# EUROPAGATE

European SR-Z39.50 Gateway

# Functional Specification

**University College Dublin**
**Danmarks Tekniske Bibliotek**
**An Chomhairle Leabharlanna**
**Consejo Superior de Investigaciones Científicas**

**CEC Libraries Programme Project LIB-EUROPAGATE/2-2062**

| | | | |
|---|---|---|---|
| Work Package: | 3 | Responsible partner: | UCD |
| Task: | 1 and 2 | Contributing partner: | DTB |
| Deliverable: | D7/D8 | | |
| Availability: | Public | Date: | 23/11/94 |

The copyright in this document is the property of the EUROPAGATE contractors. All or part of the document may be copied for educational or research purposes by individuals or by libraries acting on their behalf, provided that such copies are not sold.

# Preface

This report is the functional specification of the Z39.50/ISO-SR gateway.

This report contains contributions from

- Dimitrios Andreadis, University College Dublin
- Séamus Ó Ciardhuáin, University College Dublin
- Mogens Sandfær, Danmarks Tekniske Bibliotek
- Franck Falcoz, Danmarks Tekniske Bibliotek
- Morten Nielsen, Danmarks Tekniske Bibliotek
- Jan Bronée, Danmarks Tekniske Bibliotek

It was edited by Séamus Ó Ciardhuáin, University College Dublin.

For further information about the specification or the EUROPAGATE project, contact one of the two technical managers:

- **Séamus Ó Ciardhuáin**, *Senior Research Officer*
  Department of Computer Science, University College Dublin,
  Belfield, Dublin 4, Ireland
  Tel: +353-1-7062485, Fax: +353-1-2697262, Email: seoc@net-cs.ucd.ie

- **Mogens Sandfær**, *Director of Development*
  Danmarks Tekniske Bibliotek (National Technological Library of Denmark)
  Anker Engelunds Vej 1, 2800 Lyngby, Denmark
  Tel: +45 42 88 30 88, Fax: +45 45 88 82 56, Email: ms@dtb.dk

An electronic information service based on World Wide Web is available at

- **http://www.dtb.dk/egate/egate.html**

Publicly-available project deliverables may be obtained from

- **gopher://dallas.ucd.ie**

- **ftp://dallas.ucd.ie/pub/EUROPAGATE**

# Table of contents

1 Overview and Introduction ..... 6
1.1 Introduction ..... 6
1.1.1 Purpose ..... 6
1.1.2 Scope ..... 6
1.1.3 Definitions ..... 7
1.1.3 References ..... 7
1.2 General Description ..... 8
1.2.1 Problem definition ..... 8
1.2.2 Function and purpose ..... 8
1.2.3 Gateway Structure ..... 9
1.2.4 User characteristics ..... 10
1.2.5 Operational Environment ..... 11
1.2.5.1 Communication interfaces ..... 11
1.3 Operational Scenarios ..... 12
1.3.1 Scenario A1 (Project Scenario) ..... 12
1.3.2 Scenario A2 “Integrated Gateway” ..... 13
1.3.3 Scenario B “Special service provider” ..... 13
1.3.4 Scenario C “Network provider” ..... 14
1.4 Project Prerequisites ..... 14
1.4.1 Development environment ..... 14
1.4.2 Target Environment ..... 15
1.5 General Requirements ..... 15
1.5.1 End User Requirements ..... 16
1.5.2 Database Provider Requirements ..... 16
1.5.3 Administrative User Requirements ..... 16
1.5.3.1 Installation and Configuration ..... 17
1.5.3.2 Normal operation ..... 17
1.5.3.3 Accounting ..... 17
1.5.3.4 Logging and Statistics ..... 17
1.5.4 System Requirements ..... 17
1.5.4.1 Environment ..... 18
1.5.4.2 Communications ..... 18
1.5.5 Other Requirements ..... 18
1.5.5.1 Portability ..... 18
1.5.5.2 Extensibility ..... 18
1.5.5.3 Adherence to Standards ..... 18
1.5.5.4 Security ..... 19
1.5.5.5 Scalability ..... 19
2 Z39.50/ISO-SR Gateway ..... 20
2.1 Requirements ..... 20
2.1.1 End User Requirements ..... 20
2.1.1.1 Transparency ..... 20
2.1.1.2 Data Conversion ..... 20
2.1.1.3 Performance ..... 21
2.1.1.4 Security ..... 21

2.1.2 Database Provider Requirements ..... 21
2.1.3 Administrative User Requirements ..... 21
2.1.4 System Requirements ..... 21
2.2 Specification ..... 22
2.2.1 Basic functionality ..... 22
2.2.1.1 Supported services ..... 22
2.2.1.2 Client/Server modules ..... 23
2.2.1.3 Procedures ..... 24
2.2.1.4 Multiple connections ..... 26
2.2.1.5 Performance ..... 26
2.2.2 Enhanced functionality ..... 27
2.2.2.1 MARC Translation ..... 27
2.2.2.2 Resource management ..... 27
2.2.2.3 Accounting ..... 28
2.2.2.4 Logging of Operational Data ..... 28
3 Email Gateway to Z39.50 and SR services ..... 30
3.1 Requirements ..... 30
3.1.1 General requirements ..... 30
3.1.2 End User Requirements ..... 30
3.1.2.1 Interface ..... 30
3.1.2.2 Command language ..... 31
3.1.2.3 Query language ..... 31
3.1.2.4 MARC records display ..... 31
3.1.3 Database Provider Requirements ..... 32
3.1.3.1 User authentication ..... 32
3.1.3.2 Resource control ..... 32
3.1.3.3 Usage logs ..... 32
3.1.4 Gateway administrator Requirements ..... 32
3.1.4.1 Installation ..... 32
3.1.4.2 Security ..... 33
3.1.4.3 Resource control ..... 33
3.2 Specification ..... 33
3.2.1 Communication interfaces ..... 33
3.2.2 Basic functionality ..... 35
3.2.2.1 Connection-less vs. connection-oriented service ..... 35
3.2.2.2 Email system dependency ..... 37
3.2.2.3 Command languages and query syntax ..... 38
3.2.2.4 Resource management ..... 39
3.2.2.5 Authentication/Security ..... 39
3.2.2.6 Logging/Statistics ..... 40
3.2.2.7 MARC records display ..... 40
3.2.3 Software Layering ..... 40
4 Administrative server ..... 42
4.1 Requirements ..... 42
4.1.1 General requirements ..... 42
4.1.2 End User Requirements ..... 42
4.1.2.1 Interface ..... 42
4.1.2.2 Security ..... 43

4.1.3 Server Administrator Requirements ..... 43
4.2 Specification ..... 43
4.2.1 Communication interfaces ..... 43
4.2.2 Basic functionality ..... 44
4.2.2.1 Transaction logging ..... 44
4.2.2.2 Resource management ..... 44
4.2.2.3 Authentication ..... 44
A Supported MARC Conversions ..... 45
B Requirements Checklist ..... 47

# 1 Overview and Introduction

## 1.1 Introduction

This document sets out the Functional Specification for the Z39.50/ISO-SR gateway.

### 1.1.1 Purpose

The main purposes of the gateway service are

- To allow systems that utilise the Z39.50 version 2 and the ISO Search and Retrieve (SR) standards to interoperate transparently with each other, on top of TCP/IP and OSI communication stacks respectively.
- To give users with only an email capability access to servers supporting Z39.50 and SR.

The gateway will provide the administrator with tools that will allow him to manage available resources, control access, and operate an accounting system to record, and potentially charge for, use of the gateway.

The purpose of this report is to give a specification of the gateway. It is specified by giving requirements for the gateway users, the gateway administrators/operators and the database providers, followed by defining the functionality which must be achieved to satisfy those requirements. Certain aspects of the functionality are identified as basic and others as enhanced. However, it is the intention of the project to implement all the functionality specified, i.e. this document does not specify optional functions which might be desirable in future.

There are no detailed interface specifications given. These interfaces will be defined during Workpackage 4 (the detailed design of the service).

### 1.1.2 Scope

The gateway will offer both Z39.50 version 2 and ISO-SR clients access to the whole range of servers (utilising either of these protocols) on the Internet and on X.25 public networks (if access to an X.25 network is provided by the OSI stack in use).

It is intended to support every target/database that is compliant with either Z39.50-1992 or ISO-SR.

The gateway will offer additional services beyond basic connectivity to its users, such as MARC conversion and enhanced accounting for the database provider.

The purpose of a mail gateway to the Z39.50 and SR environment is to allow users who have only mail access to the Internet to search the large number of bibliographical databases available via SR or Z39.50. Users will send queries to the gateway by email, and the gateway will then establish an association with the server to retrieve information which is mailed back to the user.

### 1.1.3 Definitions

In this text the mail gateway will also be described as the "mail server". The mail server receives email requests and responds with email.

Two terms are related to bibliographic databases, a set and a record. A set is a union of records that matches a given query. A record is the actual bibliographic reference, holding reference information about a book, an article, a paper, a journal, etc. In the Z39.50/SR environment bibliographic records are passed encoded in one of the MARC formats.

### 1.1.3 References

[1] ANSI/NISO Z39.50 Version 2 — *Information Retrieval Application Service Definition and Protocol Specification for Open Systems Interconnection,* 1992.

[2] ISO 10162 — *Search and Retrieve Application Service Definition for Open Systems Interconnection,* 1991.

[3] ISO 10163 — *Search and Retrieve Application Protocol Specification for Open Systems Interconnection,* 1991.

[4] ISO 7498 — *Information Processing Systems—Open Systems Interconnection—Basic Reference Model,* 1984.

[5] Lynch, Clifford (1994). *Using the Z39.50 Information Retrieval Protocol in the Internet Environment*, Draft RFC. Posted to Z3950IW mailing list, 30 April 1994.

[6] EUROPAGATE Background Study — Part I: *Current implementations of Z39.50 and ISO SR*, Part II: *Z39.50 and ISO SR servers in operation*, Part III: *Current developments and future trends*, EUROPAGATE Consortium, 1994.

[7] ISO 2709 — *Documentation — Format for bibliographic information interchange on magnetic tape.*

[8] *USMARC Specifications for Record Structure, Character Sets, Tapes*. US Library of Congress.

## 1.2 General Description

### 1.2.1 Problem definition

In the Information Technology (IT) world there is a trend towards distributed systems and client-server technologies. The same trend applies to bibliographic databases where a database provider publishes his databases over a computer network, by means of a well defined interface (application protocol) to the actual database and the search engine. Two protocols have been developed for this purpose, ANSI Z39.50 and the ISO Search and Retrieve (SR). Although the two protocols are compatible, use of different communication stacks by implementations inhibits interoperability. In addition, the existence of a variety of different query types and bibliographic record formats limits potential functionality when the communicating parties cannot agree on a common query type or record format. A gateway offering communication between different stacks and conversion of data to a format acceptable to users would be a valuable service as the number of networked bibliographic databases increases.

Access to networked databases may be impossible for many users who do not have full network connections (Internet or X.25), even though they may have email facilities available. The email function of the gateway will make it possible for these users' queries to be passed to the server and responses to be returned. This will broaden the potential market for networked databases.

Another perspective for the mail gateway is to port the technology to other non-connection-oriented services, e.g. the World Wide Web (WWW) or Gopher. These services, being connectionless, are targets for an ongoing development of gateways and it will be useful to study some of the solutions from existing implementations as well as to implement some of the ideas from this mail gateway in a similar gateway with another system.

### 1.2.2 Function and purpose

The main purpose of the gateway system is to give a client that utilises the ANSI Z39.50-1992 (version 2) protocol access to a server using the ISO Search and Retrieve (SR) protocol. The relationship between the above elements can be seen in the following diagram:

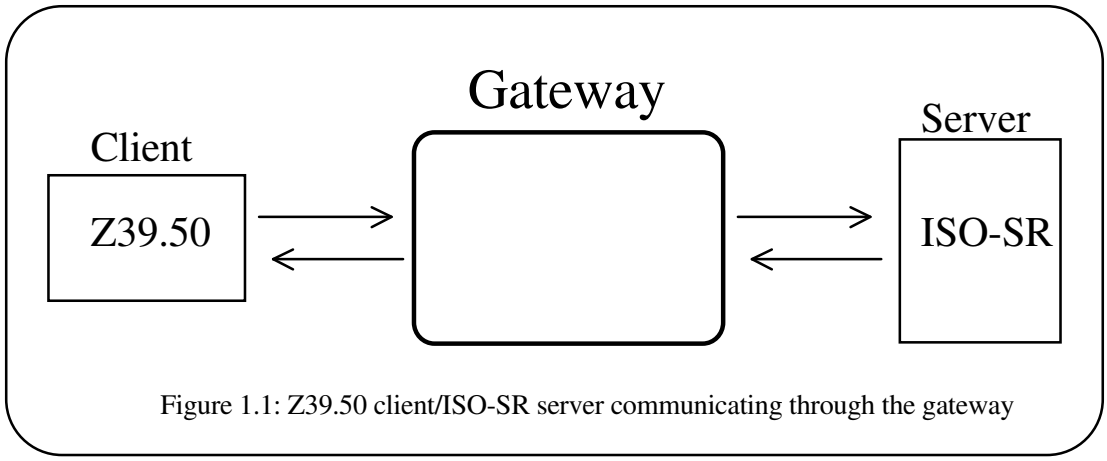


Figure 1.1: Z39.50 client/ISO-SR server communicating through the gateway

Conversely, an SR client should be able to communicate successfully with a Z39.50 version 2 server, by means of the gateway service:

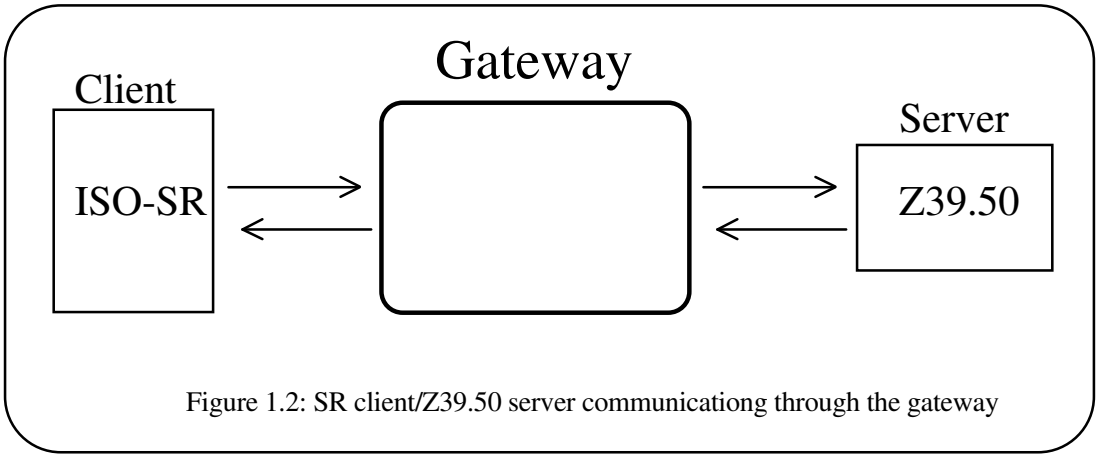


Figure 1.2: SR client/Z39.50 server communicationg through the gateway

In simple terms, the gateway will allow users with access to a certain type of client (either ISO-SR or Z39.50) to search the whole range of available bibliographic databases, no matter which of the two standards is used by the target system (server).

