Network Programming

BITS Pilani
Pilani Campus

K Hari Babu
Department of Computer Science & Information Systems

Data Representations

Data Representations

innovate achieve lead

- Language independent, Platform independent data representations
  - Abstract Syntax Notation (ASN.1)
  - External data representation (XDR)
  - eXtensible Markup Language (XML)
  - JavaScript object notation (JSON)
  - Google Protocol Buffers
  - Apache Thrift

Google Protocol Buffer

- Google adopts a minimal and efficient remote invocation service
- Recall that: Remote invocation requires – among all the other services – the following two components
  1. Serialization of data
  2. Agreement on data representation (data-type size and format)
- Protocol Buffer (PB) is a common serialization format for Google

Goal of Protocol Buffer

- In Protocol buffers, Google has designed a language to specify messages

The goal of Protocol Buffer is to provide a language- and platform-neutral way to specify and serialize data such that:

- Serialization process is efficient, extensible and simple to use
- Serialized data can be stored or transmitted over the network

Protocol Buffer Language

- Message contains uniquely numbered fields
- Field is represented by

<field-type, data-type, field-name, encoding-value, [default value]>

- Available data-types
  - Primitive data-type
    - int, float, bool, string, raw-bytes
  - Enumerated data-type
  - Nested Message
    - Allows structuring data into an hierarchy

```
message Book {
    required string title = 1;
    repeated string author = 2;
    enum Status {
        IN_PRESS = 0;
        PUBLISHED = 1;
        OUT_OF_PRINT = 2;
    }
    message BookStats {
        required int32 sales = 1;
        optional int32 citations = 2;
        optional Status bookstatus = 3 [default = PUBLISHED];
    }
    optional BookStats statistics = 3;
    repeated string keyword = 4;
}
```

Protocol Buffer Language (cont'd)

- Field-types can be:
  - Required fields
  - Optional fields
  - Repeated fields
    - Dynamically sized array
- Encoding-value
  - A unique number (=1,=2,…) represents a tag that a particular field has in the binary encoding of the message

```
message Book {
    required string title = 1;
    repeated string author = 2;
    enum Status {
        IN_PRESS = 0;
        PUBLISHED = 1;
        OUT_OF_PRINT = 2;
    }
    message BookStats {
        required int32 sales = 1;
        optional int32 citations = 2;
        optional Status bookstatus = 3 [default = PUBLISHED];
    }
    optional BookStats statistics = 3;
    repeated string keyword = 4;
}
```

A *.proto* File

- The specification of the message is contained in a *.proto* file
- The *.proto* file is compiled by *protoc* tool
  - The output of the *protoc* is a generated code that allows programmers to manipulate the particular message type
    - For example, assigning, extracting values to/from messages

```
public boolean hasTitle();
public java.lang.String getTitle();
public Builder setTitle(String value);
public Builder clearTitle();
```

- The *Builder* class:
  - Messages are immutable in protocol buffer, Builder class is mutable

Comparison of Protocol Buffer Language

- Advantages of Protocol Buffer (PB)
  - PB is 3-10 times smaller than an XML
  - PB is 10-100 times faster than an XML

- Can we compare PB with XML?
  - PB works only on Google infrastructure, which is relatively closed system and does not address inter-operability
  - XML is richer (it specifies self-describing data and meta-data). PB is not so rich. There are accessory programs that can create a full description. However, they are hardly used

Supporting RPC using Protocol Buffers

- PB produces a serialized data that can be used for storage or communications
- Most common use is to use PB for RPCs
- Example:

```
service SearchService {
    rpc Search(RequestType) returns (ResponseType)
}
```

  - RequestType can correspond to list of keywords
  - ResponseType can then correspond to a list of books matching the keywords
- *protoc* compiler takes this specification and produces
  - Abstract interface SearchService
  - A stub that supports type-safe RPC calls

Acknowledgements

Q&A

innovate achieve lead

BITS Pilani
Pilani Campus

Thank You