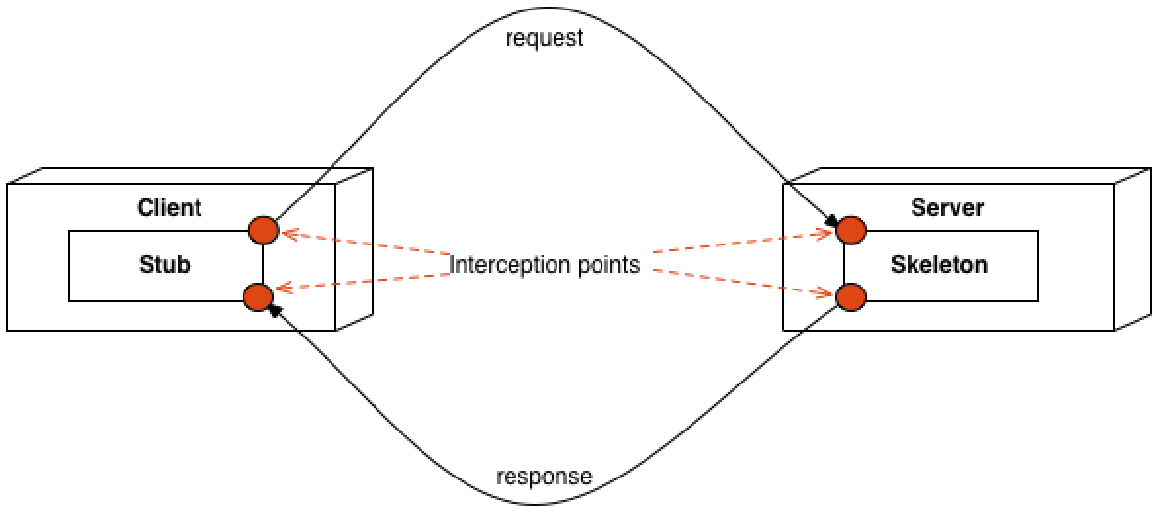
Interceptors

A page about interceptors, will be here soon.

What are interceptors

Interceptors are cool. Technically an interceptor is a piece of code that implements either ClientSideRequestInterceptor, ServerSideRequestInterceptors or both. However this *piece of code* has a great effect, since it can intercept any call from client to server at one of the following four interception points.



"Intercept" means in this context that an interceptor can do the following:

- Inspect the call, method name, target server, parameters and do nothing (continue call execution).
- Change some parameters or the return value.
- Abort the call by forcing the stub to throw an exception.
- Cancel the call and return another value instead (both prior and post call execution).
- Transfer additional data to/from server - piggy bagging.

What are interceptors for.

There are a lot of scenarios where interceptors are useful. Some of them are covered by the examples in the `org.distributeme.test.interception.interceptor` package.

Security/ Access Control

Interceptors can be used to check whether a specific caller is allowed to access the target service, transparently to the service and its developer. They can use various techniques to achieve this goal, for example check for specific ip or demand the user to execute a specific call (login) first, and re/transmit access credentials via piggy bagging.

Logging

Interceptors can log all requests between clients and servers, or between a particular subset of clients and servers, or to/from particular client/server, or requests which meets some additional criteria etc.

Debug

Interceptors can be used to submit additional information from server to the client, for example execution time or db query info.

Tracking

Track information. For example track calls which are performed in context of a specific userId over all services/networks. Turnable on/off on demand.

Availability Testing

This topic is worth it's own article.