The objective of the email gateway is to allow email users without on-line access to international networks to search databases supporting Z39.50 and SR. Below is a schematic view of the email gateway.

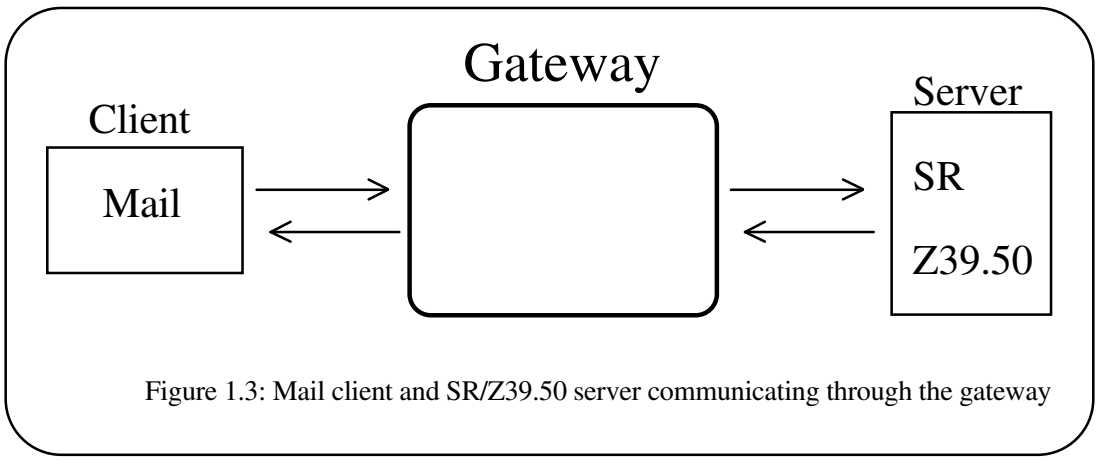


Figure 1.3: Mail client and SR/Z39.50 server communicating through the gateway

### 1.2.3 Gateway Structure

The structure of the gateway is shown in Figure 4.

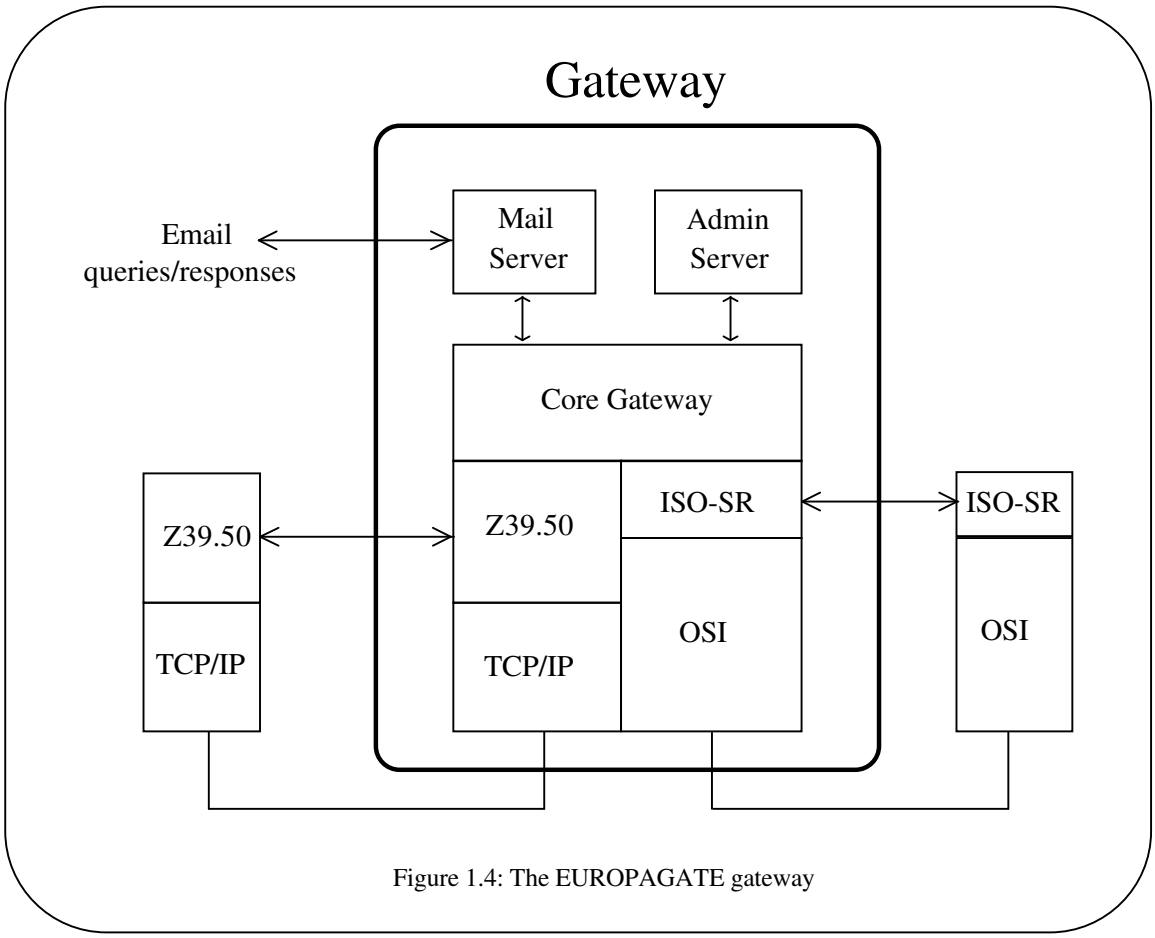


Figure 1.4: The EUROPAGATE gateway

The components identified in the diagram are discussed separately in this document.

### 1.2.4 User characteristics

In the context of a gateway system, there are two sorts of user: the *end-user* and the *administrative user*. An end-user is a person who actually makes use of the system. He will be able to use any Z39.50 version 2 client running on top of TCP/IP, in order to communicate (through the gateway) with an ISO-SR server on top of a full OSI stack (and vice-versa). On the other hand, an administrative user is a person who is responsible for ensuring the installation and proper operation of the gateway system. Both of these users have requirements of different sorts and they are considered separately below.

Using the gateway service will be straightforward for the potential end-user. He will be able to use the same origin (client) software to search and retrieve records from the desired target system, in the usual manner. Instead of connecting directly to the target, he will have to go through the gateway and supply the target address, so that the gateway can proceed with establishing the actual connection to the server.

After the initial connection is established, the gateway will work transparently. Although a good understanding of indexes, query techniques, etc., is necessary for the end-user to make effective use of the available Search and Retrieve facilities, very little extra knowledge is

required to use the gateway service effectively. The key point is that most of the knowledge which it is necessary for a user to have is determined by the client software (which is the same whether a gateway is used or not) and not by the gateway. The impact of the gateway on the user is thus minimal.

The most basic requirement for a potential email gateway user is the ability to handle an email client. The user will need to be able to specify queries in the language supported by the gateway. These queries will be sent as the body of a message, so no special requirements exist for the user's email client software, and again the impact of the email gateway on the user is minimal.

In general, a user searching for reference in a library database needs some kind of understanding of bibliographic records and indexes. This will also apply for the user of the gateway.

### 1.2.5 Operational Environment

For the end-user the operational environment will be the same, whether or not he is using the gateway service. According to the client/server model the gateway will provide a remote service to potential users. In general, the client software used by the users will not run on the same node as the gateway. Instead, the gateway will use the network to accept connections from clients and attempt connections to servers.

For users of the email gateway, the operational environment is solely the mail environment in which he/she is working.

From the gateway operator's point of view, the environment is a UNIX system running the Z39.50 and SR software.

#### 1.2.5.1 Communication interfaces

ISO-SR and Z39.50 are application layer protocols (layer 7) within the OSI reference model. They assume the presence of lower-level OSI services (in particular, the presentation layer) and of the OSI Association Control Service Element (ACSE) within the application layer. However, in the real world the primary choice for Z39.50 implementors is to layer the Z39.50 protocol directly over TCP/IP rather than to implement it in an OSI environment, or to use the existing techniques that provide full OSI services at and above the OSI transport layer on top of TCP connections.

Although the two protocols are designed to be compatible, the choice of different communication stacks inhibits interoperability. Ensuring interoperability between the communicating systems is the primary goal of the gateway. The gateway must be accessible by Z39.50 and ISO-SR clients, and must be able to access Z39.50 and ISO-SR servers respectively. Therefore, the gateway node must interface with both stacks:

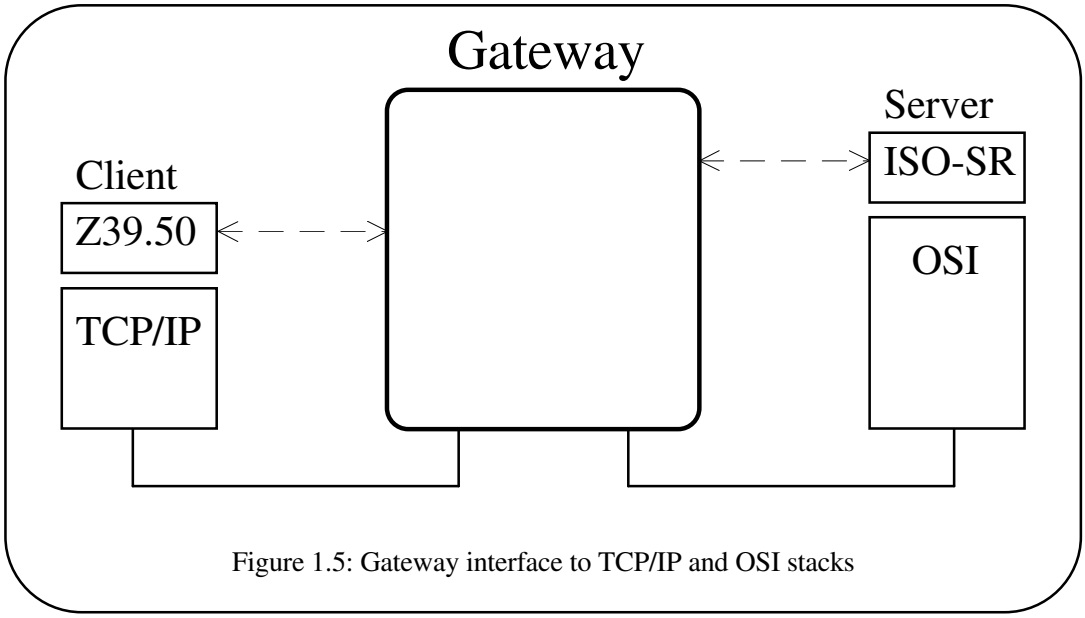


Figure 1.5: Gateway interface to TCP/IP and OSI stacks

## 1.3 Operational Scenarios

Several possible operational scenarios for the gateway have been identified, depending on factors such as who operates the gateway, where it is located, who its users are, and what value-added services it provides to those users. There are different complexities of implementation in each scenario.

For the development and pilot service, we chose Scenario A1 below because it is the simplest to implement and is easy to set up initially, with no need for user authentication and only a small number of agreements with the database providers being necessary. In future, however, the other scenarios (especially Scenario 3) seem to offer considerable scope for exploitation. There is no reason why the gateway could not offer different types of service to different users depending on their requirements.

In the description which follows, “user” refers to the user of client software accessing a catalogue to retrieve bibliographic data, and “database provider” refers to the operators of the catalogue server, i.e. libraries or commercial operators making such bibliographic data available.

### 1.3.1 Scenario A1 (Project Scenario)

In this scenario, the users have contracts directly with the database provider to obtain access to the server. The DB provider contracts with the gateway to provide access to the server through the gateway. As far as possible the gateway is transparent to users. The gateway’s customers, therefore, are the database providers who benefit by having a greater number of clients able to access the database, and potentially by having the gateway provide them with enhanced management functions (accounting and access control for example). The gateway is presumably operated by a third party. This scenario is particularly attractive from the user’s point of view because it requires little or no change to the user’s way of working compared to going directly to the server.

