Universidade Federal de Minas Gerais (UFMG)
Escola de Ciência da Informação (ECI)
Colegiado de Biblioteconomia

ECI046 – Ontologias em Organizações
Prof.: Renato Fabiano Matheus
Supervisão: Maurício Barcellos
Nome: Thiago dos Reis Soares da Silva

Atividade Avaliativa 02 - Exercício individual

Versão 20181025 (as modificações feitas após versão inicial estão marcadas)

Prazo de entrega: 05/11/2018 até 23h55 Valor: 40 pontos Entrega via Moodle.

Obs: entrega com atraso implica subtração de 3 pontos a cada dia. Entrega em 10/11/2018 até 23h55 valendo 3 pontos.

Descrição da atividade

1. Especificar um **problema organizacional** a ser resolvido utilizando **ontologias computacionais**, conforme detalhado no item **Especificação** deste documento.

Passos preliminares

2. Criar uma cópia deste documento no Google Drive e editar a cópia como seu documento de entrega
    - URL do documento:
      (URL do documento base)
      (Usar menu File ⇒ Make a Copy do Google Drive com usuário Google conectado e depois fazer SHARE ⇒ "Get Shareable Link" ⇒ "Done" e depois copiar endereço do documento a seguir)
      <Endereço deste documento no Google Drive>
    - **ENTREGA: documento com respostas em formato PDF via Moodle.**
    - Sugere-se colocar também uma cópia do PDF no seu Github: <endereço github>
3. Identifique-se: Aluno: <Aluno>

Requisitos de arquitetura da ontologia (a implementação das ontologias será objetos das Atividades 03 e 04)

4. Criar uma nova ontologia OWL básica em RDF/XML usando Protégé e/ou Protégé Web, cujo nome deve estar relacionado com a organização e o problema cuja solução você irá modelar e implementar. Os nomes das classes e propriedades de sua ontologia base devem ser em português.
5. Agregar à sua ontologia básica pelo menos outras 2 (duas) ontologias vistas durante o curso ou disponíveis na Web, e.g.: Schema.org, FOAF, DBPedia Linked Data, SKOS, BFO e OBO-Foundry, … (ver slides usados em aulas).
6. Sua ontologia base deve conter pelo menos 5 classes, cada classe pelo menos 3 atributos e 3 consultas

SPARQL. As consultas SPARQL devem consultar preferencialmente pelo menos 2 classes.

7. Lembre-se de usar restrições de propriedades OWL (InverseOf, SameAs, DistinctWith, Min/Max) (ver apresentações sobre OWL).
8. Procure usar outras características para propriedades de dados ("lang", com diferentes línguas "en", "pt"; tipos de dados "string", "integer", outros).
9. Não utilizar como base a ontologia universidade.owl.

Especificação básica

Especifique (cada especificação a seguir deve ser feita em 1 ou 2 parágrafos, com 10 a 20 linhas):

10. **(6 pts) Cenário** (descreva o contexto e a organização na qual o problema organizacional será resolvido) (e.g., biblioteca, agência bancária, loja de roupas presencial ou virtual)
11. **(6 pts) Processo de trabalho** (identifique e descreva o processo de trabalho que será foco da solução proposta) (e.g., processo de controle de usuários, processo de controle de estoque, processo de venda, …)
12. **(6 pts) Problema a ser resolvido** (descreva o problema) (e.g., "Controlar quais usuários estão com livros emprestados"; "Identificar quais usuários estão com entregas em atraso")

As especificações a seguir devem ser apresentadas em tabelas com vários itens cada:

13. **(3 pts) Requisitos de software** a serem implementados e **forma de implementação** (mínimo de 3 requisitos específicos para "**problema a ser resolvido**") (criar tabela) (e.g., Especificar formato de dados ⇔ Criação de ontologia; Identificar/Listar usuários/produtos que são do tipo X/que custam mais do que Y… (procure ser específico neste requisito no sentido de ser capaz de fazer uma consulta SPARQL na sua ontologia para resolvê-lo (nas tabelas seguintes e na Atividade Avaliativa 03), sendo que a indicação é que as consultas SPARQL acessem mais de uma classe da sua ontologia e das ontologias agregadas ⇔ Cadastro de indivíduos usando Protégé OWL ou Protégé Web; Consultar usuários em atraso ⇔ "Fazer consulta SPARQL")
14. **(3 pts) Modelagem de dados** (identificar em quais ontologias/classes/propriedades cada um dos requisitos irão impactar) (criar tabela à parte ou incorporar tabela de **Requisitos de software**)
15. **(3 pts) Perfil de usuários** (criar tabela de funcionalidades por usuário) (identificar perfis de usuários do sistema e as várias funcionalidades que cada um poderá usar) (e.g., Administrador ⇔ Criação ontologia OWL com Protégé, Gerente, Estagiário ⇔ Consultar, …)
16. **(3 pts) Requisitos de interface** (identificar como será a interface para acesso às funcionalidades) (criar tabela) (associar Requisitos de software ⇔ Usuário(s) ⇔ Descrição de requisitos de interface com identificação de ambiente) (identificar parâmetros de entrada e saída)

Solução (coloque suas respostas a partir daqui)

Para facilitar a modelagem nas Atividades seguintes, procure **marcar aquelas palavras que se tornarão classes**, **propriedades** ou **restrições** de consultas SPARQL em negrito no texto de sua solução. Por exemplo, "**Cenário**: as Pessoas relacionam-se dentro da organização por meio de um aplicativo de **Mensagens**…" ou "**Cenário**: o **parque de diversões** é um negócio no qual relacionam-se **Funcionários**, **Clientes** e **Fornecedores**…". Essa estratégia vai facilitar a identificação de quais palavras correspondem a classes da ontologia.

