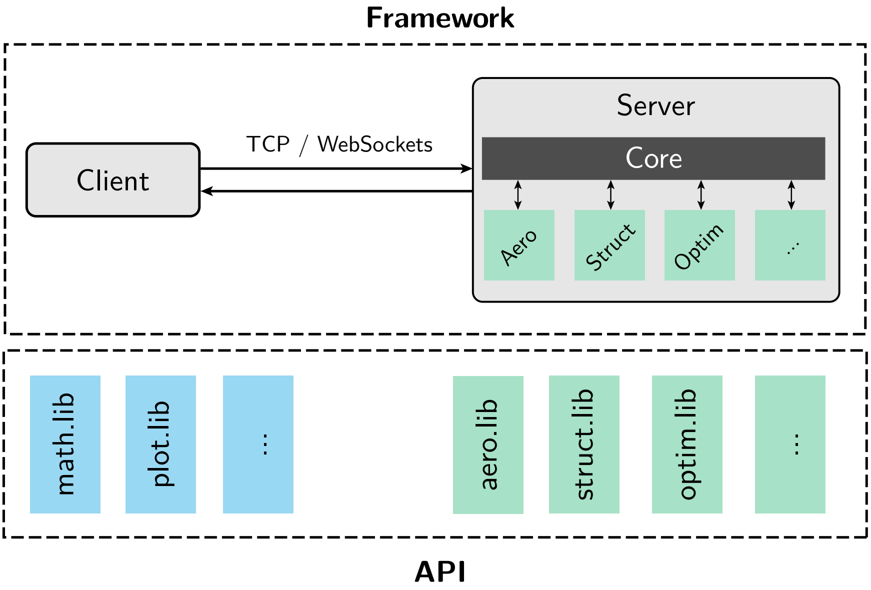
The Adaptiv Framework

Nuno Alves de Sousa

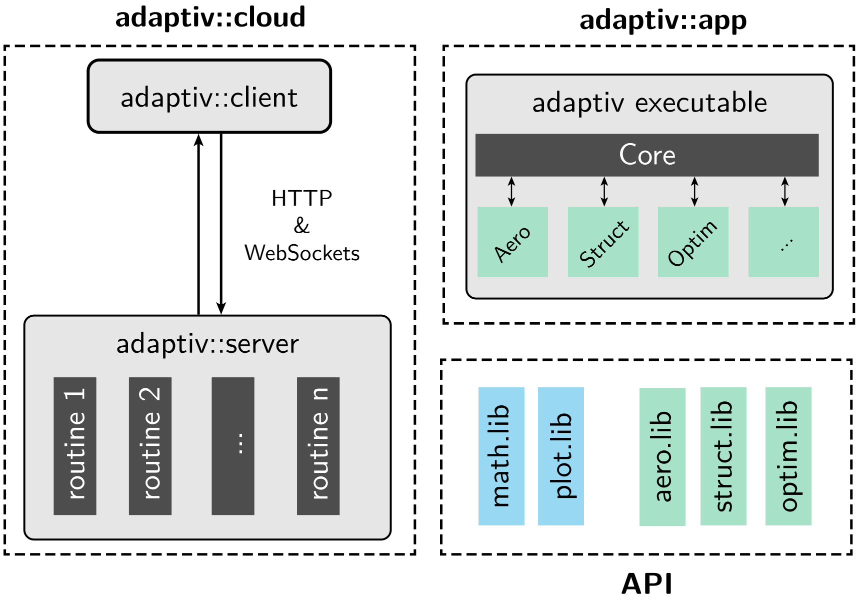
Instituto Superior Técnico
Área Científica de Mecânica Aplicada e Aeroespacial

June 26, 2019

Proposed architecture



A revised architecture



The **adaptiv::cloud** framework

adaptiv as a service:

- Remote procedure call (RPC) architecture
- Rich network communications (full-duplex)
- Collaborative platform
- Possible web app implementation

Benefits:

- Software-hardware integration
- Location agnostic
- Centralized computational resources
- Leverage user simulations to train a surrogate optimizer

The **adaptiv::cloud** framework

Main features:

- Built with Boost.Beast
- Asynchronous operations
- SSL
- Custom communication protocol

Why **adaptiv::cloud** & **adaptiv::app**?

- Inversion of control
- Avoid odd choices on behalf of the user
- Modular & reusable architecture
- **adaptiv::cloud** is an internal, continuous, development tool

Communication protocol

adaptiv::cloud network exchanges are JSON based:

- Every response/request has a target (*i.e.* remote subroutine)
- An optional message can be sent to/from that target
- The message is generated from a JSON-serializable C++ type

```
{
    "target": "rans",
    {
        "message":
        {
             "param1": 32,
             "param2": "Hello, world"
        }
    }
}
```

Listing 1: Example of an **adaptiv::cloud** network exchange

Creating a custom network message

- Almost as simple as declaring a new C++ type
- Introspection is used to check if `serialize()` is missing

```cpp
struct MyRequestMessage
{
            int param1;
    std::string param2;

    // Make 'MyRequest' JSON-serializable
    template<class Archive>
    void serialize(Archive& archive)
    {
        archieve(
            CEREAL_NVP(param1), // Register 'param1' for serialization
            CEREAL_NVP(param2)  // ... and 'param2'
        );
    }
}
```

Listing 2: Custom network message

Sending a request/response

A request/response is constructed from a target and a message:

```cpp
namespace protocol = adaptiv::cloud::protocol;

// Create a request message
MyRequestMessage message{32, "Hello, world!"}

// Create a Request
protocol::Request request("rans", message);

// Output the generated adaptiv network message in JSON format
std::cout << request.json();
```

Listing 3: Creating a request

The request/response are templated on the message type:

```cpp
template<class NetworkMessage>
class Request: public NetworkExchange<NetworkMessage>
{ /* ... */ }
```

Demo

https://github.com/seriouslyhypersonic/adaptiv_co

Progress

Library `adaptiv::`	Description
`cloud`	cloud framework
`concepts`	concepts library
`math`	random numbers, Eigen
`net`	asynchronous I/O and networking
`serialization`	JSON, XML and binary serialization
`system`	error handling
`traits`	**adaptiv**-specific type traits
`utility`	input (parsers) & output (styles)

Table: Libraries under development

adaptiv