FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Departamento de Informática

Mestrado Integrado em Engenharia Informática
Sistemas Distribuídos – exame, 3 de julho de 2017
2º Semestre, 2016/2017

NOTAS: Leia as questões atentamente antes de responder. **O exame é sem consulta. A duração do exame é 2h30min**. O teste contém **11** páginas.

Nome: _____ **Número**: _____

1. Pretende-se desenvolver um serviço web para gerir informação sobre séries de televisão. Para esse fim, um serviço REST e um serviço SOAP irão disponibilizar as seguintes operações: i) adicionar um novo tipo de série (*genre*) caso este não exista já; ii) adicionar informação sobre uma nova série associada ao seu tipo e ao nome da série, a informação sobre a série encontra-se dentro de uma instância da classe TVShow; iii) listar todas as séries conhecidas (i.e, devolve instâncias das classes TVShow). No caso do serviço REST esta operação pode ser parametrizada com um parâmetro opcional chamado *genre* que faz com que o serviço apenas reporte informação das séries daquele tipo. No caso do serviço REST as operações que adicionam um novo tipo de série ou uma nova série retornam ao cliente um erro (código HTTP 409 – CONFLICT). No caso do serviço SOAP o cliente recebe uma indicação que a operação falhou através do seu retorno (*false*).

   a) Complete o Anexo A, que apresenta a implementação do serviço descrito em cima em REST, como achar necessário, sabendo que as operações listadas acima utilizam os seguintes URLs para um servidor a executar na máquina local no porto 8080.

      i) http://localhost:8080/tvshows/{genre}

      ii) http://localhost:8080/tvshows/{genre}/{showname}

      iii) http://localhost:8080/tvshows/

      ou http://localhost:8080/tvshows?genre=action

   b) Complete o Anexo B, que apresenta a implementação do serviço descrito em cima em SOAP, como achar necessário.

2. No Anexo C, apresenta-se um excerto do código java de um serviço REST que gere uma coleção de ficheiros replicada em vários servidores idênticos, em particular mostra os endpoint REST utilizados para eliminar um ficheiro existente. A ideia é simples, quando um cliente solicita a remoção de um ficheiro, o servidor que recebe esse pedido recorre ao método auxiliar *getServerList()* para obter uma lista dos servidores ativos (incluindo o próprio) de um serviço externo. Posto isto executa uma operação em cada um dos servidores que efetivamente remove o ficheiro. No entanto quando se executa o cliente mostrado no Anexo D, que contacta um dos servidores, o cliente ao fim de muito tempo não obtém resposta. Explique o que explica este comportamento e corrija o código de forma a resolver este problema.

3. Indique se cada afirmação é **[V]erdadeira** ou **[F]alsa** (**nota: respostas incorretas descontam 2/3 da cotação de uma resposta certa**):

___ Pesquisar por um identificador é algo que um sistema P2P não estruturado consegue realizar, mas de forma pouco eficiente, face a um sistema P2P estruturado.

___ Um sistema P2P não estruturado é geralmente mais resiliente às falhas que um sistema P2P estruturado.

___ O particionamento tem como principal objetivo aumentar a disponibilidade de um sistema distribuído.

___ Um proxy num sistema distribuído pode ter diversos papéis, por exemplo mascarar falhas de conectividade.

___ Não há interesse em colocar um proxy junto do servidor, este deve ficar sempre próximo do cliente.

___ Num sistema de comunicação por middleware, a entrega duma mensagem pode ocorrer num momento distinto da receção da mensagem na máquina.

___ No Java RMI, todos os métodos que um servidor exporta devem lançar a exceção *RemoteException*.

___ A grande vantagem do mecanismo de serialização de dados ProtoBuf, quando comparado com o JSON, é que os ProtoBuf por serem implementados pela Google estão disponíveis em mais linguagens.

___ Num servidor REST, no suporte disponível no JAX-RS Jersey, o programador não necessita de se preocupar com acessos concorrentes às variáveis mantidas pelo objeto que implementa o serviço.

