The Firebird wire protocol

(As implemented in the .NET provider)

Carlos Guzman Alvarez

Revision History

Revision 0.1	31 May 2004	First draft for review.
Revision 0.2	02 Jun 2004	Fixed issues reported by Paul Vinkenoog.
Revision 0.3	03 Jun 2004	Added new subsections to the Statements section.
Revision 0.4	05 Jun 2004	Fixed issues reported by Paul Vinkenoog.
Revision 0.5	06 Jun 2004	Fixed issues reported by Paul Vinkenoog.
Revision 0.6	07 Jun 2004	Added events system documentation.
Revision 0.7	16 Jun 2004	Modifed document ID to wireprotocol.
Revision 0.8	17 Jun 2004	Added two new segmendted lists.
Revision 0.9	18 Jun 2004	Improved segmentedlist usage. Fixed rendering of important tags.
Revision 0.10	19 Jun 2004	Changed rendering of important tags using Paul Vinkenoog fix.
Revision 0.11	20 Jun 2004	Added new segmentedlist. Updated Statements.Prepare documentation. Updated Statements.Execute documentation. Updated Blobs.GetSegment documentation. Updated Blobs.Seek documentation.
Revision 0.12	21 Jun 2004	Updated services information.

Responses

Generic response

Int32

Operation code

If operation equals `op_response` :

Int32
Object handle

Int64
Object ID

Buffer
Data

Byte[]
Status vector

> **Important**
>
> Information about how to parse the status vector can be found in the Interbase 6.0 documentation set

SQL response

Int32
Operation code

If operation equals `op_sql_response` :

Int32
Message count

Buffer
Data row

Fetch response

Int32
Operation code

If operation equals `op_fetch_response` :

Int32
Status

> **Important**
>
> End of cursor is indicated with a non-zero status.
>
> A status with value of 100 means that there are no more rows.

Int32

Count

Slice response

Int32
Operation code

If operation equals `op_slice` :

Int32
Slice length

Int32
Slice length

Buffer
Slice data

Databases

Attach

Attachments to a database are done in two steps.

Identification

Client

Int32
Operation code (`op_connect`)

Int32
Operation code (`op_attach`)

Int32
Version (`CONNECT_VERSION2`)

Int32
Architecture type (*Generic = 1*)

String
Database path

Int32
Protocol versions understood (*1*)

Buffer
User identification

Important

The next block of data should be sent as many times as protocols are supported.

Int32
Protocol version (`PROTOCOL_VERSION10`)

Int32
Architecture type (*Generic* = *1*)

Int32
Minimum type (*2*)

Int32
Maximum type (*3*)

Int32
Preference weight (*2*)

Server

Int32
Operation code

If operation equals `op_accept` :

Int32
Protocol version number

Int32
Architecture for protocol

Int32
Minumum type

Attachment.

Client

Int32
Operation code (`op_attach`)

Int32
Database object id (0)

String
Database path

Buffer
Database parameter buffer

List of parameters sent in the DPB :

Parameter	Description	Value	Optional
`isc_dpb_version1`	Version		
`isc_dpb_dummy_packet_interval`	Dummy packet interval	120	
`isc_dpb_sql_dialect`	SQL dialect	3	
`isc_dpb_lc_ctype`	Character set		
`isc_dpb_sql_role_name`	User role		*
`isc_dpb_connect_timeout`	Connection timeout		
`isc_dpb_user_name`	User name		
`isc_dpb_password`	User password		

Server

Generic response - Where the object handle is the database handle.

Detach

Client

Int32
Operation code (`op_detach`)

Int32
Database handle

Server

Generic response

Create

Client

Int32
Operation code (`op_create`)

Int32
Database object id (0)

String
Database path

Buffer
Database parameter buffer

Server

Generic response - Where the Object handle is the database handle.

Drop

Client

Int32
Operation code (`op_drop_database`)

Int32
Database handle

Server

Generic response

Information request

Client

Int32
Operation code (`op_info_database`)

Int32
Database handle

Int32
Incarnation of object (*0*)

Buffer
Requested information items

Int32
Requested information items buffer length

Generic response - Where Data is holds the requested information.

