ROS 2 types on the wire

Emerson Knapp (Bonsai Robotics)

Type descriptions and hashing in Iron

...and beyond!

Terminology

ROS 2 Interface

- Message, Service, Action
- Topic Name + Type Name + QoS

Motivation: Evolving Interface Types

① Existing types change

② New types are defined

Evolving Types: End Goals

1. Dynamic comms: Subscribe and publish to types defined at runtime

2. Make the dynamic portion invisible - automatically translate changed type

Evolving Types: First Steps

1. Represent a type

2. Communicate a unique and compact ID of types to all participants

3. Communicate full type on request

…then there's a whole lot more to do, but this is an essential start!

Terminology

ROS 2 Interface
- Message, Service, Action
- Topic Name + Type Name + QoS

Type Description
- What are the fields of an interface type?

Type Hash
- Compact unique identification for an interface type, to detect changes

Type Source
- Original text used to define a type
- Could be .msg, .srv, .action, .idl

1 Representing types

Type Description Interfaces

`type_description_interfaces`

For representing types:

- `TypeDescription.msg`
- `IndividualTypeDescription.msg`
- `Field.msg`
- `FieldType.msg`

Additionally, for transmitting types

- `GetTypeDescription.srv`
- `TypeSource.msg`

Type Description Interfaces

TypeDescription.msg

```
IndividualTypeDescription   type_description
IndividualTypeDescription[] referenced_type_descriptions
```

IndividualTypeDescription.msg

```
string  type_name
Field[] fields
```

Type Description Interfaces

Field.msg

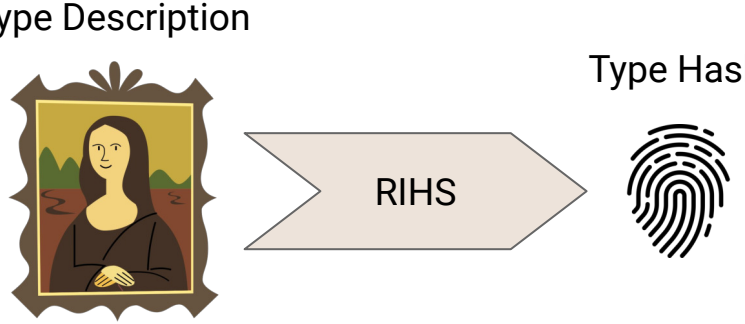
```
string    name
FieldType type
string    default_value
```

FieldType.msg

```
uint8  type_id
uint64 capacity
uint64 string_capacity
string nested_type_name
```

2 Hashing types

ROS Interface Hashing Standard (RIHS)



- Versioned, Iron has **RIHS01**
- Goals
  - Widely available tools
  - Computable from received msg
  - Versioned string representation
- Two implementations built:
  - C (in **rcl**)
  - Python (**rosidl_generator_type_description**)

RIHS01 Overview

1. Map TypeDescription -> JSON (skip default values!) -> text (specific formatting)
2. Exact bytes of text -> SHA256
3. Prefix “RIHS01_” + hex-string of SHA256 (e.g. 01a5e….)
4. Result: 71 byte fixed-length string output

```
emerson@33910e678692:~$ ros2 topic info -v /chatter
Type: std_msgs/msg/String

Publisher count: 1

Node name: _ros2cli_2644
Node namespace: /
Topic type: std_msgs/msg/String
Topic type hash: RIHS01_df668c740482bbd48fb39d76a70dfd4bd59db1288021743503259e948f6b1a18
```

Code Generation

Embedded in the C code generation for interface types, available via typesupport:

