Distributed Computing
Assignment: Serialization Frameworks

Jishnu Sri Ojaswy Akella
17MCME05

November 9, 2020

1 Notes on Serialization Frameworks

1.1 What is Serialization?

We will start with a couple of simple analogies.

ANALOGY 1:
Suppose that I'm on the phone with my friend and I am talking to him about my new pet dog and he wants to know how it looks. Now, I cannot transfer a living animal over the phone, so, I will describe the main features such as color, breed, age and my friend on the phone will try to construct an image based off of this description.

Now a more technical approach in attempting to explain Serialization.

ANALOGY 2:
Suppose that there are two applications running on two physical systems. And, these two applications communicate/exchange data amongst themselves through a medium. This medium could be a protocol that supports exchange of data in memory such as TCP/IP or UDP. Now if System A wants to send the number 10 over the network to System B, it will convert 10 into its binary form which would be 1010 and send it across the network along with the meta information that describes the number, also in binary.

In the first analogy, I have encoded my pet into a series of descriptions Age: 6 months, Color: Brown, Breed: Labrador which on decoding one can imagine how my dog would look, and that is what I wanted. I got what I wanted without having to actually go over to his place to show him. In the second analogy, systems are communicating in an organised manner as well. These analogies are simple yet powerful in depicting the main function of Serialization that is providing a way to organise and transfer an object over a connection such that the receiver can easily decode it and use it.

Definitions, Features, and Uses

Serialization is the process of converting various items such as Data Structures, Objects in memory, and unordered data to a linear binary data stream/series of tokens which depending on the function can then be transmitted over a network or recreate when needed or just store it on desk.

Deserialization is the process of converting the serialized stream of bytes taken from either the network or persistent storage to the object it was initially, while retaining all of its properties.

The most important feature of Serialization is that the state of an object doesn't get lost in the conversion. Thus, on Deserialization we will have the exact object we had before we serialised it, without any loss.

Main uses of Serialization include sending an object through a web service, passing an object from one domain to another, passing an object through a firewall, and storing data on the desk.

Question: What is the use of Serialization when all it does is convert data into a binary state which is already the state in which data will be stored on a system?

Answer: All data in a system will obviously be stored in a binary form but what Serialization does is gives us the choice to organise the data in any manner we want before converting it into binary.

Question: Why should we use Serialization instead of just doing a copy/paste of the data?

Answer: We use Serialization because when we copy/paste binary data chances are that we could have copied the reference/pointer of the data in place of the actual value and apart from that the endianness isn't carried while trying to copy/paste. It is really important to retain the state of the data.

Examples of Serialization Frameworks

There are many examples of Serialization Formats. Some of these examples include formatting structures such as CSV(Comma Separated Values), JSON(Java Script Object Notation), XML(Extensible Markup Language), YAML(YAML Ain't Markup Language), Java Serialization, Pickle(Python). Most of such serialization formats are human readable, language-neutral and platform neutral.

Here is an example of a JSON object:
{"Employee": { "Name": "Arjun", "Salary": 50000, "Position": "Project Manager", "5 years Experience": True} }

2 Understanding Serialization Frameworks from the aspect of functionality

There are many real-life applications of Serialization which we across in our day to day lives. I will be discussing some of the Serialization formats and the applications that they are best suited for.

Serialization Formats and their Applications

- Cryptography - It is a priority to establish a secure communication between two nodes of a network in the presence of a third party. **ASN. 1** is a standard language where we can serialize and deserialize data structures across platforms. ASN.1 compilers can encode and decode data structures. This encoding associates with several standardized encoding rules such as BER (Basic Encoding Rules), PER (Packed Encoding Rules), which prove useful for applications that undergo restrictions in terms of bandwidth. It is both Operating System and Language independent. One end of transfer could be written in COBOL while the other end is written in C or Java, and we will still be able to exchange information using ASN.1 with its encoding rules. It supports various data types such as Integers, Boolean, Characters, etc. with which we can build structures and lists.

- Dynamic Web Pages - AJAJ (Asynchronous JavaScript and JSON) and AJAX (Asynchronous JavaScript and XML) are two major methodologies used in creating Dynamic Web Pages. While JSON was intended as a human readable standard file format, there is a use case of it as a configuration language. JSON schema is a specification that defines the structure of JSON data. It has been developed from XML Schema yet there are many difference between JSON and XML. One being that JSON can only support text and numbers whereas XML can support text, numbers, images etc. and JSON can only support UTF-8 encoding where as XML can support various such formats. XML is also more secure than JSON. JSON parsers hav been seen to suffer from DoS attacks. Although XML might seem to give an advantage over JSON, it depends on the functionality. JSON boosts performance on a heavy load server and can keep its bandwidth costs down.