Transactions

Start transaction

Client

Int32
Operation code (`op_transaction`)

Int32
Database handle

Buffer
Transaction parameter buffer

Server

Generic response - Where Object handle is the new transaction handle.

Commit transaction

Client

Int32
Operation code (`op_commit`)

Int32
Transaction handle

Server

Generic response

Rollback transaction

Client

Int32
Operation code (`op_rollback`)

Int32
Transaction handle

Server

Generic response

Commit retaining

Client

Int32
: Operation code (`op_commit_retaining`)

Int32
: Transaction handle

Server

Generic response.

Rollback retaining

Client

Int32
: Operation code (`op_rollback_retaining`)

Int32
: Transaction handle

Server

Generic response

Prepare

Client

Int32
: Operation code (`op_prepare2`)

Int32
: Transaction handle

Server

Generic response

Statements

Allocate

Client

Int32
: Operation code (`op_allocate_statement`)

Int32
: Database handle

Server

Generic response - Where Object Handle is the allocated statement handle.

Free

Client

Int32
: Operation code (`op_free_statement`)

Int32
: Statement handle

Int32

Option	Description
`DSQL_close`	Closes the statement.
`DSQL_drop`	Releases the statement.

Server

Generic response

Prepare

Client

Int32
: Operation code (`op_prepare_statement`)

Int32
: Transaction handle

Int32

Statement handle

Int32
SQL dialect

String
Statement to be prepared

Buffer
Describe information items

List of requested information items :

- `isc_info_sql_select`
- `isc_info_sql_describe_vars`
- `isc_info_sql_sqlda_seq`
- `isc_info_sql_type`
- `isc_info_sql_sub_type`
- `isc_info_sql_length`
- `isc_info_sql_scale`
- `isc_info_sql_field`
- `isc_info_sql_relation`

Int32
Target buffer length (1024)

Server

Generic response - Where Data holds the statement description (matching the requested information items)

Describe

Describe of output parameters of a query is done using the Statements information request message

List of requested information items :

- `isc_info_sql_select`
- `isc_info_sql_describe_vars`
- `isc_info_sql_sqlda_seq`
- `isc_info_sql_type`
- `isc_info_sql_sub_type`
- `isc_info_sql_length`
- `isc_info_sql_scale`
- `isc_info_sql_field`
- `isc_info_sql_relation`

Describe bind (input parameters)

Describe of input parameters of a query is done using the Statements information request message

List of requested information items :

- `isc_info_sql_select`
- `isc_info_sql_describe_vars`
- `isc_info_sql_sqlda_seq`
- `isc_info_sql_type`
- `isc_info_sql_sub_type`
- `isc_info_sql_length`
- `isc_info_sql_scale`
- `isc_info_sql_field`
- `isc_info_sql_relation`

Execute

Client

Int32
Operation code

Operation	**Usage**
`op_execute`	DDL and DML statements.
`op_execute2`	Stored procedures.

Int32
Statement handle

Int32
Transaction handle

If the statement has input parameters :

Buffer
Parameters in BLR format

Int32
Message number (0) ??

Int32
Number of messages (1) ??

Buffer
Parameter values

If not :

Buffer
Empty (length only 0)

Int32
Message number (0) ??

Int32
Number of messages (0) ??

If the statement is an stored procedure and there are output parameters :

Buffer
Output parameters in BLR format

Int32
Output message number (0) ??

Server

Int32
Operation code

If operation equals `op_sql_response` :

SQL response

if not :

Generic response

Rows affected by query execution

Obtain the rows affected by a query is done using the Statements information request message

List of requested information items :

- `isc_info_sql_records`

Fetch

Client

Int32
Operation code (`op_fetch`)

Int32
Statement handle

Buffer
Output parameters in BLR format

Int32
Message number

Int32
Message count/Fetch size (200)

Server

Int32
Operation code

If operation equals `op_fetch_response` :

Fetch response.

If not :

Generic response.

Set cursor name

Client

Int32
Operation code (`op_set_cursor`)

Int32
Statement handle

String
Cursor name (null terminated)

Int32
Cursor type (0).

> **Important**
>
> Reserved for future use

Server

Generic response

Information request

Client

Int32
Operation code (`op_info_sql`)

Int32
Statement handle

Int32
Incarnation of object (0)

Buffer
Requested information items

Int32
Requested information items buffer length

Server

Generic response - Where Data holds the requested information.

