end of transaction

Run Time Type Information

RTTI allows you to specify the type of the object at runtime.

server

A server waits in a client/server system for a request/call from a →client. It provides →services for the client. A server can make subrequests/subcalls to other servers and can thus act as a client.

Servers are transaction-monitored under T-ORB. Non-transaction-monitored servers can also be connected using T-ORB/Client.

server process

A server can consist of one or more operating system processes. A number of client queries are generally processed by different server processes.

server stub

In DCE terminology, a server stub is the same as the →external interface under GINA.

service

A →server provides services. Services are generally provided in rpc-like interfaces in an API. Services are thus called. T-ORB provides an object-oriented API. Functions, class methods and instance methods can be called remotely (with transaction monitoring). →Stubs relieve the application of the task of dealing with communication details.

shadowed attribute

A shadowed attribute is an attribute inherited from a base class (*Base*::*attr*) that redefines (*Derived*::*attr*) the inheriting class (`class` *Derived* : `public` *Base*). In C++, this type of attribute can be addressed in the name scope of the base class (*Base*::*attr*) as well as in that of the derived class (*Derived*::*attr*).

signature

The signature of an operation consists of the type list of parameters used in the operation.

skeleton

In CORBA terminology, a skeleton is the same as the →external interface under GINA.

smart pointer

The term smart pointer is used in C++ to refer to classes which overload the dereferencing operators `*` and `->`. Smart pointers are often used to access →persistent objects (see Persistency Service), or to access remote objects (see T-ORB).

stub

A stub shields the client application from the details of a →remote call. A stub method or stub function can be called in the same way as a local method. The parameters are encoded, the remote call is actually made and the results are decoded, if necessary, within the stub method or stub function.

A representative object is needed for calling methods. We call this a stub object or a proxy. In DCE, we talk about a client stub, rather than a stub.

synchronous request/call

In the case of the T-ORB procedure synchronous call, a client waits for the result of a synchronous request/call, thereby blocking further operations. A normal C++ function or method call is a synchronous call.

Furthermore, T-ORB allows clients to make a number of synchronous, remote requests in parallel. The client waits until the results of all the calls have been received.

Druck vom 24. 01.2001 17:01.06 fachwort

T-ORB

The Transaction-monitored Object Request Broker allows local objects to send messages to remote objects with transaction monitoring. This means that not only the (persistent) states of the object, but also the messages undergo transaction monitoring. The sender does not have to worry about communication details.

time request

In the T-ORB procedure “time request”, a call is executed either at a certain time (absolutely) or with a certain time delay (relatively). A client does not wait for the result of a time request. Any events that may occur are thus discarded (see also →notification).

transaction

A transaction is an operation which is executed in accordance with the all-or-nothing rule. Transactions must comply with the ACID property (Atomicity, Consistency, Isolation, Durability):

A transaction is atomic, i.e. it is executed without “interruption” as a result of individual steps. A transaction changes an inconsistent state to a consistent state. Transactions are isolated, i.e. any inconsistent interim states that may exist are not visible from the outside. Their results are permanent, i.e. protected from errors.

see also →end of transaction

transaction-monitored request/call

A transaction-monitored request/call is carried out in a →transaction in accordance with the all-or-nothing rule.

Transaction-monitored calls can occur in a distributed way under T-ORB. They then form a →distributed transaction. A transaction-monitored call can be terminated with a commit or a rollback.

transaction manager

see →transaction monitor

transaction monitor

A transaction monitor or transaction manager coordinates a number of →resource managers and combines their →local transactions in a →distributed transaction operation to form a →global transaction.

transient

In contrast to a →persistent object, the life span of a transient object is limited by the life span of its process. Normal C++ objects are transient objects.

two-phase commit

refers to a transaction manager protocol, which ensures that all the changes in the resources are made atomically in a →distributed transaction operation, and that all the changes in all the resources are reversed if an error occurs in one resource.

value-based access

Access on a database using a search criterion.

XA protocol

The →transaction monitor and the →resource manager are connected over the standard X/Open XA interface. It coordinates the transactions using the services provided by the resource manager.

Druck vom 24. 01. 2001 17:01:06 abkuerzg

Abbreviations

ACID	Atomicity, Consistency, Isolation, Durability
ANS	American National Standard
ANSI	American National Standards Institute
API	Application Programming Interface
ASN.1	Abstract Syntax Notation One
BER	Basic Encoding Rules
BLOB	Binary Large OBject
CCITT	Comité Consultatif International Télégraphique et Téléphonique International Telegraph and Telephone Consultive Committee
CDE	Common Desktop Environment
CER	Canonical Encoding Rules
CICS	Customer Information Control System
CMISE	Common Management Service Element
CMIP	Common Management Information Protocol
CMIS	Common Management Information Service
CMX	Communications Manager in SINIX
CORBA	Common Object Request Broker Architecture
CPI-C	Common Programming Interface for Communication
CTP	Connection Termination Point