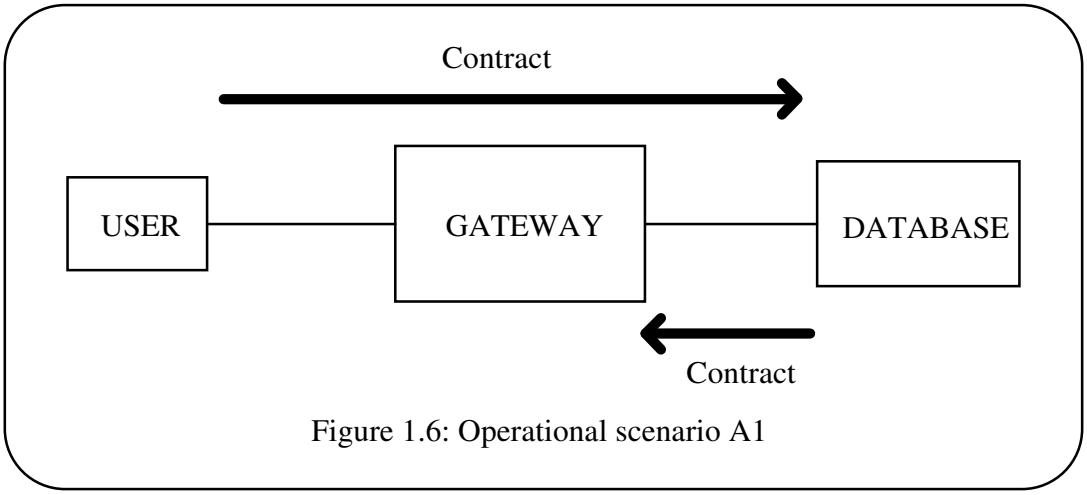


Figure 1.6: Operational scenario A1

### 1.3.2 Scenario A2 "Integrated Gateway"

This scenario is generally similar to Scenario A1, but the gateway is an integral part of the DB provider's environment rather than being operated by a third party. The gateway is invisible to the users and is replicated at each server site. The DB provider benefits by having greater accessibility to the server, making it more attractive to users, and possibly from any added functionality offered by the gateway. The user contracts with the DB provider as before.

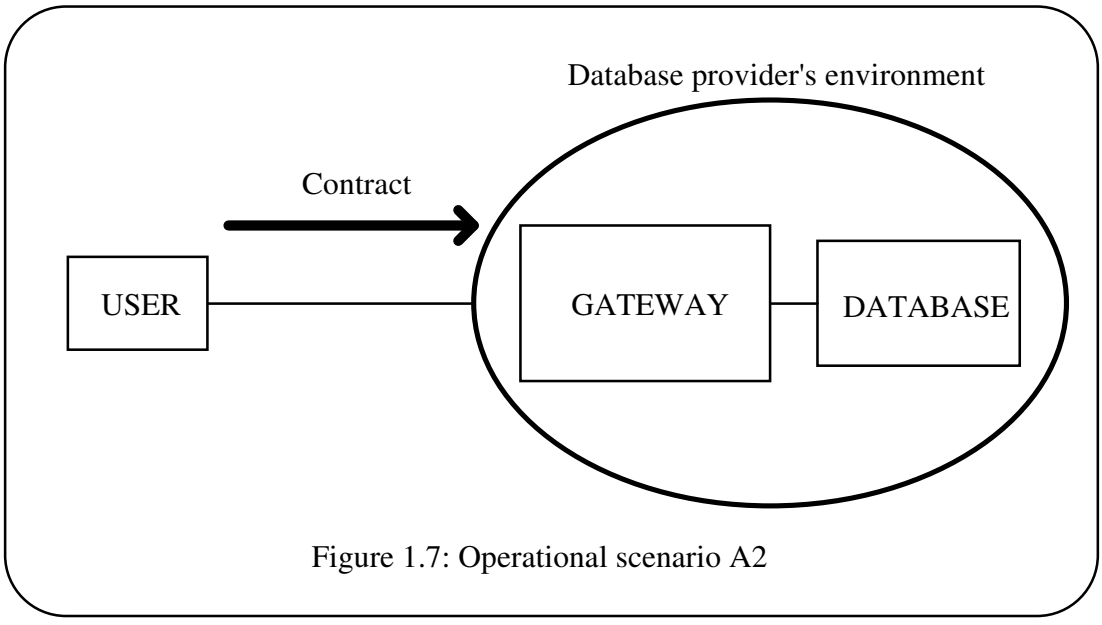


Figure 1.7: Operational scenario A2

### 1.3.3 Scenario B "Special service provider"

The user contracts with the gateway to use it, and the gateway operators contract with database providers to obtain access to the servers. The user gets a single point of access to the servers and the possibility of services such as multiple simultaneous searches (similar to IRIS). The database providers benefit from a greater user base and similar services to Scenario 1. In effect, the gateway becomes a "virtual database" service.

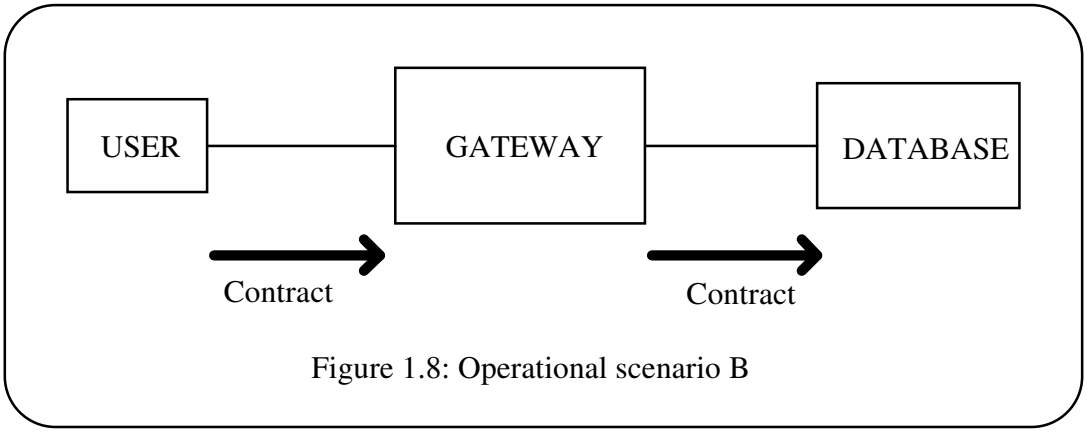


Figure 1.8: Operational scenario B

### 1.3.4 Scenario C "Network provider"

The user contracts both with the gateway and with the database provider and benefits from the gateway by getting a preferred access method. The gateway has merely a "network provider" role.

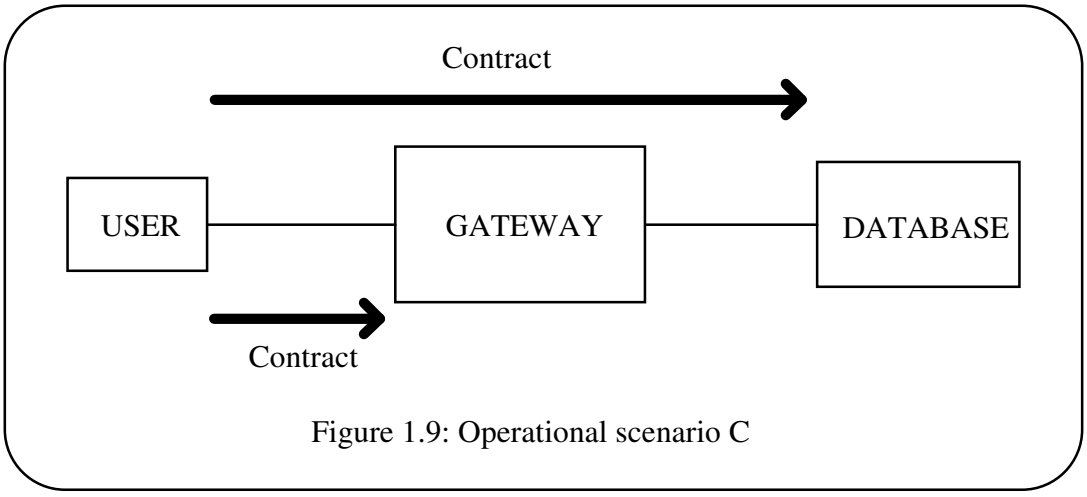


Figure 1.9: Operational scenario C

## 1.4 Project Prerequisites

The resources available to the project partners place certain restrictions on the development, and these are described in this section.

### 1.4.1 Development environment

Sun SPARCstations running Solaris 1 or Solaris 2 will be used for development. For testing purposes, a PC running a Z39.50 version 2 client and connected to the SPARCstation through a local area network will be used. UNIX-based Z39.50 and SR clients will also be used.

The GNU C/C++ compiler (gcc) will be used for all development. This has the advantage that it is ANSI standard C and will be available to others wishing to use the gateway code at no

cost, as well as being available on a large number of platforms to simplify future porting of the software.

The TCP/IP interface will be the sockets interface provided by the operating system.

The OSI stack used to support SR will be ISODE version 8, operating over TCP in accordance with RFC1006, and over SunNet X.25 if X.25 is required on the system in question.

Publicly available implementations of Z39.50 and SR will be used to provide the origin and target functions in the gateway. This avoids the need to reimplement the protocols themselves without precluding the distribution of the software developed by the project. Possible implementations for this purpose include:

- Stanford UNIX zclient, by Harold Finkbeiner
- NLC's (National Library of Canada) Information Retrieval Kernel (target)
- Nordic SR-Net package (client and server)

MARC conversion will be based on software developed as part of the Nordic SR-Net project.

### 1.4.2 Target Environment

The target platform will be similar to the development environment, i.e.

- Sun SPARCstations running Solaris 1 or 2
- Sockets library for TCP/IP access
- ISODE 8 to provide the OSI environment

The compiler will be necessary for those wishing to recompile the gateway, but it will be sufficient to have access to executables in order to install and operate it. This implies that configuration parameters should not be set at compile time but at startup or run time.

It will be necessary for the gateway operator to obtain and install the supporting software such as ISODE, and to ensure that an email system is available if email access is to be supported.

As far as possible, freely available software will be used to ensure that setting up a gateway is inexpensive.

## 1.5 General Requirements

Through the definition of requirements we try to refine our expectations from the system, define the scope of the project, give guidelines for the design phase, and set some standards against which to measure the success of the resulting system.

This section sets out the requirements which apply to the gateway as a whole. Requirements specific to a certain component are discussed in the sections dealing with that component.

### 1.5.1 End User Requirements

The project will not develop client software, so the end-user requirements of the gateway are limited. In general, they relate to allowing users to retain their existing client software and to use new clients. This means that the gateway's operation must be transparent to the client software.

The only direct interaction of users with the gateway will be in the creation of queries to be submitted by email. The query language used should be either an existing standard, or a minimally modified form of one, so that users do not have to learn a new language.

For the pilot service, there will be no requirement for user registration and similar functions (see Section 1.4). However, a fully developed gateway supporting all the operational scenarios would have to provide facilities for users to subscribe, to authenticate themselves and to obtain status information on their accounts from the gateway. It is not clear to what extent these facilities could be automated to reduce the need for interaction with the gateway operators. These and other functions will not be addressed in this project.

*R1.5.1a The gateway should interoperate with existing clients.*

### 1.5.2 Database Provider Requirements

The operational scenario chosen for the pilot service means that the gateway must support the database providers as customers. They will require facilities for

- subscription, i.e. allowing access to their servers through the gateway
- statistics, i.e. obtaining information about connections through the gateway
- accounting, i.e. details of the charges made by the gateway operator(s)

These value-added services for database providers are an important aspect of the gateway and others would be expected in a fully-developed gateway.

*R1.5.2a The gateway should allow database providers to subscribe, obtain accounting information and statistics from the gateway service.*

### 1.5.3 Administrative User Requirements

The administrative user is responsible for maintaining the operation of the gateway for end-users and database providers. Thus an administrative user might be the local system administrator in some company; or he might be an employee of a third party company, which provides the gateway service to its clients. Regardless of his position though, the administrative user will always be responsible for similar tasks; for installing the system initially, for maintaining the correct operation and use of the system, and for updating or upgrading the system from time to time.

The requirements to facilitate the administrative user are as follows.

#### 1.5.3.1 Installation and Configuration

The installation of the gateway should be easy or automated if possible. Difficulties might arise from the installation of other system parts required by the gateway (e.g. an OSI stack provided by ISODE). An installation manual addressing possible difficulties in the installation procedure should be created during the development phase.

*R1.5.3.a Installation of the gateway should be easy and well documented.*

#### 1.5.3.2 Normal operation

Normal operation of the gateway should require as little work as possible. Ideally day-to-day operation should be maintainable without any work, but if some management is needed, then it should be easy to do or automatable. This implies that management operations must be able to be performed from scripts to allow them to be carried out automatically. A basic set of such scripts should be provided with the distributed version of the gateway at the end of the project.

*R1.5.3.b The gateway should be accompanied with a basic set of administrative scripts.*

#### 1.5.3.3 Accounting

For the gateway to be operated on a commercial basis, it is necessary for the operators to have sufficient data on the use of the gateway that they can implement a charging policy. For the operational scenario chosen for the pilot service, this means that the operators must be able to bill the database providers and to provide them with information concerning the connections made through the gateway. Subscription to the gateway may be on a flat rate basis or related in some way to the number and duration of connections. Both of these schemes should be supported for the pilot service.

