Disciplina: Sistemas Distribuídos

Ano/Semestre: 2018/1

Professores: Windson Viana e Fernando 30

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Roteiro de Estudo sobre Representação de dados de comunicação em

SD

**1- Contexto**

O problema da representação de dados para comunicação aparece quando

queremos comunicar partes de um sistema que se encontram em plataformas ou

sistemas operacionais distintos. Esse problema pode ser visto de duas formas.

Primeiro, como codificar os dados do meu sistema para serem transmitidos em

um stream de bytes de forma que seus valores tenham a mesma semântica em

duas plataformas. Segundo, como estruturar (organizar, concatenar ou agrupar)

os dados para que a outra parte do sistema possa recuperá-la e utilizá-la.

A primeira parte do problema está relacionada a codificação dos dados

( *encoding* ). Vale lembrar, por exemplo, que uma sequência de bits representando

um inteiro (e.g., #AAFF) pode ter valores distintos dependendo do sistema

operacional ou da linguagem de programação (i.e., um problema conhecido como

endianess1). Além disso, o mapeamento dos bits para um caracter depende de

uma tabela de codição (e.g., ASCII, Unicode, UTF-8, Base-64). A conversão de

dados de um formato/tipo/objeto, como uma instância de String em Java, em

outro formato (e.g., um stream binário em UTF-8) é um exemplo de codificação.

Geralmente, a codificação refere-se ao uso de um codec (tabela combinada entre

as partes) para converter um objeto em um stream de bits. Uma pequena revisão

ilustrada do problema pode ser vsita em:

1 https://www.youtube.com/watch?v=B4t1lq3SlAY

https://www.youtube.com/watch?v=SbzxEt4oXGY .

**No caso específico dessa aula,** nossa preocupação maior é com a segunda parte

do problema, a estruturação dos dados a serem enviados. A estruturação de

dados para transporte ou armazenamento requer serialização de bytes em uma

ordem específica. Isto é, para que a sequência resultante de bytes seja escrita de

uma maneira que possa ser lida por outro processo/máquina. Por exemplo,

deseja-se transmitir as coordenadas geográficas, o nome e o id de usuário enter

duas partes de um sistema (e.g., aplicação móvel e o servidor Web). Em qual

sequência esses dados devem ser agrupados? Como separar essas informações de

forma que o outro processo possa recuperá-las?

Em Sistemas Distribuídos, os mecanismos que se encarregam disso são chamados

de processos de *marshalling e unmarshalling* . De certa forma, a serialização de

objetos é um exemplo desse processo. No decorrer do roteiro, veremos quatro

outras formas de representação de dados que podem ser utilizadas para esse fim.

Bibliotecas em várias linguagens de programação estão disponíveis para

transformar um objeto em uma representação nesses formatos (e.g., Java Object

to JSON, Pythn object to XML,....)

**2- Instruções**

Esse roteiro foi projetado para o ensino de representação de dados de

comunicação em Sistemas Distribuídos. As leituras e vídeos recomendados vão

guiar o seu estudo. Se você já sabe parte dos conteúdos apresentados, não é

necessário assistir a todos os vídeos.

O seguinte formulário pode ser usado para você fazer uma autoavaliação de

conhecimentos. Após responder, clique em “Ver pontuação” para receber seu

feedback e quais dos assuntos você te o domínio suficiente para não precisar ver

os vídeos.

https://docs.google.com/forms/d/e/1FAIpQLSeDpYzq9BR\_raFxGpnGCM3w980R5

X9qZhtyBfjI9LvSybwAXw/viewform?usp=sf\_link

A seguir, são apresentadas ferramentas que podem ser utilizadas para a melhor

visualização e entendimento dos conceitos apresentados neste roteiro:

Tema Link

Visualização, minimização validação de

JSON

https://codebeautify.org/jsonviewer

Ferramentas YAML https://onlineyamltools.com/

Validação de documentos XML https://www.w3schools.com/xml/xml\_validator.asp

**3- Leitura Recomendada**

A representação de dados para transmissão entre sistemas é uma característica

essencial para comunicação entre componentes de Sistemas Distribuídos. As

principais formas de fornecer interoperabilidade para um sistema são

apresentadas a seguir.

Tema Link

Serialização - Introdução https://www.devmedia.com.br/introducao-a-seri

alizacao-de-objetos/3050

Serialização com Java https://www.devmedia.com.br/serializacao-de-o

bjetos-em-java/23413

Serialização com Python https://code.tutsplus.com/pt/tutorials/serializationand-

deserialization-of-python-objects-part-1--cms-

26183

Vídeo de Introdução à

interoperabilidade e serialização

https://www.youtube.com/watch?v=uS37TujnLRw

**4 - Vídeos XML**

Tema Link

Introdução rápida ao XML https://www.youtube.com/watch?v=hBnhsr4Eyl8&

t=52s

Introdução ao XML de forma detalhada https://www.youtube.com/watch?v=9V4TFPD\_2-

U&t=10

Curso completo de XML(playlist) https://www.youtube.com/watch?v=tAN-1xUsftg

&list=PLBB413675AFBDC1F4

XML Schema https://www.youtube.com/watch?v=56pR\_5rO-m

4

DTD versus XML Schema https://www.youtube.com/watch?v=BPw-iesWa\_

8

**5- Vídeos JSON**

Tema Link

Introdução rápida ao JSON < em inglês > https://www.youtube.com/watch?v=N7svaYUZW