Important

Information about how to parse the information buffer sent by the Firebird server can be found in the Interbase 6.0 documentation set

Blobs

Create/Open

Client

Int32
Operation code

Operation	Description
`op_create_blob`	Creates a new blob
`op_open_blob`	Opens an exiting blob

Buffer
Blob parameter buffer (*optional*)

Int32
Transaction handle

Int64
Blob ID

Server

Generic response - Where :

a. Object handle is the blob handle

b. Blob id is the blob id

Get segment

Client

Int32
  Operation code (`op_get_segment`)

Int32
  Blob handle

Int32
  Segment length (*max length = 32768*)

Int32
  Data segment (0)

Server

Generic response - Where Data is the blob segment.

Put segment

Client

Int32
  Operation code (`op_batch_segments`)

Int32
  Blob handle

Buffer
  Blob Segments

Server

Generic response

Seek

Client

Int32
  Operation code (`op_seek_blob`)

Int32
  Blob handle

Int32
  Seek mode (0)

Int32
  Offset

Server

Generic response - Where ObjectHandle is the current position.

Arrays

Get slice

Client

Int32
    Operation code (`op_get_slice`)

Int32
    Transaction handle

Int64
    Array handle

Int32
    Slice length

Buffer
    Slice descriptor (SDL)

String
    Slice parameters (Always an empty string)

Buffer
    Slice (Always empty)

Server

Slice response

Put slice

Client

Int32
    Operation code (`op_put_slice`)

Int32
    transaction handle

Int64
    Array handle (0)

Int32

Slice length

Buffer
: Slice descriptor (SDL)

String
: Slice parameters (Always an empty string)

Int32
: Slice length

Buffer
: Slice data

Server

Generic response - Where BlobId is the array handle.

Services

Attach

Client

Int32
: Operation code (`op_service_attach`)

Int32
: Database object ID (0)

String
: Service name

For local connections : service_mgr

For remote connections: HostName: service_mgr

Buffer
: Service parameter buffer

Server

Generic response - Where Object handle is the services manager attachement handle.

Detach

Client

Int32
: Operation code (`op_service_detach`)

Int32
: Services manager attachment handle

Server

Generic response

Start

Client

Int32
: Operation code (`op_service_start`)

Int32
: Services manager attachment handle

Int32
: Incarnation of object (0)

Buffer
: Services parameter buffer

Server

Generic response

Query service

Client

Int32
: Operation code (`op_service_info`)

Int32
: Services manager attachment handle

Int32
: Incarnation of object (0)

Buffer
: Services parameter buffer

Buffer
: Requested information items

Int32
: Requested information items buffer length

Server

Generic response - Where Data contains the requested information.

Events

Connection request

Client

Int32
: Operation code (`op_connect_request`)

Int32
: Connection type (`P_REQ_async`)

Int32
: Partner identification (0)

Server

Int32
: Attachment handle

Int16
: Port number

> **Important**
>
> This is part of the sockaddr_in structure.
>
> It is not in XDR format

Int16
: Socket family

> **Important**
>
> This is part of the sockaddr_in structure.
>
> It is not in XDR format

Byte[4]
: IP Address

> **Important**
>
> This is part of the sockaddr_in structure.
>
> It is not in XDR format

Byte[8]
Zeroes

> **Important**
>
> This is part of the sockaddr_in structure.
>
> It is not in XDR format

Byte[4]
Garbage

Queue events

Client

Int32
Operation code (`op_que_events`)

Int32
Database handle

Buffer
Events parameter buffer

Int32
Ast function address

Int32
Ast parameters function address

Int32
Local event id

Server

Generic response - Where Object Handle holds the remote event id.

Cancel events

Client

Int32
Operation code (`op_cancel_events`)

Int32
Database handle

Int32
Local event id

Server

Generic response.

External Data Representation (XDR)

The Firebird wire protocol uses XDR for exchange messages between client and server.

Data types

Int32
Integer 32-bits

Int64
Integer 64-bits

Buffer

Composed by :

Type	Description
Int32	Length.
Byte[]	Buffer data.

Byte[]
An array of bytes

String
A text string (*Read/Written as a buffer*)