Alternative scenarios would require support for the management of user accounts with passwords to restrict access to users who have contracts with the gateway. Support for this will not be developed as part of the project.

#### 1.5.3.4 Logging and Statistics

Keeping detailed log files of gateway usage is necessary to facilitate accounting and to allow the administrator to analyse the use and performance of the gateway. Logging is also required for troubleshooting of the gateway in operation and during development. Basic tools for processing the logged data will also be required. Clear specifications of the log formats should be provided to facilitate development of other tools. The level of logging should be configurable by the gateway administrator.

*R1.5.3.c The gateway should be able to provide sufficient logging information.*

### 1.5.4 System Requirements

A number of requirements relate to the system on which the gateway will run. The general requirements of this kind applicable to all components are given below.

A number of requirements relate to the system on which the gateway will run. The general requirements of this kind applicable to all components are given below.

#### 1.5.4.1 Environment

The gateway service is intended to run on UNIX workstations connected to the Internet and optionally to a public X.25 network. The software will be developed to be portable to as many such systems as possible.

#### 1.5.4.2 Communications

The gateway will use the sockets interface for TCP/IP communications. For OSI communications, it will use ISODE over TCP and optionally over X.25.

Mail queries should be accepted from any source, i.e. the type of mail system available to a user (RFC822, X.400, etc.) should not restrict access to the gateway's functions. The only part of the mail of interest to the gateway should be the message body which should have the same format in all environments as it is created only by the user.

### 1.5.5 Other Requirements

#### 1.5.5.1 Portability

A design objective of the system is that it should be portable to other platforms supporting the underlying communications facilities (sockets and ISODE). This means that it should be possible to run the gateway on most UNIX systems. Dependencies on specific architectures or proprietary interfaces should be avoided as far as possible, and documented clearly where they occur.

*R1.5.5.a* *The gateway should be portable between UNIX platforms which support the necessary communication facilities (sockets and ISODE).*

#### 1.5.5.2 Extensibility

The Information Technology world moves so rapidly into new areas that is not sensible to believe that the gateway will satisfy all the user requirements of the future. New versions of both ISO SR and Z39.50 are under development and are expected to be in effect within a relatively short time. It is important that the system is designed in such a way that future extensions can be easily made when new requirements become apparent.

*R1.5.5.b* *The gateway should be easily extensible to accommodate a newer version of the supported protocols.*

#### 1.5.5.3 Adherence to Standards

European and international standards should be used whenever relevant. In particular, Search and Retrieve operations will use the ISO SR (ISO 10162 and ISO 10163) and ANSI Z39.50-1992 (version 2) standards. Network communications will use two sets of standard

protocols, namely TCP/IP (as standardised in RFC documents) and OSI (as standardised by ISO and CCITT). The OSI upper layer protocols will be operated over TCP/IP in accordance with RFC 1006. They may also be operated over X.25.

The system will be programmed in ANSI C.

*R1.5.5.c The gateway will support the ISO 10163 Search & Retrieve protocol on top of an OSI communication stack provided by ISODE.*

*R1.5.5.d The gateway will support the ANSI/NISO Z39.50-1992 (Version 2) Information Retrieval protocol on top of the TCP/IP transport stack.*

#### 1.5.5.4 Security

Authentication information defined in the protocols will pass through the gateway and it is necessary to guarantee its confidentiality. Special care should be taken when keeping identification data and passwords in permanent or even temporary system files and databases, and storage of these in non-volatile form should be avoided as far as possible.

The same security requirements apply to any extra authentication and accounting information deemed to be necessary, so that the gateway administration can control availability and keep track of system usage. In general, this means that the security facilities of the operating system should be properly understood and applied in the gateway

*R1.5.5.e The gateway should ensure the confidentiality of sensitive data.*

#### 1.5.5.5 Scalability

Although it is not expected that there will be a very large number of users during the pilot service phase of the project, it is important to ensure that the gateway is not constructed in such a way that it will not be able to grow to meet increased demand. For this reason, the gateway should be designed to be scalable in terms of the number of users supported, accessible databases etc.

*R1.5.5.f The gateway should be scalable to heavier use in future.*

# 2 Z39.50/ISO-SR Gateway

## 2.1 Requirements

User requirements are based on the gateway model described in Section 1.4. General requirements of the gateway are discussed in Section 1.6 above.

### 2.1.1 End User Requirements

#### 2.1.1.1 Transparency

The end-user will be able to interact with the gateway only through client applications (Z39.50 or SR origin), so there is no requirement for the gateway to have a user-friendly interface. However, it is essential that operation of the gateway be as transparent as possible from the user's point of view. For the client software it must appear as if nothing is intervening between the origin and the target. Whenever some particular information is needed by the gateway (e.g. target system's address, so that the gateway can establish a connection to the server) this has to be provided in an easy and uniform way for the whole range of clients. This implies that the method of specifying a target accessed through the gateway should be similar to the specification for direct access. The additional information needed by a user should be minimised.

*R2.1.1.a* *Operation of the gateway should be transparent for the end-user and especially for the client software.*

*R2.1.1.b* *Addressing (or any additional) information should be encapsulated in common and easily accessible fields.*

#### 2.1.1.2 Data Conversion

A client application may expect to receive records in a MARC format other than those formats supported by the server. In this case, the gateway should try to map the returned records to the desired format, so that the transferred data are semantically equivalent for both parties. The same would apply to protocol services where this can be achieved.

It is likely that in most cases the client and server will agree on a common data format and that no conversion will be required in the gateway. However, the ability to perform conversions would be a valuable additional service. The difficulty of conversion means that a full and transparent conversion may be impossible, but the gateway should be capable of returning a useful subset of the data to the user.

Attribute set conversion is not addressed in this project because there does not yet appear to be any need for it: the only attribute set used for bibliographic data is BIB-1, and it is not clear

that it is meaningful to attempt conversion to/from attribute sets for other types of data. However, in the longer term, it may be possible to have the gateway convert between attribute sets.

*R2.1.1.c The gateway should be able to perform (partial) MARC record conversion whenever appropriate.*

#### 2.1.1.3 Performance

It is clear enough that the presence of a gateway between any two communicating systems reduces the total throughput and introduces latency. The degradation in performance increases in cases where the gateway acts in a *staging* way, i.e. it waits to receive all the data contained in one request, before it can process them and finally send them off to the recipient. (The opposite would be to *pipe* the data from the origin to the target and process them *on the fly*.) However, for the gateway service to be successful it is essential to keep the response time as small as possible.

Defining the effect of interposing the gateway between the user and the server is complex. Many important response time parameters are not affected by the gateway, in particular search time on the server. Network response times are largely outside the control of the gateway operators and cannot usefully be included in the gateway specification. The specification of performance targets for the gateway should therefore be restricted to the internals of the gateway. The additional time per request and the data conversion time per record, for a given gateway load, are the parameters which it is felt will most affect the perceived delay introduced by the gateway.

*R2.1.1.d The degradation in performance caused by the gateway should be minimal.*

#### 2.1.1.4 Security

Given the operational scenario chosen in Section 1.4 for the pilot service, limited security is required on the gateway. It will not be necessary to maintain accounts and passwords for gateway users as the customers will be the database providers rather than end users. A fully developed gateway for commercial operation might require more security facilities.

### 2.1.2 Database Provider Requirements

None beyond Section 1.6.2.

### 2.1.3 Administrative User Requirements

None beyond Section 1.6.3.

### 2.1.4 System Requirements

None beyond Section 1.6.4.

## 2.2 Specification

### 2.2.1 Basic functionality

The main purpose of the gateway is to achieve interoperability between the communicating systems. In other words, the gateway is first called to bridge the gap caused by the use of different communication stacks in Z39.50 and ISO-SR implementations. After the desired interoperability has been achieved, the gateway should aim towards enhanced functionality, i.e. add value to the gateway service.

Achieving interoperability could possibly mean that the gateway simply pipes data from the client to the server, from one communication stack to the other, as soon as all initial connections have been established. This would work when transferring data from a Z39.50 client/server to an ISO-SR server/client, but not the other way round, since indirect references to external objects — which are legitimate in a pure OSI environment — do not apply to TCP/IP implementations. In other words, the Z39.50 byte-stream on top of TCP/IP is self-contained, while the ISO-SR byte-stream on top of an OSI stack can reference objects (e.g. a supported application context) which are located outside the byte-stream.

#### 2.2.1.1 Supported services

Z39.50 version 2 is a superset of ISO-SR, therefore a client that uses the gateway node to communicate with a server should be able to request any service common to both protocols, as long as the server supports this particular service. In other words, the originator (Z39.50 or ISO-SR client) should be able to:

- connect to the target system (server) through the gateway. The originator will have to provide -in some way- the target address to the gateway, which in turn will use this address to establish the second communication path to the target.
- use the INITIALIZE service to negotiate for operational parameters and supported services at the beginning of a session.
- use the SEARCH service to query available databases on the target system and receive information concerning the result of the query.
- use the PRESENT service to retrieve records from a specified result set, created by previous Search requests.
- use the DELETE service to request deletion of a specified or all result sets, held by the target system.
- terminate a session, by means of the ABORT and RELEASE services.

For those services unique to Z39.50 (Access control and Accounting/Resource control) the gateway can do very little to enhance interoperability. In the case of a Z39.50 client accessing an SR server, the gateway cannot convey to the client cost estimates about the target system,

nor can impose access control on behalf of the server, simply because the SR server does not support these features! Therefore, Access and Accounting/Resource control will not be supported (on the client side) for the Z39.50 client/SR server scenario.

Conversely, in the case of an SR client accessing a Z39.50 server, the gateway cannot communicate to the client an Access control or an Accounting/Resource control request without support from the SR protocol itself. Any intermediate solution (i.e. notifying the client in some other way) would have to skip the normal protocol procedures and use some other program to get the necessary information from the client process. This is a highly non-transparent solution and as a result unacceptable.

A sensible solution/compromise to this problem would be for the gateway to reply positively (on behalf of the client) in every Accounting/Resource control request. Similarly, a password supplied by the SR client on start-up (in the same way as the necessary addressing information) can be return automatically by the gateway as a response to every Access control challenge. The feasibility of this option will be further investigated during the system design phase.

*S2.2.1.a* *The gateway will support all services common to the ISO-SR and Z39.50-1992 protocols.*

*S2.2.1.b* *The gateway will automatically respond to Z39.50 Access and Resource Control requests, if possible.*

#### 2.2.1.2 Client/Server modules

The gateway should be able to accept connections and exchange Protocol Data Units (PDUs) with Z39.50 and ISO-SR clients that want to use the gateway service. From the origin's (client) point of view, the gateway will behave similarly to the final target system (server) responding to origin's requests. On the opposite side, the gateway will act like a client when connecting to the corresponding ISO-SR or Z39.50 target. This double client/server role of the gateway for the Z39.50 client/ISO-SR scenario can be seen in the following diagram:

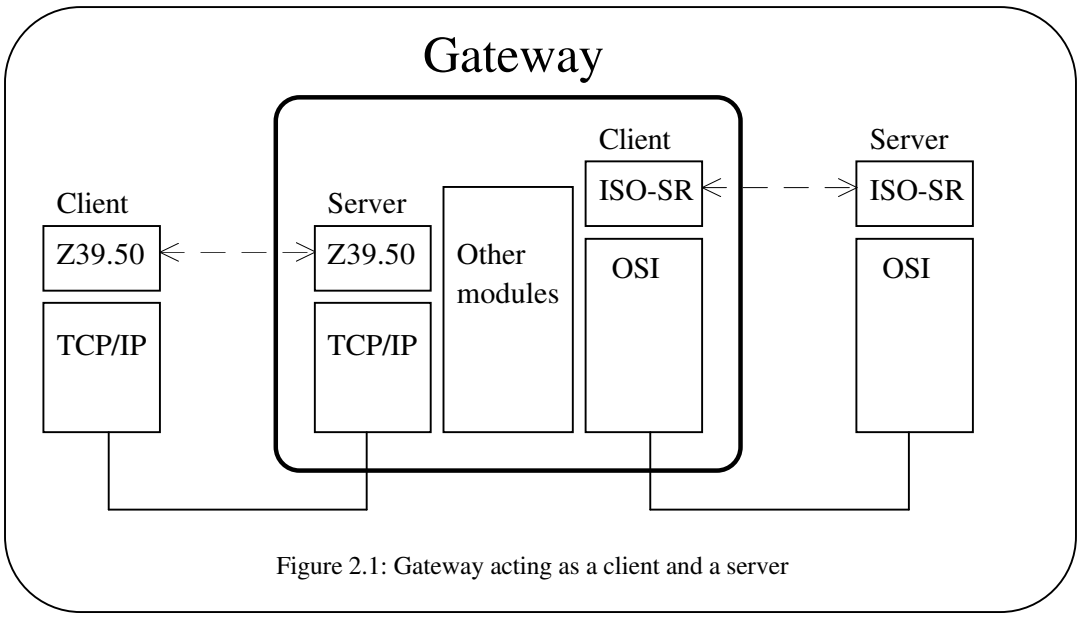


Figure 2.1: Gateway acting as a client and a server

These four client and server interfaces (Z39.50 client/server and ISO-SR client/server) constitute an important part of the gateway. In order to minimise development effort, portions of publicly available Z39.50 and ISO-SR implementation should be used.