DB	DataBase
DBMS	DataBase Management System
DCE	Distributed Computing Environment
DCN	Data Communication Network
DER	Distinguished Encoding Rules
DOMS	Distributed Object Management Service
GINA	General Interface for Network Applications
GUI	Graphical User Interface
IC	Information Conversion
IDL	Interface Definition Language
IIOP	Internet InterORB Protocol
IRONMAN	Integrated **R**egionalized **O**bject-oriented **N**etwork **MAN**agement System
ISO	International organization for standardization
ITU	International Telecommunication Union
LAN	Local Area Network
LU6.x	Logical Unit type 6.x (SNA protocols)
MCF	Message Communication Function (ITU recommendation M.3010)
MIBS	Management Information Base Service
NDR	Network Data Representation
NE	Network Element
NEL	Net Element Layer
NK	Network node (German abbreviation)

Druck vom 24. 01. 2001 17:01:07 abkuerzg

NLS	Native Language Support
NWL	Network Layer
ODMG	Object Database Management Group
OLTP	Online Transaction Processing
OMG	Object Management Group
OO	Object Orientation
OODB	Object-Oriented DataBase
OOP	Object-Oriented Programming
ORB	Object Request Broker
OS	Operating System
OSI	Open Systems Interconnection
OSI-TP	Open Systems Interconnection distributed Transaction Processing
PCMX	Portable Communication Manager
Q3	TMN interface in accordance with CCITT recommendation M.3010
RAID	Redundant Array of Independent Disks
RDBMS	Relational DataBase Management System
RPC	Remote Procedure Call
RTTI	Run Time Type Information
SNA	System Network Architecture
SQL	Structured Query Language
TCP/IP	Transmission Control Protocol/Internet Protocol
TMN	Telecommunication Management Network

TNSX	Transport Name Server SINIX
T-ORB	Transaction-monitored Object Request Broker
TP	Transaction Processing
TTP	Trail Termination Point
UPIC	Universal Programming Interface for Communication
UTM	Universal Transaction Monitor
WAN	Wide Area Network
WS	Work Station
XA	defines an Application Interface for Distributed Transaction Processing
XDR	eXternal Data Representation
XMP	X/Open Management Protocols Application Interface
XOM	X/Open OSI Abstract Data Manipulation
X/Open	is an independent worldwide open systems non-profit organization
XPG	X/Open Portability Guide

Druck vom 24. 01.2001 17:01:07 literat

Related publications

Contact your local Siemens Nixdorf office to order manuals whose order numbers begin with a U.

Information relating to version and order numbers is only valid for the period in which this manual is published. Please ask for information on the latest editions. ❍❍●

[1] BEA TUXEDO
Reference Manual
Release 6.3, BEA Systems Inc., April 1997
Part No. 801-001010-002

[2] BEA TUXEDO
Administration Guide
Release 6.3, BEA Systems Inc., April 1997
Part No. 801-001007-002

[3] Booch, G.:
Object-Oriented Analysis and Design
Benjamin/Cummings, Readwood City, 2nd edition, 1994, 589 p.

[4] Booch, G.; Rumbaugh, J.; Jacobson, I.:
Unified Modeling Language Semantics and Notation Guide 1.0
Rational Software Corp., San Jose, CA, 1997

[5] CAE Specification Distributed Transaction Processing:
The XA Specification
Reading, UK, 1991, 80 p.
X/Open Document Number XO/CAE/91/300

[6] CCITT:
Principles for a Telecommunications Management Network
Recommendation M.3010, 1992

[7] CMX V5.1
Communications Manager in UNIX
Operation and Administration
Fujitsu Siemens Computers GmbH
U6519-J-Z145-4-76
http://www.fujitsu-siemens.com/servers/man/man_de/utm_man/outm_50.htm

[8] The Common Object Request Broker: Architecture and Specification
Revision 2.2, February 1998
Object Management Group, Framingham, USA
http://www.omg.org

[9] CPI-C/LU6.2 (SINIX, BS2000, MS-DOS)
Reference Manual
Siemens Nixdorf Informationssysteme AG, November 1992, 446 p.
U9939-J-Z715-1-7600

[10] Ellis, M.; Stroustrup, B.:
The Annotated C++ Reference Manual
Addison-Wesley, Reading, 1990

[11] Generic++ 2.5
Portable C++ Foundation Class Library
User Manual
OO.Tec GmbH, Gauting, June 1999
http://www.ootec.de

[12] GINA
Introductory Guide
Siemens Nixdorf Informationssysteme AG, April 1996
http://www.oo-gina.com

[13] GINA V4.0
Developer Manual
Siemens Business Services GmbH & Co OHG, September 2000
http://www.oo-gina.com

[14] GINA V4.0
Reference Manual Persistency Service
Reference Manual T-ORB
Siemens Business Services GmbH & Co OHG, September 2000
http://www.oo-gina.com