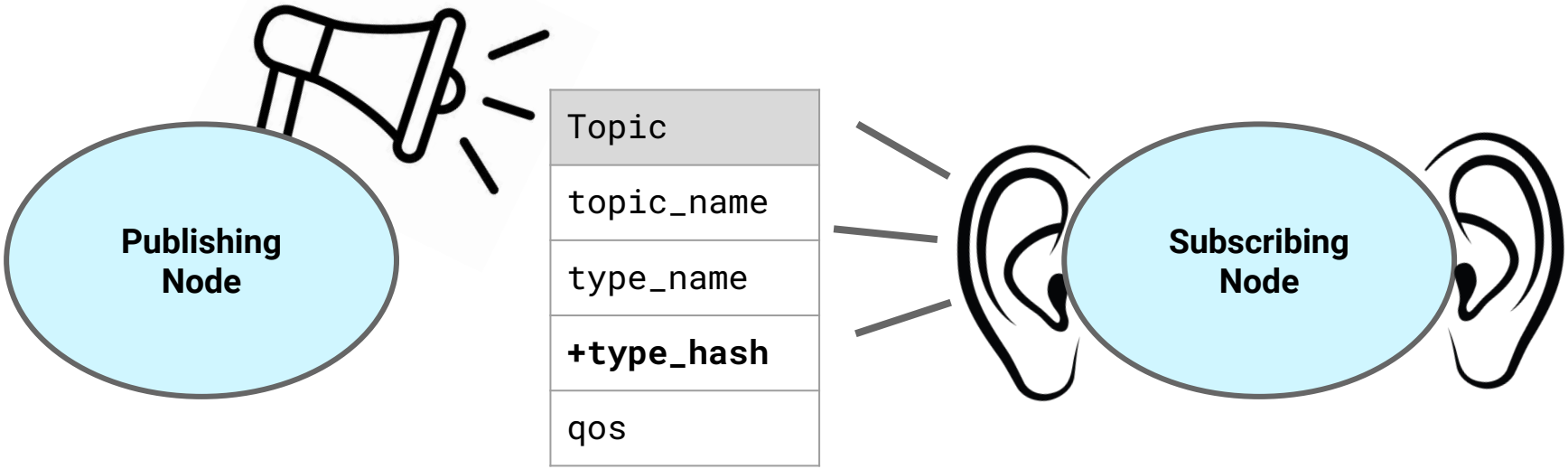
- Type Hash
- Type Description
- Type Source text

For minimized builds:
Description and source can be disabled by a preprocessor definition

```
--cmake-args -DROSIDL_GENERATOR_C_DISABLE_TYPE_DESCRIPTION_CODEGEN=ON
```

Type Hash Discovery

`rmw_topic_endpoint_info_t` contains new member **`type_hash`**



③ Distributing descriptions

Type Description Service

- On discovering a Type Hash, want to get a Type Description
- In Iron, **`~/get_type_description`** service on Nodes, enabled by default
- Controlled by Parameter **`start_type_description_service`**

GetTypeDescription.srv

```
string type_name
string type_hash
bool   include_type_sources true
---
bool            successful
string          failure_reason
TypeDescription type_description
TypeSource[]    type_sources
KeyValue[]      extra_information
```

...and beyond!

For more info

Design of these features in **REP-2016** (still under review)
https://bit.ly/ros-rep2016

What's missing?

- Type Hash discovery for Services and Actions
- Constants / "Enums" in msgs, do they change the hash? (not right now)
- Automatic hash-mismatch detection
- CLI tooling to fetch Type Descriptions

Future work:

Dynamic Types

Enable using Type Descriptions discovered dynamically to subscribe and publish any type.

Especially useful for developer tools such as RViz, Foxglove, Rosbag2, PlotJuggler, etc

Author (mostly): Brandon Ong @methylDragon

DDS XTypes for implementation in DDS-based RMWs
Map between TypeDescription ↔ DynamicType

First pass for Fast-DDS is in Iron RCL layer
(rclcpp C++ API open as experimental PR)

Future Work:

Evolved Type Translation

Author (mostly): William Woodall @wjwwood

Long term plan with large scope, delivery TBD

- Provide plugin infrastructure to automatically translate between versions of types as they evolve over time
- A translation function could be provided for any pair of hashes - allowing for arbitrary complexity of translation chains
- Type Description and Dynamic Types get closer to these goals

See REP-2011 draft for details

Acknowledgements

My work on this project done on contract for Foxglove - not possible without them!

Chris Lalancette
William Woodall
Brandon Ong

And thanks to the entire ROS 2 contributor community!
(especially if you reviewed the PRs!)

Thank you!

Emerson Knapp

Staff Robotics Engineer

Bonsai Robotics

emersonknapp@bonsairobotics.ai

linkedin.com/in/emersonknapp

github.com/emersonknapp