*S2.2.1.c The gateway will use existing ISO-SR and Z39.50 (version 2) client and server implementations.*

#### 2.2.1.3 Procedures

The gateway will work in a staging way, which means that it will wait to receive all data contained in a Protocol Data Unit (either Request or Response), before it can process them and send them off to the final recipient. The following example (diagram and textual description) illustrates a Present operation, assuming that both communication paths (client-to-gateway and gateway-to-server) have been established:

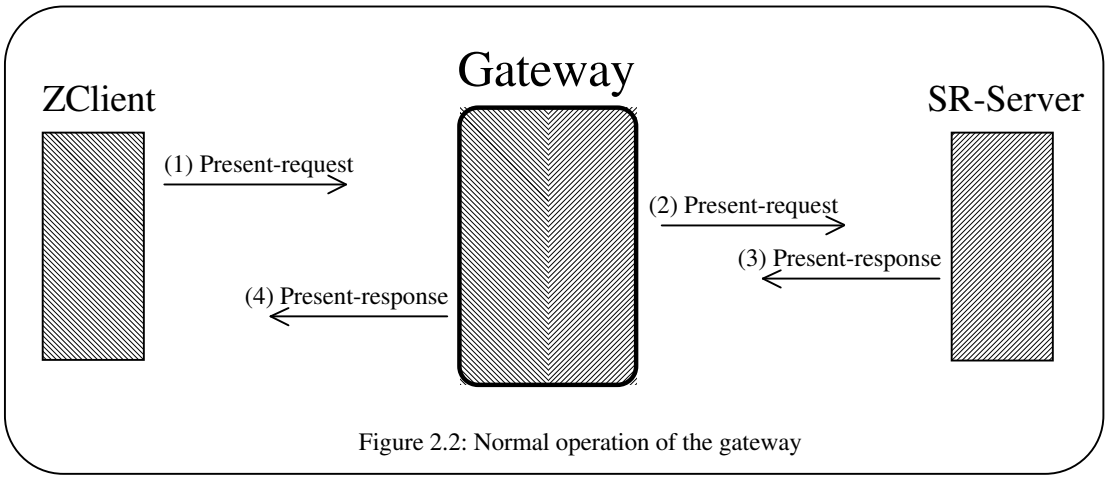


Figure 2.2: Normal operation of the gateway

a) Z39.50 client (Zclient) transmits a Present-request(1) (PDU number 1 in the diagram) to the Z39.50 server part of the gateway (G/Zserver).

b) G/Zserver receives the request, and passes this on to the control modules to perform any necessary mappings. The resulting data are given to the ISO-SR client part of the gateway (G/SRclient) for transmission.

c) Target system (SRserver) receives the Present-request(2), gathers the corresponding data and returns the resulting records in a Present-response(3) to G/SRclient.

d) G/SRclient gets the Present-response(3); control functions perform any potential transformations and the resulting data are given back to G/Zserver for transmission.

e) Zclient finally receives the Present-response(4).

This works fine in the middle of a session when both communication paths (client-to-gateway and gateway-to-server) have been established. However, at the very beginning when a client connects to the gateway, the gateway does not know the final destination (server address) in

order to forward the corresponding PDUs (protocol data units). This information has to be conveyed to the gateway in some way.

One solution would be to encapsulate the target address with some other data in the Initialize-request. The main advantage of this solution is that both connections are created at the beginning of the session (as opposed to the second solution described later on). Additionally, there is a true end-to-end parameter negotiation among the client and the server (by means of the initialization procedure), since the Initialize-request (response) is forwarded untouched to the target (origin) system. This interaction can be seen in the following diagram:

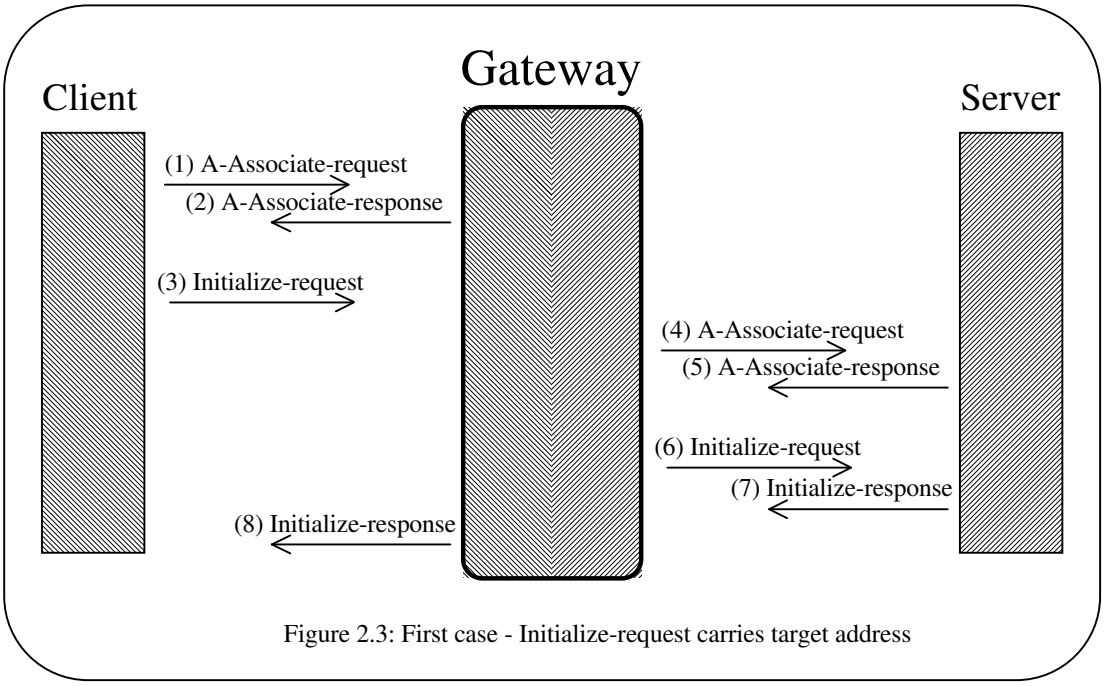


Figure 2.3: First case - Initialize-request carries target address

Another solution would be to encapsulate the target address in the first Search-request and more precisely in the database field of a Search PDU. In this way the Initialization procedure is executed in two steps; first between the origin system and the gateway:

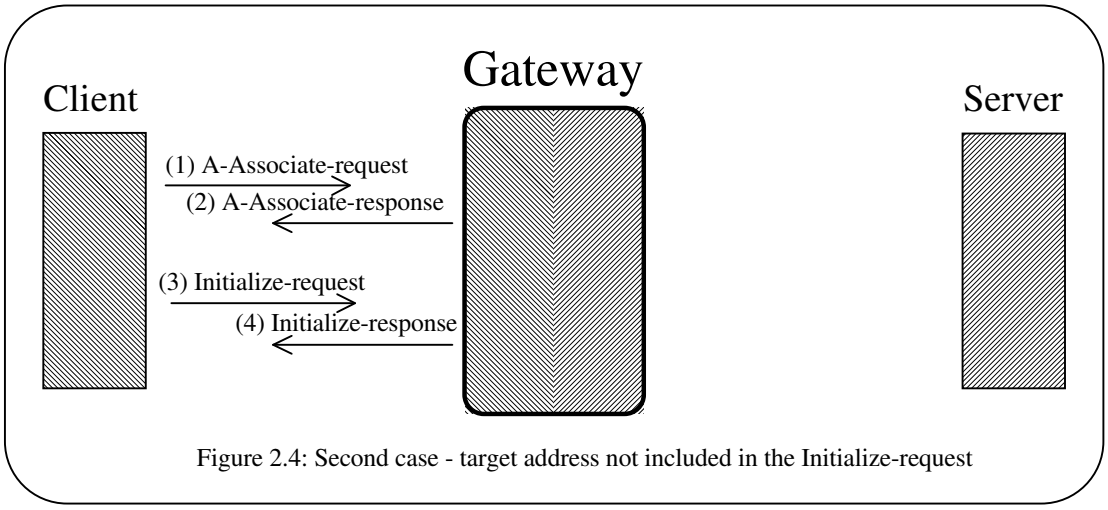


Figure 2.4: Second case - target address not included in the Initialize-request

When the first Search-request reaches the gateway, the second communication path (gateway to target) can be established. At that point, the gateway will perform the Initialization procedure with the target, send the Search-request and return back the results to the origin system with a certain delay. Of course, this delay exists only at the beginning and after that normal operation is possible. The whole procedure is depicted in the following diagram:

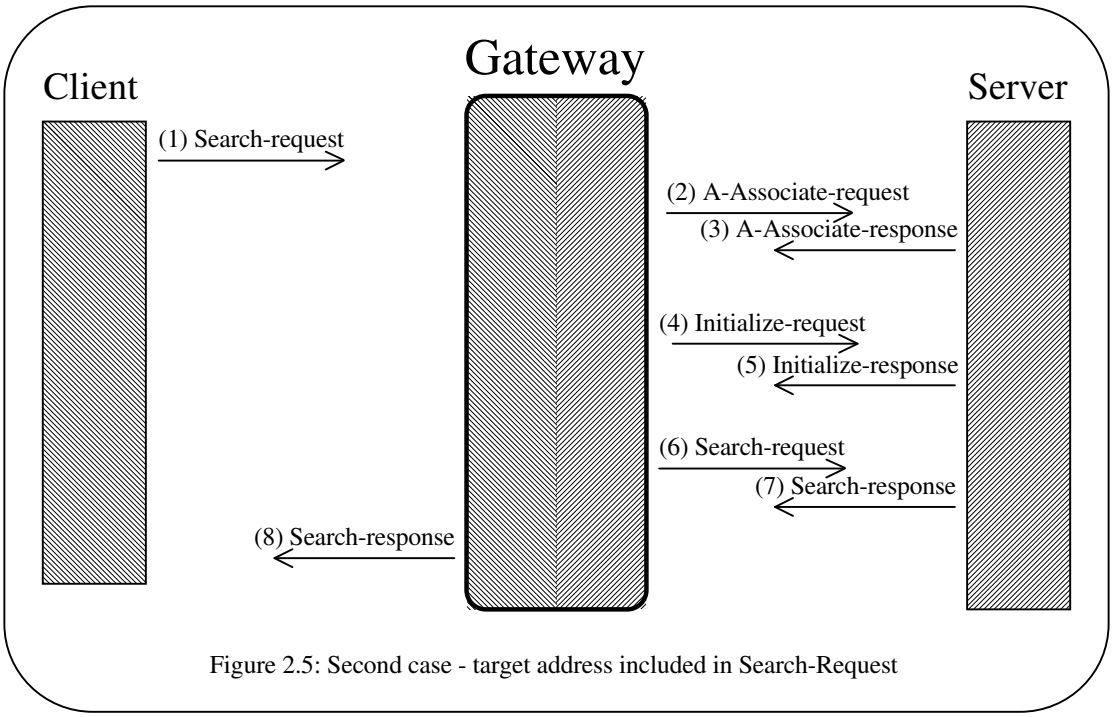


Figure 2.5: Second case - target address included in Search-Request

*S2.2.1.d* *The gateway will work in a staging way by forwarding requests and responses as appropriate.*

*S2.2.1.e* *Addressing information will be encapsulated either in the Initialise or the first Search request.*

### 2.2.1.4 Multiple connections

During normal operation, more than one Z39.50 and ISO-SR client will require the use of the gateway service at the same time. For this reason, the gateway should be able to accept multiple connections simultaneously from both communication interfaces. For the initial development, at least 10 simultaneous connections should be supported. The design of the gateway should avoid placing fixed limits on this, by using dynamically allocated tables etc.

*S2.2.1.f* *The gateway will be capable of supporting at least 10 concurrent associations, and the upper limit should not be fixed by internal aspects of the gateway.*

### 2.2.1.5 Performance

The gateway should meet the following performance targets with a load of 10 users when running on the equipment currently available (Sun SPARCstation IPC or similar speed UNIX workstation):

- 10 seconds to process a request.
- Maximum of 0.5 second per record to convert data.

*S2.2.1.g The gateway should not spend more than half a second to map a bibliographic record from one MARC format to another, at the maximum user load.*

*S2.2.1.h The gateway should not spend more than 10 seconds to receive, process and forward a request or response as a whole, at the maximum user load.*

### 2.2.2 Enhanced functionality

After the desired interoperability has been achieved, the gateway should aim towards enhanced functionality by offering additional services to both the end-user and the administrative user. These services include resource management, logging, accounting, statistics and MARC record translation. These services are not optional for the project but they have a lower priority compared to the services defined in the Basic Functionality section. Their final details will be specified during the system design phase.

#### 2.2.2.1 MARC Translation

Bibliographic data are transferred using one of the many available MARC formats (e.g. USMARC, UNIMARC). Different servers use different formats to transmit records across the network. Thus a client might not be able to interpret correctly the resulting records if it does not support the same MARC format as the server, although some servers will perform record translation according to client's preferred record syntax.

On the other hand, full MARC record mapping among the various formats is at best difficult and maybe impossible. However, it should be possible for the gateway to translate the most important information contained in a MARC record. Proper design of the gateway will provide easy extensibility in the gateway's MARC translation module, in order to accommodate future possible record mappings. In the initial implementation the following formats should be supported:

- UNIMARC
- USMARC
- UKMARC

The gateway will be able to map records from one of these formats to any of the others, at least for the fields listed in Appendix A.

*S2.2.2.a The gateway should be able to translate between the UNIMARC, USMARC and UKMARC formats.*

#### 2.2.2.2 Resource management

The purpose of resource management is to control usage and availability of the system resources used by the gateway. The resource management module should ensure that the gateway's use of certain system resources does not exceed some well-defined limits. The

administrator must be able to specify maximum and/or minimum (as appropriate) values for use of these resources, either as a configuration read at start-up time or while the gateway is running (i.e. changing values should not require the gateway to be stopped and restarted with the loss of established connections).

