発表概要

代数的データ型から型定義とバイナリ形式へのデコーダとエンコーダのコード生成

鳴海 秀人[1,a)]

2020年10月29日発表

Web アプリのクライアントとサーバの通信には HTTP，WebSockets などを使用し，データをやりとりする．実際にわたすのはバイナリ形式でクライアントとサーバでは何らかの方法でアプリでやりとりするデータをバイナリ形式にエンコード，デコードする必要がある．アプリのデータ型からバイナリ形式に変換できるものとして JSON，XML，Protocol Buffers などが用いられる．JSON，XML はテキストデータであり，画像などのバイナリデータを含めるには base64 などの形式をさらに介さなければならないため容量が大きくなり，クライアントとサーバで別の言語を使用した場合はデータ型の辻褄を合わせる必要がある．Protocol Buffers は独自のスキーマ形式で記述し，画像などのバイナリデータのサイズを維持し，言語間の辻褄が合わせを行ってくれるが，代数的データ型を直接含めることはできない．代数的データ型はありえないパターンを減らして，データを分かりやすく扱えるものだ．この研究で作成した definy-core のコード生成機能を使えば 1 つの定義で型定義とバイナリ形式へのデコーダ，エンコーダのコードを生成できる．スキーマの定義とコード生成はコードで行うため，既存言語の入力支援の恩恵を受けることができ，エディタ拡張機能を用意する必要がなく，コードを生成する順番など柔軟に対応できる．現在 TypeScript のコード生成のみサポートしている．definy-core 自身の型定義も definy-core を用いて作られている．Web アプリケーションのために開発したが，それ以外の用途でも使用可能．

Presentation Abstract

Code Generation for Type Definitions, Binary Format Decoders, and Encoders from Algebraic Data Types

Hideto Narumi[1,a)]

Presented: October 29, 2020

The client and the server communicate using communication methods such as HTTP and WebSockets to exchange data. What is passed is in binary format, and the client and the server need to encode and decode the data exchanged by the application in some way to the binary format. JSON, XML, Protocol Buffers, etc. are used as the ones that can be converted from the application data type to binary format. JSON and XML are text data, and if you want to include binary data such as images, you need to use a format such as base64, which will increase the size of the data. Protocol buffers have their schema and maintain the size of binary data such as images, and do not need to write multiple languages, can not include algebraic data types directly. Algebraic data types are designed to reduce impossible patterns and to make the data easier to understand. The code generator of definy-core, which we have developed in this study, allows us to generate type definitions and codes for decoders and encoders to binary formats with a single definition. Since the schema definition and code generation are done in code, we can benefit from the input support of the existing languages and there is no need to provide editor extensions and the order of code generation is flexible. Currently, only TypeScript code generation is supported. Definy-core's type definitions are also made with definy-core. Although it is designed for web applications, it can be used for other purposes as well.

[1] 東京電機大学未来科学部情報メディア学科実空間コンピューティング研究室
Cyber Phisical System Lab, Department of Information Systems and Multimedia Design, School of Science and Technology for Future Life, Tokyo Denki University, Adachi, Tokyo 120–8551, Japan
[a)] naruminchoo@cps.im.dendai.ac.jp