___ A semântica de invocação "at most once" pode ser implementada por um protocolo sem repetição do envio das mensagens.

___ A tolerância a falhas num sistema distribuído pode ser conseguida propagando os dados propriamente ditos ou replicando as operações sobre os dados.

___ No algoritmo primário-secundário, um cliente pode ler uma versão de um secundário mais antiga do que a versão que está no primário.

___ O sistema de *caching* usado com o NFS obriga a que o servidor e os clientes tenham os relógios sincronizados (a menos de um pequeno erro).

___ Um requisito fundamental para melhorar a segurança de um sistema informático assenta em utilizar algoritmos criptográficos abertos, cujo código esteja no domínio público.

___ Um certificado de chave pública, entre diversos dados, é composto pela chave pública do detentor do certificado cifrada com a chave pública da entidade certificadora.

___ A cifra por blocos encadeados é um mecanismo que permite cifrar/decifrar um volume arbitrariamente grande de informação com base em primitivas que cifram/decifram um número fixo de bits de cada vez.

___ A termo segurança futura perfeita aplica-se aos sistemas que estão imunes a todos os tipos de ataques no domínio da segurança dos sistemas informáticos, incluindo os de negação de serviço.

___ O DNS é sistema hierárquico para tradução de URNs em endereços IP.

___ Ao contrário do DNS, o Zookeeper oferece um serviço de nomes com consistência forte.

Nome: _____ **Número**: _____

4. O capitão Joaquim Pardal está aborrecido. Não há vento e é preciso matar o tempo. Na última aventura, lá para os lados do Amazonas, "comprou" um livro: A Origem das Espécies, de Charles Darwin. Excitado com a leitura, encontrou ali a chave para decidir quem da sua vasta descendência considerável (autenticidade duvidosa) herdaria o seu considerável legado. A natureza iria revelar o melhor pirata e o(a) verdadeiro(a) Pardal herdará o seu legado!!!

   Decidiu então enviar uma mensagem a todos os candidatos, detalhando o seu plano. Porém, com um putativo Pardal júnior em cada porto dos sete mares, com tempos de trânsito muito diferentes entre si, será necessário não favorecer nenhum deles.

   Com base num medalhão, contendo um segredo, que cada putativo (suposto) descendente recebeu à nascença, a mensagem terá que garantir: (1) todos irão obter o mapa do tesouro ao mesmo tempo, num local cujas coordenadas estão escondidas na mensagem; (2) reunidos nesse local, juntando os segredos contidos em cada medalhão, as coordenadas do tesouro serão reveladas a todos.

   Infelizmente, apesar de saber quais os medalhões que mandou, o capitão Joaquim Pardal esqueceu-se quem recebeu qual. Assim, **enviará a mesma mensagem a todos**. Esta mensagem será insuficiente para chegar diretamente ao tesouro.

   Ajude o comandante Joaquim Pardal a desenhar a mensagem que ele deverá enviar. Baseie a sua solução nos seguintes símbolos.

   P+L+D – *Respetivamente, a descrição do plano do Joaquim Pardal bastante extenso (e cheio de detalhes irrelevantes como alguns nesta pergunta...), o local e a data do encontro. (tudo isto informação secreta, para evitar falsos pretendentes...)*

   $\mathbf{M_i = S_i, T_i,}$ com i de 1 a k. (O capitão Joaquim Pardal gosta de soluções genéricas.). Medalhão partilhado previamente com o(a) putativo(a) piratinha, contendo uma chave simétrica secreta $\mathbf{S_i}$ e $\mathbf{T_i}$ informação incompleta sobre a localização do tesouro.

