GLUE Singleton-Free Concept

Problem Description

Two components using GLUE have to register their own Middelware with their respective PacketHandler to communicate. Both Middleware implementations have to be registered with the same schema to guarantee correct allocation of packets.

This procedure leads to problems within one Java Virtual Machine due to the use of singletons to manage registered Middleware implementations. When registering a second Middleware for the same schema this overwrites the already existing one.

The current implementation of VMTransportFactory solves this problem by registering only one Middleware for the server and setting the client's handler explicitly for a PacketThread. This procedure differs from the normal procedure for initializing and using GLUE. Furthermore it leads to problems with current Middleware implementations such as MOCCA.

Solution

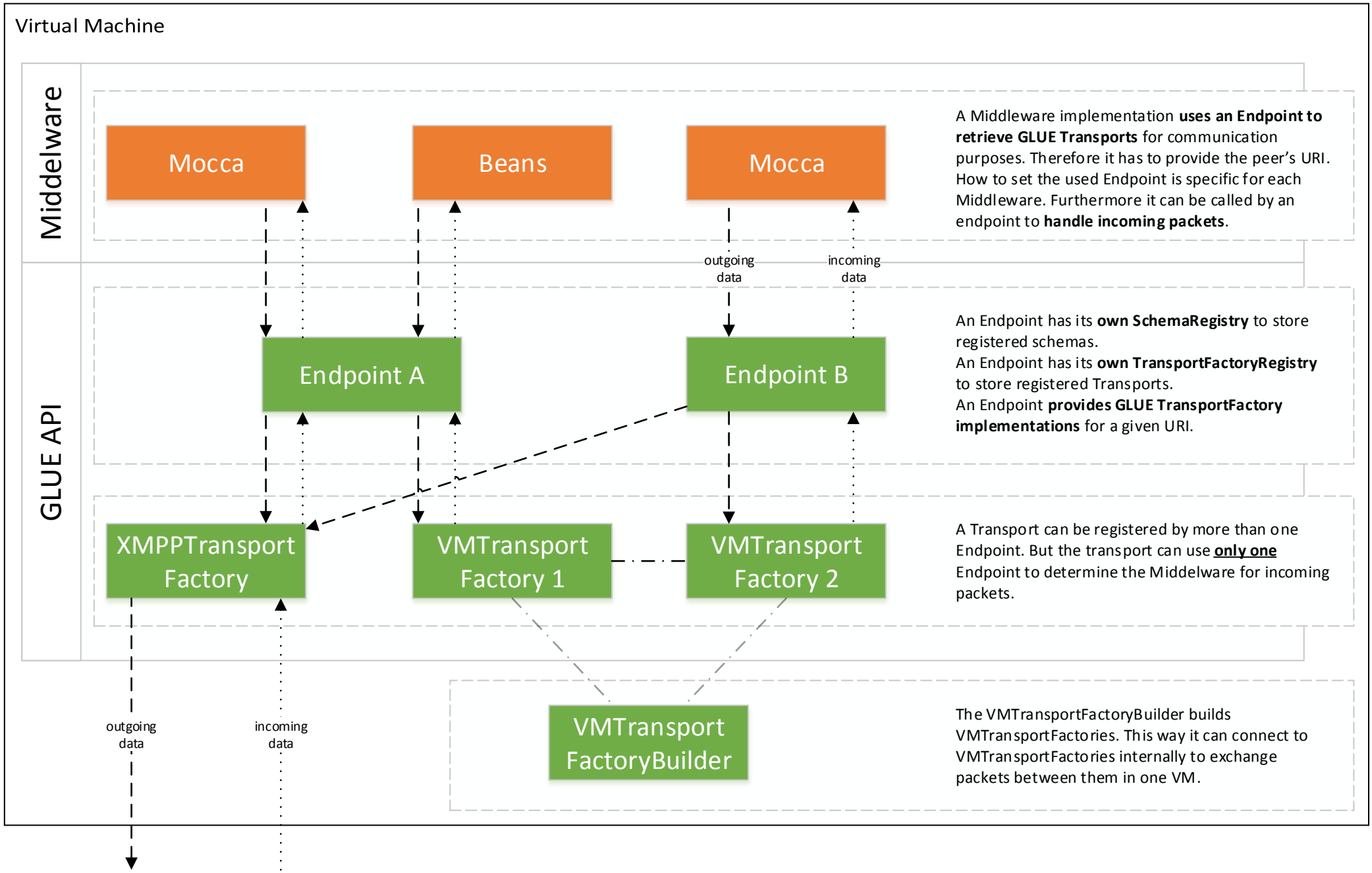
To solve the problem a singleton-free implementation is suggested. The solution is mainly based on the concept of *dependency injection*, which means that all needed resources of a component are explicitly forwarded.

The solution has to fulfill following requirements:

a) The application must be able to generate a Transport to communicate with a peer.
b) A TransportFactory must have access to relevant Middleware implementations all the time to be able allocate incoming packages.

The diagrams on the next pages describe and visualize the concept.

One Virtual Machine Scenario



Two Virtual Machines Scenario