The following limits for connections should be specifiable:

- maximum connection lifetime
- maximum idle time for a connection
- maximum total data passed on a connection in each direction

The following limits should be specifiable for the system as a whole:

- maximum number of concurrent connections (see also Section 2.2.1.4)
- maximum sizes of log files

If a connection exceeds allowed limits, it should be shut down with an error notified to the client and server, and a log message recorded by the gateway.

Connections in excess of the maximum should be refused.

Excessively large log files should be notified immediately to the administrator but responsibility for truncating them remains with the administrator to prevent loss of data.

*S2.2.2.b* *The gateway administrator should be able to specify: maximum connection lifetime, maximum idle time for a connection and maximum total data passed on a connection in each direction.*

*S2.2.2.c* *The gateway administrator should be able to specify for the system as a whole: maximum number of concurrent connections and maximum sizes of log files.*

#### 2.2.2.3 Accounting

The accounting facilities provided depend on the operational scenario selected. For the pilot service, end users will have free access to the gateway and subscription will be done by the server providers. As a result it is not necessary for the purposes of this project to provide facilities for end user authentication.

Accounts for charging use of the gateway to the server providers will be generated based on the logged information from the gateway's operation. The creation of these accounts is therefore not a function of the gateway itself.

#### 2.2.2.4 Logging of Operational Data

The purpose of keeping detailed log files of system operation is to administer gateway users, ease software debugging, fine-tune the system and prepare various statistics. An effective mechanism should provide for different levels of logging (none/basic/detailed) and should be able to record different types of information:

- General information pertaining to fundamental system operations, like start-up, shut-down, error reporting, failed connection attempts, resource limits exceeded, configuration changes, etc.

- Information specific to users and connections: connection time and duration, network address, databases searched, number of queries, hits, transferred records, etc.

- The gateway's low level interactions and exchanged protocol data units, mainly for debugging and performance analysis purposes.

The detailed information logged at each logging level (none/basic/detailed) will be identified in the design of the system. The level of logging should be configurable at start-up or while the gateway is running without the need to shut down and restart.

# 3 Email Gateway to Z39.50 and SR services

## 3.1 Requirements

The email gateway, later referred to as simply as the gateway, will make it possible to use a standard email client to communicate with SR or Z39.50 servers. We shall look at the requirement for the gateway from three points of view: the end user, the database provider and the gateway administrator. Furthermore, some general requirements will be added to cover specific aspects of this gateway. The goal of this section is only to identify the requirements; the actual specification and precise objectives of the gateway will be found in the next section, Specification.

### 3.1.1 General requirements

One of the most challenging aspects of this gateway is interfacing a connection-less service such as mail and connection-oriented protocols such as SR or Z39.50. Such interfacing is usually linked with a loss of functionality from the connection-oriented protocols. A major goal of this gateway should be to provide an interface where the loss of functionality is minimal or none. Such an interface will also be valuable for future gateways which have to deal with the same connection-less versus connection-oriented problem. The design of this gateway should be such that the part dealing with this particular problem can be reused for future projects. This of course is valid for any part which will be identified as potentially re-usable during the design phase.

*R3.1.1.a* *Interfacing between connection-less and connection-oriented should cause a minimal loss of functionality*

*R3.1.1.b* *Elements in this gateway which solve general problems should be designed to allow a potential re-use within another gateway*

### 3.1.2 End User Requirements

The end user will require to have the complete functionality of an SR or Z39.50 client using a standard email client. However, some services will not be relevant to an email gateway and others might have a null function on the gateway although this will remain transparent to the user. Due to the complexity of the SR and Z39.50 protocols, the gateway will have to act as a smart interface encoding the user requests and decoding the server replies.

#### 3.1.2.1 Interface

The user interface as such is the users' email interface (mail client). The user requirement is that any email interface can be used. The user interaction with the gateway and thus with the SR or Z39.50 servers will be done through a command language and a query syntax. The user will

expect to receive clear text replies. This implies that the MARC records returned to the user will have to be translated to clear text (display records).

*R3.1.2.a Any email client can be used against the gateway*

#### 3.1.2.2 Command language

The command language must be intuitive, fast to learn and fast to use. It must give access to the full functionality of the SR and Z39.50 protocols, although not all commands might actually be linked to a service. The SR and Z39.50 client needs a well-defined set of parameters to function correctly, e.g. target node, database name, etc. Whereever possible the parameters should have a default value thus minimising the initial communication with the user. This default value should, for instance, apply to the number of records to retrieve in a successful search. A user who changes default values will obtain the new values on subsequent requests. The modified values will become user-specific default values for the subsequent searches. A log of the syntactic analysis of the command as well as an execution log should be available for the user and returned automatically in case of error**.**

*R3.1.2.b The command language must be intuitive, fast to learn and fast to use*
*R3.1.2.c The command language must give access to the full functionality of the SR and Z39.50 protocols*
*R3.1.2.d Parameters will have system and user defined default values*
*R3.1.2.e A log of the syntactic analysis of the command as well as an execution log should be available to the user*

#### 3.1.2.3 Query language

In the SR and Z39.50 standards a query language with Reverse Polish Notation (RPN) is defined. RPN is not an obvious choice for a command line language. A more intuitive and possibly standard query language should be chosen.

*R3.1.2.f An intuitive and possibly standard query language should be chosen*

#### 3.1.2.4 MARC records display

MARC format codes are not obviously understandable by the user, and SR and Z39.50 clients usually convert these codes to clear text. The gateway should in principle handle MARC format translation. In this gateway, the MARC record are intended for display (i.e. must be displayed in a mail file to the user) which is definitely easier than conversion to another MARC format. As a minimum the gateway should display UNIMARC, USMARC and UKMARC. For other MARC formats, the gateway should translate a set of fields common to all MARC formats and leave other fields as is.

*R3.1.2.g As a minimum the gateway should display UNIMARC, USMARC and UKMARC*
*R3.1.2.h The gateway should translate a set of fields common to all MARC formats*

### 3.1.3 Database Provider Requirements

#### 3.1.3.1 User authentication

Some commercial or access restricted services will require user authentication. Authentication should be supported by the gateway inasmuch as the SR and Z39.50 protocols support it. According to the operational scenario chosen, no authentication will be required in the gateway. The gateway will only pass on the request for authentication to the user and return the user account and password to the target.

*R3.1.3.a The gateway will only pass on the request for authentication to the user and return the user account and password to the target*

#### 3.1.3.2 Resource control

The gateway should add as little overhead as possible to the database provider servers. The added overhead should be controlled to limit the usage of some resources for specific servers. This can include maximum resource usage for one user on one server and overall usage of a server by the gateway.

*R3.1.3.b The gateway should add as little overhead as possible to the database provider servers*
*R3.1.3.c The gateway overhead should be controlled to limit the usage of some resources for specific servers*

#### 3.1.3.3 Usage logs

The gateway should provide a complete log of all operations with the clients and the servers. These logs will be used in calculating and auditing the use of the gateway to access specific servers.

*R3.1.3.d The gateway should provide a complete log of all operations with the clients and the servers*

### 3.1.4 Gateway administrator Requirements

#### 3.1.4.1 Installation

The installation of the gateway will probably be system dependant as the program must receive mail from the mail system. The gateway should if possible rely on standard UNIX and limit its use of special toolkits. The gateway should not be mail system dependent but rather use an approach which allows the use of many UNIX mail systems. A user manual for the installation-procedure must be written during the development phase.

*R3.1.4.a The gateway should rely on standard UNIX and limit its use of special tool kits*
*R3.1.4.b The gateway should not be mail system dependent*

#### 3.1.4.2 Security

Security aspects must be analysed during development. The actual gateway software must not be a threat to the host computer.

A potential danger lies in the use and abuse of external database resources. From the point of view of the database administrator, the gateway stands as the resource consumer, at least in the cases where no authentication is used. Avoiding misuse by single users must be considered. It must also be investigated in which cases the originator of a mail request is traceable.

*R3.1.4.c The actual gateway software must not be a threat to the hosting computer*
*R3.1.4.d The database providers should not be threatened by the gateway nor should they be able to hold the gateway responsible for misuse.*

#### 3.1.4.3 Resource control

The administrator should be able to control and limit the resource usage of the gateway. This applies, in this case, to the use of local resources such as number of concurrent connections, disk space usage for caching, etc.

*R3.1.4.e The administrator should be able to control and limit the resource usage of the gateway*

## 3.2 Specification

### 3.2.1 Communication interfaces

As a gateway between SR and Z39.50 is also being developed, the mail gateway needs only to interface to one protocol stack as the other can be accessed through the SR/Z39.50 gateway.

The SR environment is currently working on top if the ISODE tool kit whereas the Z39.50 environment is based on TCP/IP only. The ISODE tool kit and protocol stack is a large software package that consumes quite a lot of system resources, e.g. disk space and CPU and it is non-trivial to install. To ease installation and to facilitate porting it is preferable to avoid using tool kits such as ISODE.

In order to ease the implementation and to maximise portability, we choose to implement the mail to Z39.50 gateway, i.e. SR targets/databases can be accessed through the Z39.50/SR gateway (See figure 3.2). The portability is increased as the package needs no ISODE stack on the gateway node; in fact, it will run as an independent package. The following figure illustrates the basic communication interfaces.

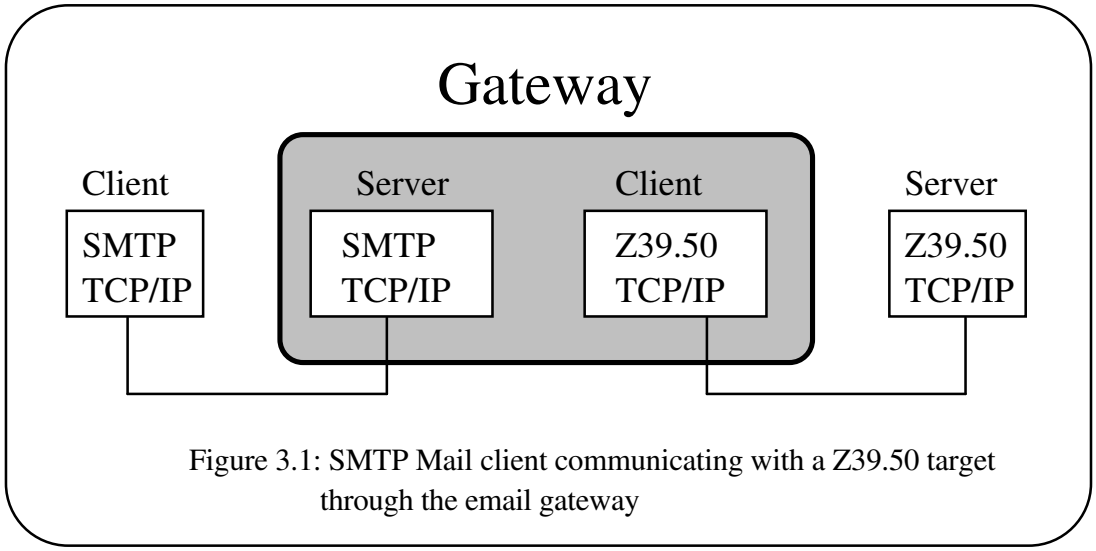


Figure 3.1: SMTP Mail client communicating with a Z39.50 target through the email gateway

The access to SR targets is through the mail gateway and further on through the Z39.50 to SR gateway :

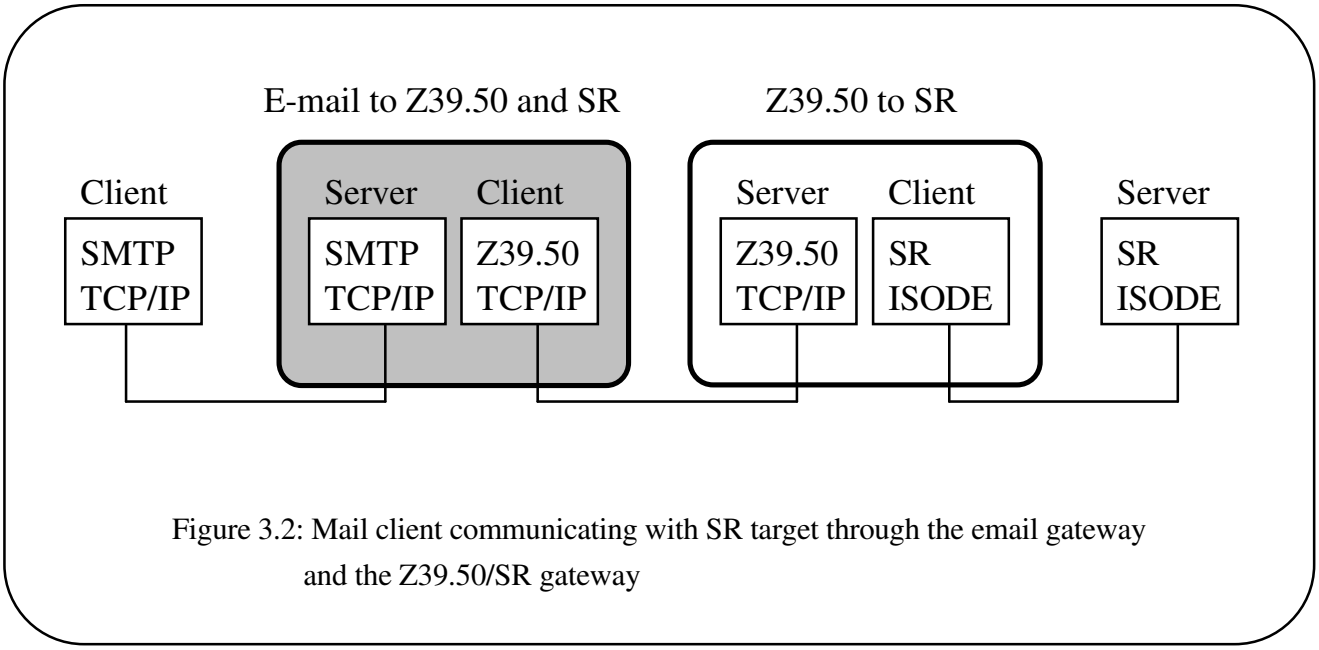


Figure 3.2: Mail client communicating with SR target through the email gateway and the Z39.50/SR gateway