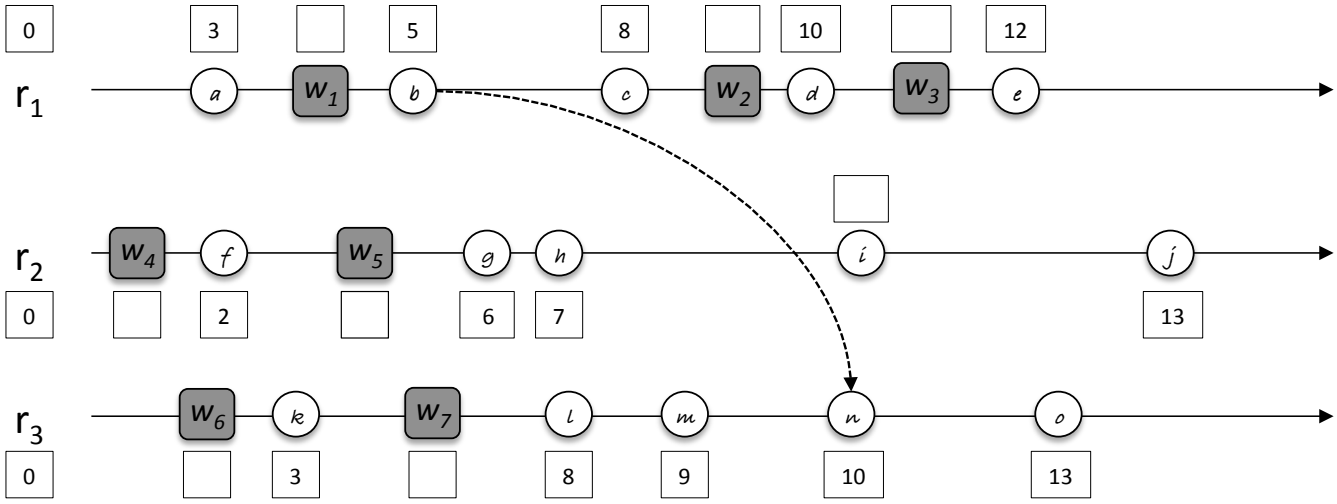
   $\mathbf{F(\ T_1, T_2, \ldots, T_k)}$ **–** uma fórmula, descrita em **P,** para combinar os $\mathbf{T_i}$ e obter **T**.

   T – coordenadas do tesouro, só podem ser obtidas após aplicar a fórmula **F** a **todos** os $\mathbf{T_i}$

   a) Comece por indicar **a parte da mensagem** destinada **apenas** a revelar aos candidatos o plano, o local e a data do ponto de encontro do desafio. A mensagem deve ser tão pequena quanto o possível. (Na altura não havia impressoras e tudo era escrito à mão, em cursivo). Os candidatos deverão poder confirmar que a informação está correta e é genuína. Todos receberão a mesma mensagem.

   b) Complete a mensagem de modo a que, no local de encontro, os candidatos possam confirmar que não foram enganados e obtiveram o valor real de **T**, após aplicarem a fórmula **F** aos $\mathbf{T_i}$ que partilharam entre si.

5. Considere um sistema armazenamento de dados distribuído composto por três réplicas: r1, r2 e r3. No diagrama temporal, abaixo, mostra-se um traço de uma execução deste sistema, onde as operações de escrita correspondem aos eventos: w1, w2, ..., w7. O sistema utiliza **comunicação multiponto assíncrona** para disseminar as atualizações das réplicas. Os eventos assinalados com um círculo e identificados por uma letra, correspondem ao **envio** ou à **entrega** dessas mensagens multiponto na réplica correspondente. As setas que representam a comunicação entre as réplicas foram omitidas, à exceção de *b -> n* que indica que a réplica r1 em *b*, enviou uma mensagem para as réplicas r2 e r3, tendo essa mensagem sido entregue na réplica r3 em n (não estando assinalado qual o evento corresponde à sua entrega na réplica r2).



a) Ignorando as caixas no diagrama (algumas delas já preenchidas) indique um conjunto de mensagens multiponto, tão reduzido quanto o possível, de modo a que as garantias de entrega da comunicação multi-ponto **não sejam** compatíveis com uma ordem causal? **Não rasure ou desenhe setas no diagrama;** responda, abaixo, na forma:
<ev_envio -> ev_recep1, ev_recep2>

b) Reconstrua o padrão de comunicação associado ao diagrama acima, sabendo que nas caixas, acima ou abaixo de cada evento, o valor já preenchido corresponde ao valor de um relógio lógico/de Lamport. O relógio tem valor inicial 0 em cada réplica e o incremento mínimo é de 1 unidade.

i) Preencha os valores do relógio lógico em falta nas caixas respetivas.
ii) Identifique as operações multiponto <x -> y, z > onde x, representa o evento de envio e y, z os eventos de entrega da mensagem nas outras duas réplicas.

