

nanofiles-c protocol specification

Revision B

Notation

- asdf: static data
- a|b: disjunction of options
- < >: mandatory field
- []: optional field
- [...]: last object N times
- All sizes are in bytes

Directory protocol

UTF-8 encoded text/plain in key:value format over simple UDP. Whitespaces are stripped.

Client requests

Ping request Test the connection and compatibility

- Operation **ping**
- Fields
 - **protocol id**
- Answer 'ping reply'

operation: ping
protocol: <protocol id>

Filelist request Get file information known by directory

- Operation **filelist**
- Fields: None
- Answer 'filelist reply'

operation: filelist

Publish request Inform directory of list available files for download from client

- Operation **publish**
- Fields: (see below), the list can be empty
- Answer 'publish response'

operation: publish
[port: <port>]
<hash1>: <filename1>; <size1>
<hash2>: <filename2>; <size2>

[...]

Directory responses

Ping reply Acknowledges back connection and informs client of compatibility

- Operation `pingok`
- Fields: None
- Answer to ‘ping request’

operation: `pingok`

Ping bad reply Warns the client its using the wrong protocol

- Operation `pingbad`
- Fields: None
- Answer to ‘ping request’

operation: `pingbad`

Filelist reply Send back list of known (filename, hash, size and peers) for every file known

- Operation `filelistres`
- Fields: (see below), the list can be empty
- Answer to ‘filelist request’

operation: `filelistres`

<hash1>: <filename1>; <size1>; <server1a>, <server1b> [...]

<hash2>: <filename2>; <size2>; <server2a>, <server2b> [...]

[...]

Publish reply Acknowledge publish request

- Operation `publishack`
- Fields: None
- Answer to: ‘publish request’

operation: `publishack`

Peer protocol

Binary little-endian over TCP buffers

All messages begin with an opcode byte

Client requests

Opcode in 0x0X

File request Requests availability of file to be downloaded

- Opcode: 0x01
- Fields:
 - filename[1]: Filename length
 - filename[filename]: Filename
- Answer: 'accepted' or 'file not found error'

```
0          1          byte
+-----+-----+
| opcode |filename|
+-----+-----+
| filename      |
| ...          |
```

Chunk request Asks server to send a chunk

- Opcode: 0x02
- Fields:
 - offset[8]: Starting byte of chunk
 - size[4]: Size of chunk
- Answer: 'chunk'

```
0          1          2          3          4          5          6          7          8          byte
+-----+-----+-----+-----+-----+-----+-----+-----+
| opcode | offset                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
| size                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
```

Stop download Terminates current file request

- Opcode: 0x03
- Fields: None
- Answer: None

```
0
+-----+
| opcode |
+-----+
```

Server requests

Opcode in 0x1X

Accepted File is available to download via chunk requests

- Opcode 0x11
- Fields: None

- Answer to: 'file request'

```

0
+-----+
| opcode |
+-----+

```

Bad chunk request error File is unavailable or not found

- Opcode: 0x12
- Fields: None
- Answer to: 'file request'

```

0
+-----+
| opcode |
+-----+

```

Chunk Data chunk of file

- Opcode: 0x13
- Fields:
 - size[4]: Size of chunk
 - data[size]:
- Answer to: 'chunk request'