- Peer to Peer Network - Bencoding is the encoding that the infamous Peer-to-Peer sharing system BitTorrent uses for transmitting data. It can support various data types such as byte strings, integers, lists and associative arrays(list of lists). BitTorrent's peer protocol operates over TCP or μTP. All integers sent in the protocol are encoded as four bytes big-endian. Each of the above data type are encoded in their own way. For Example: Strings have a length prefix (base ten) followed by a colon and the string. Eg: 5:hello corresponds to 'hello'.

- Lossless Compression - Fast Infoset is an international standard that can be used in binary encoding of any valid XML Information set. It provides smaller encoding sizes, faster processing without procuring any loss during the process. It has been adopted by the Web3D community as it suffices to their requirements. This format is considered similar to gzip. Fast Infoset parsing performance is between a factor of two to four better than XML parsing performance. Binary data can be directly stored in a Fast Infoset ouput document, without the need for text-based encodings like Base64. Characters, Integers and Float types however need to be encoded. To parse the output Fast Infoset document, POCO C++ libraries can be used.

- Gaming - Serialization is really important in various aspects of gaming such as Saving and Loading the game, Saving the state of object (user in game inventory and position). Cocos2d is a cross platform software that can be used to build games, apps and any other applications with a graphical interface. This software uses FlatBuffer, a cross platform serialization library. This library is used for performance-critical applications due to its memory efficiency, speed and the fact that it stores hierarchical data is a binary buffer without the need to parse. The schema used here is similar to the syntax of 'C' language. Various data structures such as tables, structs, arrays, values, enums can be used. Built in scalar data types include signed and unsigned byte, signed and unsigned int, singed and unsigned long, float and double.

Serialization Formats used by Major Corporations

- Amazon: Ion - Ion is a hierarchical data serialization format and a superset of JSON as it also supports timestamps, embedded binary values, symbols and expressions. It also supports infinite precision decimal which will be used in financial record keeping. Ion also takes advantage of its dual format, they are, text and binary. The text form makes it human readable, thus making it easy for diagnosis and the binary form doesn't take up much space while also retaining the state of an object. Due to all these features Amazon uses Ion to handle the large-scale, service-oriented architectures. It doesn't require any meta data as it is self describing. This provides greater flexibility over schema-based formats such as Protobuf, Thrift, and Avro.

- Apple: Property List - It is type of representation of hierarchical objects used by Apple. It offers applications with a way to make them lightweight. The data hierarchy will be retained via an object graph. It can serialize a file, which later on can be reconstituted and recreate the original hierarchy. It supports types such as integers, floating point values, date, boolean, data and string. Structures such as arrays and dictionaries can be built from this.

- Facebook: Apache Thrift - Thrift is a schema based, software framework that supports cross language serialization. It comes along with a code generator that can build services(clients and servers) in desired language. We have the option to define data types and service in a separate .thrift file which would be given as input to the code generator. It supports various types such as Boolean, Byte, signed 16,32,64 bit integers, double and string. A service consists of a set of named functions, each with a list of parameters and a return type.

- Facebook App for Android: FlatBuffer - As described in the Gaming Section above.

- Google: Protobuf - Protobuf is a language free, platform free mechanism to structure and serialize data. This schema uses protobuf, an exchange format developed in 2008 by Google. protobuf represents data as messages whose fields are indicated and aliased with a number and tag. Fields can be required, optional, or repeated. It looks like XML but is simpler and faster that it. Here, we can define the structure we want of our choice and then use the generated code to input into a variety of data streams. Protocol Buffers and Flat Buffers have both been developed by Google. But the basic difference is that Flat Buffers does not need a step for parsing where as Protobuf needs that.

- Hadoop: Avro - Apache Avro is a schema based data serialization system with a remote procedure call framework. It provides rich data structures, fast and compact binary conversion and can be easily integrated with dynamic languages. Avro schemas are defined with JSON, thus making it compatible with any language that can support JSON libraries. The converted binary format can be stored in a file along with its schema. The schema is written so that when a program comes over expecting other schema, that issue can be easily resolved. One more advantage of writing with the schema is that, the serialized object can be read directly without knowing the schema beforehand thus making it self describing. Multiple languages such as C, C++, C, Java, Haskell, Go, Python have written APIs that provide the option of direct implementation.

Honorable Mentions:

SOAP + XML: SOAP is a protocol that supports XML which can help systems with different Operating Systems in authenticating, authorizing and communicating amongst themselves.

XDC: Developed by SUN Microsystems, this serialization format is compatible with different operating systems and different platforms. It is used in the OSI 7 layer computer network protocols. It has a base unit of 4 bytes and is serialized in big endian order. It offers support to various data types such as Boolean, signed and unsigned int, signed and unsigned hyper, IEEE float and

IEEE double, strings, arrays etc. It stands for External Data Representation.

There are serialization tools which are specific to their own language and are not human readable. Some examples of such tools are: S-Expression (LISP), Pickle (Python), Java Serialization (Java). There is a tool available named Jackson, this is a lightweight JSON processor for Java.