< b -> n, _____>

c) Tirando partido dos valores do relógio lógico, defina uma ordem total para as operações de escrita que respeite a causalidade. Se não preencheu o valor do relógio de uma ou mais operações de escrita, responda usando o valor do relógio do evento seguinte na réplica em questão.

Nome: _____ **Número**: _____

6. Suponha que pretendia criar um jogo multi-utilizador com realidade aumentada para dispositivos móveis, com uma aproximação à do pokèmon go.
   Neste jogo, a posição do utilizador no jogo corresponde à sua posição real no mundo, existindo jogadores de todo o mundo. Além dos utilizadores existe um conjunto de objetos virtuais que os utilizadores podem apanhar quando estão próximos do objeto ou largar no local em que estão.
   Para servir de suporte ao seu jogo, pretende implementar um serviço que mantenha a informação do jogo, com as seguintes operações:
   (1) void **novoJogador** (id_user)
   (2) array<id_obj,coordenada> **novaPosicao** (id_user, coordenadas)
   (3) boolean **apanharObjeto** (id_user, id_obj)
   (4) void **largarObjeto** (id_user, id_obj)
   (5) array<id_obj> **listarObjetos** (id_user)
   que permitem respetivamente: (1) criar um novo jogador; (2) registar a posição do utilizador, retornando a lista de objetos próximos do utilizador; (3) apanhar um objeto; (4) largar um objeto na posição atual do utilizador (como reportado); (5) listar objetos que o utilizador tem apanhados.

   a) Suponha que tem um conjunto de centros de dados distribuídos pelo mundo ao seu dispor. Explique como guardaria a informação relativa aos utilizadores e aos objetos para que as várias operações possam ser efetuadas de forma eficiente e tolerante a falhas.

Utilizadores:

Objetos:

   b) Para implementar este serviço seria interessante usar um serviço de edge servers? Se sim, explique como. Se não, explique porquê.

Sim, porque… / Não, porque…

c) Independentemente das suas respostas anteriores, suponha que num centro de dados pretende manter informação sobre um conjunto muito numeroso de utilizadores, mantendo para cada utilizador várias informações (nome, fotografia, etc.) Supondo que o centro de dados tem 100 máquinas, explique que arquitetura usaria para armazenar esta informação.

d) Considere que pretende disponibilizar uma API REST para as operações indicadas. Apresente o URL e operação que usaria para cada operação.

(1)

(2)

(3)

(4)

(5)

e) Suponha que pretendia implementar uma funcionalidade em que os utilizadores seria notificados sempre que um outro utilizador se aproximava da sua posição. Seria interessante usar o mecanismo de web sockets na implementação desta funcionalidade? Justifique.

Sim, porque… / Não, porque…

Nome: _____ **Número**: _____

f) Suponha que pretendia manter réplicas do serviço recorrendo a um sistema de comunicação em grupo (*multicast*), em que um cliente enviaria cada uma das operações por *multicast* para um conjunto de servidores. Para cada operação indique, **justificadamente**, que propriedades devem ser respeitadas na sua propagação (fiabilidade, ordem).

(1) FIABILIDADE:

ORDEM:

(2) FIABILIDADE:

ORDEM:

(3) FIABILIDADE:

ORDEM:

(4) FIABILIDADE:

ORDEM:

(5) FIABILIDADE:

ORDEM:

7. Considere o sistema de replicação implementado no sistema Coda.

a) Explique porque razão a escrita é efetuada em duas fases (de comunicação entre o cliente e os servidores).

b) Suponha que pretende implementar a seguinte modificação ao sistema: em vez de o cliente enviar uma operação diretamente para as várias réplicas (servidores), o cliente enviaria a operação para uma réplica qualquer, que se encarregava de propagar a operação para as outras réplicas (que conseguisse contactar) antes de devolver o resultado ao cliente. Usando esta aproximação seria necessário continuar a usar um vetor para registar a versão do servidor, ou bastaria usar um inteiro com o número da versão? Justifique explicando como atualizaria a informação de forma a detetar escritas concorrentes.