For those email users not directly on the Internet, access is possible through an external mail gateway, for instance an X.400 mail gateway :

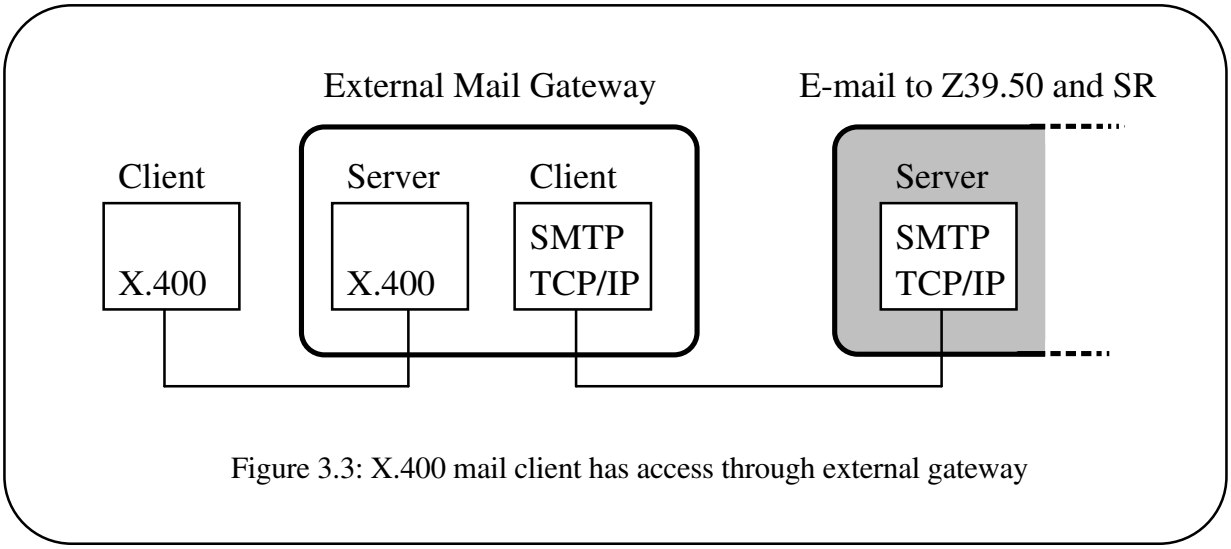


Figure 3.3: X.400 mail client has access through external gateway

*(Requirements:* R3.1.4.a)

### 3.2.2 Basic functionality

The basic functionality of a mail gateway to SR and Z39.50 targets is as follows. A user with the ability to send and receive mail can search SR and Z39.50 databases. A request will either be an actual search query (find request) or a request for records (show request).

The find request will typically hold information about which target and database to query, as well as the actual query with search terms. The reply to a find request will be the number of hits and eventually a set of records that is readable right away. The show request will hold information on which records to show and therefore refer to a previous find request. The reply is a set of records available in a readable format.

#### 3.2.2.1 Connection-less vs. connection-oriented service

The difficult part in the gateway is to handle the time period between the find request and the show request or between a find request and another find request that refers to the previous one. This is due to the fact that SR and Z39.50 are connection-oriented protocols/services. There is no trivial way to implement a connection-less service, like mail, on top of a connection-oriented service. The context of an SR/Z39.50 client is kept in the server as long as the client is connected and will be lost when the connection is closed. It is not yet possible to ask the server to store the context of a client and later restore it in order to continue the session.

There are a number of solutions to the problem:

- Re-use of physical connection
  Over a short period of time, the connection can be left open by the gateway so that a request from the same client on the same server can re-use that connection. The time

limit is imposed by the server which will time-out a connection after it has been idle for a certain time.

- Restoring the user context in a new connection
  If the connection has been closed the client context can be restored by reconnecting to the server and reissuing the previous client commands. Of course, the restoring of the context is needed only if the client, in its new requests, refers in some way to a previous search request. The full context will probably not be restored, but rather only what is referred to in the new request.

- Caching to overcome problems of closed connections
  The gateway requests the records resulting from the last search and caches them in order to be able to answer a present request even when the connection has been closed/lost. This solution can only be used in case of a small set of records and could cause some problems when dealing with commercial servers.

- Disallow reference to previous searches
  If none of the above solutions are valid for a specific server then references to previous searches could be disallowed by the query parser. This is the simplest but least elegant solution. It implies that a present request must automatically be included with a search request.

We will aim at implementing all of the above solution in the email gateway leaving the choice of which method to use for which server to the gateway administrator, in accordance with the database providers. This should allow the administrator to balance between the level of functionality and the overhead to the database providers' servers.

The scenario when a request arrives at the gateway will be the following*:

- Find request
  - Check a database of open connections to see if that client has an open connection to the requested server
  - Connection
    - If any reference to previous search are not in this connection context, restore context
    - Submit request
  - No connection
    - Initialise connection
    - If there is any reference to a previous search, restore context
    - Submit request
- Present request
  - Check if requested records have been cached
  - Cached
    - Return cached records to client
  - Not cached
    - Check a database of open connections to see if that client has an open connection to the requested server
    - Connection

- If requested set is not in this connection context, restore set
- Submit request

- No connection
  - Initialise connection
  - restore set
  - Submit request

* That scenario does not include the case where reference to previous searches is disallowed. In this case, the scenario is much simpler:

- Any request
  - Check for reference to previous request (not part of current group of commands)
  - Reference
    - Return error message
  - No reference
    - Check a database of open connections to see if that client has an open connection to the requested server
    - Connection
      - Submit request
    - No connection
      - Initialise connection
      - Submit request

*(Requirements: R3.1.1.a, R3.1.3.b)*

#### 3.2.2.2 Email system dependency

It is clear that the gateway will have to face a large number of email systems. It will receive mail from a variety of systems and possibly be installed under various flavours of UNIX mail system. As a general rule, we will insure that the email gateway will handle standard Internet email messages (RFC 822). The gateway will however be designed to be configurable and thus adaptable to different email formats. It appears that the general formatting of the mail is more dependent on the mail server that receives the mail then on the client that sends it. Thus, the variety of mail clients should not be a problem. Nevertheless, two things can be done to insure that any email client can be used against the gateway:

- The user commands and query should be placed in the body of the mail
  The body of the mail is usually quite easy to locate and it allows multiple lines thus allowing multiple command and parameter setting within one email message.
- Delimiters should mark the beginning and the end of the command block
  This is of course an inelegant way of solving the problems which might occur with a few mail clients but it is the only way to have a totally secure way to locate the commands within an email message. This of course should be optional so that it only has to be used in case of misinterpretation by the gateway.

As the formatting of the email message is influenced mainly by the mail server, making sure that the gateway is not mail system dependent is more tricky. Four point will require a co-ordination between the mail system and the gateway:

- The email message must be passed to the gateway
  Most UNIX mail systems allow an email message to be piped to a process. This is the technique we are planning to use to get the message.
- The return address of the sender must be extracted from the message
  This can be system dependent. The only solution is to allow the gateway administrator to specify the tag or tags which mark this information in the mail message (e.g. Reply-To:, From:).
- The body of the message must be extracted
  Here again the administrator should be given the possibility to specify the beginning and end of body (e.g. LFLF, EOF), if body delimiters are not used.
- The reply must sent back to the user
  The mail will be sent by spawning the mail command with the appropriate parameters and input file. Here again, this should be customisable.

*(Requirements: R3.1.2.a, R3.1.4.b)*

#### 3.2.2.3 Command languages and query syntax

The commands and query will be the end-user's way to interact with the gateway and thus with the SR and Z39.50 server.

The choice of a query syntax is a simpler matter as standards are available. We have chosen to use CCL (ISO 8777) as a query language for the gateway. The gateway will have to convert the CCL request to RPN requests which are standard for SR and Z39.50. The CCL to RPN conversion will work on a well defined subset of CCL as basic functionality and if needed on the full CCL as enhanced functionality.

The choice of a command language is a more complex matter. To our knowledge, there is no standard for this. For the commands related to SR or Z39.50 we will use what is currently used by the line mode clients. This seems to be as close as we can get to a standard. For commands directly related to the gateway, we can only do our best to choose intuitive commands. Allowing the gateway administrator to add aliases for a command will also be taken into account while designing the command parser.

The parameters to commands will have default values. The system defaults will be customisable by the gateway administrator while the user default will be customisable by the end user and kept with the user profile. System defaults will override database server defaults if specified. Because of the lack of a "session" with the gateway, the user modified parameters will be kept until the user reverts to system default or the user profile is timed out. For the same reason, even things like the target name and database name will be considered as parameters and kept until changed. This will allow the user not to have to specify these every time.

*(Requirements: R3.1.2.f, R3.1.2.c, R3.1.2.b, R3.1.2.d)*

#### 3.2.2.4 Resource management

The purpose of resource management is mainly to control usage and availability of those system resources used by the gateway. The resource management module should ensure that certain system variables do not exceed well-defined limits.

The resource management for the gateway will be of two types:

- Management of gateway host resources

  The gateway host will have finite resources to allocate to the gateway. Therefore, the gateway administrator should have control over resource usage e.g. number of concurrent connections, total connection time for each user, connection time-out, caching disk space. A complete list of these resources will be defined during the design phase as well as the way this resource control will be enforced.
- Management of service providers resources

  Resource management will also be needed to protect target systems and databases against overload and misuse. The gateway should be able to limit its resource usage on specific server. Resources to control will be the number of concurrent connections, total connection time, connection time per user, number of requests per user, etc. A complete list of these resources will be defined during the design phase as well as the way this resource control will be enforced. Of course, the gateway can only be expected to enforce resource management for resource usage apparent from the client side. When the gateway does not know the CPU usage or memory usage on the database provider server, it cannot enforce limits on these.

  (Requirements: *R3.1.4.e*, *R3.1.3.c)*

#### 3.2.2.5 Authentication/Security

Because of the operational scenario chosen for the development and pilot project, there will be no need for authentication in the gateway. Authentication will only take place (if needed) between the database provider and the end-user. The gateway will only pass on authentication request from the target and return the end-user reply to the target. This will avoid security problems linked to password storage, leaving only to consider during design the security towards the end-user accounts and passwords when they transit through the gateway.
It will be the gateway manager's task to set the record caching and connection restoring for commercial servers so that the end-user will not be charged for data he has not seen or requested. Because of the operational scenario chosen, the gateway will not have to act as a customer toward the targets nor will it have to identify itself on behalf on the end-user. The security considerations of the database provider will remain the same as when used by a standard SR/Z39.50 origin.
Some investigation will be done during design concerning the security problem linked with email, especially concerning the location and authentication of the email sender. The gateway should be designed to avoid consequences of email forgery.

*(Requirements: R3.1.3.a, R3.1.4.c, R3.1.4.d)*

#### 3.2.2.6 Logging/Statistics

The purpose of keeping detailed log files of system usage is to administer the gateway, ease software debugging, fine-tune the system, and prepare various statistics. An effective mechanism should provide for different levels of logging (Administrative, Statistic, Debugging, etc.) and should be able to record different types of information:

- General information pertaining to fundamental system operations, like start-up, shut-down, error reporting, failed connection attempts, etc.
- Information specific to users: connection time/duration, databases searched, number of queries, hits, transferred records, etc.
- The gateway's low level interactions and exchanged protocol data units, mainly for debugging and statistical purposes.

Logging will be required at various levels. Some of the module logs will be fed back directly to the user. Some log information will be sorted by server and fed back to the service providers or stored for their possible needs. Finally some logging information will be used by the gateway administrator to control and fine tune the gateway, and for purely administrative purpose.

All modules should be able to provide some level of logging information although logging itself will be centralised and controlled by the Administrative Server described in section four of this document. With the exception of information fed back directly to the user and of low level debugging information, logging information will not be available directly from the gateway but from the administrative server. Tools using this information for statistical, administrative or functional purpose are therefore independent from the gateway and will be considered as enhanced functionality.

*(Requirements: R3.1.2.e, R3.1.3.d, R3.1.2.e)*

#### 3.2.2.7 MARC records display

Records returned by a target after a search or present request are encoded in a particular *flavour* of the MARC format. A standard SR/Z39.50 client will decode these records by first unpacking them and then translating the various MARC field codes to clear text which can be easily understood by the user. The email gateway will have to substitute the SR/Z39.50 client by providing this function to the end-user. The gateway will support the complete list of fields for specific MARC formats (UNIMARC, USMARC and UKMARC) and will provide a basic translation of the most important MARC field codes which are usually similar in all MARC formats. In case of a partial translation, field codes which the gateway fail to translate will be passed along leaving the interpretation to the end-user.

*(Requirements: R3.1.2.g, R3.1.2.h)*

### 3.2.3 Software Layering

The above technical aspects are not unique for a mail gateway. Other gateways to connection-less services face the same problem. Therefore this software should be layered so that other services could easily be built on top later on. The actual layering will be defined in the design

phase but some high level logical functions should be closely related to the commands defined in the user interface. The underlying connection and context handling should be transparent.

