Galleon
Transport Interface

version 1

Purpose

The Transport Interface may be used to transmit arbitrary data between Galleon system components that may or may not be located on the same machine.

The interface provides an abstraction for the implementation details related to the location of system components and the networks on which their machines operate.

Design

The Transport Interface is designed to allow Galleon system components to exchange arbitrary data files between peers. Interface clients will send data by invoking a handler provided by the interface and receive data through a handler registered with the interface. Clients are identified using labels.

Transport Process

The Transport Interface sends data between interface clients using the following process.

1. Client A on Machine A generates a file to transmit to Client B on Machine B.
2. Client A calls the Send Handler of the Transport Interface, providing the file path and the source and destination labels as parameters.
3. The Transport Interface facilitates moving the file from Machine A to Machine B. The method of transmission is undefined.
4. The Transport Interface calls the Receive Handler registered for Client B, providing the file path and the source and destination labels as parameters.
5. Client B ingests the transmitted file.

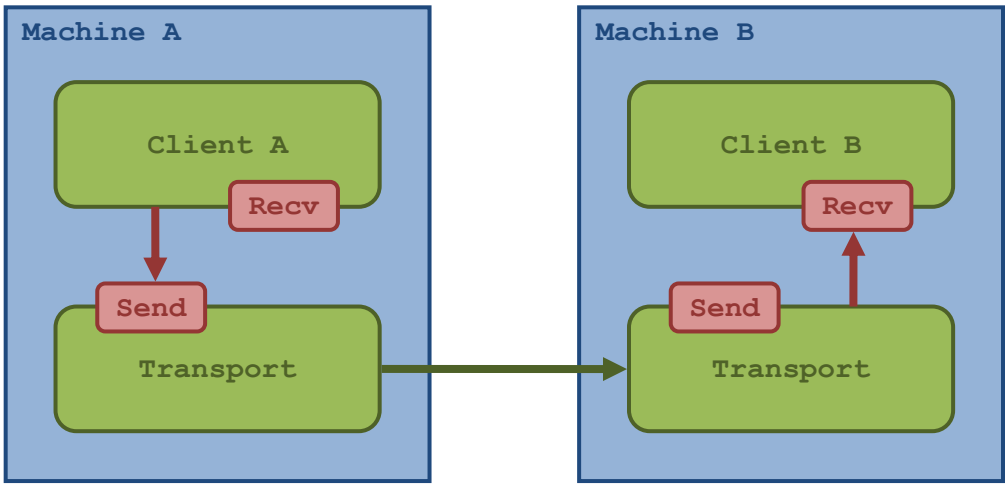

FIGURE 1: TRANSMISSION OF DATA VIA THE TRANSPORT INTERFACE

Service Level

The Transport Interface provides a best effort delivery service. The interface does not provide any guarantees of delivery or service level. All data transmitted through the interface is treated with the same priority.

If a client requires guaranteed delivery, they can build a reliable protocol on top of the Transport Interface. For example, build a super layer implementing an Automatic Repeat reQuest (ARQ) scheme.

Client Labels

System components that use the Transport interface refer to one another using a Client Label. Client Labels should be unique in any given system. Client Labels are provided as parameters to the Transport handlers.

A Client Label is a string consisting of the following characters:

- Alphabetic Characters (upper and lower case)
- Numeric Characters
- Hyphens, Periods, Underscores

Transport Chain

During the transmission of data through the Transport Interface, Handlers and their callers hand off responsibility for the transport operation. The responsible component receives ownership of the data file and is obligated to complete its transmission.

When a handler returns a success, it accepts responsibility for the operation and takes ownership of the file it is provided. The handler may modify or delete the file as necessary. The handler will be responsible for cleaning up the file when its part in the operation is complete.

When a handler returns an error, it does not accept responsibility for the operation or ownership of the data file. The caller may handle this error as appropriate, including retrying, updating a log, and/or sending an error to the user.

Layers

The Transport Interface exposes *Infrastructure* Layer functionality. The Send Handler is provided as part of the Transport Interface.

The Receive Handler is provided as part of a Client, and is located in the *Tool* or *Management* Layer, depending on the location of the Client. This specification defines the behavior that the Transport Interface expects from the Receive Handler.

Implementation

Configuration

A system implementing the Transport Interface should define the following keys within the system configuration file:

`interface.transport.version`	The version of the Transport Interface that has been implemented by the system. The current version is ‘1’. Only one version key may be defined at a time.
`interface.transport.send`	The Send Handler for the Transport Interface. Only one send key may be defined at a time.

Handlers

The Transport Interface defines two types of handlers, one for sending data and one for receiving data.

Send Handler

The Send Handler is provided by the interface implementation and is used by interface clients to request transmission of a file.

Behavior

An interface client calls the Transport Interface's Send Handler to request transmission of a file to another interface client.

The handler will transmit one file at a time. If a client would like to send multiple files, they may make multiple calls to the Send Handler or generate and send an archive file containing the files. The Send Handler does not guarantee the preservation of the filename of the transmitted file.

The Send Handler will facilitate the transmission of the file to the destination client. Upon completion, the handler should delete the transmitted file from the file system (unless it has passed responsibility for the file to another handler).

Usage

```
send_handler <src_label> <dst_label> <data_file_path>
```

`src_label` transport label for the source component (the sender)

`dst_label` transport label for the destination component (the receiver)

`file_path` local path to a file containing the data to be transmitted

Receive Handler

The Receive Handler is provided by the interface client and registered with the Transport interface. The interface will use this handler to handoff data from the interface to the client.

Behavior

An interface client implements and registers one or more Receive Handlers with the Transport. The Transport calls the appropriate handler to deliver a transmitted file to the client.

The handler will parse and process the data contained in the transmitted file. Upon completion, the handler should delete the transmitted file from the file system (unless it has passed responsibility for the file to another handler).

Usage

```
receive_handler <src_label> <dst_label> <data_file_path>
```

`src_label` transport label for the source component (the sender)

`dst_label` transport label for the destination component (the receiver)

`file_path` local path to a file containing the data to be transmitted

Handler Return Codes

The Transport Interface defines return codes that will be returned from the various interface handlers.

The code values and their meanings are provided below:

- *0* Success
- *1* Unspecified Error
- *2* Invalid Arguments
- *3* Unknown Client Label

Appendix A: Version History

Date	Description
8 Nov 2013	**Version 1, Revision 1** Transport Interface initialized and submitted for design review.
28 Jul 2014	**Version 1, Revision 2** Transport Interface revised after first Galleon pilot.
1 Dec 2014	**Version 1, Revision 3** Transport Interface revised after second Galleon pilot.
1 Jun 2015	**Version 1, Final** Transport Interface finalized and delivered.

Appendix B: Future Work

The following items have been identified as possible areas for future work:

Expose Registration Function

Provide a mechanism to programmatically register a Receive Handler with the Transport Interface.