1 Client-Server-Modell

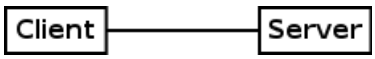

Figure 1: Client-Server-Modell

Client

Der Client ist der User, der Datenstze, Algorithmen auf den Server hochladen, diese dann dort berechnen lassen und Pakete downloaden kann.

Server

Der Server bietet dem User den Dienst an, Modelle anhand von schon vorhandenen oder vom User hoch geladenen Datenstzen und Algorithmen zu erstellen. Des Weiteren dient der Server als Datenbank von schon mit verschiedenen Datenstzen erstellten Modellen.

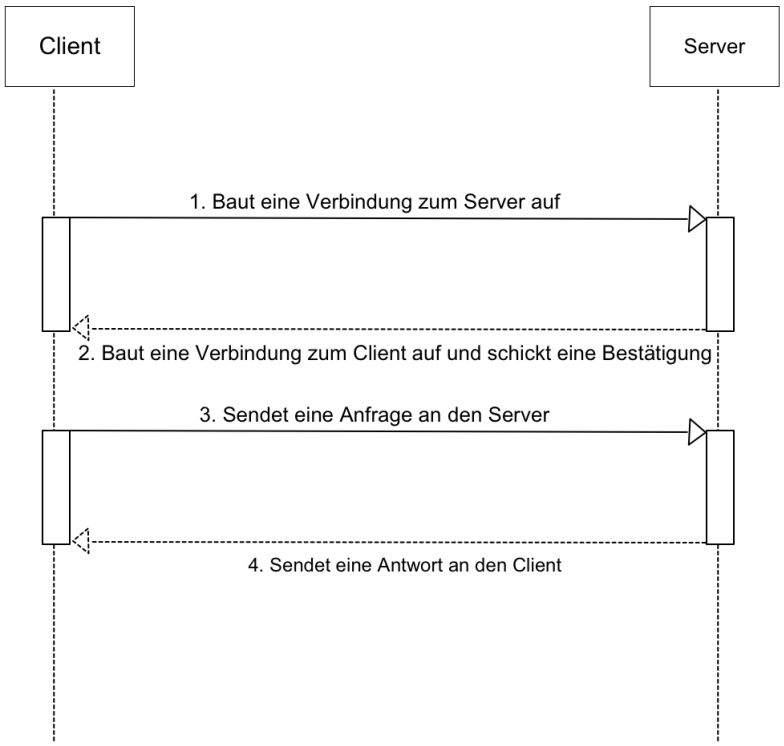

Figure 2: Client-Server-Sequenzdiagramm

Der Client versucht eine Verbindung zum Server aufzubauen, der Server versucht ebenfalls bei Anfrage eine Verbindung zum Client Aufzubauen und schickt dem Client eine Besttigung, das die Verbindung steht. Danach kann der Client Datenstze/Algorithmen hochladen und dem Server eine Anfrage zum Modelle downloaden schicken.

Was spricht fr das Client-Server-Modell?

Das Client-Server-Modell wird verwendet, wenn eine Datenbank oder ein Service von verschiedenen Orten her abgerufen werden soll. Dies ist fr beides der Fall. Der User kann global auf den Server zugreifen und Daten hoch und runter laden.
Des Weiteren sollen mehrere gleiche Server online sein, damit viele Useranfragen auf mehrere Server verteilt werden knnen und somit schneller bearbeitet werden knnen. Der User sieht aber nur den einen Server. Wenn ein Server offline (ausfllt/gewartet)ist und es sind mehrere online, so bekommt der User davon nichts mit und wird auf einen anderen Server geleitet.
Nachteile sind:
-Die Leistung des Systems ist unberechenbar, wenn die verschiedenen Service im Server-Netzwerk verteilt ist. Dieser Fall existiert in unserem System nicht.
-Es gibt Management Probleme, wenn Server in relativ Unabhngigen Besitz ist. Da die Server unabhngig arbeiten und nur zusammenarbeiten, wenn schon vorhandene Modelle/Datenstze auf einem anderen Server angefragt werden, ist dieser Nachteil zu vernachlssigen.

Was spricht gegen Pipe-and-Filter-Modell?

Dieses Modell ist zu mchtig, da die Strme zwischen Server und Client nicht gekapselt oder verschlsselt werden sollen. Server und Client mssten immer beim Aufbauen einer Verbindung eine Verschlsselung vereinbaren und die Datenstrme verschlsseln und entschlsseln. Dies vermindert oder verhindert sogar sptere Vernderungen vorzunehmen nd verbraucht unntige Rechenzeit, da keine hoch sensiblen Daten ausgetauscht werden.

Was spricht gegen Broker-Modell?

Das Broker-Modell vermindert die Performance, da alle Komponente des Systems nur indirekt angesprochen werden. Des Weiteren hngt die Kommunikation der einzelnen Systeme von vielen Komponenten ab und ist deshalb Fehleranfllig. Sinnvoll wre der Broker nur, wenn viele verschiedene Clients auf viele verschiedenen Server auf viele verschiedene Service zugreifen wollen. Die Anzahl der Services unseres Servers ist sehr berschaubar und deshalb ist der Broker zu ineffizient fr unseren Server.