*(Requirements: R3.1.1.b)*

# 4 Administrative server

## 4.1 Requirements

The Administrative server will be a utility that the gateways described in Chapters 2 and 3 of this document can use to store information in a centralised manner. The administrative server will come in two parts: a central server and a client programming interface to access this server.

### 4.1.1 General requirements

The administrative server should allow the centralised storage of a variety of information. This will include the gateways' transaction logging, resource control and authentication data. The administrative server should be designed not to fulfil the exact needs of the gateways, because these will change over time, but rather to provide a flexible way to handle data coming in a variety of formats. The server should have two distinct sets of functions. One set will be used by the gateways while the other, more comprehensive, will be used locally on the server host by the server administrator. This latter set could also be used by software dealing with further processing of the data stored in the server (e.g. statistical analysis of the transactions logging).

*R4.1.1.a* *The server should provide a flexible way to handle data coming in a variety of formats*

*R4.1.1.b* *The server should provides functions for the gateways' use as well as functions for local data management*

### 4.1.2 End User Requirements

The end-users of the administrative server will be the programmers developing the gateways, who will only use it through its programming interface.

#### 4.1.2.1 Interface

The programming interface should provide the programmers with a set of functions that covers the logging, resource control and authentication needs of the gateway. These functions should make the communication with the server transparent, the programmers using them will not need to know about the functions/server interface.

*R4.1.2.a* *The programming interface should provide functions for logging, resource control and authentication*

*R4.1.2.b* *Communication between the programming interface functions and the server should be transparent to the gateway*

#### 4.1.2.2 Security

Two security aspects should be considered for the administrative server: the security of the access to the server and the security of the data stored using the server. Access to the server should be controlled as well as the type of operations a gateway can perform on the server and the data on which it can perform these operations. Some of the data accessible through the server, like authentication data will also have to be protected. The authentication data will possibly grant access to the gateway but may also grant access to other commercial servers thus making security a critical point. The administrative server should provide some encryption mechanism to protect this data from unauthorised access.

*R4.1.2.c* *Access to the server as well as operations on the server, once access has been granted, should be controlled*
*R4.1.2.d* *Restrictions base on ownership should be available for data manipulation through the server*
*R4.1.2.e* *The server should provide data encryption for security sensitive data*

### 4.1.3 Server Administrator Requirements

The server administrator will need five sets of tools to:

- Grant or deny gateway access to the server
- Define the files and data formats
- Start and stop the server
- Manipulate the data stored through the server
- Backup and restore this data

Not all of these functions will have to be implemented within the administrative server as some of them, such as backup/restore and data manipulation, could be available in the underlying file or database management system. Nevertheless, the functionality and support for these operations should be there.

*R4.1.3.a* *The server should provide user management tools to control gateways' access and privileges*
*R4.1.3.b* *The server should provide ways of defining new data files and data custom formats*
*R4.1.3.c* *There should be some secure way to stop and restart the server without any lost of ongoing transactions*
*R4.1.3.d* *The local programming interface should allow standard data manipulation (READ/WRITE/DELETE)*
*R4.1.3.e* *Some function should offer secure backup and restore of some or all data stored through the server*

## 4.2 Specification

### 4.2.1 Communication interfaces

The communication between the server and the gateways will take place over TCP/IP using the Berkeley Sockets programming model. The communication interface should however be layered so that it could be replaced by another interface if needed.

### 4.2.2 Basic functionality

The basic functionality of the administrative server is to allow multiple gateways to store information coming in different format in a central repository. The three types of transactions the server should handle are:

- Transaction logging, where the gateways only write new transactions.
- Resource control, where the gateways can notify of an increase or decrease of a specific resource usage and can obtain a status of a resource usage
- Authentication, where the gateways can register a new account or have the server to authenticate an account.

Furthermore, the server will provide functions for further processing of some of the data, mainly transaction logs. These functions will not be used by the gateways but rather by the server administrator or software that handles this post-processing of data.

#### 4.2.2.1 Transaction logging

The transaction logging is the simplest of all the operations. It will consist of sequentially storing data coming from multiple gateways making sure that no conflict occurs. The data will be stored with information on who, which gateway, requested the logging and when this request took place. The format of the actual data will be free as far as the administrative server is concerned but it is clear that a pre-defined format will have to be agreed upon by the gateway developers. This format will be pre-defined on the administrative server which will make sure the data is stored according to this format. The content of the data will not be checked by the server and it is left to the gateways to provide the right content. Functions local to the server will allow later retrieval of this data and deletion when the data is no longer needed.

#### 4.2.2.2 Resource management

The server will allow gateways to set the usage value of a resource, increase it, decrease it and consult it. The resource values will be of two types: resources specific to one gateway and resources common to two or more gateways. Resources common to many gateways will only be viewed as common when consulting the resource usage; other type of operations will have to be specific to each gateway. It will be the responsibility of the gateway to maintain its resource usage values (e.g. a gateway upon shutting down or restarting will have to ensure that it resets its resource usage values).

#### 4.2.2.3 Authentication

Authentication will be handled as a normal input/output operation with encrypted fields. Fields mark for encryption will be encrypted before the input/output operation. Thus an account password, for example, will be encrypted before a write when storing a new password and before comparing it with the encrypted password when authenticating a password.

# A Supported MARC Conversions

The following fields appear in most MARC formats, therefore it should be possible to translate them when converting from one MARC format to another. The tags and the corresponding descriptions come from the UKMARC manual and they are presented in ascending order. Only the first two fields (001 and 008) are mandatory in any MARC record, while the most commonly used (and thus important) fields are marked as bold in the table.

| TAG | DESCRIPTION |
|---|---|
| **001** | **Record control number** |
| **008** | **Information codes, e.g. entry date, publication date, nature of publication, country of publication, illustration codes, material designation etc.** |
| **21** | **ISBN** |
| **22** | **ISSN** |
| **50** | **Library of Congress classification number** |
| 80 | Universal decimal classification number (old edition) |
| **82** | **Dewey decimal classification number (current edition)** |
| **100** | **Personal name (main entry heading)** |
| **110** | **Corporate name (main entry heading)** |
| 111 | Conference, congress, meeting, etc. |
| 222 | Key title |
| 240 | Uniform title -excluding collective title |
| 243 | Collective title |
| **245** | **Title & statement of responsibility area** |
| **250** | **Edition area** |
| 254 | Mathematical data area |
| **260** | **Publication, distribution etc. area** |
| **300** | **Physical description area** |
| **440** | **Series area - Title of series in added entry heading form** |
| 490 | Series area - Title of series not added entry heading form |
| **500** | **Nature, scope of artistic form note** |
| 501 | “with” note |
| 502 | Dissertation note |
| 503 | Edition & history note |
| 504 | Bibliography & index note |
| 505 | Contents note |
| 508 | Statements of responsibility note |
| 513 | Summary note |
| 514 | Title proper, parallel title & other title information note |
| 515 | Numbering & chronological designation |

| | |
|---|---|
| 516 | Mathematical & other cartographic data note |
| 521 | Audience note |
| 525 | Accompanying materials & supplements note |
| 528 | Publication, distribution note |
| 530 | Other versions available note |
| 531 | Physical description note |
| 532 | Series note |
| 534 | Reference to published descriptions note |
| 536 | Not relating to original |
| 537 | Program note |
| 538 | Numbers borne by the item note |
| 542 | Mode of use note |
| 546 | Language of the item |
| 554 | Frequency note (serials) |
| 555 | Indexes note |
| 556 | Item described note |
| 557 | Graphic index note |
| 600 | Personal name subject heading |
| 610 | Corporate name subject heading |
| 611 | Conference, congress, meeting, etc. |
| **640** | **Uniform title subject heading** |
| **645** | **Title subject heading** |
| **650** | **Topical library of Congress** |
| 651 | Geographical library of Congress subject heading |
| **700** | **Personal name added entry heading** |
| 710 | Corporate name added entry heading |
| 711 | Conference, congress, meeting, etc. added entry heading |
| 745 | Title added entry heading excluding uniform titles |
| 800 | Personal author series added entry heading |
| 811 | Corporate series added entry heading |
| 840 | series title added entry heading |

# B Requirements Checklist

| Requirement Number | Description |
|---|---|
| R1.5.1a | The gateway should interoperate with existing clients |
| R1.5.2a | The gateway should allow database providers to subscribe, obtain accounting information and statistics from the gateway service. |
| R1.5.3.a | Installation of the gateway should be easy and well documented. |
| R1.5.3.b | The gateway should be accompanied with a basic set of administrative scripts. |
| R1.5.3.c | The gateway should be able to provide sufficient logging information. |
| R1.5.5.a | The gateway should be portable between UNIX platforms which support the necessary communication facilities (sockets and ISODE). |
| R1.5.5.b | The gateway should be easily extensible to accommodate a newer version of the supported protocols. |
| R1.5.5.c | The gateway will support the ISO 10163 Search & Retrieve protocol on top of an OSI communication stack provided by ISODE. |
| R1.5.5.d | The gateway will support the ANSI/NISO Z39.50-1992 (Version 2) Information Retrieval protocol on top of the TCP/IP transport stack. |
| R1.5.5.f | The gateway should be scalable to heavier use in future. |
| R1.5.5.e | The gateway should ensure the confidentiality of sensitive data. |
| R2.1.1.a | Operation of the gateway should be transparent for the end-user and especially for the client software. |
| R2.1.1.b | Addressing (or any additional) information should be encapsulated in common and easily accessible fields. |
| R2.1.1.c | The gateway should be able to perform (partial) MARC record conversion whenever appropriate. |
| R2.1.1.d | The degradation in performance caused by the gateway should be minimal. |
| S2.2.1.a | The gateway will support all services common to the ISO-SR and Z39.50-1992 protocols. |
| S2.2.1.b | The gateway will automatically respond to Z39.50 Access and Resource Control requests, if possible. |

| | |
|---|---|
| S2.2.1.c | The gateway will use existing ISO-SR and Z39.50 (version 2) client and server implementations. |
| S2.2.1.d | The gateway will work in a staging way by forwarding requests and responses as appropriate. |
| S2.2.1.e | Addressing information will be encapsulated either in the Initialise or the first Search request. |
| S2.2.1.f | The gateway will be capable of supporting at least 10 concurrent associations, and the upper limit should not be fixed by internal aspects of the gateway. |
| S2.2.1.g | The gateway should not spend more than half a second to map a bibliographic record from one MARC format to another, at the maximum user load. |
| S2.2.1.h | The gateway should not spend more than 10 seconds to receive, process and forward a request or response as a whole, at the maximum user load. |
| S2.2.2.a | The gateway should be able to translate between the UNIMARC, USMARC and UKMARC formats. |
| S2.2.2.b | The gateway administrator should be able to specify: maximum connection lifetime, maximum idle time for a connection and maximum total data passed on a connection in each direction. |
| S2.2.2.c | The gateway administrator should be able to specify for the system as a whole: maximum number of concurrent connections and maximum sizes of log files. |
| R3.1.1.a | Interfacing between connection-less and connection-oriented should cause a minimal loss of functionality |
| R3.1.1.b | Elements in this gateway which solve general problems should be designed to allow a potential re-use within another gateway |
| R3.1.2.a | Any email client can be used against the gateway |
| R3.1.2.b | The command language must be intuitive, fast to learn and fast to use |
| R3.1.2.c | The command language must give access to the full functionality of the SR and Z39.50 protocols |
| R3.1.2.d | Parameters will have system and user defined default values |
| R3.1.2.e | A log of the syntactic analysis of the command as well as an execution log should be available to the user |
| R3.1.2.f | An intuitive and possibly standard query language should be chosen |
| R3.1.2.g | As a minimum the gateway should display UNIMARC, USMARC and UKMARC |
| R3.1.2.h | The gateway should translate a set of fields common to all MARC formats |

| | |
|---|---|
| R3.1.3.a | The gateway will only pass on the request for authentication to the user and return the user account and password to the target |
| R3.1.3.b | The gateway should add as little overhead as possible to the database provider servers |
| R3.1.3.c | The gateway overhead should be controlled to limit the usage of some resources for specific servers |
| R3.1.3.d | The gateway should provide a complete log of all operations with the clients and the servers |
| R3.1.4.a | The gateway should rely on standard UNIX and limit its use of special tool kits |
| R3.1.4.b | The gateway should not be mail system dependent |
| R3.1.4.c | The actual gateway software must not be a threat to the hosting computer |
| R3.1.4.d | The database providers should not be threatened by the gateway nor should they be able to hold the gateway responsible for misuse. |
| R3.1.4.e | The administrator should be able to control and limit the resource usage of the gateway |
| R4.1.1.a | The server should provide a flexible way to handle data coming in a variety of formats |
| R4.1.1.b | The server should provides functions for the gateways' use as well as functions for local data management |
| R4.1.2.a | The programming interface should provide functions for logging, resource control and authentication |
| R4.1.2.b | Communication between the programming interface functions and the server should be transparent to the gateway |
| R4.1.2.c | Access to the server as well as operations on the server, once access has been granted, should be controlled |
| R4.1.2.d | Restrictions base on ownership should be available for data manipulation through the server |
| R4.1.2.e | The server should provide data encryption for security sensitive data |
| R4.1.3.a | The server should provide user management tools to control gateways' access and privileges |
| R4.1.3.b | The server should provide ways of defining new data files and data custom formats |
| R4.1.3.c | There should be some secure way to stop and restart the server without any lost of ongoing transactions |
| R4.1.3.d | The local programming interface should allow standard data manipulation (READ/WRITE/DELETE) |
| R4.1.3.e | Some function should offer secure backup and restore of some or all data stored through the server |