Vetor, porque… / Inteiro, porque…

Anotações JAVA REST Jersey:

@Path( )
@GET
@POST
@PUT
@DELETE
@Consumes( )
@Produces( )
@PathParam( )
@QueryParam( )

Listagem de Media Types a considerar:

MediaType.APPLICATION_OCTET_STREAM
MediaType.APPLICATION_JSON

Nome: _____ **Número**: _____

Anexo A

```
public class TVShowsResources {

  private Map<String, Map<String,TVShow>> database;

  public TVShowsResources() {
    this.database = new HashMap<String, Map<String,TVShow>>();
  }




  public void createGenre(@PathParam("genre") String genre) {
    if(database.containsKey(genre))
      throw new WebApplicationException(                );
    this.database.put(genre, new HashMap<String, TVShow>());
  }




  public void createTVShow(@PathParam("genre")    String genre,
                @PathParam("showname") String tvshow,
                             TVShow show) {
    if(!database.containsKey(genre) ||
    database.get(genre).containsKey(tvshow))
      throw new WebApplicationException(                );
    this.database.get(genre).put(tvshow, show);
  }




  public List<TVShow> listTVShow(@QueryParam("genre") String genre) {
    List<TVShow> reply = new ArrayList<TVShow>();
    if(genre != null && this.database.containsKey(genre)) {
      reply.addAll(this.database.get(genre).values());
    } else if (genre == null) {
      for(String s: this.database.keySet())
        reply.addAll(this.database.get(s).values());
    }
    return reply;
  }
}
```

Anexo B

```
public class TVShowsServiceImpl {

  private Map<String, Map<String,TVShow>> database;

  public TVShowsServiceImpl() {
    this.database = new HashMap<String, Map<String,TVShow>>();
  }




  public boolean createGenre(           String genre) {
    if(database.containsKey(genre))
      return false;
    this.database.put(genre, new HashMap<String, TVShow>());
    return true;
  }




  public boolean createTVShow(              String genre,
                                            String tvshow,
                                            TVShow show) {
    if(!database.containsKey(genre) ||
    database.get(genre).containsKey(tvshow))
      return false;
    this.database.get(genre).put(tvshow, show);
    return true;
  }




  public List<TVShow> listTVShow(           String genre) {
    List<TVShow> reply = new ArrayList<TVShow>();
    if(genre != null && this.database.containsKey(genre)) {
      reply.addAll(this.database.get(genre).values());
    } else if (genre == null) {
      for(String s: this.database.keySet())
        reply.addAll(this.database.get(s).values());
    }
    return reply;
  }
}
```

Nome: _____ **Número**: _____

Anexo C

```
@Path("/storage")
public class SimpleStorageResources {
  …
  private String[] getServerList() {
   …
  }

  @DELETE
  @Path("/{file}")
  public void deleteFile(@PathParam("file") String file) {
    ClientConfig config = new ClientConfig();
    Client client = ClientBuilder.newClient(config);

    String[] servers = this.getServerList();

    for(int i = 0; i < servers.length; i++) {
      URI baseURI = UriBuilder.fromUri(servers[i]).build();
      WebTarget target = client.target(baseURI);
      try{
        target.path("/storage/" + file).request().delete();
      } catch (Exception e) {
        e.printStackTrace();
      }
  } }

  @DELETE
  @Path("/local/{file}")
  public void localdeleteFile(@PathParam("file") String file) {
    File f = new File(file);
    f.delete();
} }
```

Anexo D

```
public class FileDeleteClient {

   public static void main(String[] args) throws IOException {

      ClientConfig config = new ClientConfig();
      Client client = ClientBuilder.newClient(config);
      URI baseURI = UriBuilder.fromUri("http://fs1.di.fct.unl.pt:9090/").build();

      WebTarget target = client.target(baseURI);
      try{
         target.path("/storage/test.txt").request().delete();
      } catch (Exception e) {
         e.printStackTrace();
         System.out.println("Not OK");
         return;
      }
      System.out.println("OK");
   }
}
```