NA

Introdução ao JSON < em português> https://www.youtube.com/watch?v=H3Q63XyHk2

0&feature=youtu.be

Comparação entre JSON e XML [1] <em

inglês>

https://www.youtube.com/watch?v=FpZbb\_\_2O3

E&feature=youtu.be

Comparação entre JSON e XML [2] <em

inglês>

https://www.youtube.com/watch?v=95X-pHvGBn

w&feature=youtu.be

Comparação entre JSON e XML [3] <em

Português>

https://www.youtube.com/watch?v=hBnhsr4Eyl8

&feature=youtu.be

**6 - Leituras e Vídeos YAML**

Tema Link

Introdução ao YAML <em inglês> https://www.youtube.com/watch?v=U9\_gfT0n\_5

Q&feature=youtu.be

Sintaxe YAML< em inglês > https://www.youtube.com/watch?v=W3tQPk8DNb

k&feature=youtu.be

Tópico no stack overflow https://pt.stackoverflow.com/questions/80881/oque-%

C3%A9-e-pra-que-serve-yaml

Comparação entre JSON, XML e YAML <

em Português>

https://zenorocha.com/xml-pra-que-conheca-o-js

on-e-o-yaml/

Comparação entre JSON, XML e YAML

<em inglês>

https://www.youtube.com/watch?v=FEhOGrnN0

bM

**7 - Leituras e Vídeos Protocol Buffers**

Tema Link

Introdução rápida <em inglês> https://www.youtube.com/watch?v=AW09fAsEb

00

Tutorial em vídeo < em inglês> https://www.youtube.com/watch?v=72mPlAfHIjs

Protocol Buffer & JSON <em inglês> https://www.youtube.com/watch?v=9IUrAZHxn3

s

Documentação oficial< em inglês> https://developers.google.com/protocol-buffers/

docs/overview

Página do GitHub https://github.com/google/protobuf

**8 - Atividades a serem realizadas em casa**

Monte uma equipe de até 4 pessoas. Escolha uma linguagem de programação de

preferência da sua equipe e separe códigos exemplos de como serializar e

desserilizar dados nas estruturas de representação de dados externos

apresentadas (i.e., XML, JSON, YAML, Protocol Buffers).

**9 - Atividade a ser realizada em Sala (11/05)**

**Formato de entrega: relatório com respostas de cada questão**

1 - Imagine, então, um aplicativo móvel que mostra Trailers de lançamentos do

cinema e suas avaliações do IMDB. Para tal, ele pagina sua interface e faz o

download da descrição de apenas 10 filmes seguindo o formato listado abaixo.

Mostre como seria a representação dessa lista de 10 filmes nos quatro formatos.

Para o Protocol Buffer, mostre como seria o código da mensagem, já que o

payload seria binário.

class Filme {

int FilmeID;

String titulo;

int ano;

int Avaliacao\_IMDB;

List<Filme> tres\_filmes\_mais\_relacionados;

URL linkparaTraillerNoYoutube;

}

2- Para essa atividade, escolha duas estruturas de representação externa e

implemente o processo de serialização e desserelização dos dados. Você pode

simular a comunicação do aplicativo móvel com o servidor usando sockets e

usando o console como “interface gráfica”

3- Nessa parte da atividade em sala, vamos mesclar as práticas de PBL (Problem

Based Learning) e Aula Invertida para fazermos uma análise comparativa das 4

tecnologias de representação de dados externos. 5 casos serão apresentados e

você deve escolher uma tecnologia adequada e uma inadequada para cada um

deles produzindo uma justificativa para tal

Caso 1 – Recomendado: XML, pela boa formação de dados (usando esquemas) e a navegação entre os nós do XML é facilitada.

Não recomendado: YAML, pelo tempo elevado de load.

Caso 2 – Recomendado: YAML se os proprios usuários construissem suas próprias informações, pois a leiturabilidade humana do YAML é elevada. Caso contrário, JSON.

Não recomendado: XML, pois os dados podem crescer bastante em curto prazo e o arquivo pode estar muito grande e poluido de tags XML.

Caso 3 – Recomendado: Protobuf, pelo tamanho menor e pela velocidade de leitura, o que faz diferença em sistemas de tempo real.

Não recomendado: XML, por causa do tamanho maior e volume de mensagens recebidas pelo servidor.

Caso 4 – Recomendado: JSON, pois como não há necessidade de alto desempenho, o JSON tem vantagem por ser mais legível por humanos.

Não recomendado continua sendo o XML.

Caso 5 – Recomendado: XML, pelo suporte a esquemas e boa formação de dados.

Não recomendado: Protobuf, por ser mais difícil de ser implantado por não ser bem portável.

4- Por fim, compare, de forma superficial, as quatro tecnologias estudadas para

seu uso em *marshaling* e a técnica de serialização de objetos da sua linguagem

escolhida. Use os seguintes critérios: tamanho dos dados gerados, facilidade de

programação, interoperabilidade e desempenho.

5- Discussão/Debate em sala

Vamos discutir os resultados da questão 3