## RPC Stream Transfer Specification

Author: Terry Skotz 2/10/12

## Design Requirements

\*Easily determine start and end of individual transfers

\*No restrictions on datum names

\*No restrictions on datum data

\*Easily add a new datum type

\*Backward, Forward compatibility

## Overview

A Transfer is sent as a chunk of data. Each transfer starts with a 4 byte ID followed by a 4 byte size value and finally the data. The size value describes the size of the data (in bytes) plus the 8 bytes for the ID and Size value. The receiver can then unpack the data based upon the ID.

|  |  |  |
| --- | --- | --- |
| **Size (bytes)** | **Chunk** | **Description** |
| 4 | ID | 4 byte ID |
| 4 | Size | S (Uint32) |
| S - 8 | Data | Bucket O’Data |

The data may also contain sub chunks of data referred to as Datums. A Datum is a chunk of data (based on above Transfer format – ID, Size, Data) which contains a predefined data layout. Datums are used in Request and Reply Transfers.

New Transfer and Datum types can easily be created using this scheme. It just requires that the receiver have knowledge of the new type’s ID. Unknown ID’s may be skipped or flagged as error.

##### **Notification Types**

|  |  |
| --- | --- |
| “RQST” | Request Notification Stream |
| “RPLY” | Reply Notification Stream |
| “EVNT” | Event Notification Stream |
| “STAT” | Progress Bar/Thermometer Status Notification Stream |
| “DLOG” | Dialog Notification Stream |
| “WNDW” | Window Notification Stream |
| “XCPT” | Exception Notification Stream |

##### **Request/Reply Transfer Datum Types**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| “Stri” | String | “Shrt” | Short | “Func” | Function Name |
| “Bool” | Boolean | “USht” | Unsigned Short |  |  |
| “In32” | 32 bit Integer | “Flot” | Float |  |  |
| “Ui32” | Unsigned 32 bit Integer | “Dble” | Double |  |  |
| “In64” | 64 bit Integer | “Buff” | Data Buffer |  |  |
| “Ui64” | Unsigned 64 bit Integer | “Cmpd” | Compound Datum |  |  |

Notification Types

### Request

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Chunk ID | Text | “RQST” |
| 4 | Chunk Size | Uint32 | S = 12 + size of all added datums |
| 4 | Request ID | Uint32 | *i* : unique id of the request |
| S - 12 | The Request Data | Request Datums | *request data* |

### Reply

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Chunk ID | Text | “RPLY” |
| 4 | Chunk Size | Uint32 | S = 8 + size of all added datums |
| 4 | Reply ID | Uint32 | *i* : id of the corresponding request |
| S - 12 | The Reply Data | Reply Datums | *reply data* |

### Event

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Chunk ID | Text | “EVNT” |
| 4 | Chunk Size | Uint32 | S = 13 + T + data size |
| 4 | Optional Int32 Val | Int32 | *int32 value* |
| 1 | Event Type Size | Unsigned Char | T |
| T | Event Type | text | *event text* |
| S – (13 + T) | Data | text | *event message text* |

### Status

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Chunk ID | Text | “STAT” |
| 4 | Chunk Size | Uint32 | S = 9 + message size |
| 1 | Percentage | Unsigned Char | *percentage value* |
| S – 9 | Message | Text | *message text* |

### Dialog

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Chunk ID | Text | “DLOG” |
| 4 | Chunk Size | Uint32 | S = 9 + B + message size |
| 1 | Title Size | Unsigned Char | T |
| T | Title | Text | *title text* |
| 1 | Buttons Size | Unsigned Char | B |
| B | Buttons | Text | *button text* |
| S–(10 + T +B) | Message | Text | *message text* |

### Window

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Chunk ID | Text | “WNDW” |
| 4 | Chunk Size | Uint32 | S = 10 + T + P + Status size |
| 1 | Title Size | Unsigned Char | T |
| T | Title | Text | *title text* |
| 1 | Unique ID Size | Unsigned Char | P |
| P | Unique ID | Text | *id value* |
| S – (10+Z+P) | Status | Text | *status text* |

### Exception

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Chunk ID | Text | “XCPT” |
| 4 | Chunk Size | Uint32 | S = 8 + data size |
| S – 8 | Data | text | *exception message data* |

## Request/Reply Notification Datums

### String

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Datum Type | Text | “Stri” |
| 4 | Datum Size | Uint32 | S = 9 + N + data size |
| 1 | Name Size | Unsigned Char | N |
| N | Name | Text | *name data* |
| S – (9 + N) | Data | Text | *string data* |