Druck vom 24. 01.2001 17:01:07 literat

[15] IDL C++ Language Mapping
Specification, September 1994, 135 p.
OMG Document No. 94-9-14

[16] Informix Guide to SQL
Syntax
IDS.2000, V9.2, December 1999
Part No.: 000-6527
CD: Informix Answers Online, Product Documentation, Version 3.2

[17] INFORMIX-ESQL/C
Programmer's Manual
Client SDK 2.30, Version 9.21 April 1999
Part No.: 000-5424
CD: Informix Answers Online, Product Documentation, Version 3.2

[18] Informix Guide to SQL
Reference
IDS.2000, V9.2, December 1999
Part No.: 000-6526
CD: Informix Answers Online, Product Documentation, Version 3.2

[19] Administrators Guide
for Informix Dynamic Server 2000, V9.2, September1999
Part No.: 000-6202
CD: Informix Answers Online, Product Documentation, Version 3.2Administrator's

[20] Archive and Backup Guide
for Informix Dynamic Server 2000, V9.2, September 1999
Part No.: 000-6205
CD: Informix Answers Online, Product Documentation, Version 3.2

[21] Trusted Facility Manual
for Informix Dynamic Server 2000, V9.2, September 1999
Part No.: 000-6234
CD: Informix Answers Online, Product Documentation, Version 3.2

[22] ISO WG 21:
Draft Proposed International Standard for Information Systems
– Programming Language C++
WG21/N0687, April 1995

[23] LU6.2-OSI-TP-GATE (SINIX) V1.0
Gateway between LU6.2 and OSI-TP
Product Manual
Siemens Nixdorf Informationssysteme AG, August 1993, 68 p.
U21141-J-Z815-1-7600

[24] The Object Database Standard: ODMG-93, Release 1.1
Morgan Kaufmann Publishers, 1994

[25] *open*UTM V5.0
Concepts and Functions
Fujitsu Siemens Computers GmbH
U20683-J-Z135-3-76
http://www.fujitsu-siemens.com/servers/man/man_de/utm_man/outm_50.htm

[26] *open*UTM V5.0 (UNIX, Windows NT)
Generating and Handling Applications
Fujitsu Siemens Computers GmbH
U4083-J-Z135-6-76
http://www.fujitsu-siemens.com/servers/man/man_de/utm_man/outm_50.htm

[27] *open*UTM V5.0 (BS2000/OSD)
Generating and Handling Applications
Fujitsu Siemens Computers GmbH
U3034-J-Z135-7-76
http://www.fujitsu-siemens.com/servers/man/man_de/utm_man/outm_50.htm

[28] *open*UTM V5.0 (BS2000/OSD, UNIX, Windows NT)
Administering Applications
User Guide
Fujitsu Siemens Computers GmbH
U24410-J-Z135-2-76
http://www.fujitsu-siemens.com/servers/man/man_de/utm_man/outm_50.htm

[29] *open*UTM V5.0 (UNIX, Windows NT)
Messages, Debugging and Diagnostics
Fujitsu Siemens Computers GmbH
U21400-J-Z135-4-76
http://www.fujitsu-siemens.com/servers/man/man_de/utm_man/outm_50.htm

[30] *open*UTM V5.0 (BS2000/OSD)
Messages, Debugging and Diagnostics
Fujitsu Siemens Computers GmbH
U20632-J-Z135-4-76
http://www.fujitsu-siemens.com/servers/man/man_de/utm_man/outm_50.htm

Druck vom 24. 01.2001 17:01:08 literat

[31] *open*UTM-Client V5.0 (UNIX, WindowsNT)
for the UPIC Carrier System
Fujitsu Siemens Computers GmbH
U25770-J-Z135-2-76
http://www.fujitsu-siemens.com/servers/man/man_de/utm_man/outm_50

Druck vom 24. 01.2001 17:01:08 verwsix.doc

Index

Druck vom 24. 01.2001 17:01:09 verwsix.doc

kmk
Druck vom 24.10.01 17:11

Siemens Business Services GmbH & Co OHG
SBS MPM CPI
81730 Munich
Germany

Fax: (089) 636-48 303

Internet: gina.service@mch20.sbs.de

Comments
Suggestions
Corrections

Submitted by

Comments on GINA V4.0
System Administrator Guide

kmk
Druck vom 24.10.01 17:11

Siemens Business Services GmbH & Co OHG
SBS MPM CPI
81730 Munich
Germany

Fax: (089) 636-48 303

Internet: gina.service@mch20.sbs.de

Comments
Suggestions
Corrections

Submitted by

Comments on GINA V4.0
System Administrator Guide

Herausgegeben von / Published by
Siemens Business Services GmbH & Co OHG
D-81730 München

Printed in Germany
Bestell-Nr./ Order No. ***GINA V4.0 System Administrator Guide – September 2000***