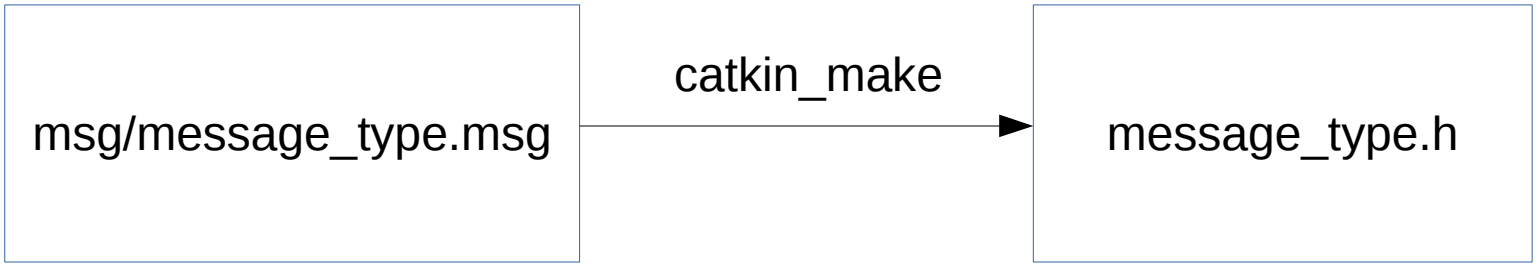
Собственные сообщения и сервисы

Сообщение

- Каждое сообщение — это объект, который существует в единственном экземпляре.

  void subscribe_callback (const package::message_type & message)

- Каждое сообщение генерируется из файла .msg в класс C++



В файловой системе

workspace/

src/

package/

msg/

message_type.msg

- int8 data
- char label

Компиляция сообщений

- find_package(catkin REQUIRED COMPONENTS
  std_msgs
  message_generation
)
- add_message_files(
    FILES
    message_type.msg
)
- generate_messages(
    DEPENDENCIES
    std_msgs
)
- catkin_package(
    CATKIN_DEPENDS message_runtime
)
- add_dependencies(node <package>_message_generation.cpp)

Очереди сообщений

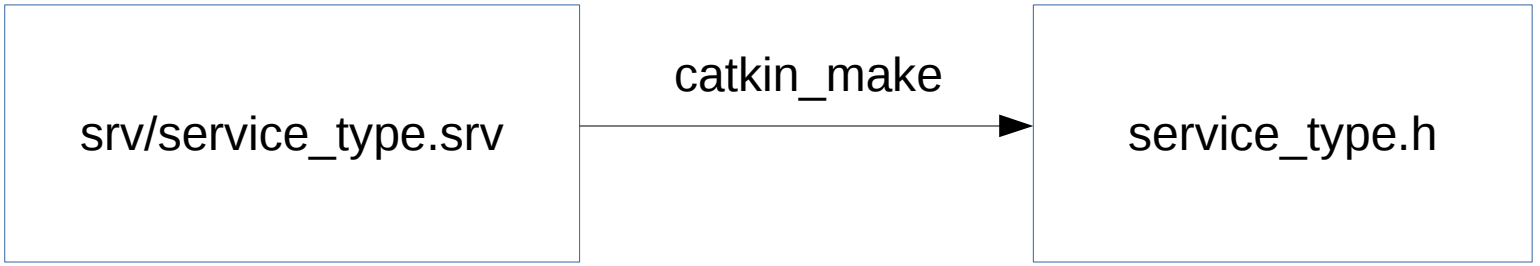
Писатель и читатель имеют собственные очереди сообщений

```
ros::Subscriber sub = n.subscribe("Name", 10, reader);
ros::Publisher pub =
                n.advertise<my_message::Message1>("Name", 20);
```

pub_queue topic sub_queue

Сервис

- Общение request-response
- Отношение типа клиент-сервер
- rossrv vs rosservice = rosmsg vs rostopic



В файловой системе

workspace/

src/

package/

srv/

service_type.srv

- ✔ int8 data
- ✔ char label

- ✔ int16 response

Компиляция сервиса

- find_package(catkin REQUIRED COMPONENTS
  std_msgs
  message_generation
)
- add_service_files(
    FILES
    service_type.msg
)
- generate_messages(
    DEPENDENCIES
    std_msgs
)
- catkin_package(
    CATKIN_DEPENDS message_runtime
)
- add_dependencies(node <package>_message_generation.cpp)

Работа с сервисом

- Можно вызывать из консоли и из ноды
- Сервис — это блокирующий вызов, поэтому тот, кто его вызывает будет ожидать ответа (возможно вечно)
- По договорённости, сервис не должен выполняться долго (более секунды), иначе лучше использовать «действие»

Как написать сервер для сервиса

```
bool concat(pkg::type::Request&  input, pkg::type::Response& output) {
        output.name3 = input.name1+input.name2;
        return true;
}

int main(int argc, char** argv) {
        ros::init(argc,argv,"server");
        ros::NodeHandle n;
        string s(ros::this_node::getName()+string("/conc"));
        ros::ServiceServer serv = n.advertiseService(s.c_str(), concat);
        ros::spin();
        return 0;
}
```

Как написать клиента для сервиса

- ros::ServiceClient clnt =

  n.serviceClient<pkg::type>(«service_name»);

- clnt.call(obj_of_service)

- bool ros::service::waitForService(«service_name», timeout)

Действия

- Такое же расположение в файловой системе:

  ./action/act_type.action

  goal

  ---

  result

  ---

  feedback