### Boolean

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Datum Type | Text | “Bool” |
| 4 | Datum Size | Uint32 | S = 9 + name size |
| 1 | Data | Unsigned Char | *bool value* = 1 or 0 |
| S - 9 | Name | Text | *name data* |

### Short

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Datum Type | Text | “Shrt” |
| 4 | Datum Size | Uint32 | S = 10 + name size |
| 2 | Data | Short | *short value* |
| S - 10 | Name | Text | *name data* |

### Unsigned Short

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Datum Type | Text | “USht” |
| 4 | Datum Size | Uint32 | S = 10 + name size |
| 2 | Data | Unsigned Short | *unsigned short value* |
| S - 10 | Name | Text | *name data* |

### 32 bit Integer

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Datum Type | Text | “In32” |
| 4 | Datum Size | Uint32 | S = 12 + name size |
| 4 | Data | Int32 | *Int32 value* |
| S - 12 | Name | Text | *name data* |

### Unsigned 32 bit Integer

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Datum Type | Text | “Ui32” |
| 4 | Datum Size | Uint32 | S = 12 + name size |
| 4 | Data | Uint32 | *Uint32 value* |
| S - 12 | Name | Text | *name data* |

### 64 bit Integer

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Datum Type | Text | “In64” |
| 4 | Datum Size | Uint32 | S = 16 + name size |
| 8 | Data | Int64 | *int64 value* |
| S - 16 | Name | Text | *name data* |

### Unsigned 64 bit Integer

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Datum Type | Text | “Ui64” |
| 4 | Datum Size | Uint32 | S = 16 + name size |
| 8 | Data | Uint64 | *unsigned int64 value* |
| S - 16 | Name | Text | *name data* |

### Float

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Datum Type | Text | “Flot” |
| 4 | Datum Size | Uint32 | S = 12 + name size |
| 4 | Data | Float | *float value* |
| S - 12 | Name | Text | *name data* |

### Double

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Datum Type | Text | “Dble” |
| 4 | Datum Size | Uint32 | S = 16 + name size |
| 8 | Data | Double | *double value* |
| S - 16 | Name | Text | *name data* |

### Buffer Data

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Datum Type | Text | “Buff” |
| 4 | Datum Size | Uint32 | S = 9 + N + data size |
| 1 | Name Size | Unsigned Char | N |
| N | Name | Text | *name data* |
| S – (9 + N) | Data | Char | *Char data* |

### Compound

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Datum Type | Text | “Cmpd” |
| 4 | Datum Size | Uint32 | S = 10 + N + T + data size |
| 1 | Name Size | Unsigned Char | N |
| N | Name | Text | *name data* |
| 1 | Type Size | Unsigned Char | T |
| T | Type | Text | *compound type data* |
| S –(10+N +T) | Data | bytes | *compound data* |

### Function

|  |  |  |  |
| --- | --- | --- | --- |
| **Size (bytes)** | Description | Data Type | Data |
| 4 | Datum Type | Text | “Func” |
| 4 | Datum Size | Uint32 | S = 8 + data size |
| S – 8 | Function Name | text | *function name* |

## Helpful Byte Info

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Type | C++ Bytes | C++ Value Range | **Java Bytes** | **Java Value Range** |
| bool | 1 | True or False | 1 bit | True or false |
| byte | N/S |  | 1 | -128 to 127 |
| unsigned char | 1 | 0 - 255 | N/S |  |
| char | 1 | -128 to 127 | 2 | 0 – 65,535 |
| unsigned short | 2 | 0 – 65,535 | N/S | Use Java Int to store this value |
| short | 2 | -32,768 to 32,767 | 2 | -32,768 to 32,767 |
| unsigned long | 4 | 0 - 4,294,967,295 | N/S | Use Java long to store this value |
| long | 4 | -2,147,483,648 to -2,147,483,647 | 8 | 9,223,372,036,854,775,808 to +9,223,372,036,854,775,807 |
| unsigned int32 | 4 | 0 - 4,294,967,295 | N/S | Use Java long to store this value |
| int33 | 4 | -2,147,483,648 to -2,147,483,647 | 4 | -2,147,483,648 to -2,147,483,647 |
| float | 4 | 3.4e +/- 38 (7 digits) | 4 | 1.40129846432481707e-45 to 3.40282346638528860e+38 |
| double | 8 | 1.7e +/- 308 (15 digits) | 8 | 4.94065645841246544e-324d to 1.79769313486231570e+308d |