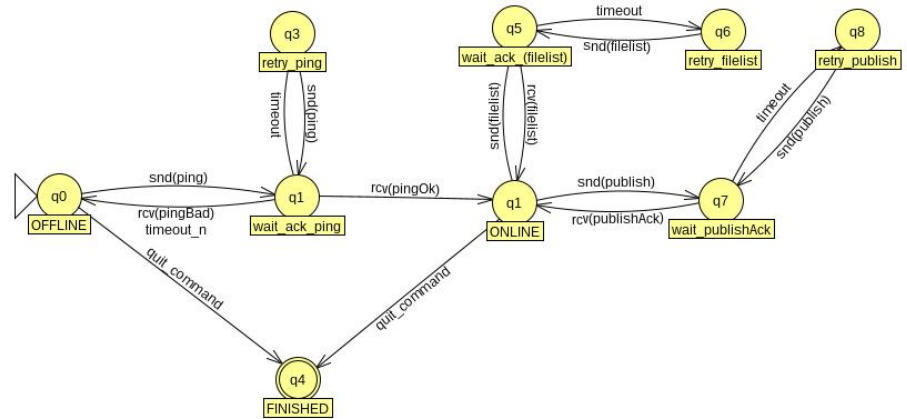
```

0          1          2          3          4          byte
+-----+-----+-----+-----+-----+
| opcode | size                                     |
+-----+-----+-----+-----+-----+
| data ...
|

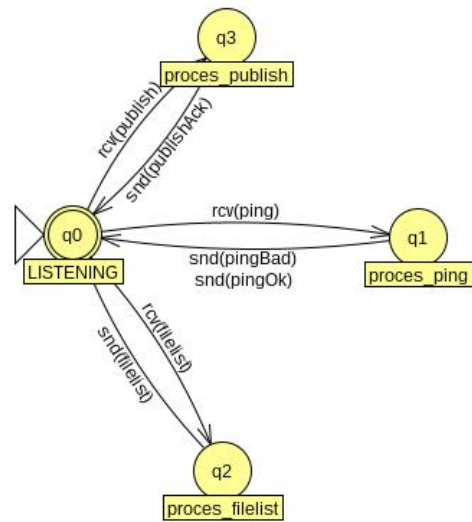
```

Application automaton

Directory protocol

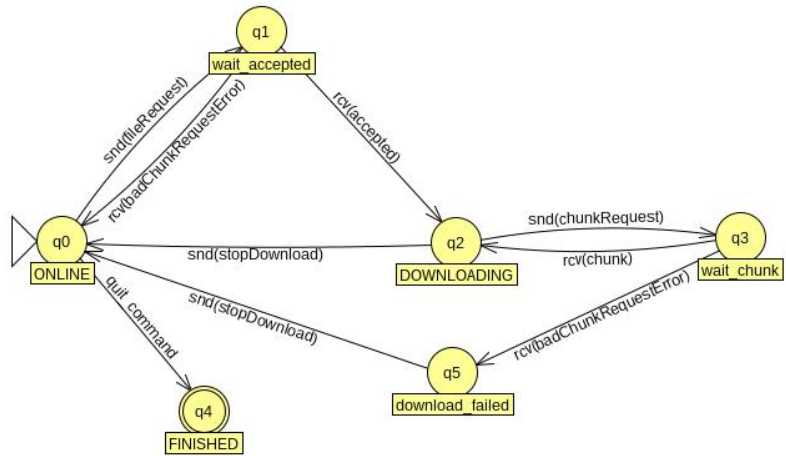


Client automaton

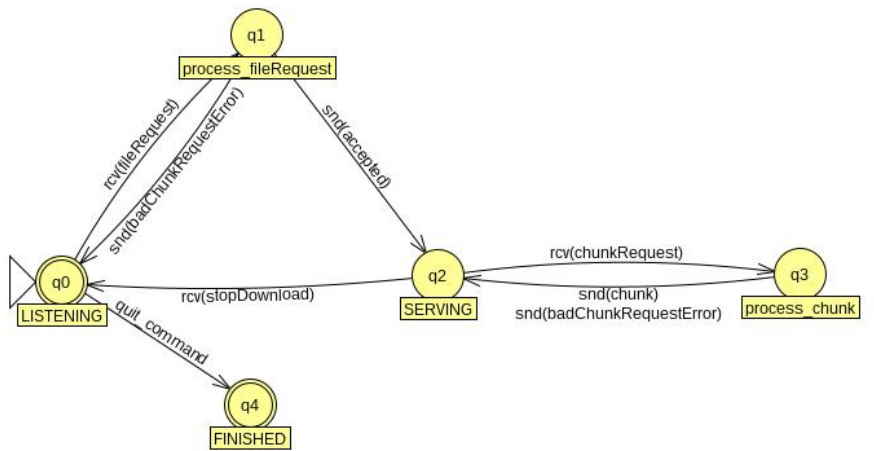


Server automaton

Peer protocol



Client automaton



Server automaton