Especifique (cada especificação a seguir deve ser feita em **1 ou 2 parágrafos, com 8 a 20 linhas**):
EXEMPLO DELINEAMENTO PARCIALMENTE RESPOSTAS POSSÍVEIS (a seguir)

Especificação básica

Cenario

17. **Cenário** (descreva o contexto e a organização na qual o problema organizacional será resolvido) (e.g., biblioteca, agência bancária, loja de roupas presencial ou virtual)

Pokémon é um jogo eletrônico japonês criado em 1996 que teve suas mais diversas interações nesses 22 anos, seja em videogames, celulares, jogos de cartas, dentre outros produtos. O jogo é uma aventura em que o personagem coleciona e captura monstros no jogo e os usa para batalhar com outros personagens. Cada monstro tem tipos, golpes, habilidades e pontos de status.

Para as batalhas existe o limite de 6 monstros por personagens, estrategicamente o recomendado é que cada monstro tenha uma variedade em seus golpes que os tornem mais efetivos aos outros monstros.

Processo de trabalho

18. **Processo de trabalho** (identifique e descreva o processo de trabalho que será foco da solução proposta) (e.g., processo de controle de usuários, processo de controle de estoque, processo de venda, …)

O processo de trabalho será o levantamento de dados para a montagem de uma equipe de 6 pokémon com golpes equilibrados.

Problema a ser resolvido

19. **Problema a ser resolvido** (descreva o problema) (e.g., “Controlar quais usuários estão com livros emprestados”; “Identificar quais usuários estão com entregas em atraso”)

O problema a ser resolvido é ser feita a escolha de forma mais equilibrada entre os tipos, tendo um equilíbrio de efetividade entre tipos de pokémon e golpes.

Tabelas de requisitos

Requisitos de software (especificar 3 requisitos)

ID	Requisito	Forma de implementação	Descrição
I001	Listar informações de movimentos e informações gerais	Consulta SPARQL	A consulta deve mostrar o **nome do pokémon** e **seus dados.**
I002	Listar informações de movimentos e informações gerais	Consulta SPARQL	A consulta deve mostrar o **nome do pokémon** e **seus dados.**
I003	...		

20. **Requisitos de software** a serem implementados e **forma de implementação** (mínimo de 3 requisitos específicos para “**problema a ser resolvido**”) (criar tabela) (e.g., Especificar formato de dados ⇔ Criação de ontologia; Fazer cadastro de usuários ⇔ Cadastro de indivíduos usando Protégé OWL ou Protégé Web; Consultar usuários em atraso ⇔ Fazer consulta SPARQL)

Modelagem de dados

21. **Modelagem de dados** (identificar em quais ontologias/classes/propriedades cada um dos requisitos irão impactar) (criar tabela à parte ou incorporar tabela de **Requisitos de software**)

ID	Modelos de dados (ontologias/classes/propriedades/relacionamentos)
I001	Pokémon: nome e tipo. Golpes: NomeGolpe , tipoGolpe **Golpe temTipo Tipor**: qualidade pertencente à Pokémon e golpes.
I002	Usar SKOS:Concept: prefLabel para descrever a efetividade dos tipos

Usuários

22. **Perfil de usuários** (criar tabela de funcionalidades por usuário) (identificar perfis de usuários do sistema e as várias funcionalidades que cada um poderá usar) (e.g., Administrador ⇔ Criação ontologia OWL com Protégé, Gerente, Estagiário ⇔ Consultar, …)

NÃO SEI COMO APLICAR NO PROBLEMA ESCOLHIDO

Perfil do usuário	Funcionalidade
Analista do Banco Central	Consultas I001 Consulta I002 (listar a área de atuação de todas as empresa)
Administrador da Empresa/Banco	Consultas I002 (no caso de consulta com usuário Administrador listar apenas a área de atuação da própria empresa)

Requisitos de interface

23. **Requisitos de interface** (identificar como será a interface para acesso às funcionalidades) (criar tabela) (associar Requisitos de software ⇔ Usuário(s) ⇔ Descrição de requisitos de interface com identificação de ambiente) (identificar parâmetros de entrada e saída)

ID	Usuário	Descrição requisito de interface
I001	Bulbasaur	Receber o **tipo** do Pokémon Devolver **nome** do Pokémon, **tipo** e **golpes** do **Pokémon**